CHAPTER 5

Studying Software Design Cognition

Jeff Wai Tak Kan

Independent Design Researcher

John Gero

Krasnow Institute for Advanced Study

CONTENTS

5.1 INTRODUCTION

What do software designers do when they design? While there have been many studies of software designers, few if any have focused on their design cognition—the cognitive activities while they design. How does a team of software designers expend their cognitive effort during a design session? This chapter addresses this question by presenting a case study of a team of software designers where their cognitive effort is measured and analyzed.

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 protocol studies of designers, there is no uniformity in the segmentation and coding of design protocols. The papers in DTRS2 [1] and DTRS7 [3] demonstrate the range of methods and coding schemes applied to studying the same raw data. This chapter uses a generic coding scheme to study a team of professional software designers. By using this generic method, the results from this study can compared with those from other studies that use the generic scheme.

5.2 FUNCTION–BEHAVIOR–STRUCTURE 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 [4] defines ontology, apart from its use in philosophy, as an explicit specification of a conceptualization. The 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 [5] viewed designing as a purposeful act with the goal to improve the human condition. Gero [5] stated: "The meta-goal of design is to transform requirements, more generally termed functions which embody the expectations of the purposes of the resulting artifact, into design descriptions. The result of the activity of designing is a design description."

The coding of the protocols will be based on a general design ontology, namely, the function–behavior–structure (FBS) ontology [5] as a principled coding scheme (as opposed to an ad hoc one) (Figure 5.1).

The FBS design ontology [5], as a formal model, represents designing in terms of three fundamental, nonoverlapping 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 5.1 shows the eight FBS design processes in relation to the function (F), behavior (B), and structure (S) state variables. The formulation ($R \rightarrow F$, $F \rightarrow Be$) is the transformation of the design intentions from requirements, expressed in terms of functions, into expected behavior. Synthesis ($Be \rightarrow S$) is the transformation of the expected behavior (Be) into a solution structure. Analysis ($S \rightarrow Bs$) is the derivation of "actual" behavior from the synthesized structure (S). Evaluation ($Bs \leftrightarrow Be$) 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 ($S \rightarrow D$) is the production of the design description.

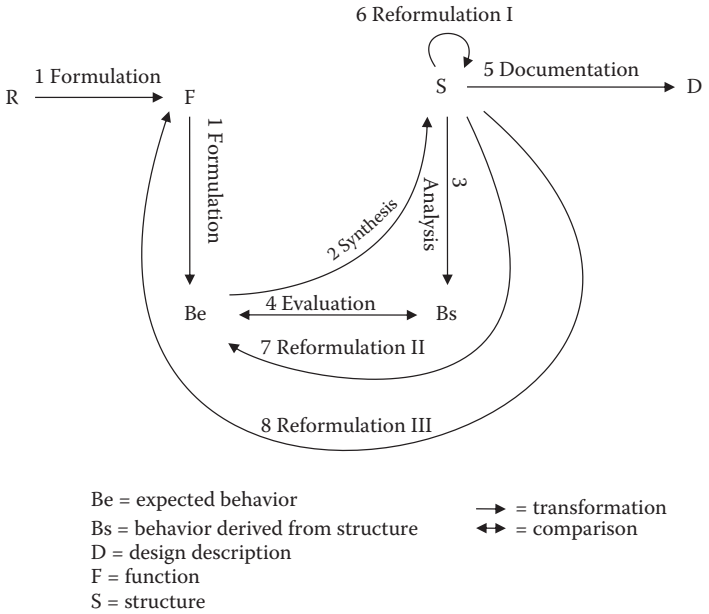

FIGURE 5.1 The FBS ontology of design processes.

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 aspects of designing. These have not been well articulated in most models because they have not been adequately understood.

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

5.2.1 Software Designing and FBS

The basis of the FBS view is that all design tasks, irrespective of their 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 the resulting behaviors (Be and Bs) and structure (S). Therefore, one significant advantage that 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 the structure (S) variables will usually be some physical reification. For example, to design a portable shelter, the function (F)

of the shelter can be formulated to an expected behavior (Be) of a space with a 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 its 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 the software they design. With these expected behaviors, they can synthesize the objects or the relationships of the codes of those objects. With these objects, they can derive their behaviors by running that part of the program, or 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 the drag-and-drop environment of the input, and the signal timing of the intersections. An example of the behavior derived from the structure is the time for cars to travel between intersections.

AU: The sentence beginning 'With these objects...' has been changed. Please check the meaning.

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. 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. 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 the 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.

5.3 OBSERVATIONS OF THE SESSION

In this section, we present some basic qualitative and quantitative observations of the Intuit protocol session in the National Sciences Foundation workshop on Studying Professional Software Design (http://www.ics.uci.edu/design-workshop/index.html). Since we are concerned with showing the applicability 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.

5.3.1 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, if not all, of the drawing and writing on the whiteboard. They spent more than 5 minutes at the beginning to protocol the design brief individually without verbalizing. Then, Male 1 suggested to separate the user interface (UI) from the underlying data structure and to focus on the types of data; a few entities/variables were proposed—"signals," "roads," "cars," etc. Afterward, they examined if it was necessary to have the "lanes" as an object, the expected behaviors of the software were discussed, and the scenarios of traffic at intersections were simulated on the whiteboard, especially with the left-turning situation. With the "signal" object, the expected behavior of the "rules" to control the signals was reintroduced by Male 2; he suggested very early on that in the program structure there are rules applied to each intersection. "Intersection" as an entity and its control were discussed. With these, they established cases with the discussion of "distance," "speed," and "travel time"; an object-oriented software structure was proposed at around 30 minutes. "Use cases" was proposed and physical structure analogies, such as "protected left," "add car," "road becomes a queue," "a map contains intersections," and "meta-controller" were deliberated. Manual simulations were carried out throughout the session to evaluate and design the required program. Near the end of the session, the structure of the program—different from the physical structure—was put forward.

5.3.2 Quantitative

The turn taking by the two designers was very even; Male 1 had 125 utterances and Male 2 had 123 utterances. The word counts of Male 1 and Male 2 were 2777 and 2894, respectively. Figure 5.2 shows the word count of the two individuals in 5-minute intervals without breaking up the utterances. It shows the variation of the number of words of both

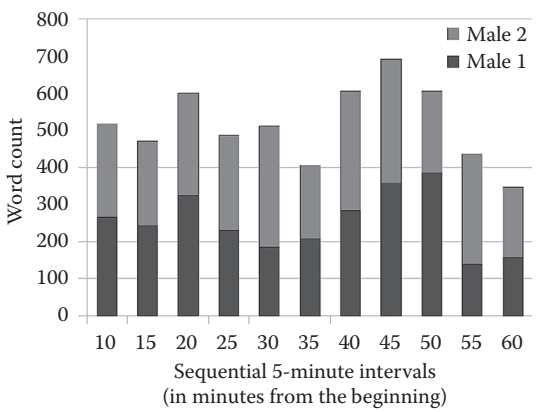

FIGURE 5.2 Word count in sequential 5-minute intervals over the protocol for Male 1 and Male 2.

designers across the session. After around 45 minutes, there was a constant drop in the word counts. At the 50 and 55 minute marks, there was a reverse of dominance.

5.4 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. Table 5.1 shows some examples of the segmentation and coding.

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 cognitive focus of designing in terms of the 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 intercoder 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 the agreement between the coders and the final arbitrated results was 90.4%. The high percentages of agreement provide additional support for the claim that the FBS ontology-based coding methodology provides a consistent and coherent way to view a protocol by different people. In the absence of such

TABLE 5.1 Examples of Protocol and Codes

Segment	Name	Transcript	Code
1	Interviewer	Feels like school again	O
2	Male 1	Well, I want to start by hearing your summary of this	O
3	Male 2	Gotcha, well. Looks like basically two pieces:	R
…	…	…	…
27	Male 1	you know I think about this as a software application.	F
28	Male 1	Looks totally like you want to pull out some kind of patterns and some kind of patterns and [inaudible] controller type of thing.	Be
29	Male 1	And so if we extract the UI piece first,	S
30	Male 1	and then let's focus on kind of the underlying stuff	S
31	Male 1	in order to support you know kind of the traffic flow.	Be
…	…	…	…
38	Male 1	So focus on the data pieces for this particular thing.	S
39	Male 1	Writes: "Data"	D

a unifying, guiding ontological framework, different coders can produce widely differing segmentation schemes, making any kind of quantitative analysis not robust and, more significantly, the results from individually generated coding schemes cannot be compared unless the codes can be directly mapped onto each other.

5.5 RESULTS AND ANALYSIS

5.5.1 Descriptive Statistics

The segmented protocol contains 640 segments, 603 of which are design issue codes, those that do not relate to design issues are coded as "O" and they are not counted further in this analysis. Figure 5.3 shows the distribution of the FBS codes. Male 2 referred to the requirements more than Male 1, but overall the "R" coded segments were sparse (17 in total). Only three function (F) issues were 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. The structure (S) and expected behavior (Be) issues dominated this session. Under the FBS ontology, the interaction of S and Be can be either a synthesis process (Be → S) or a reformulation process (S → Be′). A Markov chain will be used in the next subsection to analyze this kind of interaction.

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 5.4 shows the change of distributions of the FBS design issues across the three thirds. Changes can be observed qualitatively in this figure and statistical analyses can be carried out to confirm the qualitative results.

Requirement issues were raised mostly in the first third of the session with Male 2 contributing double the number of requirement issues. In the second and third parts of the session, each of the participants only raised one requirement issue. In this design session, function issues were rare and only occurred in the first third of the session—no function issue was observed in the second and third thirds of the design session.

Expected behavior issues represented one-third of the cognitive effort of these software designers. Male 1 contributed more behavior issues than Male 2 in the first third of the session. Their roles reversed as the design session progressed. This is an indication of a change

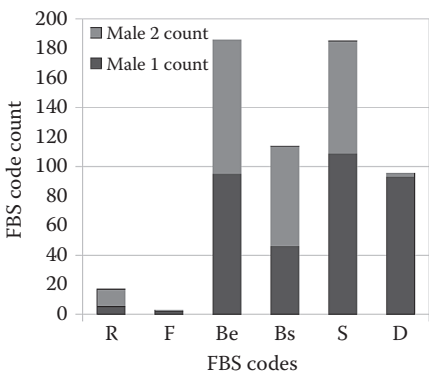

FIGURE 5.3 The distribution of design issues via FBS codes for Male 1 and Male 2.

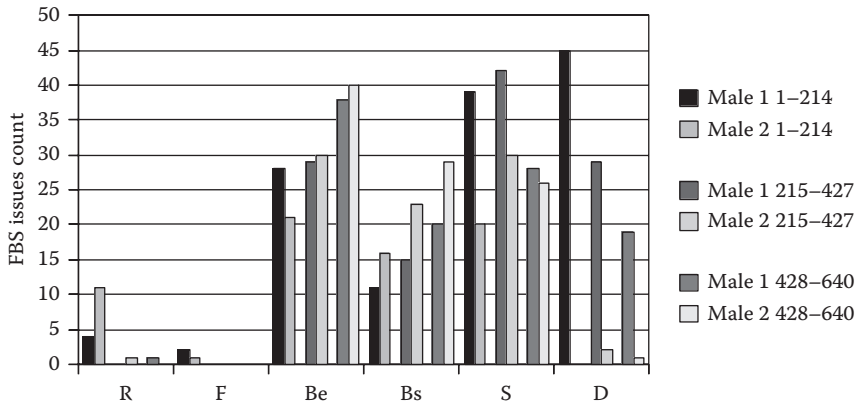

FIGURE 5.4 The design issue, via FBS code, distributions for the beginning, middle, and end of the session for Male 1 and Male 2.

in the relationship between the two team members over time. There was a steady increase in the focus on behavior from structure as the design progressed. This change in cognitive focus is in line with theories about software design.

The number of structure issues peaked in the middle of the session with Male 1 being more dominant. Like other design domains, structure is the dominant design issue in software design.

Unlike domains involving the design of physical artifacts, the action of documentation decreased as the session progressed. Male 1 did most of the documentation, while Male 2 did not have any documentation action in the first third of the session. We can also observe that Male 1 focused more on structure while Male 2 focused more on behavior. This division of contribution within a team is an indication of either background experience or role playing and requires further analysis to disentangle.

5.5.2 Sequential Analysis

Traditionally, design protocols have been analyzed by static statistical methods that are often based on the assumption that each segment is an independent event. Markov chains can be used to model or describe the probability of one event leading to another. Kan [6] proposed using a Markov analysis to examine the sequence of FBS states. Kan and Gero [7] used a Markov analysis to compare multiple design sessions. Markov chains have also been used to analyze writer's manuscripts and to generate dummy text [8]; to rank web pages by Google [9]; and to capture music compositions and synthesize scores based on analyses [10], to name some of its applications.

A probability matrix P is used to describe the transitions of the six FBS states (R, F, Be, Bs, S, and D). Equation 5.1 shows the transition matrix of this software design session.

There are only three occurrences of the F event, which result in a round off to zero in many of the rows and columns containing F. The occurrences of R are also low with some round off to zero as well. Any nonzero values of the probability of the transitions from R to

R are therefore less reliable than other transitions. *P* is one characterization of the dynamics of the entire 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 (.67).

$$P = \begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & .29 & .00 & .12 & .00 & .41 & .18 \\ F & .00 & .00 & .67 & .00 & .00 & .33 \\ Be & .00 & .01 & .21 & .23 & .39 & .16 \\ Bs & .02 & .00 & .47 & .15 & .29 & .07 \\ S & .03 & .01 & .26 & .20 & .19 & .31 \\ D & .01 & .00 & .42 & .15 & .43 & .01 \end{array} \quad (5.1)$$

AU: P, F, Be, Bs, etc. need to be consistent regarding whether they are italic or not. Would you like us to make them all italic or all not italic throughout? Please advise.

As mentioned earlier, the occurrences of Be and S issues dominated the session, and the probabilities of issues following these two issues characterized this session. The most likely issue that follows a Be issue is S (.39). The transition from Be to S is a synthesis process. Table 5.2 shows three sequences of Be and S issues that are synthesis processes. In the first one from segment 153 to segment 154, Male 1 synthesized the "hierarchy" of rules for the signals from the expected behavior of traffic at an "intersection." In the second sequence, 156–157, two program structure entities were proposed that were synthesized from the expected behavior of "has to synchronize these signals." In the third sequence, 160–161, the expected behavior of "controlling the interactions" translated to "an encapsulated entity."

AU: Please check that the edit made to the sentence 'As mentioned earlier ... session, conveys the intended meaning.

TABLE 5.2 Examples of S after Be from the Protocol that Are Synthesis Processes

Segment	Name	Transcript	Code
153	Male 1	The intersection needs to say that (gesturing over drawing and words) only one—only safe ones go on at a time so if these are green (gesturing over drawing) these have to be red and …	Be
154	Male 1	… so there's this kind of this hierarchy where the intersections own (draws …) the signals and …	S
155	Male 1	Draws lines and writes S1 S2 S3 that linked to intersections	D
156	Male 1	… so we have this behavior where somebody has to synchronize (drawing circle to enclose S1 S2 and S3) these signals, and so does that occur in the intersection?	Be
157	Male 1	As a containing data entity? Or conceptual entity?	S
158	Male 2	It sounds like more and more like the intersection comes from [inaudible] …	Be
159	Male 2	… because basically it's going to have given (gesturing over intersections and S1) S1 goes green, it's going to have to delegate the actions of what S2 and S3 are; is it safe from stuff like that	Bs
160	Male 1	Exactly, exactly. Somebody is controlling the interactions	Be
161	Male 1	If you think of this as kind of an encapsulated entity	S
162	Male 1	Draws circle enclose S1	D

The most likely issue that follows an S issue is D (.31), which is a documentation process. An example is the transition from segment 154 to segment 155 in Table 5.2, where the drawn lines represent the hierarchy and the ownership of the "intersections" entity. Another example is from segment 161 to segment 162, where the "encapsulated entity" was drawn as a circle.

The second most expected issue that follows an S issue is Be (.26); segments 157 and 158 in Table 5.2 are an example. We infer from the context that Male 2 was suggesting/expecting the behavior of the intersection based on the proposed entities in segment 157.

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.

$$P_1 = \begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & .33 & .00 & .13 & .00 & .33 & .20 \\ F & .00 & .00 & .67 & .00 & .00 & .33 \\ Be & .00 & .02 & .09 & .16 & .42 & .30 \\ Bs & .04 & .00 & .40 & .16 & .28 & .12 \\ S & .07 & .04 & .16 & .14 & .16 & .44 \\ D & .03 & .00 & .40 & .18 & .43 & .00 \end{array} \tag{5.2}$$

$$P_2 = \begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & .00 & .00 & .00 & .00 & 1.00 & .00 \\ F & .00 & .00 & .00 & .00 & .00 & .00 \\ Be & .00 & .00 & .26 & .08 & .49 & .17 \\ Bs & .00 & .00 & .40 & .23 & .34 & .03 \\ S & .01 & .00 & .26 & .24 & .23 & .26 \\ D & .00 & .00 & .37 & .17 & .43 & .03 \end{array} \tag{5.3}$$

$$P_3 = \begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & .00 & .00 & .00 & .00 & 1.00 & .00 \\ F & .00 & .00 & .00 & .00 & .00 & .00 \\ Be & .00 & .00 & .24 & .39 & .31 & .07 \\ Bs & .02 & .00 & .57 & .09 & .25 & .07 \\ S & .00 & .00 & .38 & .21 & .17 & .23 \\ D & .00 & .00 & .53 & .05 & .42 & .00 \end{array} \tag{5.4}$$

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 .67 and in the second and final thirds it is .00. 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 quantitative model of designing behavior.

If we assume that each segment is cognitively related to its immediately preceding segment and that the transition probabilities indicate design processes, the first third of the design session had high transition probabilities of formulation (F → Be: .67), documentation (S → D: .44), reformulations I and II (D → S: .43, D → Be: .40), and synthesis (Be → S: .42). The second third of the design session had high transition probabilities of synthesis (Be → S: .49), reformulation I (D → S: .43), and evaluation (Bs → Be: .40). The last third had high transition probabilities of evaluation (Bs → Be: .57) and reformulation II (D → Be: .53).

If we only examine the highest transition probabilities of each third, they are P_1: .67 (F → Be), P_2: .49 (Be → S), and P_3: .57 (Bs → Be). This follows the FBS ontology of designing that starts with formulation (F → Be: .67), then synthesis (Be → S: .49) followed by evaluation (Bs → Be: .57). This also confirms the analysis-synthesis-evaluation model of designing, where the formulation in the FBS model is a more articulated form of "analysis" as described in other models.

If we select the probabilities of those transitions that represent the eight design processes in Figure 5.1, we can plot the design processes of this design session for the beginning, middle, and end of the session (Figure 5.5).

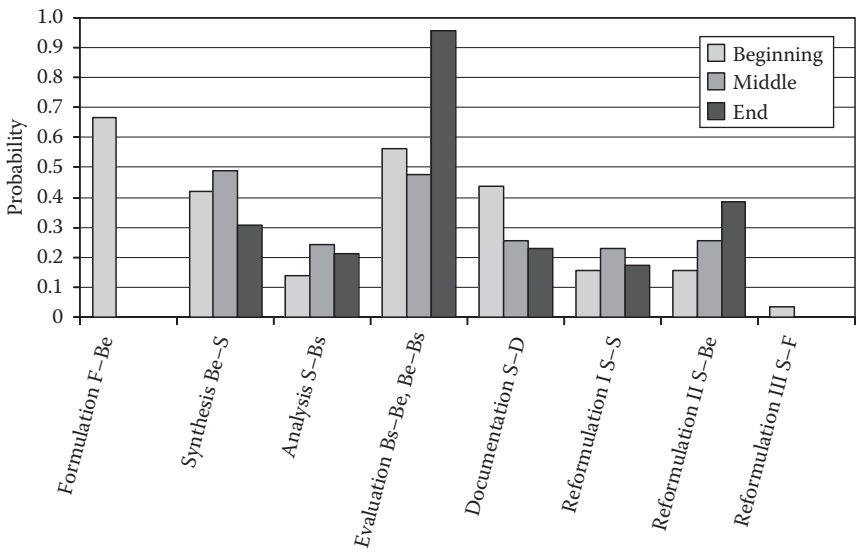

FIGURE 5.5 The transition probability distributions for the eight design processes for the beginning, middle, and end of the session.

From the transition probabilities in Figure 5.5, we can infer the behavior of this design session in terms of design processes. Formulation only occurs at the beginning of the session. Synthesis increases in the middle of the session and then drops off. Analysis is low at the beginning. There is a significant increase in evaluation and reformulation II at the end of the session. The transition probability matrix results are concordant with the qualitative analysis and they provide the basis for further quantitative characterizations of this design session such as the mean first passage time.

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 [11] proved that the mean first passage matrix M is given by

$$M = \left(I - Z + EZ_{dg}\right)D \tag{5.5}$$

where

I is an identity matrix

E is a matrix with all entries of 1

D is the diagonal matrix with diagonal elements $d_{ii} = 1/\alpha_i$

Z is the fundamental matrix such that

$$Z = \left(I - (P - A)\right)^{-1} \tag{5.6}$$

Z_{dg} is the diagonal matrix of Z

A is the probability matrix, with each row consisting of the same probability vector $a = (\alpha_1, \alpha_2, \ldots, \alpha_n)$, such that $\alpha P = \alpha$.

Equation 5.7 describes the number of FBS states expected to be passed through from one state before reaching other FBS states.

$$M = \left(\begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 49.1 & 168.4 & 3.9 & 6.7 & 2.5 & 5.0 \\ F & 70.0 & 168.4 & 1.9 & 6.2 & 3.7 & 4.5 \\ Be & 69.3 & 167.3 & 3.2 & 5.0 & 2.7 & 5.3 \\ Bs & 68.3 & 168.2 & 2.5 & 5.4 & 2.9 & 5.8 \\ S & 67.6 & 166.7 & 3.1 & 5.2 & 3.2 & 4.6 \\ D & 68.6 & 168.1 & 2.7 & 5.4 & 2.6 & 6.0 \end{array}\right) \tag{5.7}$$

Equations 5.8 through 5.10 are the unnormalized mean first passage times matrices of the design session divided into thirds: 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 the 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 time from R to R in the first third is 20.1 and it then increases to around 190 in the remainder of the session; the mean length of the passage time from F to F in the first third is 61.3 and it then increases to infinity (meaning there is no return to F) in the remaining two thirds; 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.

$$M_1 = \begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 20.1 & 61.6 & 4.4 & 8.3 & 2.8 & 3.6 \\ F & 30.4 & 61.3 & 2.1 & 7.7 & 3.6 & 3.1 \\ Be & 29.6 & 60.0 & 4.1 & 6.7 & 2.6 & 3.2 \\ Bs & 28.8 & 61.3 & 3.1 & 6.8 & 2.9 & 3.8 \\ S & 27.9 & 59.6 & 3.8 & 6.9 & 3.3 & 2.8 \\ D & 29.1 & 61.1 & 3.2 & 6.7 & 2.6 & 4.2 \end{array} \tag{5.8}$$

$$M_2 = \begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 192.8 & \infty & 4.3 & 6.6 & 1.0 & 6.4 \\ F & 194.2 & \infty & 4.2 & 6.9 & 3.4 & 6.8 \\ Be & 193.9 & \infty & 3.3 & 6.4 & 2.1 & 5.6 \\ Bs & 194.3 & \infty & 2.9 & 5.6 & 2.5 & 6.6 \\ S & 191.8 & \infty & 3.3 & 5.6 & 2.8 & 5.4 \\ D & 194.1 & \infty & 3.0 & 6.0 & 2.3 & 6.5 \end{array} \tag{5.9}$$

$$M_3 = \begin{array}{c|cccccc} & R & F & Be & Bs & S & D \\ \hline R & 188.8 & \infty & 3.3 & 5.0 & 1.0 & 8.8 \\ F & 187.7 & \infty & 3.3 & 4.8 & 4.4 & 9.8 \\ Be & 187.1 & \infty & 2.6 & 3.3 & 3.3 & 9.1 \\ Bs & 183.8 & \infty & 2.0 & 4.3 & 3.4 & 9.2 \\ S & 187.8 & \infty & 2.3 & 4.0 & 3.7 & 7.8 \\ D & 188.2 & \infty & 2.1 & 4.4 & 2.9 & 9.6 \end{array} \tag{5.10}$$

What these mean passage times for this design session show is how formulation decreases over the duration of the session, while at the same time analysis increases. These are the quantitative measures of qualitative design behaviors.

AU: The sentence beginning 'What these mean...' has been changed. Please check the meaning.

Using different assumptions, we can obtain additional results in other dimensions. If we assume that design processes occur not with consecutive segments but with related segments, we can produce other data for design process behavior. For example, when there

is an F segment, it needs a connection to a later Be segment to form a formulation process (F → Be). With the mean first passage time, we can predict when the next Be will appear. Using Equations 5.8 through 5.10, we can measure those average first passage times that form the eight design processes shown in Figure 5.6.

The measured shortest mean passage time: for a formulation process (2.1) it was at the beginning of the session; for a synthesis process (2.1) it was in the middle of the session; for analysis (4.0) and evaluation (3.3) it was at the end of the session; and for a documentation process (3.3) it was at the beginning of the session. These are quantitative measures of qualitative behavior. Considerable detailed information can be derived from the data in Figure 5.6.

Figure 5.6 also shows that there is a change in the behaviors of analysis, reformulation II, and documentation when an S design issue occurs. At the beginning of the session, there was an average of 2.8 steps for a documentation process, which increased to 7.8 steps at the end of the session. Whereas the steps in the analysis process after an S design issue decreased from 5.2 to 4.0; and the steps in reformulation II decreased from 3.8 to 2.3. This matches the qualitatively observed behavior of a large number of structure entities being proposed and documented at the beginning of the session, while toward the end there was more simulation, evaluation, analysis of the proposed structure, and reformulation of the expected behavior.

5.5.3 Comparison with Another Domain of Designing

We compared 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 as the first third of that design session for comparison with the first third of the software design session. The design issue distributions of the first one third of the two design sessions are shown in Figure 5.7.

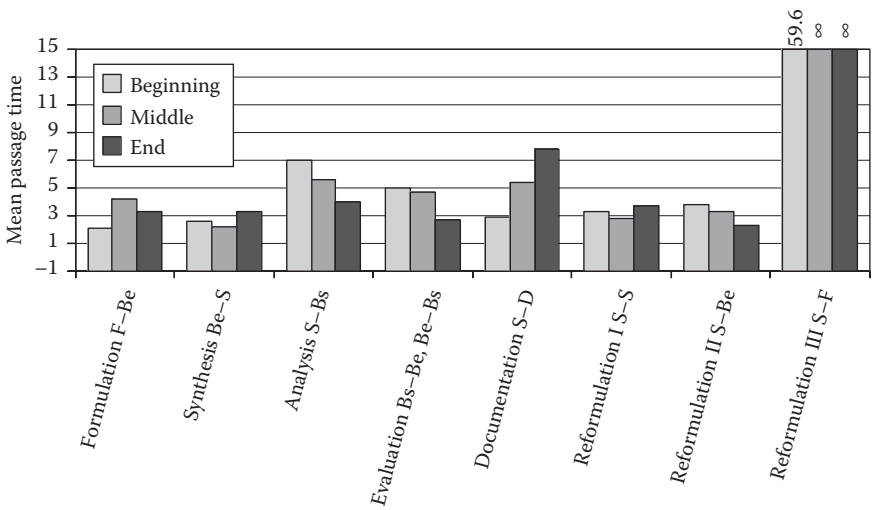

FIGURE 5.6 The distribution of the mean first passage times of the eight FBS design processes for the beginning, middle, and end of the design session.

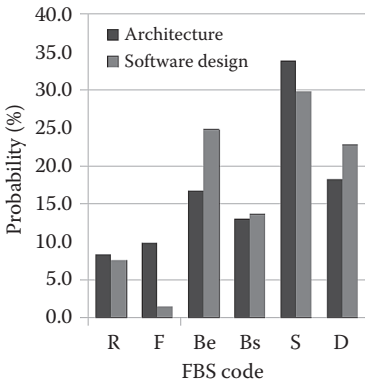

FIGURE 5.7 The design issue distributions of the software design and the architectural design sessions.

Since the same generic coding scheme was used for both domains, the results are commensurable. Figure 5.7 shows that the architectural design session had considerably more F design issues, less Be design issues, and more S design issues than the software design session.

These are interesting observations: they indicate 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 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 the 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.

AU: Please check that the edit made to the sentence 'These are... discussion' retains the intended meaning.

We are now in a position to compare quantitative results from a wide range of measurements both within a single domain and between domains.

5.6 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 have been shown to be derivable from the basic design issues that are the consequence of the segmentation/coding. Distributions of design issues and design processes across the 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 can be used to develop an understanding of the design cognition of software designers.

In this design session, the designers' cognitive resources were mostly focused on expected behavior and structure issues. These two design issues accounted for 60% of the cognitive

effort of this session. Formulation processes only occurred at the beginning of the session. There was an increase in synthesis processes in the middle of the session, which subsequently dropped off. Although the probability of analysis processes was lower at the beginning of the session, this session approximately reflected the analysis-synthesis-evaluation model of designing. There was a significant increase in the probability of evaluation and type II reformulation processes at the end of the session.

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 chapter, 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 a powerful common ground from which to analyze designing activity in various domains. It aids in developing rich insights into the 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.

AU: Latin expressions such as 'in vivo' and 'in vitro' are usually made italic. Would you like us to implement this in this chapter?

This ontology-based FBS coding scheme has been applied to statistically significant cohorts of designers and has been shown to be robust [12,13]. This chapter has demonstrated its utility in developing data from design protocols, data that form the basis of an understanding of the design cognition of software designers.

ACKNOWLEDGMENT

This research is supported in part by a grant from the U.S. National Science Foundation grant no. IIS-1002079. Any opinions, findings, and conclusions or recommendations expressed in this chapter 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.

REFERENCES

1. N. Cross, H. Christiaans, and K. Dorst, Introduction: The Delft Protocols Workshop, in *Analysing Design Activity*, Wiley, Chichester, pp. 1–15, 1996.
2. J. McDonnell and P. Lloyd, Eds, *DTRS7 Design Meeting Protocols: Workshop Proceedings*, Central Saint Martins College of Art and Design, London, 2007.
3. J. McDonnell and P. Lloyd, Eds, *About Designing: Analysing Design Meetings*, Taylor & Francis, London, 2009.
4. T. R. Gruber, A translation approach to portable ontology specifications, *Knowledge Acquisition*, 5(2), 199–220, 1993.
5. J. S. Gero, Design prototypes: A knowledge representation schema for design, *AI Magazine*, 11(4), 26–36, 1990.
6. W. T. Kan, Quantitative methods for studying design protocols, PhD dissertation, The University of Sydney, 2008.
7. 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, Springer, Berlin, pp. 123–134, 2009.
8. H. Kenner and J. O'Rourke, A travesty generator for micros: Nonsense imitation can be disconcertingly recognizable, *BYTE*, (12), 449–469, 1984.

AU: Ref. 2. Please verify publisher's name.

9. A. N. Langville and C. D. Meyer, Updating Markov chains with an eye on Google's page rank, *SIAM Journal on Matrix Analysis and Applications*, 27(4), 968–987, 2006.
10. M. Farbood and B. Schoner, Analysis and synthesis of Palestrina-style counterpoint using Markov chains, in *Proceedings of the International Computer Music Conference*, Computer Music Association, San Francisco, CA, pp. 471–474, 2001.
11. J. G. Kemeny and J. L. Snell, *Finite Markov Chains*, The University Series in Undergraduate Mathematics, D. Van Nostrand, Princeton, NJ, 1960.
12. 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 function–behavior–structure coding scheme, *Design Studies*, 32(1), 1–29, 2011.
13. J. S. Gero and U. Kannengiesser, Commonalities across designing: Evidence from models of designing and experiments, *Design Studies*, 2013 (in press).

AU: Ref. 10. Please verify the publisher's name and location and the page range.

AU: Please provide publication date, volume number and page range for Ref. 13 if now available.