LINKOgrapher: AN ANALYSIS TOOL TO STUDY DESIGN PROTOCOLS BASED ON FBS CODING SCHEME

Morteza Pourmohamadi[1] and John S Gero[2]
(1) The University of Sydney, Australia (2) Krasnow Institute for Advanced Study, USA

ABSTRACT
This paper presents LINKOgrapher, a software tool that carries out analyses on coded design protocols. LINKOgrapher is implemented building on an ontologically-based coding scheme utilising the Function-Behaviour-Structure (FBS) ontology. It aims at enabling cross-comparisons of different protocol studies through utilising a re-usable coding scheme and standardizing the analysis methods. Current measurements include tabular statistics, dynamic modeling of design issues and design processes, Markov models, first passage models and entropy models. The calculation and visualization of the results on the screen is near real-time, saving time and effort needed to analyse long design protocols. The results are exportable as graphic models as well as textual outputs. The measurement procedures and features of LINKOgrapher are discussed along with exemplary results.

Keywords: Protocol study, analysis toolkit, Function-Behaviour-Structure ontology, FBS coding scheme, design cognition

1. INTRODUCTION
Protocol analysis is the most commonly used method for studying design cognition [1-3]. It is a rigorous methodology that utilises verbalised thoughts of designers as empirical data to acquire knowledge about their cognitive activities [3]. It has been used extensively in design research to assist in the development of the understanding of the cognitive behaviour of designers [4-8].
A typical protocol analysis method consists of the following seven phases [9]:

1. Coding development
2. Recording verbalisations of designers
3. Transcribing the recordings
4. Segmenting and coding the transcriptions
5. Analysing the coded protocols
6. Generating the conceptual links
7. Analysing the linkograph

Quantitative analysis of design protocols is a costly research method, both in terms of time and resources. One of the possible ways of reducing the time and cost of such methods is to develop software tools to automate phases of the process. The tool presented here is developed to assist with the fifth and seventh steps of the above procedure by generating statistical models and graphs to be interpreted by the researcher. The ad-hoc nature of traditional protocol studies limits their use to the specific cases they have been developed for. Even the results of different studies over a single data set are not comparable in many cases [5, 10]. This case-dependency has been a major barrier for developing standardized measurement toolkits for the analysis of coded design protocols, as well as re-usability of the coding schemes. The toolkit described in this paper utilizes an ontological coding scheme, aiming to establish a common ground for analysis of design protocols.
Designing is not a unitary activity and it is unlikely that a single coding scheme will be capable of capturing all its nuances. One early attempt at producing uniform support for protocol analysis was the Protocol Analyst's Workbench [11]. It made use of a set vocabulary that could be extended by the user. Once the vocabulary was extended the results were no longer commensurable. MacSHAPA [12, 13] is another coding tool which is used in engineering design cognition studies. It was developed for use with sequential coding of videos and allows multiple overlapping codes. AFECS - A Flexible Expandable Coding Scheme [14] was proposed as a general-purpose approach. It was not considered

complete and its coding scheme has been extended when used in different situations resulting in incommensurability of its results.
However, as in all science, the claim is made that there is a regularity in designing that transcends any individual or situation and it is that regularity that is being studied. An ontology is one means to provide a framework for that regularity. Design cognition is based on the notion that designing involves processes that transform one design state or issue to another. Therefore, an ontology of designing should cover both issues and processes. Previous coding schemes were moving towards this. The AFECS approach [14] made use of 22 "Activities" that were mapped onto 8 "Objects" that appeared as an early ontology of designing. AFECS however, did not distinguish between issue and process in its Objects. Hughes and Park [15] modified AFECS to produce 7 Objects with 40 Activities within a four-level hierarchy. This means that there are 40 codes to choose from which generates cognitive overload in the coder and would appear to be too fine-grained for comparative analyses.
An ontologically-based design issues coding scheme founded on the Function-Behavior-Structure ontology of designing has been proposed for protocol analysis [16, 17]. In this ontology the codes are the design issues and the connections between the codes directly map onto design processes, as a result design processes are a consequence of the ontology rather than a separate part of the ontology. That the design processes are a consequence of the design issues is an important advance in a design ontology. The use of this ontology is grounded both in its utility and coverage in protocol studies and in the increasing references to it by design researchers.
This paper presents the Function-Behaviour-Structure (FBS) ontologically-based design issues coding scheme as a potential re-usable coding scheme which enables development of standardised analysis toolkits. One such toolkit called LINKOgrapher will be introduced as a software tool that has been developed on the basis of this coding scheme. The features and currently implemented analysis methods of LINKOgrapher will be discussed and future expansions for improving the tool will be described.

2. FBS-BASED DESIGN ISSUES PROTOCOL CODING SCHEME

The codes in this coding scheme are structured in accordance with the design issues defined by the FBS ontology [18, 19]. There are six codes that denote the design issues: Functions (F), Expected Behaviours (Be), Structures (S), Structural Behaviour (Bs), and Documents (D), as well as the design issues that arise from sources other than the designer (e.g. design brief) that are coded as Requirements (R). The FBS design issues coding scheme strictly allows for only one design issue for each segment producing a strict isomorphism between codes and segments. This feature eliminates any overlapping codes or multi-code segments in the coded protocol.
The aim of designing is to develop and transform functions into structures and finally documents. The design processes are defined as transitional processes between code pairs, Figure 1. Formulation (1) is the process of inferring expected behaviours from the functions and requirements. Synthesis (2) is the process of transforming expected behaviours into structure. Analysis (3) is the process of transforming structure into behaviours derived from structure. Evaluation (4) is the process of comparing analysed behaviours with expected behaviours. Documentation (5) is the process of external representation. Reformulations (6, 7 and 8) are the processes of changing the space of possible designs by changing the structures, behaviours or functions.

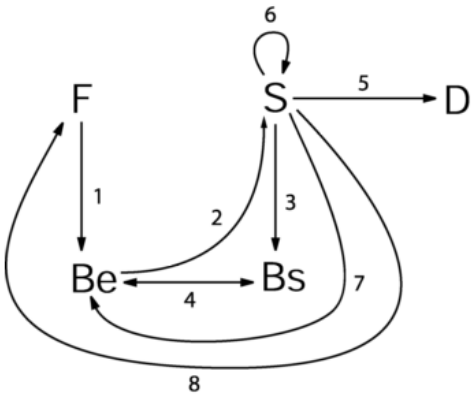


Figure 1. Design issues and their transition processes [18]

The FBS coding scheme has been adopted to code and analyse protocols in different design studies. The range of these studies stretches to various design disciplines, different number of designers and diverse design tasks [7, 8, 17, 20, 21]. This coverage and popularity suggests the potential of this coding scheme as a unifying analysis methodology across different design domains.

2.1 Ontological basis

FBS design issues coding scheme is based on the FBS ontology of design and its expanded version, situated FBS (SFBS) design ontology [18, 19]. These ontological theories are widely accepted in the field of design and engineering [22]. According to Google Scholar, the two introductory papers of this ontology are approaching 1,000 citations between them. This ontology creates a useful ground for interpretation of design issues across different design domains.

2.2 Distinct Segments

In the FBS design issues coding scheme, the segments in a coded protocol strictly map onto only one code, i.e. there is no overlapped or multi-coded segment [23]. This feature not only accounts for clearer sequential structure, but also it allows for better text-code relation. While a sequential data structure leads to many event-based or statistical analyses, a clear text-code relation creates the potential for different machine learning and language analyses in addition to the standard descriptive statistical analyses.

2.3 Consistent Code Values

The FBS ontology of design has defined only one level of design issue [18, 19]. Consequently, the codes in the FBS design issues coding scheme belong to the same level of importance, i.e. there is no high-level or low-level code. This property reduces the range of granularity of codes, but improves the consistency of the data, which is important in event-based and statistical analyses. In addition the nature of the codes is always consistent. All of the codes in FBS coding scheme are defined as design issues. There is no other type of the code. The design processes are transitions between codes and are a consequence of the coding and additional semantics. They are not a separate ontology.

2.4 Semantic/Syntactic Modes

The FBS design issues coding scheme allows both syntactic and semantic relations between codes. In syntactic mode, related codes are defined by their position in the sequence, i.e. the neighboring codes are considered to be related to each other. In the semantic mode it is the semantic relationship of the codes that defines their positions [9]. This deep insight into the conceptual connections of the codes is driven from the linkograph of the coded design protocol. Though linkography in design protocols is not a new method [24], using a linkograph network to extract semantic transitional processes is an innovative way of capturing the apparently unstructured nature of design sessions.

2.5 Reformulations

The codes in the FBS design issues coding scheme are pre-defined. However, fixed codes might be problematic if we consider how design spaces evolve and transform during a design session. Reformulation processes resolve this problem by addressing changes in design state space [20]. In other words, they allow for changes in design issues without requiring changes in codes.

These features of the FBS design issues coding scheme support its application as a re-usable coding scheme. Its ontological foundation allows for cross-domain and cross-case comparisons. Its semantic/syntactic modes and reformulation processes increases internal flexibility in each study. And finally, the consistency and distinction of codes in a sequential structure facilitate a broad range of statistical analyses.

The next step in generalising a coding scheme is to standardise the results of its analyses in order to make them comparable. LINKOgrapher, a software system, is an analysis tool that has been developed with this goal in mind [9]. Its input protocol is assumed to have been coded using the FBS design issues coding scheme, LINKOgrapher calculates general and tabular statistics, performs probability analyses and draws the linkograph as well as the resulting charts. The following is an outline of the features and structure of LINKOgrapher.

3. LINKOgrapher:

LINKOgrapher is a software tool that aims at automating the calculations in design protocol analysis and hence reducing the cost and time for doing such studies. It has been implemented using Processing Java IDE and is based on the FBS design issues coding scheme. In similar protocol analysis tools the coding scheme needs to be entered into the tool before it can carry out any analysis on the input data. LINKOgrapher has been developed on the foundation of the FBS design issues coding scheme and allows the researcher to initiate the analyses of data directly. LINKOgrapher inherits the re-usability properties of the FBS design issues coding scheme in analysing different protocols without any further manipulation. It also works as a visualisation tool to generate visual models from the calculated results and draws the linkograph from the conceptual links. The interface for LINKOgrapher is shown in Figure 2. The left navigation block shows the analyses available. The bottom navigation block allows the user to trim the protocol and control the window size for dynamic models. The central canvas is both the drag and drop import window and the output window.

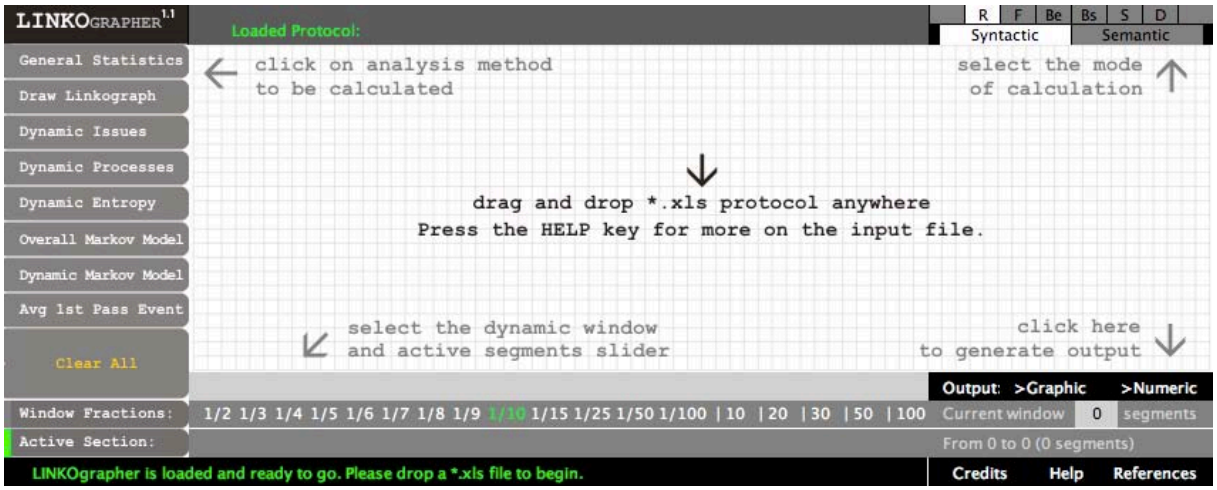


Figure 2. Screenshot from LINKOgrapher initial interface

The input data structure is transformed into a sequential string of codes that is the backbone representation of all analyses. Any segment with non-FBS codes will be ignored. In addition to its code, each segment is annotated with a segment number, backlink and its text passage. LINKOgrapher reads the input data and calculates general statistics for the imported protocol. Other analysis methods are available through navigating the keys in the left side of the window. Table 1 lists the analysis methods available in the version 1.1 of LINKOgrapher.

Table 1. Implemented analyses methods in LINKOgrapher version 1.1

Approaches	General Statistics	Probability Analyses	Dynamic Models
Analysis Methods	Segment and link counts Design issue distribution Design process distribution Backlink/Forelink counts	1st order Markov model for both syntactic and semantic modes 2nd order Markov model Average 1st pass event Linkograph entropy	Design issue distribution Design process distributions for both syntactic and semantic modes Dynamic entropy 1st order Markov model

3.1 General Statistics

At a basic statistical level, LINKOgrapher calculates descriptive statistics of the dataset such as counts, distributions and central tendencies, Figure 3. The number of segments, valid FBS codes and the means for distribution of link nodes in both vertical and horizontal axes [25] are the most general parameters of interest about any coded protocol. Where the semantic linkograph has been constructed the vertical distribution of nodes in the linkograph is an index for the overall distance of the links (how far the linked segments are apart). The horizontal distribution of the nodes denotes the issue activity during the design session.

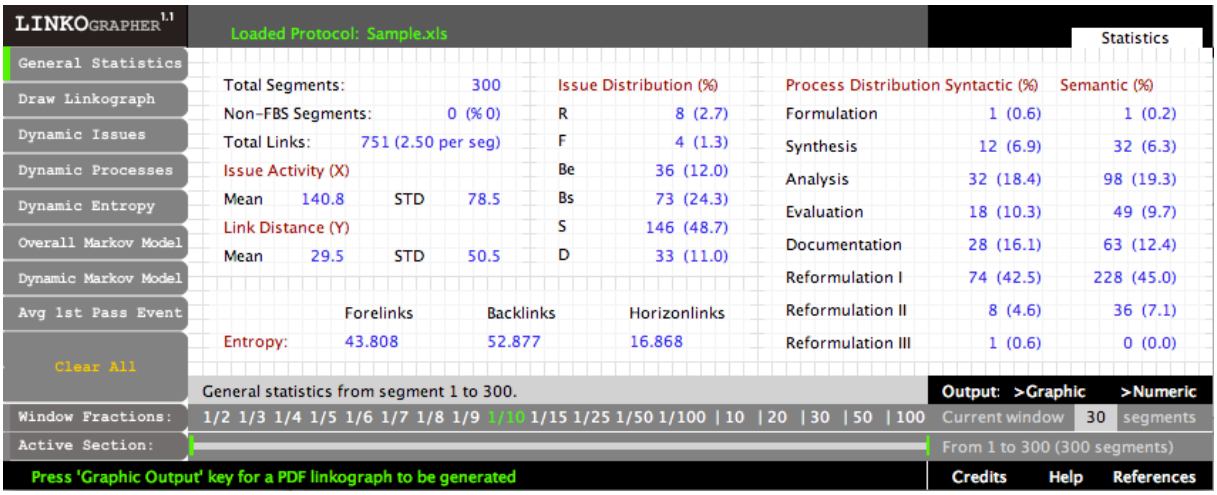


Figure 3. Screenshot from LINKOgrapher presenting the general statistics.

At a more detailed level, LINKOgrapher calculates the distributions of design issues and processes. The distribution of processes is calculated in both semantic and syntactic modes. In syntactic mode, the coded protocol is treated as a plain string of FBS codes: B>A is a valid transition process if A is the immediate segment before B. In semantic mode however, the conceptually linked segments are considered for counting the transition processes: B>A is a valid transition process if B is linked back to A in the linkograph.

3.2 Markov Models

Every coded design protocol is a chronological sequence of codes occurring through the design session. Considering each code as an event enables a series of probability analyses to study the possibilities of each coded event occurring after another event. LINKOgrapher utilizes this approach in calculating Markov models, average first pass events and entropy.

LINKOgrapher generates 1st and 2nd order Markov models. In the first order Markov model, the next state of the system only depends on its current state [26]. In the FBS design issues coding scheme, this is considered a weak model and is proposed to predict the probability of design issues coming after each other in strict time-order sequence [20]. The 1st order Markov model of a protocol is shown by a 6x6 matrix. Given that the current segment has the same code as the row label, the probability of the next segment having the same code as each column label is shown in the crossing cell.

The semantic and syntactic modes are used to generate two separate models for each dataset, Figure 4.

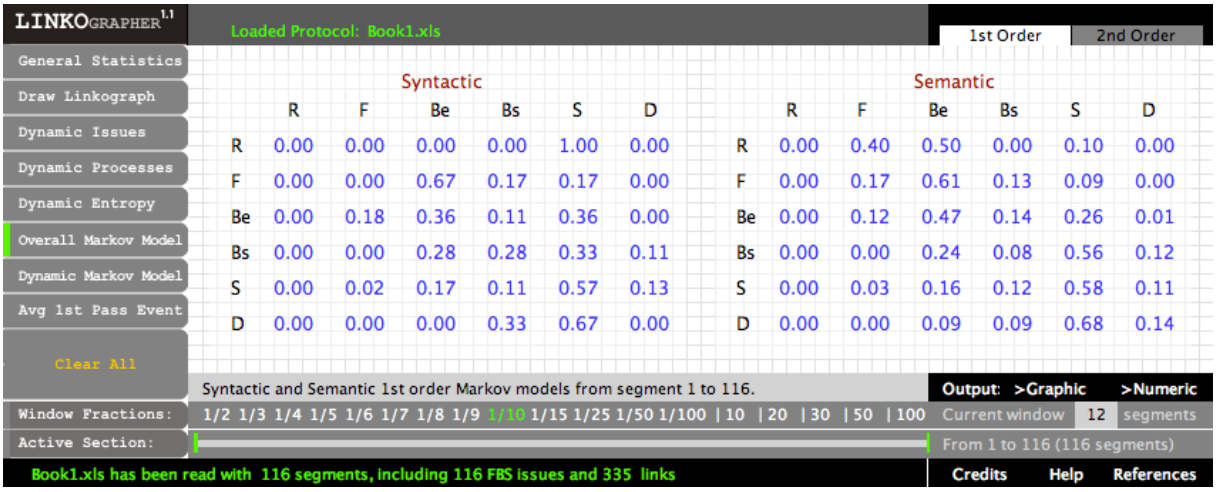


Figure 4. Screenshot from LINKOgrapher depicting the matrices for 1st order Markov models.

In the 2nd order Markov models, not only the current state but also its previous state affects the next state of the system, i.e., the system has a memory of its activity. With the FBS design issues coding

scheme, the move from a previous state to a current state is considered a design process [9]. Consequently, the 2nd order Markov model of a coded protocol describes the probability of a particular design issue following a particular design process. LINKOgrapher illustrates this model in an 8x6 matrix.

The average first pass event model, also called the average first passage time model, is another Markov model [26] to describe the probabilities of transiting states of stochastic systems. It determines how many states it takes to transit from one specific state to another. In analysing design protocols, the average first pass event model is used to define how many segments the designer needs to move from one design issue to another one. Since in coded design protocols, the sequence of codes is considered instead of the time, Gero et al [20] name this model as the average first passage event model instead of average first passage time model.

LINKOgrapher generates the average first pass event model for any given protocol in a 6x6 matrix. It also lists the longest and shortest runs between the design issues as ordered lists to facilitate easier reading.

A linked protocol is a rich source of information that can be analyzed in many different ways. In analyzing the entropy, the linkograph is considered to be a system in which every conceptual link between two segments is an event. In a syntactic manner, the nodes in the linkograph are messages that carry information about the occurrence of events. According to Shannon [27], the amount of the information carried by a message is defined by the probable states of the system. The more stochastic a system is, the more informative is a message about its state. Kan and Gero [28] argue that there is a potential correlation between the entropy of a linkograph and the productivity of the design activities.

LINKOgrapher is able to calculate the overall and dynamic entropies of the given dataset. The number of probable states at any moment is one of the parameters in calculating the entropy. In linkographs, this parameter could be calculated in three different ways, namely, using backlinks, forelinks and horizonlinks, Figure 5 [28]. Kan and Gero discuss each calculation mode and the conceptual interpretation of the resulting entropy. In backlink mode, each segment can be linked back to any of its previous segments. Consequently, the number of possible links at any moment equals the segment number minus one. In forelink mode, the number of remaining segments to the end of the protocol is considered as possible links for each segment. Horizonlink is not a link itself but it is an indicator of the distance between two linked segments. There are n-1 rows in a linkograph with n segments and the number of possible states in each level equals the length of the protocol minus the level of the links.

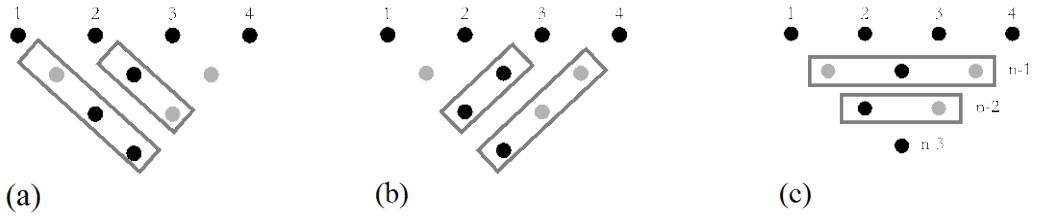


Figure 5. Measuring (a) backlink, (b) forelink and (c) horizonlink entropy [28].

3.4 Dynamic Models

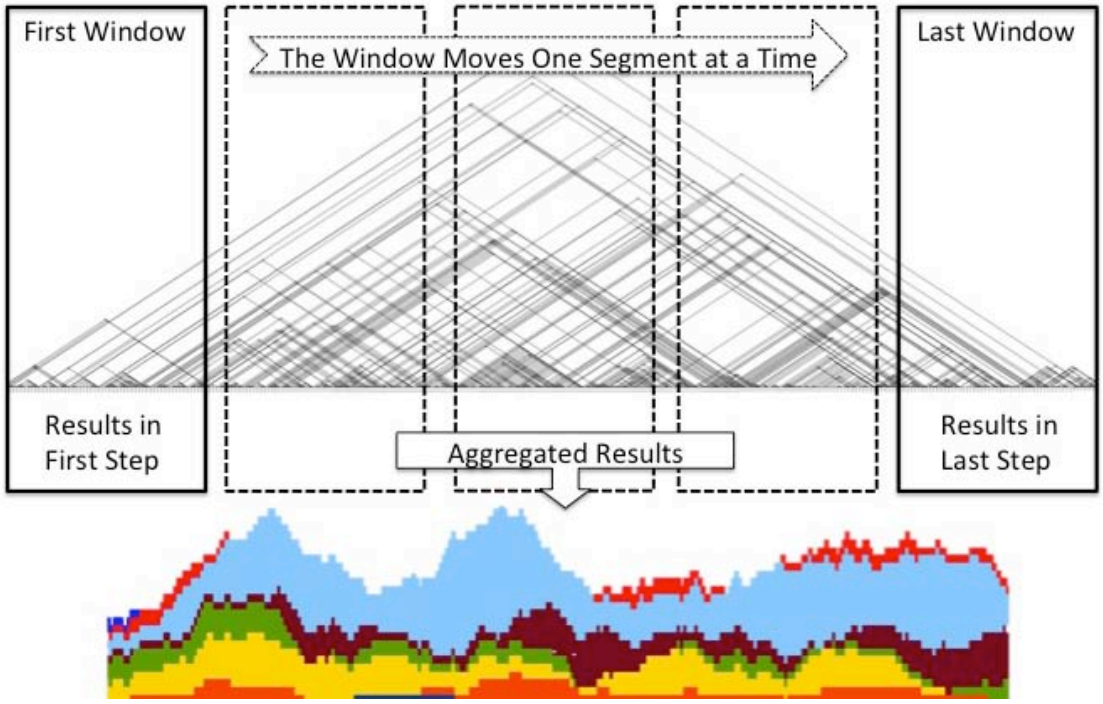
Though overall models are useful to illustrate general properties of design protocols, they miss a lot of information about changes that occur during any design session. In order to capture the dynamic nature of designing, LINKOgrapher treats the coded protocols using three additional approaches:

1. fractioning
2. windowing
3. trimming

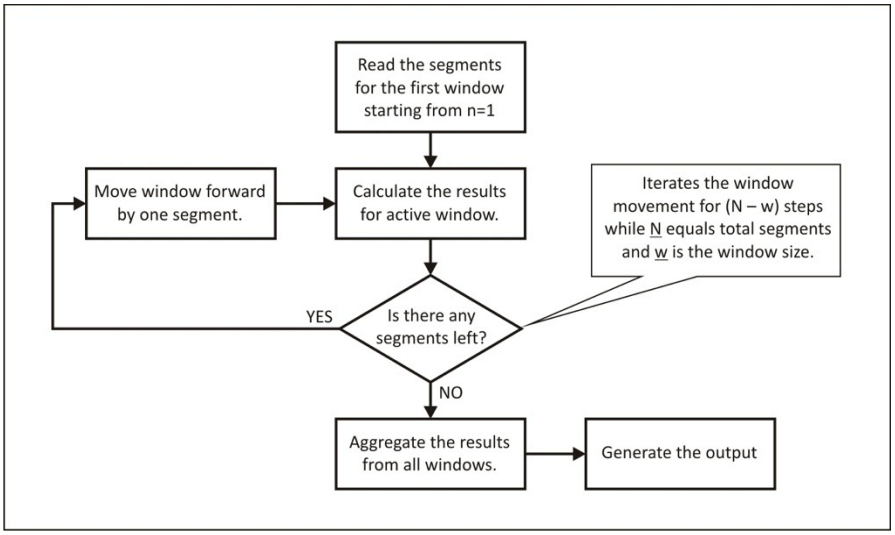
In fractioning, the input dataset is divided into sections and each section is treated individually. The resulting measurements account for the situation at the related fraction of the protocol, e.g., the beginning of the design session. Comparisons are possible between the fractions of a single case or similar fractions from multiple cases.

In windowing a fixed number of segments are selected and the analyses carried out for those as a window. The window then is moved from the beginning to the end of the protocol, one segment at a time. Aggregating the results gives an overall insight into the dynamism of the design session for each

parameter. A visual interpretation of the method is presented in Figure 6(a). The flowchart for this process is presented in Figure 6(b).



(a)



(b)

Figure 6. (a) Visual interpretation of windowing method with resulting graph, and (b) flowchart for windowing method to calculate dynamic results.

One of the problems in comparing multiple cases is differences in the lengths of design sessions. This feature allows for regulating the results of protocol studies to generate equally sized datasets. For example, if dataset A has 1000 and B has 1200 segments, performing an analysis with window sizes of 100 for A and 120 for B will generate similarly sized results, hence direct comparison becomes possible.

Trimming is the process of selecting a set of contiguous events and excluding all events preceding and following that set. This allows for analyses of a subset of the protocol by itself.

Applying the above treatments to different standard analyses generates the potential for the development of insightful results about the dynamism of designers' behaviours during the design sessions. LINKOgrapher calculates dynamic issues, processes, entropies and Markov models for the same input data.

Applying the same concept of dynamic models, LINKOgrapher calculates the dynamic distributions of design issues and processes through the design session, Figure 7.

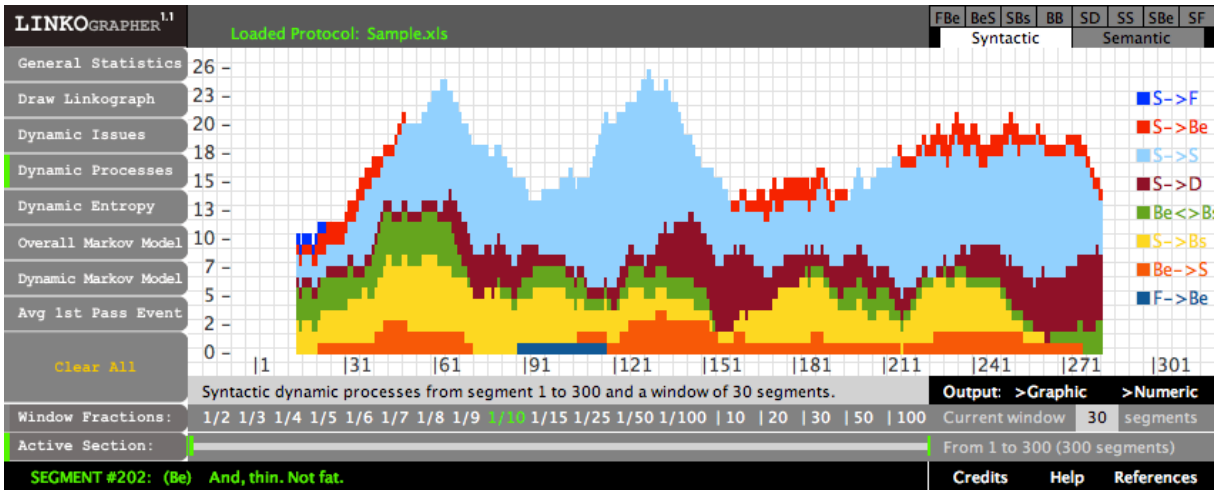


Figure 7. Dynamic distribution of design processes in syntactic mode. The results are viewable in both stacked and individual graphs.

3.5 LINKOgrapher Inputs and Outputs

A typical usage scenario using LINKOgrapher starts with recording the video/audio of design sessions and transcribing the verbalizations. The next step is to segment and code the transcriptions based on the coding scheme. Manual segmenting and coding is still one of the most elaborate and resource-consuming steps of protocol study. There are issues about coder bias, which are addressed by using two or more independent coders and arbitrating the final result. However, after finishing the coding and linking of the design protocol, LINKOgrapher can carry out the analysis and modeling of the data in a short time compared to a human and in a standardized manner.

Preparing the coded/linked design protocol is easy once it has been coded based on FBS coding scheme. The input is a Microsoft Excel spreadsheet with a defined format. Each line of the input file includes a verbalized phrase from the protocol along with its code in a separate column. The input file can be imported to the tool by dragging and dropping it on the tool's window. The general statistics of the imported dataset will be viewable once the dataset has been imported. Other analysis modes can be selected after this stage.

One of the main features of LINKOgrapher is its ability to generate dynamic graphs and present them visually. The visualized results change as any measurement parameter changes. This feature allows the researcher to explore the dataset in a direct way without concern about the calculations. The graphs can be shown as stacked or individually for each issue/process.

LINKOgrapher generates two types of outputs for all its results: textual and visual. The textual output is formatted as a plain text file with *.txt extension and is directly importable to spreadsheet software such as Microsoft Excel. The visual output is a Portable Document Format (PDF) file, which can be used directly or as an image in word processing software.

One of the useful graphical outputs of LINKOgrapher is the linkograph drawing. It uses the conceptual links defined in the input data set to generate the linked graph of the protocol in a PDF file. In addition to the links and their nodes, the drawing includes segment utterances and their annotations such as codes.

4. CONCLUSION

The Function-Behaviour-Structure coding scheme has properties that support its potential as a re-usable coding scheme to be applied to different cases independent of their domain, topic and number of designers. It relies on the FBS ontology of design to define a heterogeneous set of codes with discrete definitions. It allows both semantic and syntactic relations between design issues and allows for reformulation of issues during the design session without requiring any changes of the codes.

These properties are the basis for developing a software tool to automate analysis of coded design protocols.
LINKOgrapher is developed around the concept of enabling cross-case comparisons by standardizing the input and output format of analyses, as well as the generation of the models. Consequently, any results of any studies done using the FBS design issues coding scheme and LINKOgrapher will be comparable as they use the same basis for their analyses.
The LINKOgrapher tool presented in this paper reduces the time and effort needed to analyse coded design protocols. The direct generation of the results allows the researcher to focus on the research instead of calculation issues. Near real-time visualization of the results by the tool improves the capacity for explorative design protocol studies. LINKOgrapher is published as a free tool at www.LINKOgrapher.com and can be downloaded and used in design protocol studies.
LINKOgrapher can provide commensurable results for the design cognition of:

- individual designers
  - student designers
  - professional designers
- design teams
  - heterogeneous design teams
  - homogeneous design teams
- designer environments
  - co-located designers
  - remotely located designers
- designers using tools
- designers from different disciplines.

5. FUTURE DEVELOPMENTS

LINKOgrapher is a preliminary step toward creating a generic protocol analysis toolkit that is based on a re-usable coding scheme. The next version will be expanded in three areas. 1. To cover more steps from the process of protocol study, including the coding and linking phases that are the most time consuming steps in the preparation of protocols for analyses. 2. To structure the software code and data as an open-source tool which will create the opportunity for custom expansions such as introduction of new codes or different analyses. 3. To improve the visualization of the results and the quality of outputs.

REFERENCES

[1] Van-Someren, M., Barnard, Y. and Sandberg, J., *The Think Aloud Method: A Practical Guide to Modelling Cognitive Processes*. (Academic Press, 1994).
[2] Svenson, O., A note on think aloud protocols obtained during the choice of a home. (Psychological Laboratories, University of Stockholm, 1974).
[3] Ericsson, K. and Simon, H., *Protocol Analysis; Verbal Reports as Data*. (MIT Press, 1993).
[4] Cross, N., Dorst, K. and Roozenburg, N., *Research in Design Thinking*. (Delft University Press, 1992).
[5] Cross, N., Christiaans, H. and Dorst, K., *Analysing Design Activity*. (Wiley, 1996).
[6] Ennis, C. and Gyeszly, S., Protocol analysis of engineering systems design process. *Research in Engineering Design*, 1991, 3(1), pp15-22.
[7] McNeill, T., Gero, J. and Warren, J., Understanding conceptual electronic design using protocol analysis. *Research in Engineering Design*, 1998, 10(3), pp129-140.
[8] Kavakli, M. and Gero, J., The structure of concurrent cognitive actions: A case study of novice and expert designers. *Design Studies*, 2002, 23(1), pp25-24.
[9] Gero, J.S., Kan, J.W.T. and Pourmohamadi, M., Analysing design protocols: Development of methods and tools. In *Research into Design.* India. pp3-10 (Research Publishing, 2011)
[10] McDonnell, J. and Lloyd, P., eds. *About: Designing. Analysing Design Meetings*. (CRC Press, 2009).
[11] Fisher, C. Advancing the study of programming with computer-aided protocol analysis, *Empirical Studies of Programmers – Second Workshop* pp198-216 (Ablex, 1987)

[12] Sanderson, P., Scott, J., Johnston, T., Mainzer, J., Watanabe, L. and James, J., MacSHAPA and the enterprise of exploratory sequential data analysis (ESDA). *International Journal of Human-Computer Studies*, 1994, 41(5), pp633-681.
[13] Atman, C.J., Bursic, K.M. and Lozito, S.L., An Application of Protocol Analysis to the Engineering Design Process. In *ASEE Annual Conference Proceedings.* 1996.
[14] von Mayrhauser, A. and Lang, S., A coding scheme to support systematic analysis of software comprehension. *IEEE Transactions on Software Engineering*, 1999, 25(4), pp526-540.
[15] Hughes, J. and Parkes, S. , Tends in the use of verbal protocol analysis in software engineering research, *Behaviour & Information Technology*, 2003, 22(2), pp127-140.
[16] Gero, J.S. and Mc Neill, T., An Approach to the Analysis of Design Protocols. *Design Studies*, 1998, 19(1), pp21-61.
[17] Kan, J.W.T. and Gero, J.S., Using the FBS ontology to capture semantic design information in design protocol studies. In McDonnell, J. and Lloyd, P., eds. *About: Designing. Analysing Design Meetings*, pp213-229 (CRC Press, 2009).
[18] Gero, J., Design prototypes: a knowledge representation schema for design. *AI Magazine*, 1990, 11(4), pp26-36.
[19] Gero, J.S. and Kannengiesser, U., The situated Function-Behaviour-Structure framework. *Design Studies*, 2004, 25(4), pp373-391.
[20] Kan, J.W.T., Gero, J.S. and Sarkar, S., Using a Generic Method to Study Software Design Cognition. In *Workshop on Studying Professional Software Design.* 2010, pp1-7
[21] Tang, H.-H. and Gero, J., A cognitive method to measure potential creativity in designing. In *Workshop 17 - Creative Systems: Approaches to Creativity in AI and Cognitive Science, ECAI-02*. Lyon, 2002, pp47-54
[22] Gero, J.S., Generalizing Design Cognition. *Design Thinking Research Symposium*, Sydney, 2010).
[23] Kan, J.W.T. and Gero, J.S., A generic tool to study human design activity. In *Human Behavior in Design.* pp123-134 (Design Society, 2009)
[24] Goldschmidt, G.: 1990 Linkography: assessing design productivity", *in* R. Trappl (ed.), *Cybernetics and System '90*, World Scientific, Singapore, pp. 291-298.
[25] Kan, J. and Gero, J., Acquiring information from linkography in protocol studies of designing. *Design Studies*, 2008, 29(4), pp315-337.
[26] Kemeny, J. and Snell, J., *Finite Markov Chains*. (Springer, 1976).
[27] Shannon, C., A mathematical theory of communication. *The Bell System Technical Journal*, 1948, 27, pp397-423.
[28] Kan, J.W.T., Bilda, Z. and Gero, J.S., Comparing entropy measures of idea links in design protocols: Linkography entropy measurement and analysis of differently conditioned design sessions. *AIEDAM*, 2007, 21(4), pp367-377.