WHY SOFTWARE ENGINEERS ARE RELUCTANT TO USE STRUCTURED METHODS

Mashaushi K.R.S.

Abstract: This paper looks into the reasons for an apparent boycott of theoretically proven structured techniques and methodologies by most developers. It begins by discussing the fact that when a methodology for developing software contradicts with the way human brains solves problems, the result is stress and thence burnout to the software engineer. Points that explain the boycott are identified and explained. The paper then proposes further study considerations needed to improve the use of structured tools.

INTRODUCTION

There is evidence that many software developers do not like very structured tools for developing softwares. One indication of this is the massive use of prototyping techniques. Many of today's rapid-prototyping projects have completely abandoned traditional software engineering methods,diagramming techniques, and methodologies and have decide to plunge directly into Visual Basic, for example, to provide the user with a demonstrable prototype as soon as possible. For some environments this may be acceptable. For example for small and medium size client/server systems. But for large-scale systems, we still must use the more conventional approach to software engineering. We must be able to answer questions like these: (a) what is it that the user wanted us to do? (b) Does this system implement those requirements? (c) If we changed the priorities of various requirements, how do we just keep track of them? The argument that structured software methodologies are not liked, is provoked by evidence of their minimal use despite a perceived importance of their underlying concepts (Tesch, et al.1995). Table one summarizes Tesch's findings on the attitudes of Information Systems (IS) professionals in using methodologies and CASE tools.

The Problems of Organization of Thoughts

The process of cognizing is one of the complex phenomenon whose theoretical explanations about its nature are still welcome. The questions are: What exactly happens during this process? What controls it? Such questions are pertinent to this work where we examine the element of cognitive restrictions on contemporary and exiting structured software engineering methodologies.

Mental and Conceptual Models

Alistair Sutcliffe (1988) discusses the concept of the schemata in describing working of the human brain. He defines it in terms of the memory processing model. In this model the human memory comes in two varieties. Short-term working memory and long-term permanent storage. The information-processing model is used to place memory in the perspective of perception and cognition.

According to the model, each perceptual sense has a processor and associated short-term memory. The memory form the input and output buffers of the human system, storing abstract short-term visual and audio images and other captured information. Each memory is also associated with a sensory processor. The sensory processors analyse the contents of their memories and pass the resulting information to the cognitive processor for identification of the sensory input. Meaning from an input is generated when information in the input short-term memories is passed on to the central cognitive short-term memory for interpretation. The cognitive processor is thought to be responsible for object identification and manipulation. This is effected by matching the incoming information with past experience and then attaching semantic meaning to the image or sound.

This description of the human thinking process does not provide us with a reasonable argument without considering another view of what happens inside the brain: mental and conceptual models. Fischer (1991) summaries mental models thus:

> The user's model of a complex system is a cognitive construct that describes a user's understanding of a particular content domain in

> the world. These modelsare formed by experience, self-exploration, training, instruction, observation, and accidental encounters.

The mental models may not be accurate, that is to say it may not be a good representation (copy, abstraction, subset, or summary) of the conceptual model used to design the system or the system itself. Norman (1983) describes the difference between the conceptual and mental models:

> Conceptual models are devised as tools for the understanding or teaching of physical systems. Mental models are what people really have in their heads and what guides their use of things. Ideally, there ought to be a direct and simple relationship between the conceptual and mental model. All too often, however, this is not the case.

The mechanics of the human brain as presented by Sutcliffe (1988) and Norman (1983) above can now be used to examine why software engineers are not happy with structured methods.

Software engineers interpret and predict the results of using a methodology by using their mental model of it. The content and format of a mental model can not be discovered by simply asking software engineers how they view or understand a methodology to work. The verbal

Both of the above models (Sutcliffe's (1988) and Norman's (1983) models of the human brain operation) indicate that human information processing involves matching of the input data or information with what is in the long term memory. This is done in the cognitive processor.

The cognitive processor retrieves data and patterns of data from the long term memory. During this process of matching, also known as thinking or reasoning, humans employ heuristics in getting a solution. This means that human reasoning is not strictly logical, rather it is a process of making comparisons against a series of propositions which make up a mental model.

Heuristic reasoning requires considerable efforts. Comparison of facts gathered from the environment also requires freedom in choosing pairs of object images for comparison. Here we can identify two factors that can cause stress in software engineering using structured methods:

1. The process of comparing against a series of propositions making up the mental model may become long. This happens if the model does not easily match with the way the methodology requires the problem to be solved. This prolonged reasoning is a source of stress.(Sutcliffe 1988)

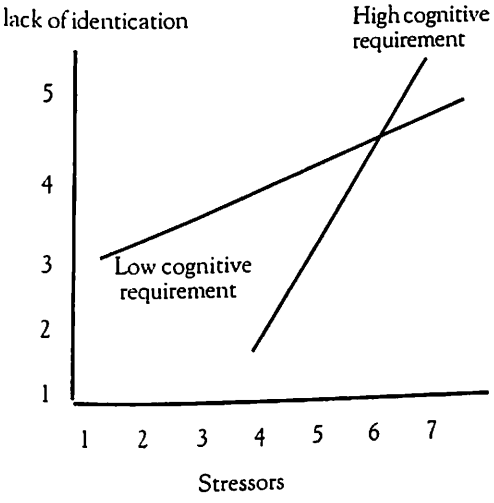
Table1: *Professionals' attitudes regarding tools*

	Agree (%)			Disagree (%)			
Questions regarding tools	Strongly	Moderately	Slightly	Slightly	Moderately	Strongly	Average Score
Traditonal 3.88 design methodologies fail to provide a systematic approach to the design process	9.7	17.1	11.1	15.2	31.3	15.7	
Computer aids including tools improve design quality	4	4.7	4.3	25.1	46.0	17.5	4.60
Structure chart is an effective programme representation	1	2.4	12.4	23.9	40.2	20.1	4.60

Average scores commputed with indicated weights, strongly agree =1; moderately agree =2; slightly agree = 3 slightly disagree = 4: moderately disagree = 5 strongly disagree =6. (Adapted from D.B. Tesch *et al* 1995)

2. Structured methods limits the use of Heuristics by software engineers. This contradicts with something which is a natural way of reasoning. This limitation makes problem solving difficult uninteresting. Sutcliffe (1988)points that 'Mundane, non-stimulating tasks are likely to cause user fatigue precisely because they do not stimulate interest and hence do not hold attention' The result of this is fatigue. This is why software engineering has been identified to cause stress and hence burnout among developers.

Figure1: *Moderator effect of cognitive requirements on the relationship between stressors and lack of Identication*



(Adapted from Brodbeck, Heinbokel and Stolte, 1994)

Stressor scores of software engineers have also been reported by Rubin and Hernendes (1989) as higher than those of other professionals. They also reported that there is high intrinsic work motivation in software engineering professionals, which make them prone to development of stress if they work in stressful situations (Pines and Arson, 1989).

Brodbeck, (1994) indicates that cognitive requirements are among the elements of control at work. These cognitive controls were assessed by asking the subjects to give percentages of time spent in thinking (in contrast to performing routine work) The findings of Bordbeck's study (figure 1) are adapted here in the following discussion.

Following structured methodologies means subjecting the engineer to certain cognitive corridors prescribed by the methodology or technique itself. This is a very high degree of control because it does not just direct thinking but it goes further to define the pattern of 'though ' during a 'thinking session'. Yourdon (1993) underscores the semantic effect of graphical tools;

> graphical tools used in essential modeling are mainly of semantic nature, is butle, yet important The communication grammar, in other words, the graphical conventions, predetermines the way of thinking about the problem (Yourdon, 1993).

For example in using Entity Relationship Diagrams (ERDs) to model requirements the analyst is forced to think in terms of ERDs. This has two implications; It can produce stress provoking environments in an already concluded stressful profession. Second, it can demotivate the analyst by denying her of challenging assignments (because work is made more clerical).

While high cognitive requirements in working environments are expected to motivate engineers, Brodbeck (1994) concluded that software professionals with high cognitive requirements revealed higher burnout scores if the level of stressors was high, and low scores if the level of stressors was lower than the case for those with lower cognitive requirements.

However there was an unexplained variance when Brodbeck analyzed his data of burnout against the various elements that caused it. This unexplained portion is associated/hypothesized here as being attributable to 'cognitive freedom'

User dissatisfaction causes burnout

On user satisfaction we define the professional satisfaction in using a tool as resulting from the following:

- Perceived success of implementation results
- Actual incorporation of design principles in the course of development; and
- The perceived "fit" between professionals' natural problem solving process and the one imposed by a methodology.

The third element which addresses the issue of fit, seem to conform with our earlier discussion on the human information processing. This lack of fit that might exist, seem to be the difference between a software engineer's mental model of the a methodology and the conceptual model of the methodology. This drift can increase with

the degree of control that is imposed by a methodology. Control in a methodology is manifested by rigidity steps or stages to be followed, tools that can be used in modelling, shapes, format and styles of notations.

However, because structured methods have contributed very significantly in software development, it is not proper to argue that these methods are useless. So, software engineers may be more confortable with structured methods at a certain level of structuredness. On the basis of this we develop the following hypotheses:

1. Because control (Rules and Procedures) at work, elements of structuredness, have positive correlation to burnout (Brodbeck and Hainboke, 1994) it can hypothesized the lack of cognitive freedom results in burnout.
2. Software engineering is a field that is (or to be fair, expected to be) challenging and requiring high cognitive inputs. It can be plausibly assumed that various platforms and workbenches (Methods and CASE tools) reduce personal accomplishment (Lack of identification) see figure1). This is because they reduce this highly respected profession to a clerical, routine work. Hence we hypothesize that, there is an optimum degree that a methodology can be structured. As a methodology becomes highly structured;
   - It becomes less appealing to software engineers
   - It becomes less effective.

CONCLUSION

This discussion has led to the development of hypotheses that need to be pursued by a more experimental research. The ideal experiment may utilize talk-aloud techniques to explore the answers to the question. It is also the feeling of the author that there has not been sufficient amount of behavioral studies on IS professionals especially in as far as use of technologies is concerned.

On the basis of the suggestion of Tesch 1995, that a consideration of issues of satisfaction may lead to the development of more effect system development tools, the author strongly encourages more research towards this direction.

REFERENCES

Brodbeck, F.C; Hainbokel, T. and Sttolte, W. (1994) "Stressor burnout relationship in software development teams." *Journal of Occupational and Organizational Psychology*, Vol.76, iss. # 6, p. 327-341

Tesch, D.B., Klein G.Sobol, M.G.1(1995) "Information system professional attitudes to development tools." *Journal of Systems and Software*, Vol.28 issue # 1, p. 1343-50

Maslach, (1982) **"Understanding burnout; Definitional issues in analyzing a complex phenomenon."** in W.S.Paine (ed). *Job Stress and Burnout: Research, Theory and Intervention Perspectives* p. 29-40 Beverly Hills, CA:Sage

Stacy, W. and MacMillian J, (1995) "Cognitive bias in software engineering." *Communications of the ACM*, Vol.38. iss 6. p. 34-38

Phol, K (1994) "Three dimensions of requirements." *Information Systems*, Vol 19, iss. 6 p. 56

Leonard -Barton, D. and Deschamps, I. (1988) "Managerial influence in the implementation of new technology." *Management Science*, Vol.34, iss. # 4, p. 1252-1265

Yourdon, E. (1993) *Yourdon Systems Method - a Model Driven Approach*. Prentince Hall International, New York

Rubin, H.I. and Henarndes, E.F. (1988) "Motivation and behaviours of software professionals." in E. Award (ed.), *Proceeding of the 1988 ACM SIGCPR Conference*. p. 62-71. New York:

Pines, A.and Arson, E.(1988) *Career Burnout: Causes and Cures*. New Your: Frees Press

Davis, G.B. (1982) "Strategies for information requirements determination." *IBM Systems Journal*, Vol 21, Iss. #2, p. 4-30

Land, F. F and Somogyi, E. (1986) "Software engineering: the relationship between a formal system and its environment." *Journal of Information Technology*, Vol 1 Iss. # 1, p. 246 - 48

Markus, L.M. (1984) *Information Systems in Organizations; Bugs and Features*. Pitman Marshfield.

Burns, R.N. and Denis, A.R. (1985) "Selecting the appropriate application development methodology." *Data Base*, Vol 17, iss. # 1, p. 1924 -29

Capper, L. (1985) "The use of analysis and design methodologies in the working environment: an experimental approach." in Bemelmans (ed) .....?, p. 37-48

Fischer, G. (1991) "The importance of models in making complex systems comprehensible." in M. J. Tauber and D. Ackermann (eds), *Mental models and Human-Computer Interaction* 2. Amsterdam: Elsevier North Holland

Sutcliffe A. (1988)
Human Computer Interface Design, MacMillan Education LTD, London.
Norman, D. A. (1983)
"Some observations on mental models." in D. Gentner and A. L. Stevens (eds), *Mental Models*, Hillsdale, Nj: Lawrance Erlbaum