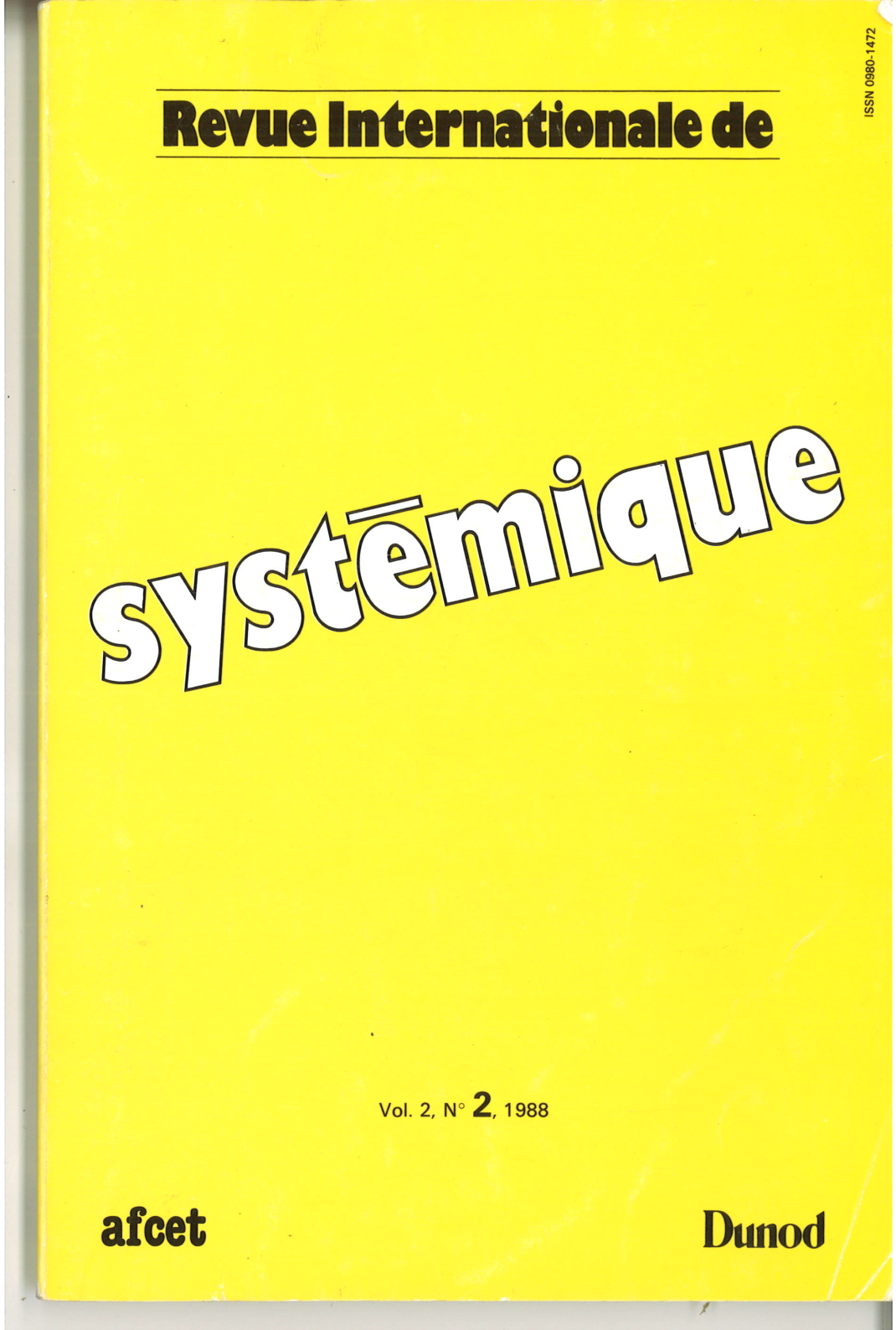

AFSCET

Revue Internationale de systēmique

Revue
Internationale
de Sytémique

volume 02, numéro 2, pages 195 - 214, 1988

Cognitive Models and User Interfaces for an Advanced Software and System Engineering Development Environment

James D. Palmer, Andrew P. Sage

Numérisation Afscet, janvier 2016.

REVUE INTERNATIONALE DE SYSTEMIQUE
Vol. 2, N° 2, pp. 195 à 214

COGNITIVE MODELS AND USER INTERFACES for an ADVANCED SOFTWARE and SYSTEMS ENGINEERING DEVELOPMENT ENVIRONMENT

James D. PALMER, Andrew P. SAGE

Georges Mason University [1]

Abstract

This paper is concerned with issues associated with enhancements to software productivity. An Advanced Software and Systems Engineering Development Environment (ASSEDE) for software productivity is suggested as a vehicle for enhancement of software productivity. One of the major ingredients in supporting software productivity is the development of a Model Base Management System (MBMS) that comprises a number of modules that support efficiency and effectiveness throughout the system development life cycle. Prominent among these are software prototyping and software reusability approaches for what we call macro-enhancement of software productivity. Associated with this MBMS and ASSEDE design effort are other issues concerning knowledge representation and dialog management. An overarching question concerns cognitive issues associated with the ability of humans to cope with the requirements potentially imposed through introduction of information and knowledge support technologies for software productivity improvements. We suggest allowing those using ASSEDE to structure software specification tasks in an initially unconstrained way such as to allow determination of useful descriptions of individual differences in information structuring and use as a function of experien-

1. School of Information Technology and Engineering, Fairfax, VA 22030, Etats-Unis.

Revue internationale de systémique. 0980-1472 Vol. 2/88/02/195/20/$ 4.00/

tial task familiarity. One of the hoped for objectives in so doing is a general theory of software systems life cycle engineering that incorporates viable cognitive information processing realities. Another important objective is the development of tailorable user MBMS that are best matched to user needs.

Résumé

Cet article s'intéresse aux problèmes liés à l'augmentation de la productivité de logiciels. Un Environnement Evolué pour le Développement de Logiciels et d'Organisation de Systèmes (en bref ASSEDE), est suggéré comme un moyen d'améliorer la productivité de logiciels. L'un des principaux ingrédients de l'aide à leur productivité est le développement d'un Système de Gestion d'une Base de Modèles (en bref MBMS) qui comprend un certain nombre de modules pour contribuer au rendement et à l'efficacité du système tout au long de son développement. Parmi les plus importants de ceux-ci se trouvent des approches du prototypage de logiciels ainsi que de leur réemploi, que nous désignons comme la macro-augmentation de la productivité de logiciels. En liaison avec cet effort pour concevoir le MBMS et l'ASSEDE, d'autres questions concernent la représentation des connaissances et la gestion du dialogue. Un point clef est celui des éléments cognitifs liés à la capacité des humains de se débrouiller avec les exigences potentiellement imposées, par l'introduction d'information et de technologies du support de la connaissance, en vue de l'amélioration de la productivité des logiciels. Nous suggérons de donner la possibilité à ceux utilisant ASSEDE de structurer les tâches de spécification de logiciels de façon initialement libre, par exemple, grâce à la possibilité de déterminer des descriptions utiles de différences individuelles dans la structuration de l'information et en utilisant la familiarité expérimentale avec la tâche. L'un des objectifs espérés d'une telle pratique est l'élaboration d'une théorie générale du traitement des cycles de vie des logiciels, incorporant les réalités de la gestion de l'information cognitive. Un autre objectif important est le développement d'un MBMS spécialement adapté aux besoins des utilisateurs.

1. Introduction

An ideal software and systems engineering development environment is one in which the enf-user is able to identify system requirements [Davis, 1982 ; Sage, Galing and Lagomasino, 1983] through an interactive human-machine interface that enables visualization of the

impact of alternative requirements specifications. Through this, the user should be able to refine these requirements interatively and throughout the process, so as to produce the software system closest to that required to fufill identified needs. The ultimate operational goal of a software development environment is to enable automated production of requirements specifications and to be able to ultimately enable these to be translated into operational code that meets all of the operational functionality requirements of reliability, maintainability, availability, portability interoperability, verifiability, validability, and trustability [Beam *et* al, 1987 ; Frankel, 1985 ; Harrison, 1958]. This process implie the need to look beyond the conventional «waterfall life cycle» [Boehm, 1981] and other production life cycle models to take advantage of the opportunities afforded by emerging information technology advances : expert systems, artifically intelligent interfaces, open systems architectures, distributed data bases, graphic interfaces, and cognitive interfaces.

Progress is being made in the incremental development of such a software design and management environment, as indicated by the following advances :

1. The Common Ada Missle Packages (CAMP) project has developed an environment taht demonstrates the ability to develop and use Ada parts which are reusable, tailorable, efficient, and portected [Czarnik, 1987]. This demonstrates the ability to provide automated assistance to the part user ; and applys DOD-STD 2167 [US DoD 1985] to Ada^{tm} and the parts selected.

2. Object Oriented Language applications to identify requirements specifications have been implemented [Nordby, 1986] and produce full specifications according to de Marco's [1982] rules including data flows, hierarchical structure, and date definition.

3. Fourth generation languages (4GL) and automatic program generators have been applied to the production of code [GSA, 1986]. These have proved to be able to produce code faster and with significanty fewer errors than coding in a high level language. Problems with 4GL and automatic program generators genarally are that the code so produced often requires longer run-time, and has more lines of code, although it does otherwise improve productivity [Balzer, 1986 ; Green, 1985].

There are other signifiant environment developments that involve the translation of user needs to detailed requirements specifications [Wasserman, 1986]. These latter advances are clearly precursors for the

future total software systems engineering design and management environment, and are essential to the work necessary to produce this environment.

Important work continues in the explication of a coherent software development environment, especially through the development of reusability via libraries of parts, the development of intelligent interfaces for database management systems [Kerschberg, 1986], direct translation to code of detailed specifications automatically developed, implementation of validation and verification procedures early in the development process, and others. These research and development programs will have significant impact on the utility of the software development environment of the future.

Absent from these research and development programs is the development of the interactive cognitive interface between the user and the system analyst that will enable the translation of the system requirements in an effective and efficient manner. The notion of the ideal software development environment would provide for a *MODEL OF THE USER* at the interface so that the system would be tailorable to meet the needs of the user. This would enable the transfer of information between the user and the system development processes in a continuous way, without serious penalty to the entire process, such as to facilitate system evolution and modification as user perception of these needs develop.

In ordre to model the user at the system interface and, as a result, to tailor the interface to the capability of the user ; it is necessary to know and understand better those cognitive issues that affect the interactive process of information elicitation and transfer. Several such issues have been identified. We will discuss several of these here.

2. Cognitive issues affecting software productivity

Decisions may be described as structured or unstructured depending upon whether or not the decision making process can be explicity described prior to the time when it is ncessary to make a decision. This taxonomy would seem to lead directly to one in which expert skills [wholistic reasoning], rules [heuristics], or formal reasoning [holistic evaluation] are descriptively, or should be normatively, used for judgment [Sage, 1981]. Generally, operational performance decisions are more likely than strategic planning decisions to be pre-structured. It is important to note that expertise is a relative term which

depends upon familiarity with task and the operational environment into which it is imbedded. Since decision environments do change, and since novices become experts through learning and feedback, it is clear that there should exist many areas in which the proper form of knowledge base support is a hybrid of an «expert system» and a «decision support system». This suggests that there will be a variety of decision-making processes in practice and that an effective support system should support multiple decision processes. In a similar way, information requirements for decision-making can be expected to be highly varied, and an effective support system should support a variety of date (and knowledge) base management needs.

There are a number of cognitive abilities that a knowledge support system for software productivity enhancement should embrace. It should support the software systems engineer in the formulation or framing of the decision situation in the sense of recognizing needs, identifying appropriate objectives by which to measure successful resolution of an issue, and through aiding in the generation of alternative courses of action that will resolve the needs and satisfy objectives. It should also provide support in enhancing the abilities of the system user to identify possible impacts of alternative courses of action. This analysis capability must be associated with capability to enhance the ability of the support system user to provide an interpretation of these impacts on needs and objectives. This interpretation capability will lead to evaluation of the alternatives and selection of a preferred alternative option.

Real life software systems engineering problems and associated design and management issues are so complex, unstructured, and poorly understood initially, that an automated process for knowledge support must allow and encourage learning so that system users gradually obtain increased understanding of the decision situation, as well as the personal value perspectives which led to the decision, such as to be able then to make a morre informed and hence typically better decision. Thus, support to formal based reasoning results in the learning of rules and ultimately the skills that augment, and perhaps even replace, formal reasoning in later performance of these same tasks. This appears to firmly introduce skill and rule based knowledge into the judgment situation and to make it very desirable that processes for model based management and knowledge support incorporate these forms of knowledge.

It is the desire to provide evaluation and recommendation capabi-

lity in a knowledge support system that leads to the need for a model base management systems [MBMS], especially and specifically one intended to enhance software productivity [Dolk, 1984]. It is through the use of model base management systems that we are able to provide for sophisticated analysis and interpretation capability in a knowledge support system. The single most important characteristic of a model base management system is that it should enable the decisionmaker to explore the design situation throught use of the data base by a model base of algorithmic procedures. This can occur through the use of modeling statements, in some procedural or nonprocedural language. It can occur through use of model subroutines, such as mathematical programming packages, that are called by a management function It can also occur thourgh the use of data abstraction models. This latter approach is close to the expert system approach in that there will exist element, equation, and solution procedures that will together comprise an inference engine. Advantages to this approach include case of updating and use of the model for explanatory and explication purposes. This is especially valuable when a multi-level perspective is appr-priateto decentralized and distributed decisionmaking.

3. Knowledge support requirements

There are a number of behavioral implications to knowledge support system introduction into a decision situation, such as software systems engineering, that are very important. User involvement in the design process, management support for the design effort, and the availability of user training activities are but a few of the many requisites for success [Sage and Rouse, 1986 ; Sage, 1987a]. It is especially important that potential system users not regard a support system as too difficult to learn to use, too hard or too time consuming to actually use, or as producing inaccurate, incomplete, or dated recommendations. Perhaps the most damning charges of all that affect potential user willingness to use a knowledge support system are feelings that the support system significantly interferes with the «normal» way of thinking, can not adapt to changes in problem specifications, does not produce intermediate results of value, or does not really address the actual problems that exist. Appropriate system designs avoid these impediments. To determine the extent to which the results of a software systems design effort yields satisfactory support systems, operational evaluation of prototype support systems generally necessary.

There are various kinds of knowledge that comprise expertise. Among these are : facts about the specific domain of the problem solvers that are to be aided by an expert or knowledge support system ; fixed rules and procedures that will always be valid ; problem situations in which there is imprecision and uncertainty and in which heuristic rules need to be appled with caution ; skill based perceptions based on experiential familiarity with the task at hand ; and formal theories of the domain that need be used when experiential famililiarity with the task at hand and the environment into which it is imbedded is insufficient to enable wise use of skill and rule based knowledge. Because of this great variety of knowledge, it is most likely that experts will be *unable* to present a complete and consistent set of fact files and knowledge relationships that will enable construction of a «complete» knowledge base sybsystem. Thus the process of knowledge acquisition must allow for iterative and incremental identification of knowledge, and for the potential use of formal reasoning based approaches. It must allow for expansion, contraction, replacement, and residual shifts of existing knowledge in a knowledge base as *new* knowledge is acquired.

For this reason, the process of knowledge acquisition cannot be totally separated form taht of knowledge representation or that of knowledge utilization. In order to determine consistency and completeness of the knowledge base, it will be necessary for acquired knowledge to be subjected to appropriate tests. These will ncessitate representation and use of the acquired knowledge, as it is on the basis of representation and use that such measures as consistency and completeness are determined. This requirement for iterative and modular knowledge acquisition will require that a software systems engineering support facility allow for the use of rapid prototyping and reusability constructs as needed tools throughout the system design life cycle.

The process of knowledge acquiisition is driven by the identification of existing deficiencies in a knowledge base. Without the use of meta-knowledge to drive knowledge acquisition, the acquisition process is undirected and there is little that can be done to insure effectiveness fo the knowledge base, or efficiency in its acquisition. It is this preliminary interactive use of the knowledge base that enables an expert to detect errors in the performance of the expert system. These occur in the form of reasoning or responses that the expert finds inappropriate. There must be some capability to work backwards throught the system in order to detect the location of flaws and allow modification or augmentation of knowledge at that point.

Knowledge may be acquired from experts in the form of fact files, action and event knowledge, performance knowledge (knowledge about «how to do» things), and meta-knowledge about what is known and what is not known. Meta level knowledge can be used to prioritize the relevancy of rules in the data base such that those rules most likely to be useful, given a current knowledge on the part of the expert system, are used first. Through the use of strategies such as this, acquisition of knowledge can be made effective and efficient. Our efforts concerning the use of meta-level knowledge to direct the knowledge acquisition process will include detection of knowledge base representational deficiencies, and diagnosis of sources of deficiencies, and correction of the knowledge base by interactive and iterative cycling through the acquisition, representation, and utilization portions of knowledge base subsystem design.

4. Design requirements for ASSEDE

The ultimate goal of systems design is to provide a certain function, product, or service within reasonable cost and other constraint conditions such as to enable the fulfillment of some desired performance goals [Freeman, 1983]. These overall performance goals or objectives are typically translated into a set of expected values of performance, reliability, and safety [Brooks, 1987 ; Parnas, 1983 ; Pettijohn, 1986 ; Hecht, 1986]. On the basis of this, it is the task of the system planner attempting conceptual specification of system architecture, the system designer attempting concept realization in the form of operational system specifications, as well as that of a man-machine intelligent system undertaking operational control, to examine future issues in such a way as to be able to [Sage, 1977, 1982]. :

1. identify task requirements, such as to be able to determine sub-issues to be examined further and those not to be considered further,

2. identify a set of hypotheses or alternative courses of action which may resolve the identified issues that are to be resolved,

3. identify the impacts of the alternative courses of action,

4. interpret the impacts in terms of the objectives for the task at hand,

5. select an alternative for implenmentation and implement the resulting control or policy, and

6. monitor performance such as to enable determination of how well the integrated system is functioning.

There is a major focus and emphasis on human-system, or human-machine, interaction concerns in the foregoing. Information, and the appropriate use of information to aid and enhance system planning, design, and operation is also stressed. There are many questions that relate to a successful interaction among humans, the integrated tasks in which they are engaged, and the software system that is to be designed [Sage, 1987b]. These questions relate to the control of technological systems. They concern the degree fo automation possible from an intelligent machine, especially in flexible task allocaion situations. They concern the design of computer generated displays that aid the human supervisory controller. They relate to a number of management tasks at a variety of organizational levels : strategic, tactical, and operational. System information requirements exist to varying degrees in each of these activities. All of them concern, in a very fundamental way, the effective and efficient use of information. These concerns appear of particular importance relative to the improvement of software productivity.

5. Imperfect information concerns in ASSEDE

Our research is concerned with the appropriate use of information for software systems engineering support in planning, problem solving and decision making activities. Of necessity, we must be especially concerned with enhancing the value of imperfect information towards these ends. By imperfect information, we mean information that is incomplete, imprecise, uncertain, inconsistent, unreliable, or some combination of these [Stephanou, 1987]. We must be especially concerned with the use of information to enhance requirements specifications identification for purposes of software prototype design.

Our assumptions of imperfect knowledge or information may refer to available information that may be imprecise, relative to the degree of refinement with which the asessment is made, or information which may be inconsistent in the sense of being in disagreement with presumed principles or laws of an assumed decision situation model, or information which may be incomplete in that needed elements are missing.

We need also be especially concerned with representation and automation of knowledge that arises from various reasoning perspecti-

ves including skill, rule, and formal reasoning based approaches [Rasmussen, 1983, 1985, 1986]. This will enable better representation and utilization of perceptual knowledge, as well as formal knowledge, in aids that enhance human performance [Sage and Lagomasino, 1987].

We will alos be concerned with interactive formulation of knowledge support processes as learning processes in which the software systems engineering team through interaction with the support system, is able to successively gain a better understanding about the operational situation in which the software should function, and to adapt value judgements accordingly. This is to be accomplished in such a way as to enable selective resolution of inconsistencies in the knowledge base during the search for an appropriate representation of knowledge, perhaps in the form of schema or a dominance structure among alternative courses of action, that is sufficient to enable judgement and choice. Inconsistencies are, for the most part the result of biases and inadequate heuristics used in the acquisition of knowledge. Existing methods will be extended, and new approaches to identify, avoid, and resolve, or minimize the effect of, inconsistencies due to the use of flawed heuristics and cognitive biases [Kahneman, Slovic and Tversky, 1982 ; Sage 1981] need to be explored. The role of evidential and epistemic information, as well as aleatory information must be considered.

We are discussing, in a very fundamental way, the used of very general «value of information» techniques to direct and control the acquisition, representation, and organization of information required for judgment and choice. In this, we are necessarily very concerned with developing inquiry processes and information presentation aids that allow for multiple perspectives concerning issues and problem situations [Sage and Lagomasino 1984]. The resulting knowledge support system is envisioned to have several major components.We will be concerned with approaches to data base management and model base management that allow for integrated use of these to aid in human information processing and, potentially, machine information processing as well. Development of appropriate theories that enable appropriate information processing and judgmental task sharing among human and machines is an important issue in software systems engineering. The generalized «value of information» concepts suggested here should provide useful guidelines towards this development.

In a design of this sort, one must be especially concerned with meta level model management of an inquiry system. This inquiry system will be in charge of directing and controlling the interactive

nature of a system that allows adaptation and learning. In effect, this will allow for higher level meta-knowledge, perhaps represented as formal reasoning based knowledge, to direct and control the use of skill and rule based knowledge which will typically be schema-like. Through this approach it should be possible to resolve problems that often inhibit understanding across several knowledge domains, perhaps confounded by differing perpsectives relative to the ultimate operational issues at hand. Of major importance is the accommodation of multiple knowledge perspectives and the integration of these into a «theory» of information presentation.

There are several additional research needs that are suggested by the observations made here. These can be stated in the form of hypotheses :

1. *The design of support systems that assist humans in cognitive tasks, such as software design for productivity, requires substantial comprehension of the human intellectual activities supporting judgement and choice*

This involves studies of skill based, rule based, and formal reasoning based judgement and how use of these types of knowledge depends upon the contingency task structure. It also involves a study of ways in which humans process information, especially in distributed environments that are subject to considerable message delays, node failutres, and associated uncertainties and imprecision in the resulting database.

2. *Identification of appropriate ways to represent knowledge in a multiple-agent support system is a critical issue*

This requires attention to studies that deal not only with the formal reasoning methodes common to decision analysis, but also to studies of how humans can be aided in reasoning wholistically, such as reasoning by analogy. Perhaps most importantly, it requires studies of how holistic and wholistic knowledge can each normatively support one another to aid decision-making processes.

3. *Information fusion studies and studies of the diverse interpretations that can result from the same knowledge presentation are important current issues*

There exists the need to blend descriptive and prescriptive approaches in doing this such that the resulting process is behaviorally acceptable.

4. *Acquisition and representation of knowledge from multiple perspectives are needed, as well as studies of how to accommodate this within the framework of specific model base and cognitive engine constructs that enable effective use of the support aids available for the software development environment*

This is needed in order to provide the input to the model management, cognitive engine and associated inference mechanisms that access the knowledge base of a software design and management support system. These must be designed in such a way as to consider the special needs associated with the query language structure that will result in the database, or knowledge base, and the physical locations for various portions of the data or knowledge base.

5. *There exist needs relative to integration of the various knowledge bases, model bases, cognitive engines, and the users of the system, for appropriate communication within the distributed knowledge environment*

In the situation associated with most large and complex application software design issues and environments, it will typically not be possible, due to time constraints and other complexities, to evolve a «complete» set of potential alternatives nor even a complete software requirements specifications so needed for ultimate success in design and development. For this reason, and others, the ability to deal with imperfect and ambiguous information is very necessary.

6. Some hypotheses concerning the ASSEDE facility

The function and purpose of this design support facility is such that it will enable rapid and interactive cycling through the steps and life cycle phases of a software design and management activity. The

facility will enable and enhance application of the macro-enhancement approaches discussed in Beam, Palmer, and Sage [1987]. It will enable these approaches to be imbedded into a methodology for systems design and systems management of trustworthy software.

Contemporary needs associated with the systems engineering design and management of software for enhanced productivity suggest that much benefit can be achieved through integration of systems engineering, cognitive science, and computer science approaches now made possible by developments in information technology. A principal goal fo software systems engineering is the organization of knowledge for the realization of trustworthy software [Fairley, 1985 ; Somerville, 1985]. Human initiative and creativity are best enhanced when the user of a knowledge support system is able to self direct the system towards skill based, rule based, or formal reasoning based assistance rather than having to respond to the dictates of a behaviorally insensitive and inflexible paradigm that is often used as the basis for support system development.

We have presented a overview of our conceptual approach to design of a knowledge support system for software productivity enhancement. This, as we have discussed, is basically a software repository consisting of reusable software modules and non procedural approaches that enable rapid system specification and evaluation in a prototype like fashion [Palmer and Nguyen, 1986]. A critical needed task to obtain a successful system design product is the integration of these findings into a cohesive whole that will enable the sought for system design principles to be evolved into the design of an operational knowledge support system. One of the major tasks associated with the design of a software systems engineering development environment is the determination of a set of information presentation principles and a resulting strategy for information presentation that enables judgment and choice in software productivity areas.

The hypotheses that follow relate to the cognitive aspects of group and individual interactive processes involving human-computer interfaces. One of the major goals of the interactive processess at the human-machine interface is the ability to elicit and transfer information from the user for implementation in the development of the software. One of the hoped for achievements in so doing is a better understanding of the important factors governing the effective, timely, and quality information transfer characteristics that influence the utilization of the software systems engineering development environment and the related tools and activities.

There are many issues that must be resolved during the course of design of the facility suggested here. These include : the formulation of the methods and processes for information transfer, model definition within the model base management system, reusability trade-offs, tailoring of the processes, security and control access, interfaces, expert systems and the application of knowledge based systems and controllers, interactive languages, comfiguration management, distributed systems architectures, automation of processes and all management control areas. The evaluation methodology must address the goal formulation statements, estimation of appropriate metric,s planning, designing and performing tests and analyzing the outcomes. The procedural questions that arise relate to the quality of the use and the ability to characterize the quality in definitive terms ; the nature and extent of the domain ; the cost of the effort compared to conventional approaches, and the effectiveness of the operational aspects under actual usage.

The basic scenario will be to provide the development environment, establish user goals in the formal sense, and supply data for the implmentation process. The user will then tailor the interactive interface to the comfort level of the individual and utilize the generated internal model of the user for all interactive sessions. The process will be closely monitored to ascertain the selection of tools and methodologies with a trace provided through the expert system at the interface. Concurrently the expert system will trace the preferred tool and method selection as determined from the internal inference engine and knowledge base. These will be compared and the outcomes will be used to drive improvements on the continuing development of the interactive interface. The major goals will be to determine an assessment of the situation regarding the ease of interaction and the efficiency and effectiveness of the tool and methods selection process. Strategic and tactical assessment of the situation will be generated for evaluation purposes.

The Hypotheses that support design and use of a facility such as thire are :

H1. The quality of the requirements phase of sotfware development will improve markedly if the information presentation aids and other parts of the human-machine interface (HMI) affect a symbiosis between the user and the interactive interface.

H2. Productivity and quality will improve if the HMI models the user and tailors the interface to the user needs both intellectually and operationally.

H3. Small groups develop in a known (based on observations from small group dynamics studies) and prescriptive way to maximize individual contributions in a group situation if tailored HMI's are utilized.

H4. Use of an Advances Software Systems Engineering Development Environment (ASSEDE) will increase the information available and enhance the effectiveness and efficiency of the software design and management process.

H5. Use of an ASSEDE will increase the span of user control throughout the software life cycle.

H6. Use of an ASSEDE will decrease the number of hierarchical levels needed in the approval process for identification of requirements specifications and system level architectures.

H7. Use of an ASSEDE will increase the range of software and system decomposition levels (phases in the software life cycle) at which design decisions can be made without increasing the costs that are caused by imperfect information about the problem situation.

H8. The use of an ASSEDE will enhance the efficiency and effectiveness of the information systems integration engineering efforts.

6. Summary

In this paper, we have discussed a number of conceptual issues associated with the integration of systems engineering and information systems concepts for the design and management of productive and trustworthy software. An *Advanced Software Systems Engineering Development Environment* facility was described that embodies the principles discussed.

References

ARKES, H.R., and HAMMOND, K.R. (Eds.) *Judgment and Decision Making,* Cambridge University Press, 1986.

BALZER, R., «A 15 Year Perspective on Automatic Programming», *IEEE Transactions on Software Engineering,* Vol. 11, No 11, pp. 1257-1268, Nov. 1985.

BARNARD, H.J., METZ, R.F., and PRICE, A.L., «A Recommended Practice for Describing Software Designs : IEEE Standards Project 1016» *IEEE Transactions on Software Engineering,* Vol. 12, No 2, pp. 258-263, 1986.

BEAM, W.R., PALMER, J.D., and SAGE , A.P., «Systems Engineering for Software Productivity, *IEEE Transactions on Systems, Man and Cybernetics,* Vol. 17, No. 2, March, 1987.

BEREGI, W.E., «Architecture Prototyping in the Software Engineering, Environment», *IBM Systems Journal*, Vol. 23, No. 1, PP' 4-18, Jan. 1984.

BERGLAND, G.D., «A Guide Tour of Program Design Methodologies», *Computer*, Vol. 14, No. 10, pp. 13-37, October 1981.

BOAR, B.H., *Applications Prototyping : A Requirements Definition Strategy for the 80s,* Wiley, 1983.

BOEHM, B.W., *Software Engineering Economics,* Prentice Hall, 1981.

BOEHM, B.W., «Verifying and Validating Software Requirements and Design Specifications», *IEEE Software,* Vol. 1, No. 1, pp. 75-88, January 1984.

BORGIDA, A., GREENSPAN, S., and MYLOPOULOS, J., «Knowledge Representation as the Basis for Requirements Specifications», *Computer,* Vol. 18, No. 4, pp. 82-104, April 1985.

BROOKS, F.P., Jr., *The Mythical Man Month,* Addison Wesley, 1975.

BROOKS, F.P., Jr., «No Silver Bullett : Essence and Accidents of Software Engineering», *Computer,* Vol. 20, No. 4, pp. 10-20, April 1987.

BUDDE, R., KUHLENKAMP, K., MAHTIASSEN, L., & ZULLIGHOVEN, H., *Approaches to Prototyping : Proceedings of a Working Conference on Prototyping,* Springer Verlag, 1984, [D, E. 1 !.

CHURCH, V.E., et. al., «An Approach for Assessing Software Prototypes», *ACM Software Engineering Notes,* Vol. 11, No. 3, pp. 65-76, July 1986.

CURRITT, P.A., DYER, M., and MILLS, H.D., «Certifiying the Reliability of Software«, *IEEE Transactions on Software Engineering,* Vol. SE-12, No. 1, pp. 3-11, January 1986.

CURTIS, B., «Software Metrics», *IEEE Transactions on Software Engineering,* Vol. SE-10, No. 11, pp. 637-638, Nov.1983 [F].

CURTIS, B., et. al., «Software Psychology : The Need for an Interdisciplinary Program», *Proceedings of the IEEE,* Vol. 74, No. 8, pp. 1092-1106, August 1986.

CZARNIK, M.R., «Software Engineering Challenges - CAMP», *Proceedings, 1st Software Productivity Workshop,* Herndon Virginia, April 1987.

DAVIS, G.B., «Strategies for Information Requirements Determination», *IBM Systems Journal,* Vol. 21, No.1, 1982, pp. 4-30.

DEMARCO, T., *Controlling Software Projects : Management, Measurement and Estimation,* Yourdon Press 1982.

DOLK, D.R., and KONYSNKI, B.R., «Knowledge Representation for Model Management Systems», *IEEE Transactions on Software Engineering*, Vol. 10, No. 6, November 1984, pp. 619-628.

DREYFUS, H.L., and DREYFUS, S.E., *Mind Over Machine : The Power of Human Intuition and Expertise in the Age of the Computer,* Free Press, 1986.

FAIRLEY, R.E., *Software Engineering Concepts,* McGraw Hill Book Co., 1985.

FREEMAN, P., and WASSERMAN, A.I. (EDS), *Software Design Techniques,* IEEE Computer Society Press, 1983.

FRENKEL, K.A., «Towards Automating the Software Development Cycle», *Communications of the ACM,* Vol. 28, No. 6, pp. 578-589, June 1985.

GSA Office of Software Development and Information Technology, *Fourth Generation Languages : Issues and Impacts,* Report OSDIT/FSMC-86/008, Falls Church VA, June 1986.

GREEN, J., «Productivity in the Fourth Generation : Six Case Studies» *Journal of Management Information Systems,* Vol. 1, No. 3, pp. 49-63, Winter 1985.

HARRISON, T.S., «Techniques and Issues in Rapide Prototyping», *Journal of Systems Management,* Vol. 36, No. 6, pp. 8-13, June 1985.

HECHT, H. and HECHT, M., «Software Reliability in the System Context», *IEEE Transactions on Software Engineering,* Vol. SE-12, No. 1, pp. 51-58, January 1986.

INGLIS, J., «Standard Software Quality Metrics», *AT&T Technical Journal,* Vol. 68, No. 2, pp. 113-118, March 1986.

JACKSON, M., *System Development,* Prentice Hall Book Co., 1983.

KAHNEMAN, D., SLOVIC, P., and TVERSKY, A., (Editors), *Judgments under Uncertainty : Heuristics and Biases,* Cambridge University Press, New York, 1982.

KERSCHBERG, L. (Ed), *Expert Database Systems : Proceedings from the First International Workshop,* Benjamin Cummings, Menlo Park, CA, 1986.

LAGOMASINO, A. and SAGE, A.P., «Imprecise Knowledge Representation in Inferential Activities», in M.M. Gupta (EDS) *Approximate Reasoning in Expert Systems,* North Holland Elsevier, 1986, pp. 473-498.

LANTZ, K.E., *The Prototyping Methodology,* Prentice Hall, 1986.

MILLS, H.D., et. al., «The Management of Software Engineering», *IBM Systems Journal,* Vol. 19, No. 4, pp. 414-477, April 1980.

NORBY, K., «The Design Generator», Computer Science Corporation Report to RADC, July 1986.

PALMER, J.D., and NGUYEN, T., «A Systems Approach to Reusable Software Products», *Proceedings IEEE Systems, Man and Cybernetics Conference*, pp. 1410-1419, October 1986.

PARNAS, D.L., and CLEMENTS, P.C., «A Rational Design Process : How and Why to Fake It», *IEEE Transactions on Software Engineering*, Vol. SE-12, No. 2, pp. 251-257, February 1986.

PERAL, J., *Heuristics : Partially Informed Strategies for Computer Problem Solving,* Addison-Wesley, Reading, Mass., 1983.

PETTIJOHN, C.L., «Achieving Quality in the Development Process», *AT&T Technical Journal*, Vol. 66, No. 2, pp. 85-93, March 1986.

RASMUSSEN, J., «Skills, Rules, Knowledge ; Signals, Signs, and Symbols ; and Other Distinctions in Human Performance Models», *IEEE Transactions on Systems Man and Cybernetics,* Vol. SMC 13, No. 3, May 1983.

RASMUSSEN, J., «The Role of Hierarchical Knowledge Represnetation in Decisionmaking and System Management», *IEEE Transactions on Systems, Man and Cybernetics,* Vol. SMC-15, No. 2, March/April 1985, pp. 234-43.

RASMUSSEN, J., *Information, Processing and Human-Machine Interaction : An Approach to Cognitive Engineering,* North Holland, 1986.

ROBERTSON, L.B., and SECOR, G.A., «Effective Management of Software Development», *AT&T Technical Journal,* Vol. 65, No. 2, pp. 94-101, March 1996.

SAGE, A.P., «Behavioral and Organizational Considerations in the Design of Information Systems and Processes for Planning and Decision Support», *IEEE Transaction on Systems, Man, and Cybernetics,* Vol. SMC-11, No. 9, September, 1981, pp. 640-678.

SAGE, A.P., and LAGOMASINO, A., «Knowledge Representation and Man Machine Dialogue», a Chapter in *Advances in Man-Machine Systems Research,* William B. Rouse, (ED), JAI Press, 1984, pp. 223-260.

SAGE, A.P., *Methodology for Large Scale Systems,* McGraw Hill Book Company, 1977.

SAGE, A.P., «Methodological Considerations in the Design of Large Scale Systems Engineering Processes», in Haimes, Y.Y. (ED), *Large Scale Systems*,North Holland , 1982, pp. 99-141.

SAGE, A.P., and LAGOMASINO, A., «Computer Based Intelligence Support : An Integrated Expert System and Decision Support Systems Approach», in *Expert Systems for Mangers,* B.G. Silverman (ED), TIMS series in Artificial Intelligence and Management Science, Addison Wesley, 1987, pp. 338-357.

SAGE, A.P., «Knowledge, Skills and Information Requirements for Systems Design», in Rouse W.B., and Boff, K.R., (Eds), *System Design , Behavioral Perspectives on Designers, Tools and Organizations,* Elsevier, North.

SAGE, A.P. (ED), *System Design for Human Interaction,* IEEE Press, 1987b.

SOMMERVILLE, I., *Software Engineeering,* Addison Wesley, 1985.

STEPHANOU, H., and SAGE, A.P., «Perspectives on Imperfect Information Processing», *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. SMC 17, No. 3, May 1987.

U.S. Department of Defense, DOD-STD-2167, Military Standards for Defense Systems Software Development, June, 1985.

U.S. Department of Defense, DOD-HDBK-287, Handbook of Defense System Software Development, 1985.

YAU, S.S., and TSAI, J.J.P., «A Survey of Software Design Techniques», *IEEE Transactions on Software Engineering,* Vol. 12, No. 6, oo. 713-721, June 1986.

YOURDON, E., and CONSTANTINE, L.L., *Structured Design,* Prentice Hall, 1979.

REVUE INTERNATIONALE DE SYSTEMIQUE
Vol. 2, N° 2, 1988, pp. 215 à 231

ARCHIVES

LA MECANIQUE COMPAREE [1]

(suite et fin)

Louis COUFFIGNAL

Nous publions ici la fin de l'article de Louis Gouffignal dont nous avons donné le début dans le précédent numéro de cette revue. Ce texte, emprunté au numéro de *Thalès* consacré à la cybernétique (tome 7, 1951, paru en 1953, Presses Universitaires de France), présente, en particulier, un intérêt historique. Il montre, entre autres, quel était le point de vue de l'auteur sur certains aspects de la théorie de l'information. Par ailleurs, il y est parlé d'une machine à calculer, de type arithmétique (donc non analogique), alors en cours de conception (1950), machine dont la construction fut, par la suite abandonnée.

L'ouvrage «Cybernétique», cité pour signaler un texte de Pierre Chavasse, intitulé plus précisément «la Cybernétique, théorie du signal et de l'information» (Editions de la Revue d'Optique, Théorique et Instrumentale, Paris, 1951) réunissait les communications présentées au cours du Cycle de conférences organisées, au printemps de 1950, sous la direction de Julien Lœb, dans le cadre du «Séminaire de Théories Physiques de l'Institut Henri Poincaré», sous la Présidence de Louis de Broglie (un cycle de ce genre, sous la même présidence, avait alors

1. Texte publié dans *Thalès*, tome 7, année 1951 (paru en 1953), pp. 24-36. Nous remercions les Presses Universitaires de France de nous avoir autorisés à reproduire ce document.