D1

SOME THEORETICAL PRESPECTIVES ON DEVELOPING A SOFTWARE LIFE CYCLE DATA BASE

CNTS

Dr. Bill Curtis
Software Management Research
Information Systems Programs
General Electric Company
Arlington, Virginia

In developing a software life cycle data base, we make the rather obvious assumption that research is not a scavenger hunt. Although software development efforts generate an enormous assortment of numbers, research is not an attempt "to salvage usable goods by rummaging through refuse or discards" (Webster's definition of "scavenge"). Rather, data collection for research purposes should be designed from a theoretical model of the phenomena to be studied. It is important to identify the data to be collected from the factors in the model, rather than contorting the model to fit the data that happen to be lying around. The quality of the resulting data base may also hinge on assigning an individual the responsibility of data collection and editing.

In identifying data relevant to each factor in a theoretical model, there are a number of important considerations. First, the data should be collected at the appropriate level of explanation. The following list represents three possible levels of explanation:

- Software development project
- Programming team
- Individual programmers

Data collected at the project level is not sufficient by itself to explain processes occurring at the level of the individual programmer. Thus, average lines of code per man-month at the project level is not an adequate criteria for investigating the productivity of individual programmers. Performance at the project level involves effort spent integrating the work of programmers and programming teams above and beyond the work initially expended by programmers in developing their code. In analyzing data across levels of explanation it is important to specify rules for aggregation which identify how the work of the parts is integrated into the whole.

Performance itself is an ambiguously used term. Rather than attempting to identify an ultimate criterion, the wise approach might be to identify multiple criteria at several different levels of explanation. Managers like to talk of meeting schedules within budgets, delivering high quality products. Jim McCall and Gene Walters of our Sunnyvale, CA office have identified myriad attributes constituting software quality such as reliability, maintainability, portability, efficiency, etc. Regarding criteria relevant to schedule and budget, one can collect machine records (runs, errors, changes, cpu time, etc.), personnel and payroll records (manpower loadings, labor costs, absenteeism, etc.), and managerial performance ratings. Identification of multiple criteria represents to software development managers the tradeoffs they must frequently make between schedule, budget, and quality.

In software life cycle research it is important to distinguish between objective data which are direct measurements of the phenomena under consideration and subjective data which are reports by project participants. While subjective data are important, they should be collected along with rather then instead of objective data. To evaluate the effect of a modern programming practice solely on the basis of managerial reports is to introduce into the analysis the perspectives and biases of the managers which flavor their conclusions about the technique. A warm feeling in the tummy should be backed by analyses of objective data.

There are two primary research strategies which can be employed in testing hypotheses from theorectical models depending on the type of controls which can be exercised over the variables affecting performance. In a laboratory situation, experimental controls can be exercised by manipulating the independent variable(s), holding all other situational factors constant, and minimizing the effect of individual differences among participants by randomly assigning them to different conditions of the experiment. The strongest causal statements can usually be made from rigorously controlled experiments. On the other hand, research in field settings must usually reply on statistical controls to study the effects of different variables. Through the use of multivariate correlational methods such as structural equation models or time series analysis, underlying relationships can be teased from the data and different causal models can be compared to determine which is most consistent with the data at hand. In using either experimental or statistical controls, it is always important to identify both the factors which may moderate the relationships observed and the populations to which the results can be generalized.

The Software Management Research Unit at General Electric is currently conducting research projects using each type of control discussed above. In subsequent articles, Sylvia Sheppart will report on our experimental work for the Office of Naval Research on human factors in software engineering and software complexity metrics. Phil Milliman will report on our work for Rome Air Development Center evaluating the effects of modern programming practices on software development projects.

In summary we propose the following guidelines for software life cycle research:

- Begin with a theoretical model
- Identify an appropriate research strategy
- Appoint someone responsible for data collection
- Collect data which is
  - appropriate to the level of explanation
  - objective
  - longitudinal

- Identify multiple criteria

- Hire a good statistician

ACKNOWLEDGEMENT

Work from which this paper was drawn was performed pursuant to contracts #N0014-77-C-0158 with Engineering Psychology Programs, Office of Naval Research and #F30603-77-C-0194 with Rome Air Development Center. The views expressed in this paper, however, are not necessarily those of either the United States Navy or Air Force or the Department of Defense.