Preface

Over the past several years, this journal has published a number of papers describing effective programming techniques, including the Chief Programmer Team, Structured Programming, HIPO, Top-Down Development, and Design and Code Inspections. Assessing the effectiveness of each of these techniques has been difficult for the following reasons.

- Effective programming techniques are frequently used in combination, so it is difficult to analyze the contribution made by each one.
- No standard of measurement has been so widely accepted as to serve as a yardstick for comparing projects from different locations.
- Few researchers have had the opportunity to measure enough projects, using standard techniques, to accumulate a substantial base of data for comparison.

In this issue, Walston and Felix describe preliminary research in methods of estimating programming project productivity. The authors analyze data collected during the past three years by the IBM Federal Systems Division. These data may be used in evaluating the productivity techniques described above, and may be applied in determining other factors seriously affecting productivity, such as programmer experience or program complexity. Measuring projects on a standard basis allows one to identify the major factors contributing to productivity, and may be used, in turn, to estimate the time, manpower, and expense required for future projects more accurately than has been possible in the past.

To a user who is inexpert in the use of computers, a system's data-base definition and manipulation capabilities, which he sees, are more important than its underlying techniques for organizing and maintaining the data base. The requirements of this user are addressed by the Data Base Retrieval System described by Alice Jones, which consists of a set of APL functions for data storage and retrieval designed for the user with little knowledge of APL.

Alfonseca, Tavera, and Casajuana describe the design and implementation of an APL system for the System/7 computer, placing emphasis on fitting a full time-sharing version of APL into a small computer. The authors hope that by extending APL to this event-driven system, they will be able to analyze the suitability of the language for process control and laboratory automation applications.

Another study of event-driven applications is that of Cole and Guido. A 5100 portable computer in conjunction with the experimental Research Device Coupler is used to monitor laboratory experiments. This configuration allows the researcher to use either of two high-level languages, APL or BASIC.

George McQuilken
Editor