Structured Techniques for Successful End User Spreadsheets

HARRY BENHAM
Montana State University

MICHAEL DELANEY
Saint Joseph’s University

ANDREW LUZI
California State University, Fullerton

This paper presents specific suggestions for managers who supervise end users who develop spreadsheets. It proposes the adoption of a specific, operational analysis and design tool for spreadsheets, a compatible spreadsheet structure, and concrete spreadsheet standards. End User computing literature identifies a number of risks associated with end user computing and it attributes many of those risks to end users’ lack of familiarity with good software development practices. Good software development practices begin with analysis and design. The proposed spreadsheet analysis and design tool fosters end user analysis and design activities. The proposed spreadsheet structure follows from that tool and reinforces good software development techniques. Finally, the standards proposed encourage good programming practices. Adoption of this tool, its related structure, and the standards would be an instance of providing the type of end user support and end user tools required for successful end user computing.

End user computing (EUC) has spread rapidly and relentlessly for at least the last decade. With a recent survey of Fortune 500 firms indicating that current investment in EUC is absorbing between 60 percent and 80 percent of information technology budgets (Amoroso & Cheney, 1991), it would appear that EUC will become even more important and pervasive.

Because of its importance, EUC has been studied extensively. Early EUC researchers were primarily concerned with documenting and extrapolating EUC’s rapid growth (Benjamin, 1982) or developing EUC classification schemes (Rockart & Flannery, 1983). As the number of end user developed applications increased, researchers began to focus on the risks associated with EUC. Alavi and Weiss (1986) linked end user development risks to stages of the traditional life cycle. Pyburn (1987) addressed the risks of end user developed personal, departmental, and corporate applications. Huff, Munro, and Martin (1988), Cheney, Mann, and Amoroso (1986), and Munro, Huff, and Moore (1988) were all concerned with controlling EUC risks while continuing to enjoy EUC benefits. These authors indicated a need to align EUC strategies with overall corporate strategies and advocated both end user training and the adoption of standards for data management, systems development and application testing. Recent EUC research has focused on a concern for effective managerial approaches to EUC. Magal (1991) reports on Informa-
tion Center success factors while Amoroso and Cheney (1991) look at a causal model of end user application success. These authors find information center services, organizational support for EUC, and the quality of EUC tools to be positively associated with success.

Although the EUC literature has been instrumental in developing a reasonable managerial view of EUC, it has offered relatively few concrete prescriptions for managers with direct EUC supervisory responsibilities. Which information center services or what EUC tools contribute to the success of EUC? In calling for end user training or the adoption and enforcement of standards, there is little guidance as to the type of training or standards to adopt. Cronan and Douglas (1990) have empirically verified productivity gains resulting from end user training. However, Nelson and Cheney (1987) were prompted to ask “What are the company’s training programs in LOTUS 1-2-3?” (pg. 557) when their study found that companies’ training tended to be technically oriented towards operating systems, data communications, and hardware.

Although EUC encompasses more than spreadsheet usage, spreadsheets have surely been fundamental to the growth of EUC. Ronen, Palley, and Lucas (1989) link the availability of spreadsheet software to the popularity of microcomputers. Between 50-70 percent of the near 1100 applications developed by respondents in the Amoroso and Cheney (1991) survey appear to have been spreadsheet applications. Additionally, studies such as DeThomas, Ray, and Rowe (1991) document the use of spreadsheets to perform decision support functions.

Drawing from the literature on EUC and software development, this paper generates specific, operational prescriptions for the management of end user developed spreadsheets. These concrete suggestions can be easily implemented by managers who directly supervise spreadsheet based EUC. The next section relates the software development literature to spreadsheet applications. Next, a simplified spreadsheet analysis and design tool is recommended. It is followed by a section setting forth and justifying a spreadsheet structure standard. Given a spreadsheet structure, standards for entries within that structure are presented and justified. Then the issue of end user compliance with spreadsheet standards is addressed. The final section provides a summary.

**Spreadsheets as Software Applications**

There is general agreement that software systems have a life-cycle: a sequence of phases through which each completed system has passed. The major phases of this life-cycle are:

- Conception
- Feasibility
- Analysis
- Design
- Implementation
- Testing
- Maintenance.

The explosive growth in EUC, much of it involving spreadsheets, demonstrates that end users have had little difficulty conceiving of potential spreadsheet applications. A spreadsheet’s ease of use and powerful programming environment have allowed end user developers to go from conception to implementation without serious consideration of the feasibility, analysis, and design phases. Inadequate attention to feasibility, analysis, and design can adversely affect spreadsheet quality.

There have been a few systematic studies of the quality of end user developed spreadsheets. Ronen, Palley, and Lucas (1989) cite frequent incidence of (1) mistakes in logic; (2) unreliable output; (3) unauditable applications; (4) inability to change or modify applications; and (5) an overall lack of comprehensibility. Roberts (1988) found 80 percent of the spreadsheets audited to contain one or more errors. And Alavi and Weiss (1986) found that end users often used inappropriate analytical techniques. Anecdotal evidence suggesting that end user developed spreadsheets can be characterized as unreliable, inflexible, unmaintainable, and unmanageable is abundant.

This characterization of end user developed spreadsheets is very similar to the characterization of early data processing systems. The techniques now known as structured analysis and design and structured programming were developed to address the shortcomings of those early data processing systems. Adoption of these techniques to spreadsheets would increase the emphasis on the feasibility, analysis, and design phases of the lifecycle. As was the case with earlier data processing systems, increased analysis and design effort should result in higher quality spreadsheet applications.

Software analysis is concerned with determining What the software system is supposed to do. Software design deals with determining How the system will do it. Structured analysis and design techniques are formal methods for specifying What and How. They seek to reduce complexity and increase comprehensibility by partitioning problems into smaller, less complex prob-