Preface

Systems Network Architecture (SNA) provides a framework for combining hardware and software components into an efficient, coherent teleprocessing system. To put this architecture into an historical perspective, the first paper in this issue, by McFadyen, is an overview in which the functional layers of SNA are defined as follows:

Transmission management controls the actual flow of traffic through the network; *functional management* controls the format of information sent to and from the application layers; *application* consists of user application programs in the host computer and equivalent functions in other nodes.

These functional layers are discussed in more detail in the three following papers. Cullum describes the transmission subsystem and its organization, its logical and physical aspects, and its components. Hobgood elaborates on part of the functional management layer, the Network Control Program. This program interfaces with application programs and access methods in the host computer, on one side, and with a variety of SNA components, including other Network Control Programs, on the other.

Communication between the application programs and the SNA network is controlled by the Virtual Telecommunications Access Method, a component of the host operating system described in the paper by Albrecht and Ryder. VTAM provides the primary operating system support for SNA functions, including overall network management, management of the host node, data transformations necessary for end-user communication, and data transmission.

The remaining papers in this issue discuss topics other than SNA. *LABS/7—A distributed real-time operating system* demonstrates the feasibility and practicality of a hierarchical set of computers for real-time event-driven applications. Function is effectively distributed between a large central host facility, which is able to provide a range of function and power not economically available in a small computer, and a local satellite, which is able to provide a degree of responsiveness and stability that is difficult to achieve in a central installation.

Protecting operating systems against unauthorized penetration remains a serious problem. *Penetrating an operating system: a study of VM/370 integrity* discusses a methodology for discovering operating system design flaws as an approach to learning system design techniques that may make possible greater data security. The authors found relative design simplicity to be the source of greatest protection against penetration efforts.

This issue begins a series of theme issues, where the majority of space, but not the entire issue, will be devoted to a topic of substantial interest to the data processing community. For help with the SNA portion of this issue, the Editor wants to thank Paul Green and Al Davis.

George McQuilken
Editor