Preface

The creation of software from existing resources is a well-established part of programming and software engineering for reasons of quality, productivity, and rapid development and deployment. Progress on this broad approach to reuse began at the lowest levels of programming, such as code, and has slowly reached toward the highest levels of software, such as architecture. The ability to reuse architectural assets is a vital step in the effective, efficient development of new systems and solutions from all the work that has gone before.

This issue presents the current work of the Enterprise Solutions Structure (ESS) project, a major IBM initiative focusing on the reuse of high-level software components, assets, and solutions in the large and complex systems environments of major customers. The assets being reused include architectures, methods, business patterns, and technology templates, among others. There is also a technical note on simplification of data flow diagrams. We are indebted to P. T. L. Lloyd of the IBM Object Technology Practice in Warwick, England, for his planning and coordination of this issue.

Plachy and Hausler introduce ESS through their paper on the motivation, purpose, architectural framework, development model, architectural contents, and advantages of ESS. They present the subject of high-level software reuse of solution assets, and introduce the other papers on ESS in this issue.

McDavid takes a high-level view of systems as a set of interconnected, business-oriented architectures and resulting software solutions. This leads to the description of a business systems architecture for ESS that uses key business concepts in the style of other architectures as a means for understanding and codifying systems aspects such as business requirements, boundaries of the business, and the delivery of business value.

The ESS model for reuse depends on the ability to express, in terminology and notation, the architectural aspects of existing reusable assets. The form of that expression is called the Architecture Description Standard (ADS). With ADS, assets that have been collected for reuse, or "harvested," can be systematically found and properly reused. The authors, Youngs et al., show that common descriptions of architectures in an operational model, and of components in a functional model, set the stage for active reuse.

Given the philosophy of standardization and description for reuse that surrounds ESS, Lloyd and Galambos take the next step by describing five standardized reference architectures, which are meant to cover the administrative systems needs of large businesses across the entire enterprise. These five have been carefully gleaned and abstracted from successful enterprise systems implementations.

Leishman moves beyond the basic ESS approach to describe her on-going research on solution customization. Two properties are found to be fundamental—commonality and variability—and the author describes how these properties have been applied in six examples of customizable systems. She also explains mechanisms for introducing commonality and variability, and shows in which phases of the customization life cycle the mechanisms can be applied.

Previous experience with enterprise-wide reuse has not been generally favorable. So, practical experience with such high-level and broad reuse is a necessary part of explaining the value of ESS. Harris, Rothwell, and Lloyd present the results of a number of uses of ESS in actual business situations. The conclusion is positive: ESS made it easier to rapidly and effectively provide work products that address business needs, and to support those work products later on. The authors also present the challenges they

found in further sophistication of ESS and the ESS approach.

In a technical note, Millet presents a modification to usual data flow diagram notation and use in which each data store symbol represents a whole database instead of a table. The claim is that such a modification makes it easier to create, understand, and maintain data flow diagrams.

As the *Journal* begins its 38th year, we would like to acknowledge the support of readers, authors, and referees that makes such a long history possible. We thank you and encourage you to continue your interest and participation in this publication. It also seems appropriate at such a time to state a few facts that sometimes escape us as we focus on a single paper, theme, or special issue. First, this quarterly publication is a refereed technical journal, which means that the integrity of each paper is ensured by a process that depends upon peer reviews of content, currency, and value by recognized experts within and outside IBM. Second, it is intended for the software and systems professional and applied research community worldwide. The papers are written for a technically aware readership, and we welcome submissions by knowledgeable authors around the globe, within and outside IBM. Third, the *Journal* has over 50 000 subscribers worldwide. Of those, approximately two-thirds are technical professionals and researchers outside IBM and one-third are IBM employees; two-thirds are in the United States and one-third are outside the United States.

The next issue of the *Journal* will be a unique, retrospective issue on some of the key turning points in the evolution of computing during the twentieth century, as reflected in the pages of the *Journal* since 1962. There will also be commentary on the importance of those turning points and the related papers by key individuals who saw it happen, written from their perspective at the close of the century.

Gene F. Hoffnagle
Editor