# Introduction

Transaction processing has been at the heart of information technology since the 1950s when the first large online reservation systems went into operation. Because the operating systems and programming languages of the day did little to support transactional applications, both business logic and control logic had to be manually coded into the application. This was an error-prone, laborious process that not surprisingly led to these applications acquiring a well-deserved reputation for complexity.

To address this complexity IBM pioneered and nurtured a category of software originally called transaction processing monitors but now called application servers. These products provided systems support for simplifying the development and management of transactional applications. Due to the complexity and cost of early transactional applications, transaction processing monitors were affordable only for the largest classes of applications (reservation systems, online banking, etc.). Now, however, advances in runtimes, along with better systems management and programming tools, allow usage to range from small commerce Web sites that permit catalog browsing, ordering, and payment processing to the largest airline-reservation, automotive-inventory-management, banking, and trading applications.

Despite the diversity of end-use applications, the infrastructure that supports transactions must provide a core platform for executing application logic and the flows of communication (often involving people) to and from that logic. The first generation of transaction processing monitors (CICS® and IMS™) established the semantics of processing and managing transactions. These semantics were generally expressed through extensions to the popular programming languages of the day (COBOL and PL/1). The second generation (Encina®, Distributed CICS) advanced distributed processing, focused on the C and C++ languages, and provided a very tight link between the transaction monitor and the relational database. The modern generation of Web application servers (such as WebSphere Application Server) employs many of the capabilities from the first two generations while moving to an inherently distributed, federated model based on Web standards. WebSphere® also moved from a model based on traditional programming language development to a model based on modern integrated and visual development tools. This has helped reduce the complexity of the development process and speed the time to application deployment.

All online transaction processing has stringent requirements that application servers need to satisfy, including response time, throughput, concurrent access, security, availability, and reliability. Application logic has moved from the simple scenarios of early transaction processing to complex workflow-based logic spanning multiple heterogeneous environments. Transaction scope has moved from that of a single database to workflows utilizing multiple information sources, many of which are themselves first or second generation transaction processing systems. And systems management has become far more important than previously, usually through connection to standard offerings like Tivoli® products.

In the last 20 years, the transactional infrastructure has built outward from its operating-system- and database-oriented heritage to incorporate much broader capabilities. For example, ever more advanced distributed computing technologies (initially based on SNA, then the Distributed Computing Environment and CORBA™, and most recently, Web services) have been incorporated to federate access to application logic from multiple systems. In the last decade, there has been a focus on object-oriented, and now on component-based development methodologies, to reduce application development time and maximize code reuse. Security technologies have grown ever more sophisticated, and autonomic computing technologies are increasingly automating systems operation. Today, the focus for transactional systems is to incorporate the latest programming technologies to permit the easiest possible specifications of policies, business rules, and methods for integrating disparate applications.

Although building new transactional systems remains an important focus for application servers, much of the use of these systems is moving towards business integration; that is, integrating disparate processes and systems across the enterprise. IBM's e-business on demand™ agenda focuses on helping customers address the inefficiencies that arise at the boundaries of vertically oriented functional applications. By using business integration products based on transaction systems to provide the tools and runtime capabilities, businesses can flexibly meld systems into true cross-enterprise business processes. Transaction systems must also rise to the challenge of providing people with uniform access to collections of applications that were most likely never written with shared usage in mind. This capability, using so-called portal-based technologies, contributes to what IBM calls the on demand workplace, and it has the opportunity to greatly simplify the use of ever more advanced computing capabilities from anywhere on any available device.

The papers in this issue all relate to transaction systems and to IBM's Websphere product family. While the initial products in this family focused on the core of transaction processing, subsequent innovation and engineering has been focused on the growing capabilities that transactional applications need, including all of the topics we have described above. In particular, the papers in this issue address autonomic operation, techniques for integrating diverse applications, advanced development tools, and portal technologies. By providing a snapshot of the state of the art in the development of the WebSphere Application Server, this special issue elucidates IBM's focus on this critical area.

Alfred Spector

Vice President
Services and Software
Research Division

John Swainson

General Manager
Applications and
Integration Middleware
Division

Danny Sabbah

Vice President
Strategy and Technology
Software Group