FOREWORD

Welcome to the world of IMS, my world. I have been involved with IMS since 1966. My goal has always been to make IMS as useful as possible. To that end, we in IMS have solicited input from our customers, which we used as requirements for new functions and features to make IMS a more useful product.

Today's IMS has only a superficial resemblance to the product (IMS 360 V1) that we shipped in 1969. An application program that ran on IMS in 1969, however, will still run today, unchanged, on the latest release of IMS. The most important rule for IMS developers all along has been to provide application program compatibility; we should never require a customer to change their application programs in any way to run on a new version of IMS.

I believe a lot of the success IMS has enjoyed is due to our commitment to our customers that their applications will continue to run on all the IMS versions released.

IMS is available as both a transaction manager (IMS TM) and a database manager (IMS DB), which can be used individually or together. IMS DB can be used with other transaction managers, such as IBM's Customer Information Control System (CICS). IMS TM can be used with other database managers, such as DB2 Universal Database for z/OS.

In the mid-1960s, IBM announced and released the System/360, a system that allowed a large number of terminals to be connected to the mainframe. System/360 initiated the start of a revolutionary change in the way data processing was performed in our customers' shops. It was the start of online processing.

In 1966, I was part of a team that was assigned to help a customer connect hundreds of terminals to a single mainframe computer. One of the problems we faced was the complex programming

that was required in every application to manage terminals. This was the state-of-the-art in programming design back then. It became obvious to us that requiring each application programmer to provide all the coding necessary to manage the terminals would greatly impede the growth of applications. A different approach was needed.

We came up with a design that defined transactions as the basis for online processing. Transactions in IMS were specified by a transaction code, a character string of length one to eight bytes, and staged for processing on a queue on disk storage. The idea of transactions:

- Enabled the customers to focus the design of their applications on manipulating the data and implementing their business goals.
- Liberated the customers' application developers from having to deal with the complexities of terminal communications.

The user on the terminal would enter the transaction code followed by the required data and send it to IMS. IMS handled all the complexities of managing the terminal and placed the message containing the user data on the appropriate queue. IMS then would schedule the correct application program, which only had to deal with data from the queue.

This design from 1966 is still the basic structure of IMS online systems.

Another major concern was the integrity of message processing and database changes. We journal all events to be sure that all messages that are received are actually processed and that all output messages are actually delivered. We allow the customer to specify that some messages can be designated as non-recoverable (meaning that we do not need to keep track of them). We also provide utility services so our customers can make back up copies of data, which can be combined with copies of data changes (from the journals) to recover data that is lost through hardware failures or other types of failures.

We feel that the growth in IMS usage over the past 35 years provides evidence that our strategies were correct. The integrity and reliability of IMS and its databases are major reasons for its current popularity.

IMS is a growing product. Each new version brings with it a list of new functions and features that have been requested by our customers to make the product more usable. Far from being a dead product, it is a vital and changing product that stays current.

IMS maintains its interfaces for user programs throughout changes to preserve the customer investment in application development. As an example of our support for older applications, I offer the following story that we experienced with one of our early customers:

IMS provides a Language Interface module that is linked with application programs to pass calls into IMS. The first version of this module (in 1968) was not reentrant. We corrected that in 1970 when we provided a reentrant copy of the Language Interface. We assumed that over time, all our customers' application programs would eventually be modified and would then be relinked

with the new interface. Meanwhile, we supported both versions of the Language Interface. Finally, in 1973 or 1974, we removed support for the non-reentrant version of the Language Interface. We were immediately notified by a customer that one of their application programs that had not changed since 1969, was failing. We quickly restored support for the non-reentrant module and the customer was able to continue using the old program. To the best of my knowledge, this application program continues to run today exactly as it did in 1969. I am not aware of any other active program offering that has preserved the user interface, like IMS has, through so many versions.

Our original target customer set was manufacturing, but we were careful to keep IMS as general as possible so as not to tailor our system to only that environment. Thanks to this effort, IMS has been successful in many environments such as finance, banking, government, and many others.

We have always encouraged our users to emphasize their applications that run on IMS, rather than focusing on IMS itself. The result has been that many of our customers (as well as our customers' customers) do not know that their work is being performed on IMS. This is one of the reasons this book is important; to help people understand the importance of IMS in today's world of data processing.

—Vern Watts
IBM IMS Distinguished Engineer Emeritus