An overview of three relational data base products

by S. Kahn

This issue of the* IBM Systems Journal *focuses on aspects of three recently announced IBM relational data base products. They are IBM Database 2, Query Management Facility, and Data Extract. This essay illustrates the requirements that these products were designed to address and gives a brief overview of their content and history. Its objective is to provide an introduction to the more specific and detailed papers that follow.

In this essay, three recently announced program products are described in an overview context. These products are IBM Database 2 (DB2),[1] a relational data base management system for the MVS/XA (Multiple Virtual Storage/Extended Architecture) and MVS operating system environments; Query Management Facility (QMF),[2] an end-user query and report-writing facility for the MVS/XA, MVS, VM (Virtual Machine), and VSE (Virtual Storage Extended) environments; and Data Extract (DXT),[3] a data extraction utility for the MVS/XA and MVS environments. Figures 1A and 1B depict the three products in the MVS and VM environments.

This paper first discusses the requirements on which the design for the products was based, and then gives a brief overview of each of the products. The other papers in this issue provide more detailed information about the products.

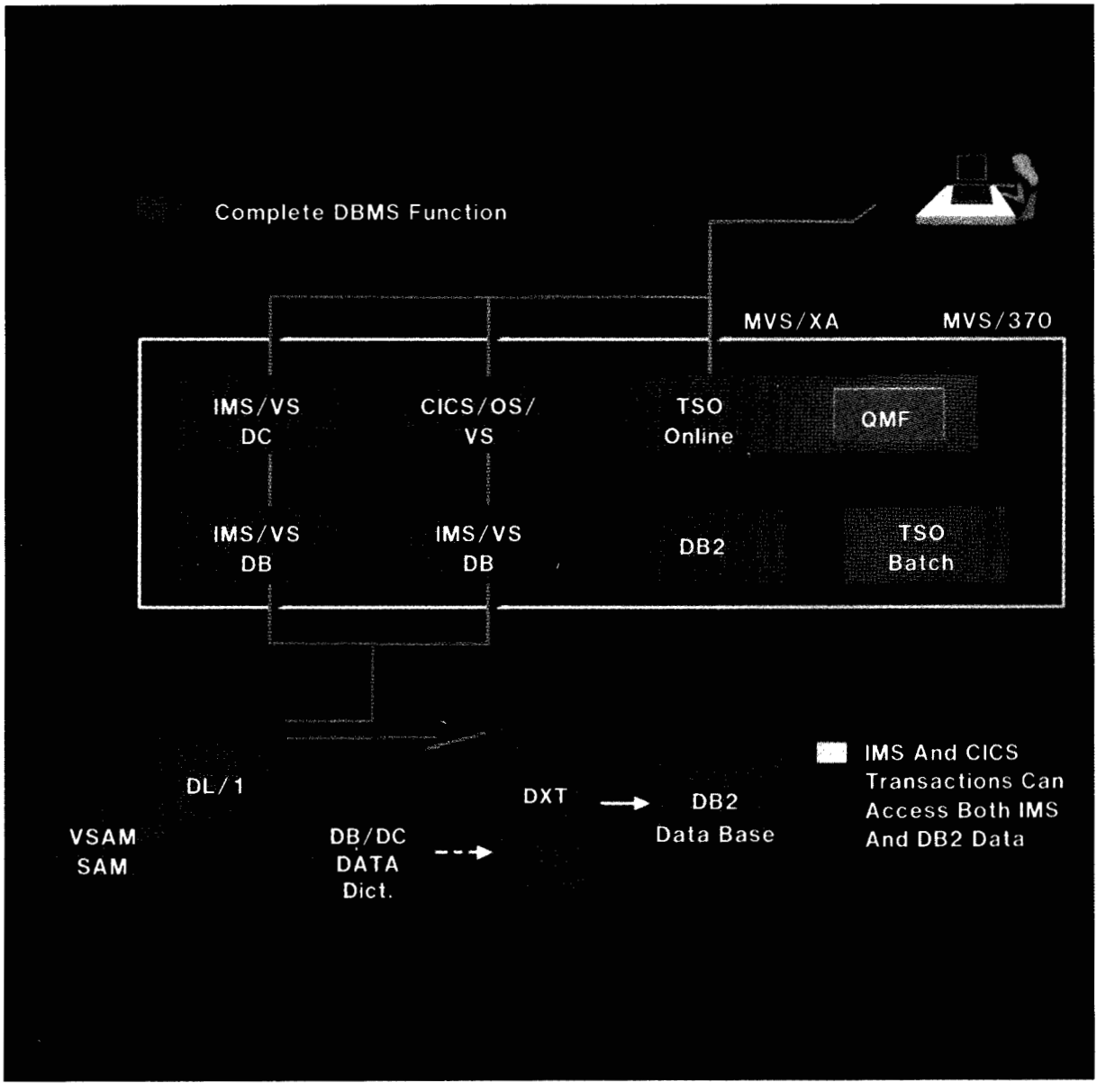
Requirements as the basis of design

Data base requirements. Data processing professionals have understood the need for data base management systems for many years. During the 1960s and 1970s most data base management systems (DBMSs) were aimed at the transaction and batch environments and placed a heavy emphasis on minimizing the computing resources used per unit of work, which was an appropriate direction given the computing economics of the times. Thus, Data Base/Data Communication (DB/DC) evolved as a single entity, and the sagas of the highly skilled data processing professionals who supported these systems became legends.

Computing economics changed dramatically in the late 1970s and early 1980s. The costs of computer professionals began to outweigh the costs of hardware. Also, the sheer power of processors available at affordable costs made the tuning of all but the highest-volume applications a lesser concern.

The infamous application backlog was growing at alarming rates. The term "invisible backlog" was coined to characterize the computer services that end users needed but never formally requested because of their frustration with the data processing organization's slow turnaround time on previous requests. The results of all of these trends were two clear requirements that had to be addressed by system software designers: (1) provide the tools to increase the productivity of data processing profession-

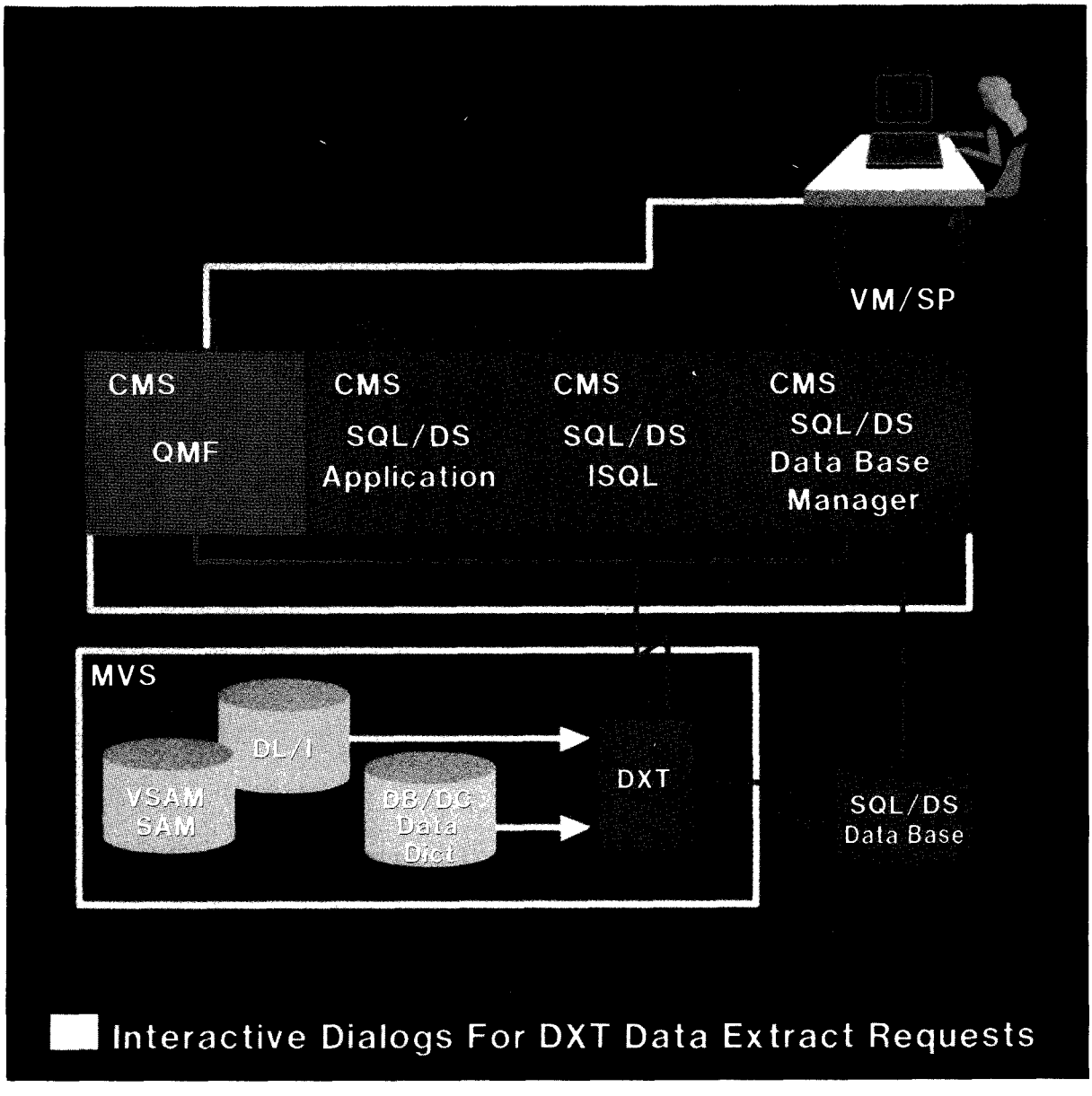
Figure 1A DB2, QMF, and DXT in the MVS environment



als by an order of magnitude, and (2) provide the tools to permit end users to directly access and manipulate data and actively participate in the development of the application solution. (See Figure 2.)

A number of products were developed in response to these needs, many of them successful to a degree. Most of those aimed at end users avoided the use of existing DBMSs, which were heavily oriented to the transaction environment. The result was a collection of products that could neither exchange nor share data, and in many cases could not support the volumes of data that existed in user organizations. In addition, many end-user-oriented products evolved

Figure 1B SQL/DS, QMF, and DXT in the VM environment

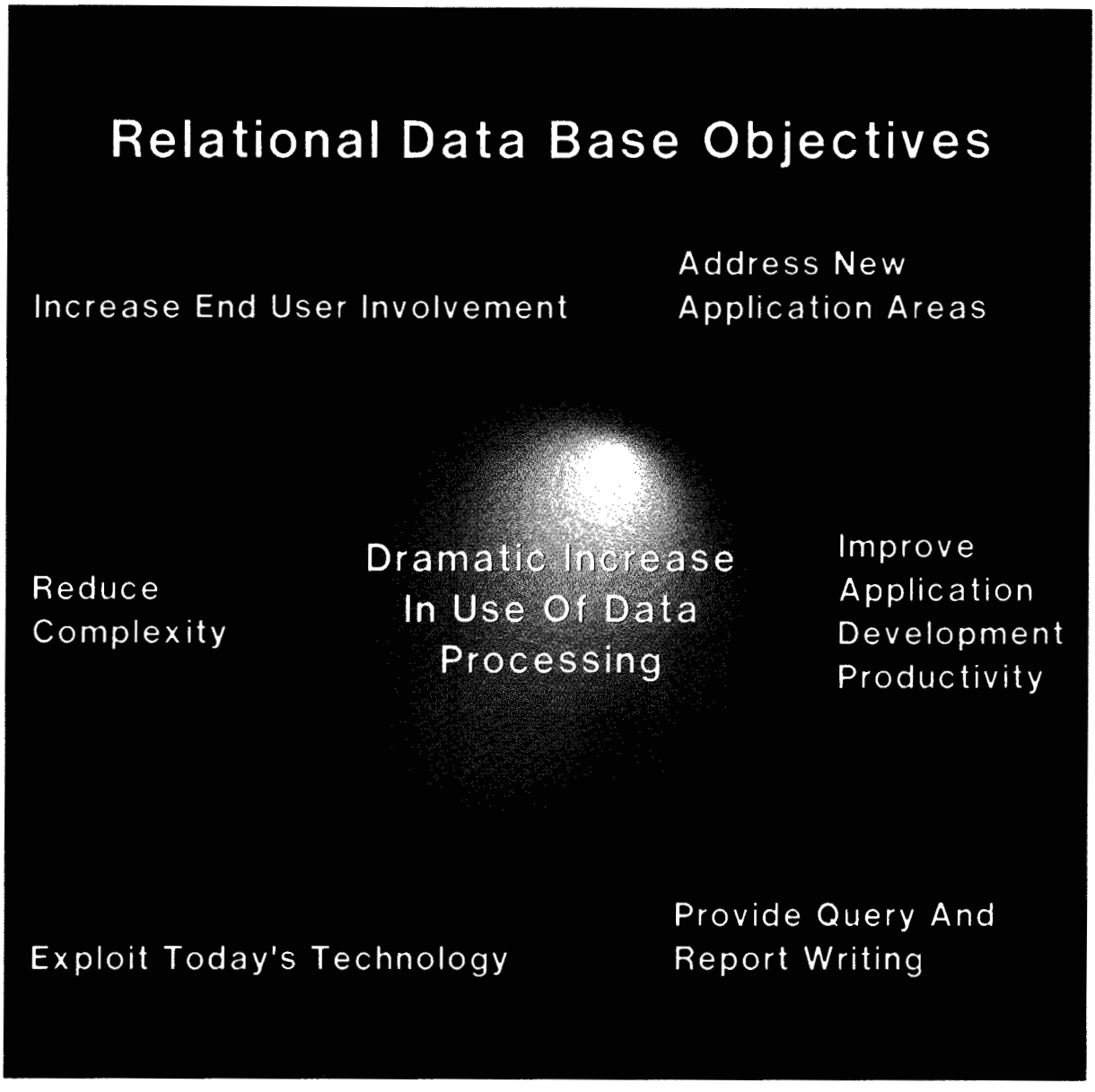


from tools intended for the data processing professional and were usable only by the more motivated business professionals.

Thus, the following user needs remained unsatisfied:

1. Access to data from all environments: transaction, batch, and interactive, including the ability to share a single data base among all of these environments.
2. A simple view of data that could be easily understood by both those who are data processing professionals and those who are not. This view of data had to shield the user from all the nuances

Figure 2 Data base requirements



of how data is stored and accessed. The user should have to specify only what he or she wants, *not* how to retrieve it.

3. The ability for this same easy-to-use DBMS to handle large volumes of data. This included the ability not only to physically address the data but also to partition it into pieces of manageable size for day-to-day housekeeping, such as data backup functions.
4. End-user tools supported by a DBMS that was robust enough to handle the workload growth that would result from the successful use of these tools. Studies indicated that the invisible backlog would surface very quickly after initial users of-

fered testimonials to their pilot usage. Thus, the data base management system had to be able to support all of the growth that the end users could afford to pay for, including very large N-way processors. In order to accommodate many end users on large processors, the software had to support a high degree of concurrency and include sophisticated failure detection, isolation, and recovery facilities.

Query/report writer requirements. Requirements for report writer software go back as far as those for higher-level programming languages. The original requirement was for a specification language to produce reports from data in existing tape or disk files, with higher productivity than could be achieved via procedural languages like COBOL. A number of products were developed to respond to this requirement, including IBM's Report Program Generator (RPG) and Generalized Information System (GIS). These tools were generally batch-oriented and aimed at professional programmers. A key implicit assumption was that the end user understood the report format that he or she wanted, whereas the professional programmer understood the data available in computer files. Although these tools addressed many of the requirements, they also generated some new ones.

End users wanted more flexible tools that would allow them to iteratively analyze a problem.

End users wanted more flexible tools that would allow them to iteratively analyze a problem. They wanted to see the data available in files and make decisions about how to proceed based on it. They wanted quick turnaround and preferred not to deal with the data processing professional as an intermediary. They also wanted to decide on the final format of the report *after* they had determined the precise data that they needed. They wanted an on-line query system and an interactive report writer!

A number of products evolved in this direction. Some batch programming tools were augmented with line-at-a-time, command-driven languages oriented to teletypewriter terminals. Often, these tools lacked the flexibility that the end user wanted, at least partly because of the rigid structure of the underlying data management facilities. Thus these new requirements remained unsatisfied.

Therefore, the requirements for the IBM query/report writer product that evolved were stated as follows: (1) An easy-to-use query interface between the computer system and the end user that fully utilized the power of a visual display terminal. (2) The ability to iteratively search the data base in real time until the desired data was found, since the end user might not know precisely what data had to be retrieved when he or she made the initial search. (3) The ability to present the selected data as formatted reports and to iteratively customize report formats. (4) The ability to produce business graphics, in addition to formatted reports. (5) The ability to create private data bases for individual or department use, and to share that data when required, in addition to accessing corporate data bases. (6) The ability to save queries, the results of queries, and report formats for later use and sharing. (7) Use of the above facilities to require minimal consulting and/or system administration by data processing professionals.

Data extract requirements. The requirement for generalized utility programs to move data between both similar and dissimilar file structures has existed since the first attachment of storage devices to computers. The data extract requirements focused on the data processing professional who was familiar with the data already available in machine-readable format. The ability to efficiently execute a number of separate extract requests via a single pass through the data and the power of the selection specification were key considerations. The ability to take advantage of existing data definitions, such as those stored in the Data Dictionary, was recognized. Finally the ability to move data easily across multiple CPUs was added as a key requirement.

Addressing the requirements. Research and development efforts were started in a number of IBM laboratories to address these requirements in the mid-1970s. As is typical, the requirements became clearer and changed as time progressed, and the objectives of the corresponding research and/or development efforts changed with them. What follows is a brief overview and history of the three IBM

program products that were developed to address these requirements: IBM Database 2, Query Management Facility, and Data Extract.

IBM Database 2

IBM Database 2 (DB2) is a relational DBMS for the MVS environment. It allows concurrent access to data bases by IMS/VS-DC (Information Management System/Virtual Storage Data Communication), CICS (Customer Information Control System), and TSO (Time Sharing Option) users, both interactive and batch. Data under DB2 control can be accessed by application programs written in COBOL, PL/I, FORTRAN, and Assembler and/or directly from TSO terminals.

DB2 is a relational DBMS for the MVS environment.

Relational data base theory originated in the IBM Research Laboratory at San Jose, California.[4] The theoretical work was followed by several prototype efforts in IBM, including the widely publicized System R[5-7] at Research in San Jose, and the less well known Peterlee Relational Test Vehicle[8] at the IBM United Kingdom Scientific Centre, then located at Peterlee, England. Several IBM customers used these systems as part of joint studies that demonstrated the commercial viability of relational data base management systems.

Some of the key pieces of IBM's relational data base technology that evolved were the Structured Query Language (SQL)[9] and techniques for generating executable code from SQL statements. Other important results were obtained in query optimization, concurrency control, authorization mechanisms, and techniques for dynamically changing the structure of stored data, i.e., adding columns (fields) to existing rows (records), without unloading and reloading the data.

During the same time period, IBM's Santa Teresa Laboratory developed techniques to more tightly couple DB/DC systems to MVS. These techniques included logging, locking, and recovery protocols and a subsystem structure compatible with MVS Cross Memory Services and MVS/XA, both of which were being developed by IBM's Poughkeepsie Laboratory.

Another group within Santa Teresa developed techniques to manage very large data bases. A number of diverse facets of this problem were explored, including expanded physical addressing limits, algorithms for efficiently managing and searching very large buffer pools residing in virtual storage, and utilities that could access data bases concurrently with user applications. In addition, a physical data management structure was designed that minimized both the need to reorganize data and the time required to recover it in the event of a storage device or media failure. Finally, techniques were developed to allow partitions of a data base to be separately activated/deactivated, recovered, and reorganized. The physical data management structure was designed to utilize data management services provided by the Data Facility Products (DFP/XA and DFP/370).

The DB2 program product emerged from all of these technology efforts. As detailed design progressed, attachment protocols were developed cooperatively with the CICS/VS development team located at IBM's Hursley Laboratory near Winchester, England, and the IMS/VS development team at Santa Teresa.

DB2 was implemented using the quality-oriented programming process technology of reviews, inspections, and selected prototyping. DB2 was coded using the Programming Language/Systems (PLS) language and submitted to formal test via a series of operating subsets of the overall product staged over several years. Functional testing began with the early subsets and continued through full system testing with TSO, IMS/VS Release 1.3, and CICS/OS/VS Release 1.6 (the latter two were being developed in parallel), load/stress testing, and performance measurement.

The task-oriented library of publications for DB2 was developed in parallel with the code. User profiles, task definitions, and outlines of the proposed publications were cross-checked to ensure that all of the required information was included and that no more than two (and preferably only one) publications were required to perform any task. Drafts of the publications were extensively reviewed and revised to ensure technical accuracy and usability. A large number of

on-line HELP panels were developed to provide the user of a display terminal with a subset of the reference material on demand. Experience-derived information, such as tuning guidelines, was added based on feedback from early users.

The high quality of the DB2 code and publications is evidenced by both the results of testing in the laboratory and early customer experience.

Relationship between DB2 and SQL/DS

Structured Query Language/Data System (SQL/DS) is IBM's relational data base management system for the VSE and VM environments. It was developed at IBM's Endicott Laboratory.

SQL/DS is a direct descendant of the System R project. The Endicott development team started with the System R design and prototype and adapted the design for the VSE environment. Code was rewritten and recovery facilities were extended in order to transform the System R prototype into a program product. Some of this rewritten code became part of the DB2 program product after being adapted to fit within the structure of DB2. The development team added Interactive SQL (ISQL) to support ad hoc query and report writing and a utility to extract data from DL/I data bases and put it into SQL/DS tables. SQL/DS was released as a program product in 1981. The SQL/DS design was then extended to support the Virtual Machine/Conversional Monitor System (VM/CMS) environment. Prerequisite enhancements to VM/SP were made both to minimize and facilitate the changes required to SQL/DS itself. SQL/DS Release 2, which supports both the VSE and VM/CMS environments, was released during 1983.

DB2 and SQL/DS both support SQL as their primary interface. IBM's SQL Control Board, composed of representatives of IBM's San Jose Research, Santa Teresa, and Endicott Laboratories, was formed in the late 1970s to coordinate the precise semantics (functions to be performed) and syntax (language specification) for SQL. The SQL Control Board also defined a common subset and approved product-specific deviations from and extensions to the standard. The compatibility objective is to allow portability of application programs and queries between SQL/DS and DB2 while recognizing the differences in their physical data management components.

In general, SQL Data Manipulation Language (DML) and Data Definition Language (DDL), for logical objects such as *tables*, is compatible between SQL/DS and DB2. In addition, DB2 DDL supports physical objects such as DB2 *tablespaces* which have no analog in SQL/DS. Programming techniques such as the use of common subroutines to process SQL error return codes have been identified to further reduce the effort required to transport application programs between DB2 and SQL/DS.

Different physical data management components were developed for DB2 and SQL/DS.

Different physical data management components were developed for DB2 and SQL/DS to satisfy the differing requirements of MVS and VSE users. For example, support of the extended addressing and functional recovery architecture of MVS/XA, a logging and recovery protocol to handle large data bases and heavy workloads, and the need to support N-way multiprocessors, are requirements that are unique to MVS users.

The Interactive component of DB2 (DB2I) supports the data processing professional in tasks such as installing DB2, running DB2 utilities, and debugging SQL statements. The DB2I user is assumed to be familiar with commonly used facilities of both the Operating System utilities and the Interactive System Productivity Facility (ISPF). Thus, DB2 relies on QMF to provide facilities for end users, whereas SQL/DS provides some end-user facilities via its ISQL component.

In summary, DB2 and SQL/DS provide very similar interfaces for application programming and query. Different physical data management components were developed to satisfy the different requirements of large-system (MVS) versus smaller-system (VSE) users. The differences between SQL/DS and DB2, while important, are generally visible only to support personnel such as data base administrators, systems programmers, and information center specialists.

Query Management Facility

The Query Management Facility (QMF) is an end-user query and report-writing facility for the MVS/XA, MVS, VM, and VSE environments. In the MVS/XA and MVS environments it accesses data managed by DB2, whereas in the VM and VSE environments the data is managed by SQL/DS. QMF supports two styles of query: Structured Query Language (SQL) and Query-by-Example (QBE).[10] The result of a query is displayed on the screen with default formatting. The format can be interactively customized via the form using a fill-in-the-blanks approach. The user may iteratively refine his or her query and/or report form and immediately see the results displayed. Queries, forms, and results may be saved for future use or sharing. A command facility is provided to perform functions such as initiating queries and reports. A series of commands may be stored for future use as a procedure. A sample program is provided with QMF to illustrate how the Interactive Chart Utility (ICU) of the Graphical Data Display Manager (GDDM) program product may be used to generate business graphics from the results of a query.

QMF is an end-user query and report-writing facility.

In the development of QMF at the IBM Santa Teresa Laboratory, an early objective was to design an end-user query facility on top of System R, such that it could be easily implemented for whatever strategic relational data base products evolved. The focus was on delivering a reasonable set of functions with usability characteristics that would appeal to someone who was not a data processing professional. The early design efforts defined a structure that used the facilities of the operating system and the DBMS for multiuser support in order to simplify the QMF implementation. With this structure in place, the designers were able to concentrate on finding the most usable way to present powerful function to those who are not data processing professionals. Early prototypes were tested at the IBM San Jose Human Factors Laboratory. Numerous changes to both product externals and publications were made based on these human factors evaluations.

A user view of the system, or model, was developed. The QMF user deals with objects such as queries, forms (report specifications), results (the output of queries), and reports (results in the formats specified by forms). Commands were designed to allow users to manipulate these objects without having to know anything else about the computer system. This simple, consistent, yet powerful model of the user workspace is the key to an easy-to-use system. The use of an interactively built form, instead of a linear report specification language, is also a key to usability.

During the development cycle the QBE query language, invented at IBM's Thomas J. Watson Research Laboratory, was added, and the technology to translate QBE queries into SQL statements was perfected. QBE draws a two-dimensional picture of a table on the screen and allows the user to express a query by placing abbreviated commands and values under the appropriate column headings.

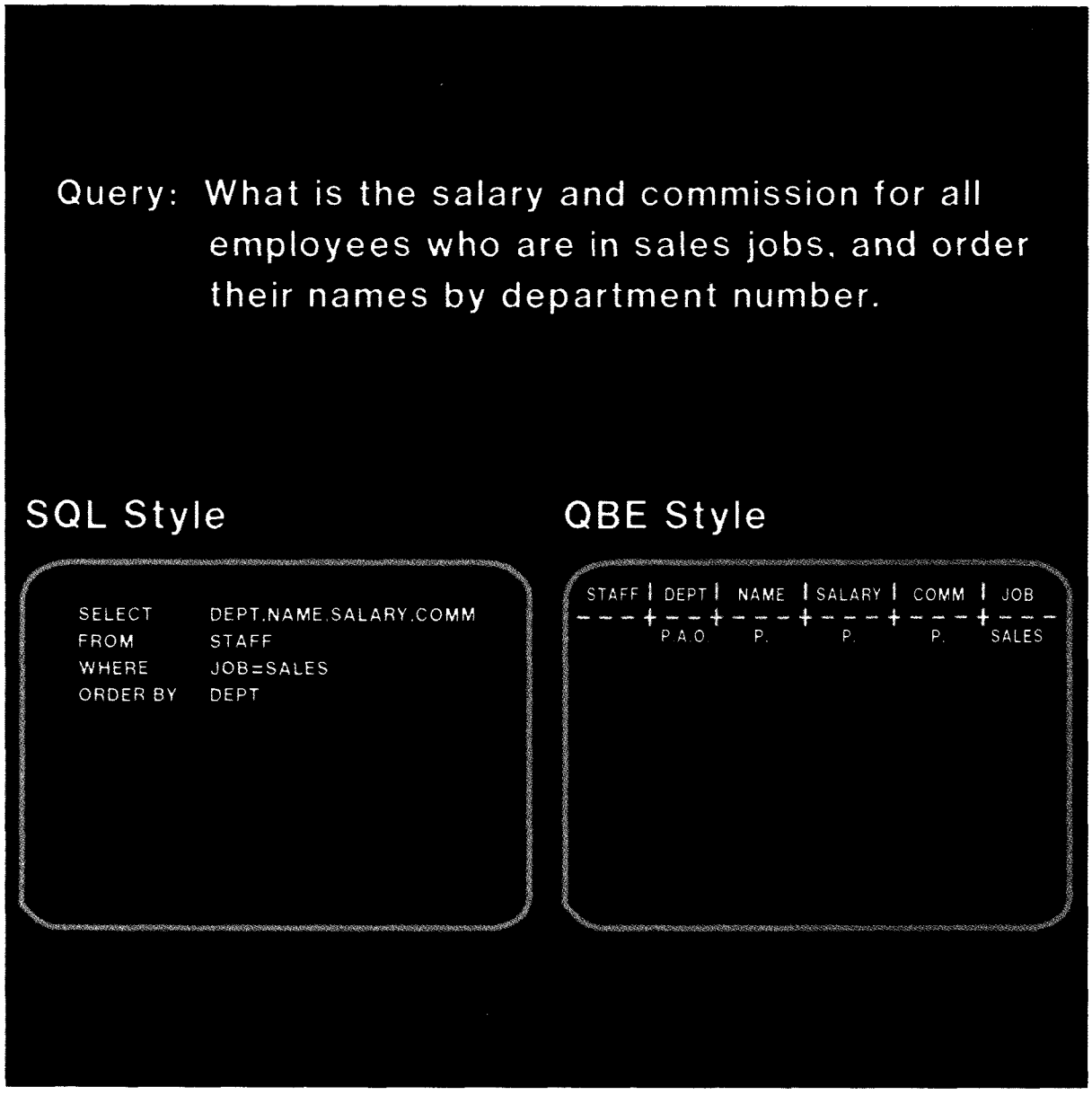
The Watson Research Laboratory QBE prototype, including a data management component in addition to the query facility, was released as an Installed User Program (IUP) in 1978 and was used by a number of customers. Early experience indicated that the QBE style of query was powerful, yet easy to learn and use. Many early users wanted to see QBE offered as part of a program product with full function DBMS and report writer capabilities (Figure 3).

The inevitable debate as to whether SQL or QBE provided the "best" user interface style was settled by comparative testing. The results showed that both were easy to use and that many users had strong preferences for one or the other. Thus both are offered in the QMF product.

QMF has a consistent end-user interface across the MVS/XA, MVS, VM, and VSE operating systems, using either DB2 or SQL/DS as its relational data base management system. QMF was implemented at the IBM Santa Teresa Laboratory for the MSV/XA, MVS, and VM environments and was adapted for the VSE environment by the IBM Endicott Laboratory.

The QMF program product that resulted from this iterative process of prototyping and human factors testing was then subjected to extensive formal functional testing. Early feedback from its users indicates

Figure 3 An SQL and a QBE screen for the same query

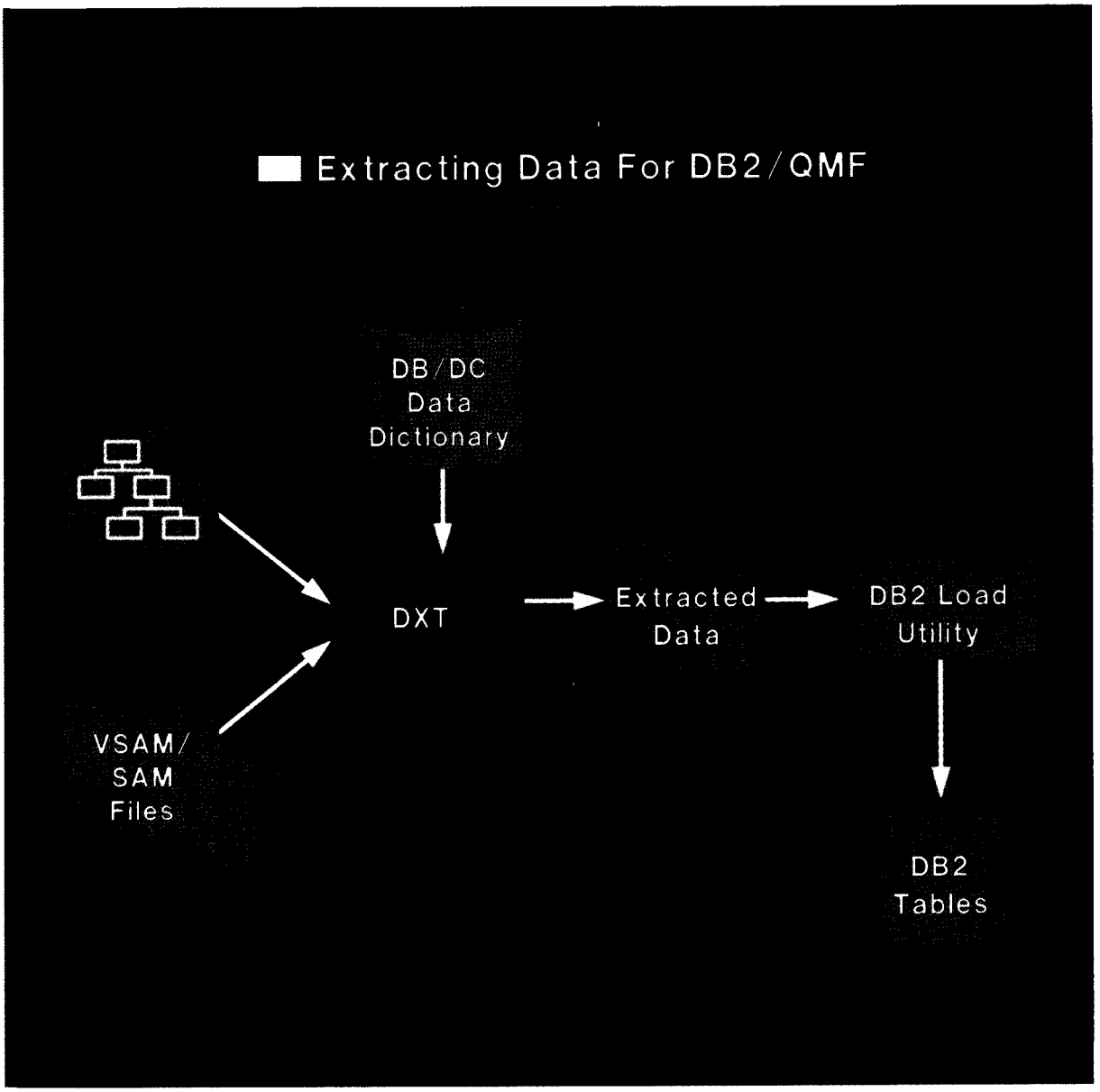


that QMF is a high-quality product and is indeed powerful, yet easy to use.

Data Extract

Data Extract (DXT) is a utility program that extracts data from existing IMS/VS DL/I data bases and Virtual Storage Access Method (VSAM) or Sequential Access Method (SAM) data sets and converts the data to a format suitable for loading into SQL/DS or DB2. Alternatively, the extracted data, which is stored in a standard data set format, may be accessed for other uses, for example by APL-based statistical analysis tools. Extract requests are expressed in a subset of SQL. Multiple extract requests may be batched for execution, allowing them to be satisfied by a single

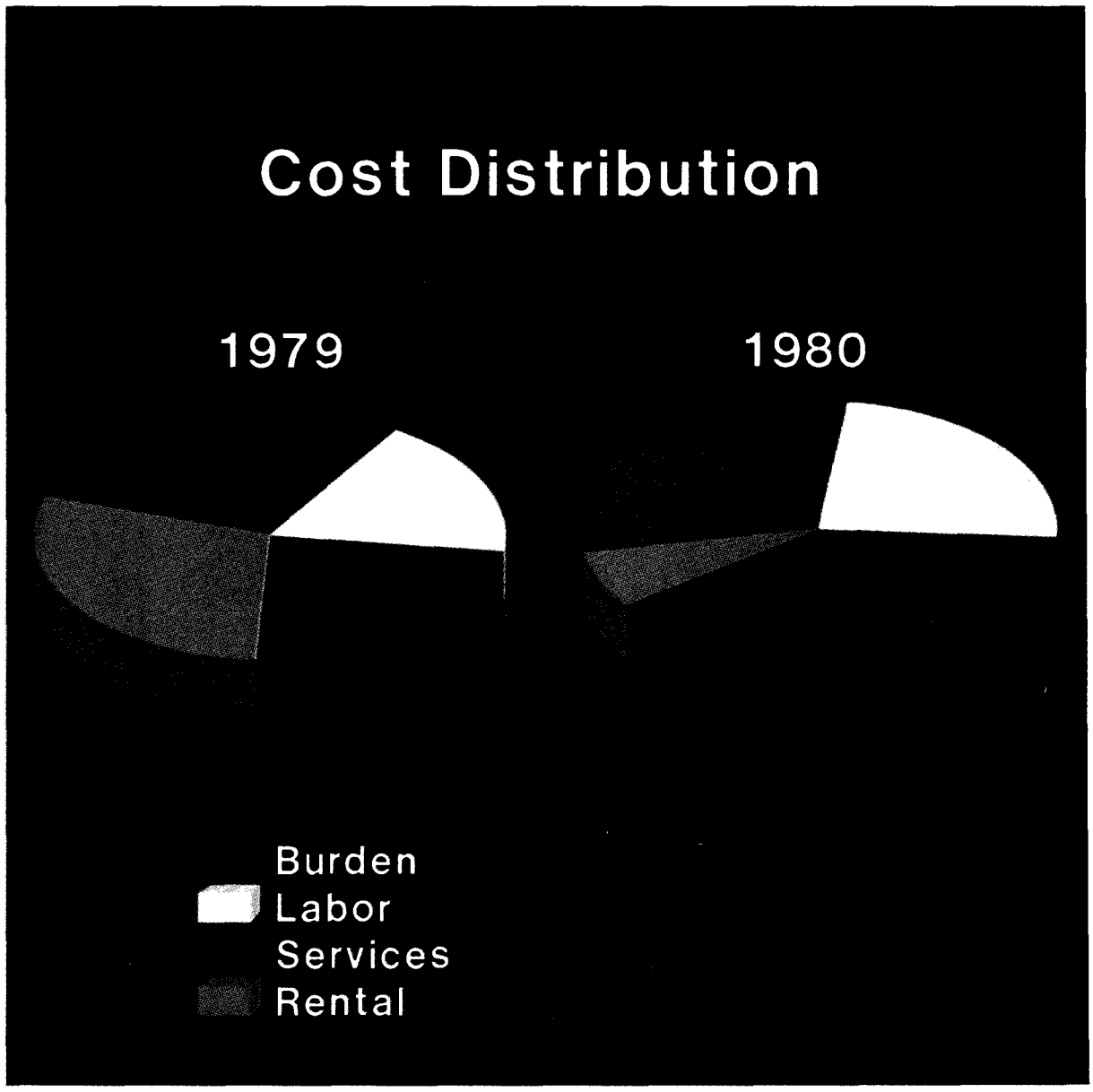
Figure 4 Extracting data for DB2/QMF



pass through a data base or data set. The extract operation itself is executed asynchronously to the extract definition and optionally on a different CPU, using the facilities of Job Entry Subsystem (JES) networking. In addition, existing definitions of data elements may themselves be extracted automatically from the IBM DB/DC Data Dictionary.

DXT was developed at the IBM Santa Teresa Laboratory, where early design efforts focused on developing an efficient generalized search algorithm, including the capability to satisfy multiple extract requests during a single pass through a data base or data set. The DXT structure—an input manager and an execution manager connected via descriptor and request

Figure 5 Business graphics on screen



data sets—was then defined. The design was expanded to use standard JES networking to allow specification and execution to occur on separate CPUs. The Dictionary Access Program (DAP), which executes as a Data Dictionary command, was designed to allow existing record/segment and field definitions to be used as input to a DXT extract specification. Interactive System Productivity Facility (ISPF) dialogs that assist the user in generating control statements were also designed and implemented (Figure 4).

After the individual pieces were designed, the complete product was coded and tested. The result is a usable and efficient tool to extract data from IMS/VS DL/I data bases and VSAM and SAM files and convert

it to a form suitable for loading into a variety of tools, including DB2 and SQL/DS.

Future trends and directions

IBM Database 2, Query Management Facility, and Data Extract represent significant milestones in the evolution of data base and query/report writer products. Early user feedback indicates that they do in fact address the key requirements and that each is a high-quality product. These products will continue to evolve as feedback from users identifies areas where tuning is required to meet overall function, performance, reliability, and usability expectations.

No one, not even the product manager, can accurately predict the ultimate content of these products. However, as one looks at the intersection of user requirements with available technologies for both these and related IBM products, reads the trade press, and talks with users, certain trends appear.

DB2 will evolve to extend both the relational data model and its implementation. Extensions to the relational data model will be pursued to more fully address requirements for intertable relationships. Additional data types for applications in such diverse areas as engineering and scientific products and office automation will evolve. The implementation of the relational model will take advantage of a wide range of physical storage structures and more efficient path selection algorithms. Improvements to the facilities for managing very large data bases will continue.

Continual improvements to the report-writing capabilities of QMF will be made, and its integration with other key tools for business professionals, such as business graphics and text processing, will continue (Figure 5). Application generation and data entry facilities for the business professional are natural additions via either product extensions or complementary products. Support for languages other than English will be added. The technology to handle less-structured query languages, called natural language in the literature, will be pursued.

DXT will evolve to allow greater flexibility in controlling the formatting of the output data, in editing the content of the selected record or segment, and in itself specifying the search criteria.

Both IBM and other suppliers are likely to take advantage of the capabilities offered by DB2, QMF, and DXT to address a wide spectrum of applications. The scope of the target environments will expand to include networks of mainframes and intelligent workstations.

Conclusion

Neither the full impact of DB2, QMF, and DXT nor the evolution to their ultimate content will occur overnight. However, it is expected that by the end of the 1980s the introduction of IBM Database 2, Query Management Facility, and Data Extract will be seen as a significant milestone in the evolution of data management, data access, and data presentation capabilities.

Cited references

1. *IBM Database 2 General Information Manual* (Program Number 5740-XYR, Release 1.0), GC26-4073, IBM Corporation; available through IBM branch offices.
2. *Query Management Facility General Information Manual* (Program Product, Program Number 5668-972, Release 1), GC26-4071, IBM Corporation; available through IBM branch offices.
3. *Data Extract General Information Manual* (Program Product, Program Number 5668-973, Release 1), GC26-4070, IBM Corporation; available through IBM branch offices.
4. E. F. Codd, "Relational database: A practical foundation for productivity," 1981 ACM Touring Lecture, *Communications of the ACM* **25,** No. 2, 109–117 (1982).
5. M. M. Astrahan et al., "System R: Relational approach to database management," *ACM Transactions on Database Systems* **1,** No. 2 (1976).
6. M. W. Blasgen et al., "System R: An architectural overview," *IBM Systems Journal* **20,** No. 1, 41–62 (1981).
7. D. D. Chamberlin et al., " A history and evaluation of System R," *Communications of the ACM* **24,** No. 10 (1981).
8. S. J. P. Todd. "The Peterlee Relational Test Vehicle—A system overview," *IBM Systems Journal* **15,** No. 4, 285–308 (1976).
9. *An Introduction to SQL* (Program Product, Program Number 5740-XYR, Release 1.0), GC26-4082, IBM Corporation; available through IBM branch offices.
10. M. M. Zloof, "Query-by-Example: A data base language," *IBM Systems Journal* **16,** No. 4, 324–343 (1977).

Reprint Order No. G321-5212.

Sam Kahn *IBM General Products Division, Santa Teresa Laboratory, P.O. Box 50020, San Jose, California 95150.* Mr. Kahn is currently Advanced Database Products Manager at the Santa Teresa Laboratory and is responsible for the planning, design, development, and testing of the DB2, QMF, and DXT Program Products. He joined IBM in 1967 as a systems engineer. He began working in the data base area in 1978 as a planner for SQL/DS. Prior to his current assignment he was the Planning Manager for IMS/VS V1, the IBM DB/DC Data Dictionary, and IBM Database 2. Mr. Kahn received the Bachelor of Engineering Science degree from Rensselaer Polytechnic Institute and the Master of Science degree in industrial administration from Purdue University.