DBMS for Distributed Computers & Networks

DIMITRIS N. CHORAFAS

DBMS for Distributed Computers & Networks

DIMITRIS N. CHORAFAS

a petrocelli book
new york / princeton

Designed by Diane L. Backes
Typesetting by Backes Graphics

Printed in the United States of America
1 2 3 4 5 6 7 8 9 10

Library of Congress Cataloging in Publication Data

Chorafas, Dimitris N.
DBMS for distributed computers and networks.

"A Petrocelli book."
Includes index.
1. Data base management. 2. Electronic data processing—Distributed processing.
I. Title. II. Title: D.B.M.S. for distributed computers and networks.
QA76.9.D3C47 1983 001.64'2 83-4220
SBN 0-89433-184-1

— FOREWORD —

This book has been written for the information scientist who faces a dual challenge: integrating intelligent machines into a local or long haul network, and employing a database management system (DBMS) at each site. The theory and examples presented herein come from the author's personal experience, yet the reader will be well advised to remember that we are still at the beginning of this new art.

The text addresses itself to both the systems specialist and the experienced end user. Its emphasis is on explanation, background, description and understanding. The typical reader will have had computer experience but is always keen to upgrade his skills along the road which is being opened by technology.

For the past decade, database management systems have received a rather cool reception from data processing managers. Few of the traditional systems analysts and programmers knew how to use them; most saw DBMS as a solution to a very specific applications problem—which is not the case. Like nearly everything else in the computer world, the DBMS evolved as disjointed, isolated tools that only systems programmers could or would understand. However, times have changed. DP managers have warmed to the idea of database management systems and the query type, user-friendly languages supported by them, and so have the users who are increasingly getting in contact with bigger and bigger chunks of the system's operation due to microcomputers in the workplace.

Functional characteristics, economic considerations, systems security, distributed locations, access authorization, journaling, data communications, and the effective distribution of processes, information elements and applications underline the DBMS need. These subjects are just as valid for business/administrative management as for technical design.

Even the approach we take toward the DBMS has changed. Early efforts focused on the tools. The first systems became commercially available in the mid-1960s; they incorporated file management, sort and

merge, retrieval, and the early query language developments. Through them, the computer profession was presented an operating systems-type faculty to handle the database. However, the fundamental theory was still missing and this handicapped systems support.

As long as we spoke of mainframes, and the DBMS user population was still thin, the lack of a proper understructure had no widely felt effects. By the early 1970s, though, a movement began toward placing distributed data processing minicomputers in industrial factories, sales offices, and banking branch offices–operated by nonprofessional DP personnel.

Suddenly, the perspective was altered. With distributed power available at the periphery, minicomputers came in close contact with the user population much more than had the centralized realtime operations. With the cost of CPU and memory devices dropping quickly while communications remained expensive, by the late 1970s a trend started toward distributed databases.

Only when we organize an integrated database, partitioning and distributing it as the environment implies, can each end user's requirements be answered in a timely, interactive, and easily accessible manner. But who is going to manage the database? DBMS, a subject which 15 years ago was reserved to the best professionals suddenly became everybody's concern. Micro-DBMS structures suggest a *one day* training period.

The concepts characterizing the micro-DBMS design and usage are a leap forward, not a step backward from those of mainframes. Micro-DBMS structures run on a variety of 8- and 16-bit machines in thousands of installations in some 40 countries. They have been used to develop a wide variety of application systems having features that are not supportable through file management. Their databases range from relatively small sizes, spread across a few floppies, to quite substantial sizes spread across megabyte hard discs.

Most importantly, by being based on recent advances in DBMS technology, they provide features as yet unavailable with many mainframe DBMS: The security, integrity transaction logging, recovery, and performance control capabilities are comparable to the best offerings on mainframes and minis. The same is also true of the applications concepts.

For this reason, this text stresses fundamentals. Its aim is to present, in a factual and comprehensive manner, how we can use the DBMS to hold such diverse, and in many cases dispersed, information systems in

one integrated whole. This brings into perspective the prerequisites of a database design tool that applies to operations as a whole and the specialized software that manages the information elements, no matter where they are located or to which operation they pertain.

In the new applications environments, both databases and the DBMS are distributed. "Distributed" means a scaling of computer resources: hence, the emphasis on mini- and microcomputers. Distributed also calls for an online capability. This is being answered through networks, both local and long haul. Networking makes available a database utility.

Along this structure of critical issues, the first chapters stress DBMS implementation in which are reviewed milestones in the development of the DBMS (Chapter 1), and text processing needs, underlining the importance of text and data definition and manipulation (Chapter 2). Chapter 3 presents the DBMS available for mini- and microcomputers. Chapter 4 emphasizes a polyvalent design approach to the projected use of the a DBMS.

Data structures–hierarchical, networking, and relational–are the subject of the next four chapters. The logical organization and types of data models are presented in Chapter 5, while Chapter 6 reviews the better-known data structures. Relational models and query capabilities are treated in Chapter 7; both strengths and weaknesses are emphasized. Then, making use of what has been said on DBMS and data structures, Chapter 8 discusses the database engine, including logic over data and rearend.

The last six chapters further the reader's understanding of the need for enhancing the user environment. This involves user level protocols (Chapter 9); Session Control, Presentation Control, and Virtual Terminals (Chapter 10); the concept of a DBMS supported distributed network (Chapter 11); how to manage the distributed database (Chapter 12); and integrity in database systems (Chapter 13).

Chapter 14 is a case study. There has always existed in our profession a misunderstanding of the great role of preparation. This led, and still leads, to "throwing money at a problem" rather than facing it squarely. Still, many people think of a DBMS in terms of what it is not–they want it to do what it cannot do–while they fail to capitalize on its strengths and advantages. Such an attitude has wronged our profession. At times it has slowed development, and can be found in the background of many

failures. The explanation given in this text aims to balance the scales. Men and systems not part of the solution are part of the problem.

Let me close by expressing my thanks to everyone who contributed to making this book successful: from my colleagues for their advice, to the organizations I visited in my research, for their insight and to Eva-Maria Binder for the drawings, typing and index.

1983
Valmer and Vitznau

Dimitris N. Chorafas

— Contents —

— 1 —

MILESTONES IN THE DEVELOPMENT OF DBMS

— INTRODUCTION —

Database management systems (DBMS) have evolved to the point of general acceptance, yet a major problem still facing the user is their effective utilization. A vital element in achieving good database usage and responsiveness is the design of this database. This calls for a methodology that provides a DBMS compatible structure from a set of information elements. Such effort includes logical design, requirements analysis, and physical design.

Performance tradeoffs will always exist among multiple users and the relationships between them should be fully described. This will be aided through the proper answers to key questions: What are user requirements and how can they be expressed? How can they be translated into an effective database structure? How can this text and database structure be adapted to a changing environment? What qualities should the DBMS possess to ably answer user needs?

As Figure 1-1 shows, at the center of any DBMS is a core of text and data. Everything around it serves this core. We will return to this issue when we consider design perspectives for text and databases, but throughout the discussion which follows we should never forget that the DBMS serves the core, not vice versa. Another basic principle is that a DBMS cannot be implemented and then forgotten; it must be steadily monitored and periodically adjusted in accordance with the developing needs for its usage.

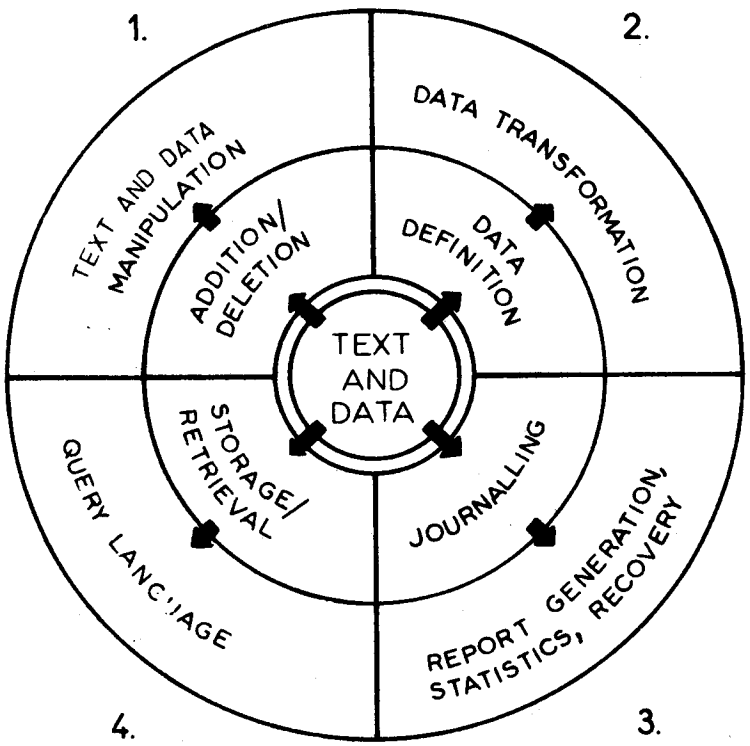
THE TOTAL DBMS INVOLVES:
1.
2.
TEXT AND DATA MANIPULATION
DATA TRANSFORMATION
ADDITION/ DELETION
DATA DEFINITION
TEXT AND DATA
STORAGE/ RETRIEVAL
JOURNALLING
QUERY LANGUAGE
REPORT GENERATION, STATISTICS, RECOVERY
4.
3.
ALL 4 PARTS ARE VITAL.
THEY CAN BE ORGANIZED IN A FUNCTIONAL NETWORK OF IE AND DB SERVICES

FIGURE 1.1

A third fundamental consideration is that if we choose a DBMS, we must stick with it. Under no conditions should we mix different DBMS in the same network–which leads to the postulate that we should choose our DBMS with great care. This becomes most challenging with the introduction of DBMS for microcomputers, as we will see in the proper section.

The fourth consideration regards the often forgotten fact that we should create the proper environment for DBMS implementation: develop AP (applications program) and IE interfaces, train people in its usage, and provide the proper infrastructure. It is a fallacy to believe that because the organization used DP for, say, 25 years, the infrastructure and the proper environment already exist. They *do not* necessarily exist, and this is usually the rule.

Even the much lower level of file management utilities includes the prerequisites of file creation: reorganization, file insert and delete, sort, dump, copy; transfer to other support media; transmission upline and downline; and loopbacks. The preparatory work will necessarily be more stringent when we look at the DBMS to provide complete file management, reporting, and inquiry capabilities, all with a minimum of user interaction but with fast inquiry for many production requirements.

We invariably find that more user interaction is required with more direct control, the more so if we call for special report formats and complex file maintenance. This last reference remains valid as we increasingly aim to add new non-procedural capabilities to a system, providing for text and data manipulation and a wide range of functions.

—— THE DBMS CONTRIBUTION ——

Database management merges the user's need for more sophisticated data manipulation techniques with the technological capabilities of computers. Functionally, it includes all issues associated with retrieving, sorting, updating, and filing text and data; answering the demands of applications; running concurrently with the AP on the same computer; and contending for the same resources: memory space, CPU time, peripheral availability.

The logical approach to a DBMS stems from the fact that text and database functions are fairly repetitive and often require large resources out of proportion to that of the application program itself. A simple merge operation will call for a large amount of memory if it has to be done within a reasonable time.

A DBMS will make it easier to update and modify existing text and database. It will substantially reduce the space needed to store and process

the data, and many database management systems contain provisions for backup and recovery. Other benefits include a reduced processing time, a sorter program development time, ease in correcting or changing existing programs, increased integrity and security of the text and database, and ease in adding or deleting variables.

The use of a DBMS significantly improves the text and database management capabilities, compared to ordinary file-handling methods. It allows tailored AP to change to a different concept of file handling than the old system with multitudes of files, pointers, updates, and structures, replacing it through simple representation of the IE and the logical relationships.

DBMS software should insulate programs and IE from definitional changes in one another. Since program modules often cannot be permanently bound to particular information elements, commands make the IE name indirectly available to the program and to database management at execution time (Figure 1-2). A set of commands includes the invocation of the program and can establish the conditions for the program, such as the IE to which it is addressed.

This facilitates device independence and permits a broad range of services for a user AP by enabling a program without change to use various resources. It also assures ease of use for application development, a reasonable extensibility of customer configurations by being able to substitute resources, and a valid means of assuring minimal cost for user program maintenance. In short, it is a valid practice.

Let's repeat these references. The DBMS would:

1. Look after data integration.

The data is no more considered the private property of an applications program; it is regrouped as an integrated resource made available to different users (persons or programs) under system control.

2. Offer AP independence relative to the data.

Since the data exist as separate entities, called by the program and fetched by the DBMS, modifications to the AP do not alter their structure as long as certain standards are observed.

3. Offer personalization of the data view.

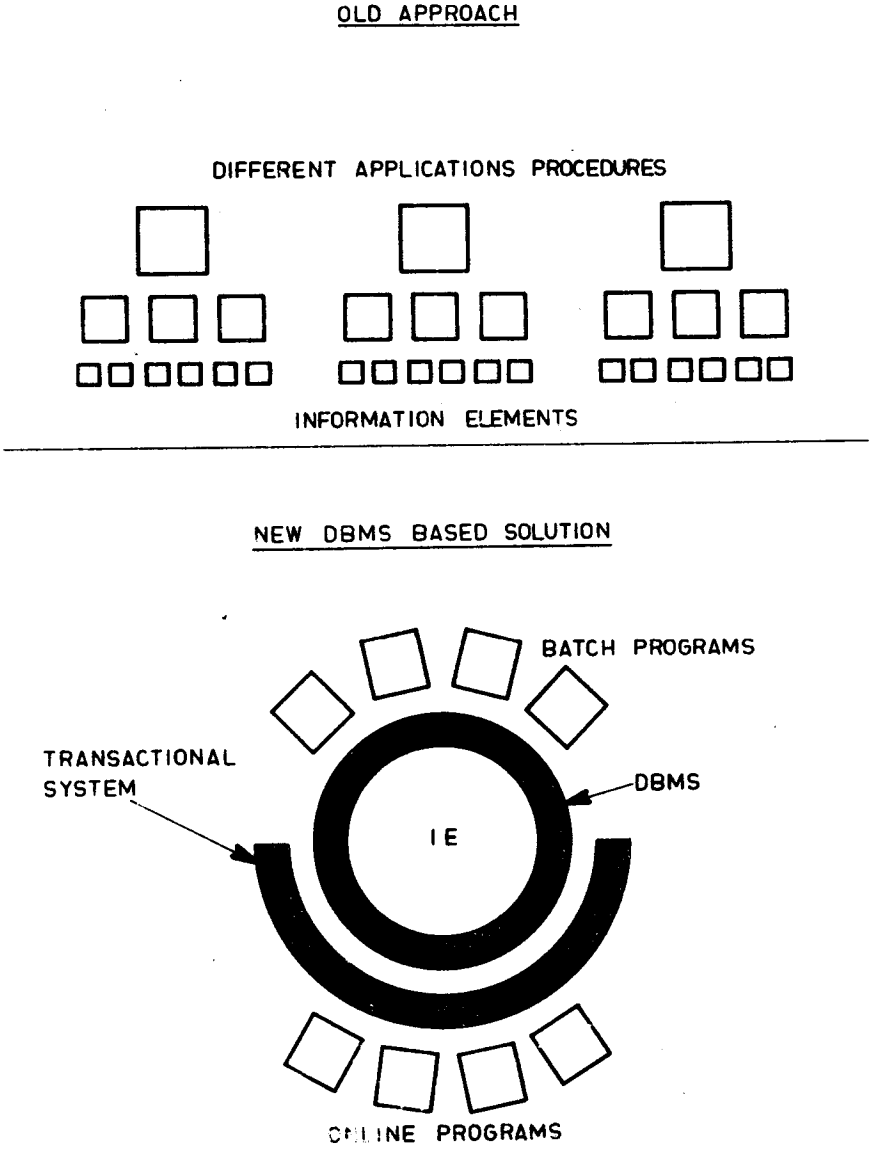
OLD APPROACH
DIFFERENT APPLICATIONS PROCEDURES
INFORMATION ELEMENTS
NEW DBMS BASED SOLUTION
BATCH PROGRAMS
TRANSACTIONAL SYSTEM
DBMS
I E
ONLINE PROGRAMS

FIGURE 1.2

Using formalisms, it is possible to structure data so that they can be seen by different programs under different descriptive characteristics, supporting a one-to-many correspondence, under DBMS control.

To assure such capabilities, the DBMS should support a number of functions including portability, storage control, and IE naming conventions. They should contribute to ease of use by providing for management of the constituent IE, thus allowing a user to control his data in a variety of ways. Naming convention and portability assistance can ensure that new applications may be easily installed or transported to a network's component system, while redundant facilities are avoided.

A DBMS is necessary to assure the right management of successive TDB (text and database) generations. We should always remember that we use the data many times for simulation, extrapolation, projection, and forecasting the future. For this job to be successful our data must become timely, accurate, precise, updatable, properly organized and kept in the TDB in a professional way.

Each one of the DBMS functions we have enumerated help to promote user transparency in regard to the mechanics of a computers and communications system. Other DBMS functions:

- Interpret physical storage requirements for a store/retrieve action.
- Assure security/protection needs.
- Give the necessary instructions for fetching data.
- Receive the requested text and data.
- If necessary, make conversions (format, code, etc.) to fit the specifications of the data request.
- Prepare the data for presentation to the AP.
- Present the data to the processing programs.

These are direct action references. In parallel, the DBMS has a host of indirect faculties to perform. They include the management of TDB resources; a number of ex-host functions (sort/merge, etc.); finer mechanics (add/delete); a host of file manipulatoin activities; integrity and authentication prerequisites; journaling and statistics; recovery; and supervisory control. Some DBMS support further functions such as query language capabilities.

Yet, though the use of a DBMS for handling text and data is rapidly growing, it has not always been supported by a precise understanding of

their functions and of those characteristics which deeply affect the traditional way of designing information systems. Among such relevant characteristics we distinguish the integration of large amounts of IE into an articulated TDB schema, and the capabilitiy of reorganizing the data description relevant to each application (logical data independence).

Thus, the preparatory work should definitely include the definition of the information system characteristics which make a database approach worthwhile; the knowledge, modelling and programming capabilities, offered by the particular DBMS which would be used; the technical, organizational and economic tradeoffs; and the understanding of how the TDB approach affects the analysis, design, and implementation cycle.

User transparency (person or program) has its cost. Among the costs of using DBMS we distinguish CPU time, central memory allocation, and the need for extra capacity to do housekeeping. Possible solutions to these constraints include the assignment of a separate processor as a rear-end (RE) text and database engine, and chip-level design for the handling of microfiles. This being stated, it is proper to add that the art of using a DBMS within a computers and communications environment is still in evolution. It will not be settled until:

1. The TDB organization is standardized, and
2. Further knowledge is acquired on what should be done through hardware and firmware, rather than software, in handling distributed resources.

That is why, at the current state of the art, and in order to gain experience, it is a good idea to use a DBMS—but after choosing one the user organization will be well advised to stay with it rather than spreading its human resources on different packages. The vital issue is knowhow; after this are the standards to be developed and supported. When the design premises are valid, there is a good chance that evolution in the DBMS structure will enhance the overall system without altering its manner of operation.

—— FROM OPERATING SYSTEMS TO DBMS ——

When we look at new developments in the management of text and databases, we must appreciate their historical perspectives. They are responsible more than anything else for available capabilities, supported functions, and possible drawbacks.