Techniques

Evolutionary Database Design and Development in Very Large Scale MIS

M.C. Filteau
The BDM Corporation, Dayton Regional Headquarters, 1900 Founders Drive, Kettering, OH 45420, USA

S.K. Kassicieh and R.S. Tripp
Anderson Graduate School of Management, University of New Mexico, Albuquerque, NM 87131, USA

Developing flexible databases that can accommodate the changes and enhancements necessary in development of large scale Management Information Systems (MIS) is crucial to the success of the system and ultimately to the survival of the organization. Since the MIS is constantly being modified and enhanced, it is necessary for the database to be designed to adapt to change as it occurs. Numerous articles point to the need for large scale systems decomposition into smaller subsystems for implementation purposes. They also require an incremental development strategy which coincides with the database evolutionary strategy.

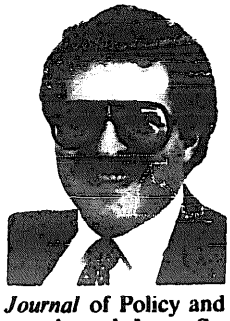
This paper describes a methodology for large scale database development that assists MIS managers in building databases for large scale systems. It describes how these concepts were used for implementation of the US Air Force's largest distributed processing MIS – "The Requirements Data Bank" (RDB).

Keywords: Large scale MIS systems, Incremental development approach, Database design, MIS prototype, MIS development methodology.

1. Introduction

There is widespread belief, as stated in the Management Information Systems (MIS) literature, that a rigorous framework is needed to ensure that sound technical design principles are introduced in the MIS development process. The traditional system development life-cycle (SDLC) approach is one framework which has evolved to meet these needs. With respect to the development of application software, the SDLC approach generally involves the following sequential steps: requirements specifications; preliminary design; detailed design; code development; testing and evaluation; and implementation. The development

Mark C. Filteau is Vice President of Information Systems and Technical Director fo the RDB Program for the BDM Corporation. Prior to the RDB Program, he managed development of information systems for NASA, the U.S. Naval Air Test Center, the U.S. Department of State, AT&T Technologies, Citibank, The World Bank, and the International Finance Corporation. The NASA Space Telescope Decision Support System Received a NASA Achievement Award and has been the subject of articles on decision support system design in *Federal Computer News*. He holds a Master of Science Degree from Florida State University and a Bachelor of Arts Degree from the University of Massachusetts.



Suleiman "Sol" K. Kassicieh is Associate Dean and Professor of Information Systems at the Robert O. Anderson Schools of Management at the University of New Mexico. He hoids a Ph.D. in Operations Research from the University of Iowa. His research interests are in the areas of artificial intelligence modeling, decision support systems and development. Some of his articles appear in *Operations Research, Information & Management, OMEGA, International Journal* of Policy and Information, Annals of Operations Research, and *Large Scale Systems* among others.

North-Holland
Information & Management 15 (1988) 203–212

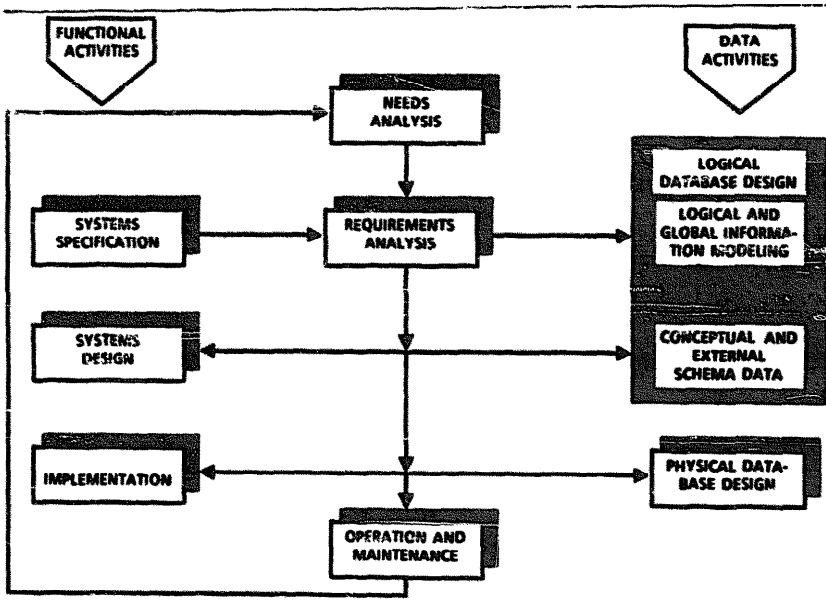


Fig. 1. Information Systems Life Cycle.

of the database for the application software follows a parallel path in this approach. These phases generally call for the development of a conceptual data model, during or immediately following the requirements definition phase, followed by development of a detailed conceptual model, logical design, and physical design of the database. The implementation of the physical design of the database should precede or coincide with the testing of the application software. *Figure 1* shows the relationships between software development activities and database development activities in the SDLC approach.

Robert S. Tripp is an Associate Professor of Management Information Systems at the Robert O. Anderson Graduate School of Management at the University of New Mexico. Before joining the University of New Mexico, he was the Program Director for the Air Force Requirements Data Bank (RDB) System Program Office (SPO). His responsibilities included the design, development, and implementation of the RDB System. He has also directed the successful development of two other large scale MIS developments: the AFLC Command, Control, Communications, and Intelligence (C{3}I) System, and the Weapon System Management Information System (WSMIS) at Headquarters Air Force Logistics Command (AFLC). He has published numerous articles which have appeared in *Information & Management*, Rand Corporation publications, *Cybernetica, International Journal of Physical Distribution, The Logistics and Transportation Review, Defense Management Journal, The Air Force Journal of Logistics*, and *Systems and Cybernetics*, among others. He holds a Ph.D. from the University of Minnesota.

Many authors, including Appleton [1986], Ahituv and Neumann [1984], King [1982], and Synnott and Gruber [1981], have outlined the shortcomings of using the sequential process, especially for the development of very large scale MIS. Many of these authors point out that it is not practical to attempt to establish detailed requirements specifications and detailed system designs before beginning the development of any portion of the system. Tripp and Filteau [1987] point out that these phases can take years to be completed on large scale systems, and that before they are completed environmental factors will generally change, requiring a change in the requirements and design of the system. To avoid some of these problems, Synder and Cox [1985], Juergens [1977], Wong [1984], and Snow [1984] advocate decomposing large scale systems into smaller increments for evolutionary design, development, and implementation. They indicate that this evolutionary approach reduces the risks associated with very large projects and can accommodate changes in requirements by adopting changes to the system as the increments are released to users.

While the adoption of this evolutionary development strategy does offer an opportunity to reduce the risks associated with the development of large scale systems significantly, it is extremely important that an evolutionary database development strategy be adopted to support large scale

MIS development. The strategy must facilitate the integration of the incremental or "local" databases, that are developed for each software increment, into a flexible and expanding "global" database accommodating changes and enhancements over an extended period of time. In other words, since the MIS principal subsystems will be constantly modified and enhanced under an evolutionary development strategy, it is necessary for the database to be designed to adapt to these changes. This paper describes an evolutionary database design methodology for large scale systems that builds a flexible and expanding integrated database as each software increment is added. It describes how this methodology was used in the implementation of the US Air Force's largest distributed processing MIS – The Requirements Data Bank (RDB).

2. An Evolutionary System Development Approach

Thus, an evolutionary product development strategy is needed for multi-million dollar, multi-year MIS developments. It is also absolutely necessary to develop and release products to users on a frequent schedule, because of the long development effort. Since people, conditions, and systems change, the development team must convince the users that their development goal is being met. Users should not have to wait for lengthy periods for some improved capability. Equally important, users and top corporate management need constant reassurance that investments are, in fact, paying off.

We propose a three step evolutionary design approach for the development of large scale MIS which has been implemented on two large scale multi-year developments resulting in successful user systems. Its major contribution is the control mechanism that affects the direction of product design and keeps large developments within reasonable cost boundaries. The three step approach is:

(*1*) A fairly complete top-down analysis of the functional requirements should be conducted to determine the strategic direction.
(*2*) The system is then broken into subsystems for detailed requirements specification, fairly rapid development, and implementation. The manner in which the decomposition takes place is extremely important: horizontal decoupling should separate the development into relatively independent subsystems with no substantial interaction. In practice this may not be possible, but attempts should be made to do so. These subsystems can then be developed separately, while planning for their eventual interfacing. This process must also be coupled with an evolutionary database design that answers the needs of evolving software applications. The greater the interaction among subsystems, the greater the importance of having an evolutionary database development strategy that

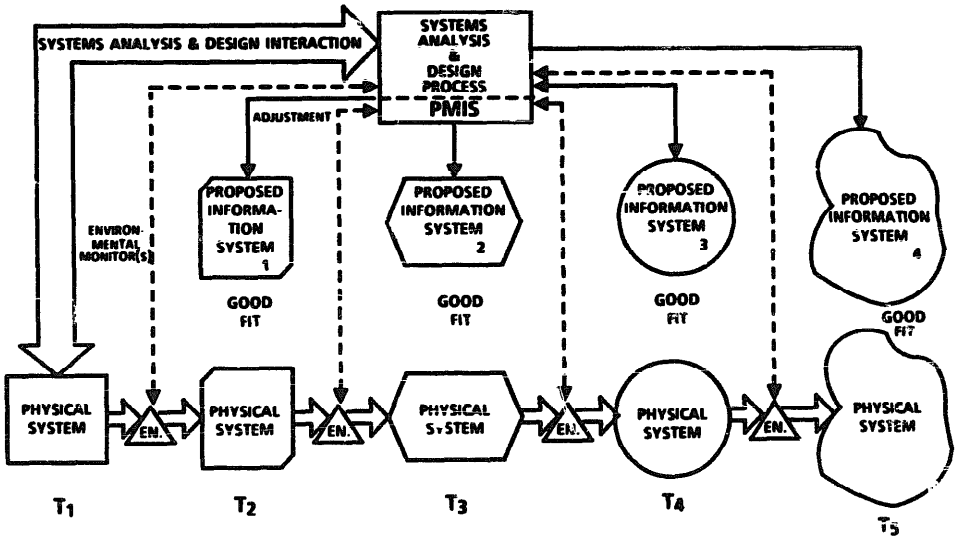


Fig. 2. A Project Management Information System for a Dynamic Analysis and Design Environment (after Snyder and Cox [13]).

allows integration of the database as increments are added. Vertical decoupling should also separate portions of the development within a functional area and allow a portion of the function to be developed and used while its expansion and enhancement is scheduled for later development. This reduces risk and allows for incremental implementation.

(*3*) Enhancements to the basic capabilities are tackled after the initial subsystems are in place. This aids in keeping users satisfied and allows the most important aspects of the system to be developed first. *Figure 2* shows a schema which supports our views.

3. An Evolutionary Database Design and Development Strategy

The database design and development strategy must be consistent with the application software development methodology used to design and develop the system. We now describe a database design strategy capable of coping with frequent changes and enhancements to the large scale system.

The suggested strategy draws heavily on the data modeling work of Chen [1976] and Batini and Lenzerini [1984]. Martin [1982, 1987], Howe [1985] and Inmon [1986] have popularized this work and coined the term "Information Engineering" (IE), which deals with data management, stability, information availability, usability, accuracy, user involvement, and user satisfaction. The ultimate goal of IE is to build a sound foundation for the design of an organizational data structure and associated application software. There are a number of techniques, methods, and rules that can be followed; these include identifying boundaries for the data, strategic data planning, enterprise modeling, and entity–relationship (E–R) modeling. The overall structure of IE includes development of the logical model, development of the physical model, and physical implementation. The steps are:

1: Develop the Logical Model
- a. Create an enterprise model.
- b. Define the data dictionary.
- c. Create/revise the local conceptual data model.
- d. Set up and update the E–R dictionary.

2: Extend the conceptual model to form the logical model

3: Develop the physical model

4: Physical implementation

These steps are minor modifications to those proposed in many articles of the 60s and 70s for a standard project life cycle model for database design that corresponds to the IS life cycle model. Many authors have also suggested the use of regular system analysis techniques for database design.

For large scale systems developments, these steps need to be modified to deal with their size and complexity. No evidence could be found to show whether IE concepts had ever been applied to very large systems developments. In addition, most functional experts in large bureaucratic organizations have limited knowledge of how their systems fit into the overall or integrated organizational effort. Thus, such systems need to be partitioned into segments with clearly identifiable boundaries. Each segment then develops a local information model and concentrates on that segment. Local models can then be combined to form an integrated global model for the entire development.

The evolutionary database design and development strategy involves the following:

Step 1: Develop the Logical Model

Phase 1: The Conceptual Modeling Process
1. Get top management commitment.
2. Break the database into organizational parts.
3. Assign systems specialists to a task force.
4. Define the data dictionary.
5. Update the E–R dictionary.
6. Validate the E–R diagram.
7. Review and evaluate the resulting model.
8. Integrate the local E–R model into the global information model (GIM).

Phase 2: Extend the conceptual model and forming the logical model:
1. Create logical model based upon the GIM.
2. Assign attributes.
3. Create third normal form reports.

Step 2: Develop the Physical Model

Step 3: Physical Implementation

Thus, the major difference between the IE approach and the large scale database development approach is in the conceptual modeling process,

where it enforces the decomposition and later integration methodology as opposed to the enterprise model of the IE approach.

4. Implementation of the Strategy

This database design and development strategy was implemented in a major Air Force development, the Requirements Data Bank (RDB), which is a multi-million dollar ten year development to modernize the Air Force Logistics Command (AFLC) Materiel Requirements Planning (MRP) Process. It involves managing the acquisition and repair requirements of approximately 900,000 spares, repair parts, and equipment items worth nearly $28 billion [USAF RDB Master Functional Description, 1986]. The cost of acquisition and maintenance of these items is approximately $15 billion annually [USAF RDB Economic Analysis, 1982].

In the 1970's, the materiel requirements process came under criticism as being inundated with problems and inefficiencies. The 22 data systems employed by AFLC to manage this enormous task were antiquated. AFLC realized that in order for the system to meet the logistics objectives and technological needs of the era, certain changes, modifications, and enhancements must be applied to the existing MRP process. Through systematic analysis and design, a major developmental program, known as the Requirements Data Bank (RDB), was established to develop and implement the needed improvements. The RDB involves an enormous development effort of nearly 3.7 million lines of code that will replace the primarily batch oriented 22 main data systems. There will be over 5,600 users using smart terminal interfaces on several geographically dispersed sites. This program will cost $300 million to develop and operate over the next ten years. The development began in 1985 and some portions of the system are currently operational.

Several factors were taken into account in building the database for the system. They included the number of data elements in the system, the number of independent MIS that were absorbed and replaced, and the cost of adoption. Modifications to the conventional design approach were made to account for the size and complexity of the development.

The RDB data base is partially complete and is being expanded to meet application software needs incrementally. The incremental design and development of the data base is performed in three steps. In step one, the focus is on developing enterprise models and associated conceptual data models. Step two involves refining the conceptual model and developing the physical models. In step three, the physical implementation activities take place.

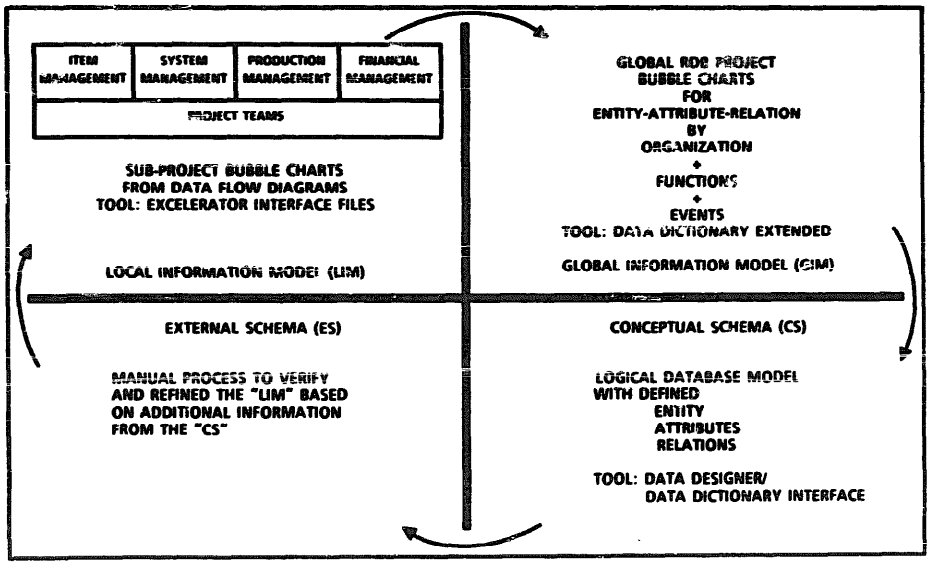


Fig. 3. Logical Data Design Process.

Because of the tremendous size and scope of the RDB, it was soon recognized that it was highly unlikely that a single global logical and physical model could be built. In addition, most Air Force functional experts did not know how their systems fit into the overall RDB effort; it was therefore decided to partition RDB into sixteen segments with clearly identifiable boundaries. Each segment would develop a local information model and concentrate on it. Each local model when completed was then combined to form an integrated global model for RDB, as shown in *Figure 3*.

The first step thus involves the development of the RDB logical models in two phases. The first phase consists of eight activities for developing a conceptual model. Phase two refines this model and develops a logical model, assigning data elements to their proper tables. The first phase started by getting top management commitment and setting up the organization. A cadre of top functional people began a seven-year planning effort leading to the selection of a professional services contractor to develop the RDB and making financial resources available for a ten year effort. Following contract award, in 1985, AFLC assigned its top systems people to a System Program Office (SPO) to manage the efforts of the contractor; AFLC also collocated the best functional specialists with the SPO and contractor for the life of the development. These specialists were assigned from headquarters but had authority to ask for aid in each area from the decentralized operational sites of AFLC whenever their expertise was needed.

Shortly after the award of the contract, AFLC created the Command Data Administration (CDA) office, which advocated the use of IE principles and provided training. Consequently, several SPO people received IE training and took an aggressive role in educating the user communities. The SPO and functional management were also supportive of the IE concepts, but several functional specialists had to be persuaded that it would benefit their projects.

Because development started before IE concepts were well understood, a formal enterprise model was never established for the RDB. It is, however, debatable whether an enterprise model could have been built and consensus developed for this size of activity. The intent and objectives of the traditional IE enterprise model were met by developing a top down concept of operations for RDB, jointly developed by the SPO and users. This functionally described the "new world" RDB was expected to achieve and how it would change operations. In addition, the RDB process functional descriptions (PFDs), that defined the details for all requirements had to show consistency with the concept of operations. The portfolio of completed PFDs included over 6500 pages describing more than 2200 processes to be automated. These PFDs took over two years to develop, involving the participation of hundreds of functional people from all sites. The PFDs not only spell out the functional requirements, but offer details of all these activities. To insure effective communications among the users, SPO, and development contractor, design teams composed of members from these groups held weekly walkthroughs of the PFDs to insure that incremental progress was being made towards completion.

The next step involved the definition of detailed PFDs that contain definitions of all data elements used within the PFD. All data elements were entered into the RDB data element dictionary extended (DEDE) as soon as the PFD was baselined. The DEDE is an Applied Data Research (ADR) Datacom/DB database [Applied Data Research Corporation, 1986]: this is used to capture all RDB metadata (data about data). An independent database was used instead of the ADR data dictionary to allow for future migration to another DBMS, if that should prove to be desirable. It should be pointed out that an automated interface with the AFLC Command or Corporate Dictionary/Directory (CD/D) was established during the RDB development. This interface was critical to the design of the RDB; it not only provided the source of several of the data elements but served as the clearing house for new definitions proposed by RDB. The CD/C interface insured that RDB would be able to "draw" data from other "on-line" developments and *vice versa*. In the incremental development approach, this interface with the CD/D is an on-going activity.As each increment is defined, the data elements are identified, baselined, and entered into the CD/D.

Entity–Relationship diagramming techniques are next used to capture the inherent structure of the data for each process that is entering the implementation phase. This shows the major groups (entities) of data and the relationships and

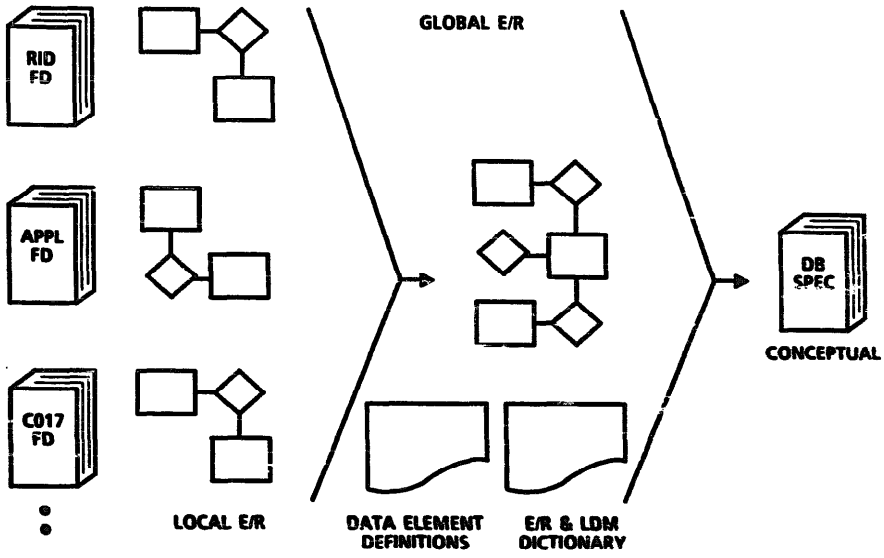


Fig. 4. Conceptual Data Modeling Phase.

associations between the groups. Excelerator [Index Technology Corporation, 1984] is used to expedite the creation of the E–R models, with each entity and relationship uniquely numbered. The E–R dictionary uses this unique number to provide a primary identified key, and a definition, including the type of data in the entity or relationship and the processes that use it. The design team validates the model checking: that there are no missing entities; that all associations between entities are valid; that all entities are true entities; and that the PFD processes correctly map to the E–R diagram. The E–R model is iteratively reviewed for changes until it is in a stable state; then it is integrated into the global information model (GIM). The GIM is a consolidation of all local models and forms the baseline for the incrementally expanding RDB integrated database design. *Figure 4* illustrates this process.

Phase 2 extends the conceptual model to form the logical model for each increment; these are created and attributes are assigned to each normalized table. Since the number of data elements is very large and elements are defined in different PFDs with different baselining schedules, this phase is accomplished concurrent with the devel-

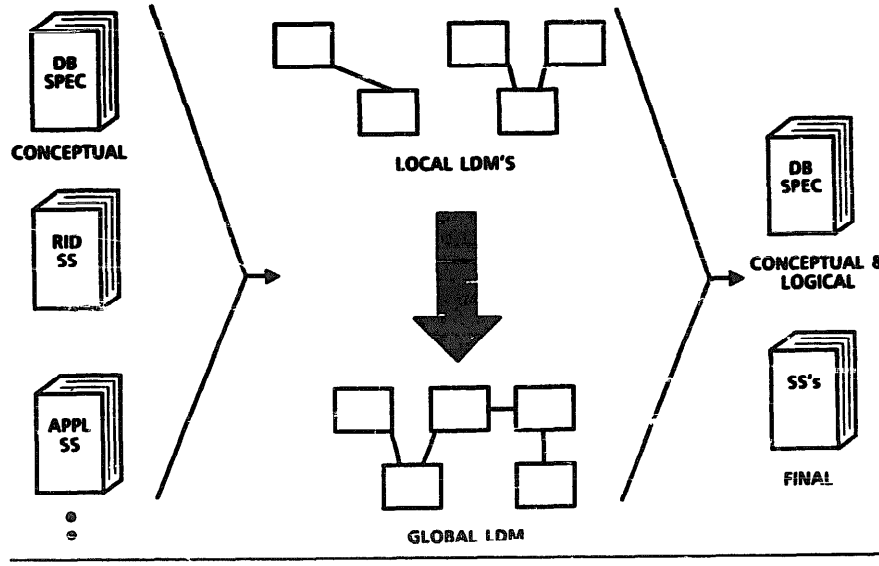


Fig. 5. Logical Data Modeling Phase.

LOCAL E/R NUMBER	GLOBAL E/R & LDM NUMBER	PHYSICAL TABLE NUMBER	GLOBAL E/R & LDM OCCURRENCES	GLOBAL E/R & LDM DASD LENGTH	GLOBAL E/R & LDM TOTAL BYTES	PHYSICAL TABLE OCCURRENCES	PHYSICAL TABLE DASD LENGTH	PHYSICAL TABLE TOTAL BYTES
1	1	30, 31, 32, 33	200	50	10,000			
2	2							
3			5	10	0			
4					0			
5			142	4,580	650,360			
6			1	9	9			
7					0			
9					0			
10					0			
11	11				0			
12			4,000	102	408,000			
13			350	7	2,450			
14					0			
15					0			
16					0			
17	17		400	41	16,400			
21			280,000	57	15,960,000			
22			800	16	12,800			
23	24		22,400	32	716,800			
25	25		11,200	28	313,600			
26			11,200	29	324,800			

Fig. 6. Database Engineering Traceability.

opment of each subsystem specification which are time phased throughout the development. The products from the logical design include: Bachman diagrams that represent the model; a set of normalized tables; and a cross-reference from the GIM to the logical model. *Figure 5* illustrates this process. To manage and ensure integrity of design, software interfaces are used between the various products. The specific steps followed in this phase are summarized next.

(*1*) Each of the entities and relationships are mapped to their appropriate logical table(s). Because there is not a perfect match between the GIM and each logical model, a traceability matrix was created to map deviations from the GIM. *Figure 6* shows an example of this.

(*2*) Following the rules of normalization, attributes are assigned to each table in the model. If any attributes do not map to a table, either: (1) an existing table in the logical model is updated or a new table is created; or (2) decisions are made whether new entities or relationships are required; or (3) updates to the traceability matrix are made as necessary. As an extra measure, to ensure integration, the normalized tables are entered into ADR Datadesigner to obtain a completely synthesized view of the model.

(*3*) To aid in the analysis step, the tables are reformatted through an interface with the Datadesigner product to create third normal form table reports.

The second major step in the RDB approach is to create physical models of the databases and application software. As in the conceptual modeling step, this is done incrementally as local information models are completed. The RDB data environment includes over 7000 unique elements and an intermix of two data distributions schemes. The first scheme partitions the data value to each of the distributed sites and thus ensures fast response times, immediate data currency, and basically puts the necessary data where it is needed. The second scheme replicates a minimum essential set of core data across all sites and thus allows site data visibility, quick response, and minimization of network overhead. A complete copy of all RDB databases is also held at two sites.

The reader can appreciate the complexity of the synchronization and communications of databases within RDB. It has been estimated that RDB will

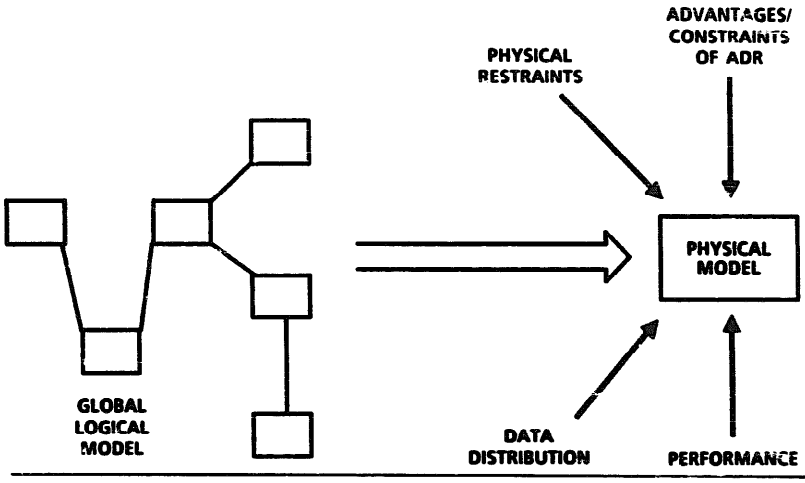


Fig. 7. Physical Data Base Modeling.

have over 60 billion bytes of data at four sites and 90 billion bytes at the two sites with complete copies of the entire database. It is estimated that the RDB will contain approximately 3.7 million lines of codes with 90 percent of that on-line and 10 percent only for batch. During this step, each process within the PFDs is assigned to a program module to automate the process. Data flow diagrams for each process are developed and related to the physical data model.

Physical implementation of the RDB software and database population is now taking place in an incremental fashion. Up to this point, the discussion has focused strictly on the data environment and has not taken into consideration any external factors. The physical model is the design phase where factors such as knowledge of the processes within the business, performance, and the special features or limitations of the DBMS come into play.

As an example, consider the following RDB example. Two of the RDB major processes, EOQ and REPARABLE spares, have 1000 common data elements that have been normalized and placed in the ITEM TABLE. The EOQ and REPARABLE items will never be processed together and most of the REPARABLE transactions will be processed only at one site. For efficiency reasons, the data elements can be separated and placed in separate physical tables. Thus, the logical model can and should serve as the guide for physical implementation, but deviations should be expected to occur, as indicated in *Figure 7*.

5. Summary: The Design and Development Experience

The importance of applying correct design methodologies cannot be overemphasized, especially in an extremely complex and large information system like RDB. Sometimes the size of a project makes it difficult to keep sight of the basic principles of systems development. The RDB is not such a case. Since it is committed to using proper data modeling and IE methodologies, the path of "where RDB is going and how to get there" can plainly be seen. A large part of the success will be directly attributable to realizing that the strategy incorporating IE techniques had to be used. This paper outlined the IE approach tailored to the RDB.

By proceeding with incremental implementation, the system keeps growing to meet user needs and shows progress against the total system goals. It is absolutely necessary to show progress continuously to retain top management and user involvement in the program. Without a sound IE approach, building a system as large and complex in an incremental fashion would have led to chaos. Without incremental releases, RDB would not have been politically viable: users and top management would not wait ten years for total system capability.

References

[1] Ahituv, Niv and Seev Neumann, "A Flexible Approach to Information System Development", *MIS Quarterly*, pp. 69–78, June 1984.

[2] Appleton, Daniel S., "Very Large Projects", *Datamation*, pp. 63–70, January 15, 1986.

[3] Applied Data Research Corporation, *"ADR DATACOM/DB and the ADR / DATACOM Environment Video (DB401)"*, Princeton N. J., 1986.

[4] Batini, C. and Lenzerini, M., "A Methodology For Data Schema Integration in the Entity Relationship Model," *IEEE Transactions on Software Engineering*, November 1984.

[5] Chen, Peter Pin-Shan, "The Entity-Relationship Model – Toward a Unified View of Data", *ACM Transactions on Database Systems*, Vol. 1, No. 1, March 1976, pp. 9–36.

[6] Howe, D.R., *Data Analysis for Database Design*, Edward Arnold, London, Great Britain, 1985.

[7] Index Technology Corporation, *Excelerator User Guide*, Cambridge, MA, 1984.

[8] Inmon, W.H., *Information Systems Architecture*, Prentice-Hall, Englewood Cliffs, New Jersey, 1986.

[9] Juergens, Hugh F., "Attributes of Information System Development", *MIS Quarterly*, pp. 31–41, June 1977.

[10] King, William R., "Alternative Designs in Information System Development", *MIS Quarterly*, pp. 31–42, December 1982.

[11] Martin, James, Strategic Data-Planning Methodologies, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[12] Martin, James, *Managing The Database Environment*, Prentice-Hall, Englewood Cliffs, New Jersey, 1987.

[13] Snow, Terry C., "Use of Software Engineering Practices at a Small MIS Shop", *IEEE Transactions on Software Engineering*, Vol. SE-10, No. 4, pp. 408–413.

[14] Snyder, Charles A. and James F. Cox, "A Dynamic Systems Development Life-Cycle Approach: A Project Management Information System", *Journal of Management Information Systems*, pp. 61–75, Vol. II, No. 1, Summer 1985.

[15] Synnott, William R. and William H. Gruber, *Information Resource Management: Opportunities and Strategies for the 1980s*, John Wiley and Sons, New York, N.Y. 1981.

[16] Tripp, Robert S. and Mark Filteau, "Blueprints: Adopting A Construction Trade Approach in Designing Large Scale Management Information Systems", *Information & Management*, pp. 55–70, 13 1987.

[17] US Air Force, *The Air Force Requirements Data Bank Master Functional Description (MFD)*, Revision B, BDM Corporation, December 12, 1986.

[18] US Air Force, *The Air Force Requirements Data Bank (RDB) Economic Analysis*, HQ AFLC/LO(RDB), Wright-Patterson AFB, Ohio, October 29, 1982.

[19] Wong, Carolyn, "A Successful Software Development", *IEEE Transactions on Software Engineering*, pp. 714–727, Vol. SE-10, No. 6, November 1984.