1984

A System Architecture for Temporally Oriented Data Management

Gad Ariav
New York University

Jim Clifford
New York University

A System Architecture for Temporally Oriented Data Management

Gad Ariav and Jim Clifford
Computer Applications and Information Systems
New York University

ABSTRACT

Attention to the temporal aspects of data management has intensified in recent years, focusing on data models and related systems that are sensitive to the ubiquitous temporal aspects of data. Both the growing need for easier access to historical data, as well as the imminent availability of mass storage devices, are making this a promising branch of database research, both practically and theoretically.

In this paper we summarize the main results of recent research on temporally sensitive data models, discuss the lessons learned in their development, and assess the prospects and difficulties involved in incorporating a temporal dimension into database management systems (TODBs). In particular, three system levels are identified: the external, user view of the database; an intermediate view closer to the structure of an existing data model; and an internal or implementation view defined in terms of low level data structures. This general architecture coherently incorporates a variety of related research results and development experiences, and serves as the framework for theoretical and implementation research into such systems.

Introduction

It seems not only natural but even somewhat tardy that in our never-ending quest to capture more semantics in formal information systems, we are beginning to augment our conceptual models with a temporal dimension. Indeed, there is growing research interest in the nature of time in computer-based information systems and the handling of temporal aspects of data. Roughly 50 references to the subject were identified and annotated by Bolour (1982), addressing four major topical areas:

1. Conceptual data modeling—an extension to the relational model to incorporate a built-in semantics for time (Clifford, 1983 (a)).
2. Design and implementation of historical databases—the organization of write-once, historical databases (Ariav, 1981), and implementation of temporally oriented medical databases (Wiederhold, 1975).
3. 'Dynamic databases'—the modeling of transition rules and temporal inferences from these rules (May, 1981).
4. AI related research—the temporal understanding of time-oriented data (Kahn, 1975).

The underlying premise of this expanding body of research is the recognition that time is not merely another dimension, or another data item tagged along with each tuple, but rather a more fundamental organizing aspect that human users treat in very special ways. The results of this research reinforce the perception that designing temporal features into information systems requires new and different conceptual tools.

A recent panel brought together many researchers in the field to discuss their work and identify promising research areas (Ariav, 1983 (a)). At the panel, four areas of research were indentified, and in this paper we focus on two of these issues, namely the implementation of temporal DBMS and the data models underlying them.

In most existing information systems, aspects of the data that refer to time are usually either neglected, treated only implicitly, or explicitly factored out (Tsichritzis, 1982). None of the three major data models incorporates a temporal dimension; users of systems based on these models who need temporal information must resort to patchwork solutions to circumvent the limitations of their systems. Furthermore, most information systems typically differentiate between present- and past-related questions in terms of data accessibility (e.g., online and offline storage, current database and log tapes.) It is important to note that this situation prevails *not* because of the scarcity of temporal references in common data, but rather in spite of their abundance. These current practices clearly simplify the task of data management, but result in reduced functionality and a less 'correct' or 'faithful' representation of reality.

Interest in the temporal aspects of data management is prompted by a number of developments. Prominent among them is the concept and growing usage of decision support systems (DDS)—the capacity to deal adequately with time and historical information is a core requirement in DSS (Ariav, 1983 (b)). Historical information is also an essential component in business planning in general, and especally in retrospective analysis (Ackoff, 1981). Applications as varied as medical tracking, personnel record systems, document control, reservations handling, and maintenance monitoring are all rich in temporal information; in fact, it is difficult to imagine operational application areas devoid of a temporal dimension.

Since the storage potentially needed for "complete histories" is doubtlessly huge, the practicality and feasibility of storing complete historical data is likewise contingent upon recent hardware developments that relax some of the constraints that have traditionally prevented the creation of similar information systems. In particular, optical storage technologies provide inexpensive access to a vast amount of data, in an access time comparable to that of current popular direct access storage devices (Copeland, 1982; Chi, 1982).

This paper explicates the theoretical basis for developing temporally oriented information systems and databases by focusing on temporally sensitive *conceptual* models and examining relevant *database* oriented data models. We then comment on the common issues raised by these inquiries into temporal models, and assess some of the difficulties encountered. A framework for integrating much of the above research in the area into a general foundation for the implementation of a temporally oriented DBMS is proposed and we conclude the paper with an examination of further research issues.

Time in Infological Models

Conceptual enterprise data models link phenomena of interest in the environment of the information system with the way these phenomena are structured and formally recorded within that system. In this section we focus on three temporally sensitive, conceptual data models that aim to capture the information requirements of the enterprise. The purpose of this analytic survey is to identify temporal issues on a conceptual level; data models that are more concerned with the structure and storage of data are discussed separately in the next section.

The earliest of the conceptual data models, the infological data model (Langefors, 1973; 1975) is based on the naural human perception of elementary facts (e-facts). Specifically, the concepts of an object, a property (or relation), and time are associated to form an "atomic" e-fact, which is assumed to be the "building block" of human knowledge. Recording such an e-fact in the information system results in an elementary message (e-message), which includes the *references* to an object, a property (attribute or relation), and a time point.

Basically, the infological time is an attribute that provides a built-in temporal context for every recorded fact in the information system. For instance, if O(p) is the set of all the objects that can potentially—regardless of time—have property p, then a "time slice," $O_t(p)$, is the set of all objects having property p at time t, which is fully contained in O(p).

The basic infological operation is the entry of a new e-message, of type insert, delete, or update. An e-message bears the information that an "elementary situation" prevails, either at a time point or during a time period. With this conceptual mechanism, which is supposed to bear a close resemblance to the human memory, the system can retain information about properties or relations that are no longer valid.

This enhanced capacity comes with a price: it complicates the nature of the (validity) constraints used in the infological model. For example, the convenient "closed world assumption" (Reiter, 1978) is complicated as, say, a person can be correctly recorded as married and single in the same instance of the database, though in reference to different time. In an example of a perfectly valid database response, the statement "X works for Y" could be false while at the same time "X work*ed* for Y" could be true.

Although time is identified as one of the three primary components of human knowledge, its role in the initial infological approach was limited to the mere indexing of recorded facts. This notion of 'infological' time was nevertheless refined in a later extension of this work (Bubenko, 1977).

For instance, a further distinction is made here between *extrinsic* and *intrinsic* times. Specifically:

1. Extrinsic time reflects the fact that every statement is passively embedded in a temporal context—whether or not this context is formally preserved. For example, the statement made in October 1983: "G, moves to, New York" has an extrinsic time-index "October 1983" associated with it.

2. Intrinsic time is part of the *content* of a statement. For instance, the statement made in January 1983: "G, moves to, New York, 9/82," includes the intrinsic time "9/82."

A much broader perspective of information modeling and time as a modeling construct was developed by Bubenko (1980). In this perspective a time model is specified *before* any other concept or data modeling construct is defined, reflecting the primary nature of the temporal contect. This time model focuses on hierarchical aspects of the structure of time and provides a formal calendar system, with a strict, unambiguous hierarchy of time periods.

Further extension of this concept recognized the need for existence criteria for objects. The approach is that objects exist between two specially designated events (e.g., hiring and firing of an employee). In general, events play a fundamental role in determining the behavior of objects, and they remain indefinately recorded in the conceptual model.

Like its predecessors, Bubenko's discussion fails to indicate how to reconcile infinite conceptual models with finite machines and finite memories. A later attempt, the temporal extenstion of the entity-relationship model (TERM) (Klopprogge, 1981), addressed this issue, and focuses on the definition of a *history-structure*—a data construct designed to capture special properties of time.

TERM concentrates on a set of modeling primitives based on the constructs of the entity-relationship model (Chen, 1976). The history structure thus augments the basic R-R constructs to create new, extended ones, (e.g., attribute-history, role-history). An entity-history is represented by the histories of the attributes that make up this type of entity (a composite history).

Time in TERM is captured within the regular E-R modeling constructs (i.e., under each basic fact (attribute or relationship) one can find its history). This is a different perspective from the earlier models mentioned above, where each point in time 'owned' the corresponding description of the "state of the world" at that time.

An entity-history automatically includes a 'latent' attribute of existence-description, indicating the periods of time during which the corresponding entity "exists." Setting the existence attribute of an entity to "off" simulates the effect of the ordinary deletion operation, but in the context of a constantly growing information system.

The basic operation in the model is, again, the addition of a state to a history structure: an *observer* can issue a database registration transaction either to initiate or complete a history. However, a designated privileged authority, the *referee*, has the power also to issue a correction transaction to erase or alter facts already in the information system. This introduces the (rather tricky) notion of *recording history*, thereby raising the issue of the DBMS having to manage two different time components. This view recognizes that the database *itself* is an object whose history is of interest; the DBMS should not only model the history of the changes to the objects in its domain, but should also track the development of the content of the database over time, preserving the sequence of changes in the database.

Another conceptual issue dealt with in TERM is the representation of an infinite history by a finite set of states. For that purpose the TERM language recognizes a (finite) set of *characteristic states* and a *derivation function* by which states that are not enunciated explicity are infered (e.g., linear interpolation, exponential smoothing, splines, or step-functions).

Time in Datalogical Models

In this section we examine the database oriented models that underlie the architecture developed later. These models fall into two general catagories. First are those in which a theorectical basis is developed, but no design is explicated (specifically (Bubenko, 1977) and (Clifford, 1982)). Second are those in which a theoretical basis is explicated and a design outlined, but no actual implementation attempted (specifically (Ben-Zvi, 1982) and (Ariav, 1983 (b)). In our examination of these works we will concentrate on major database-related findings that they explored.

Bubenko (1977) illustrates the problems in handling temporal data through an example—an inventory management system. The system keeps track of available quantities-on-hand, and the events affecting this information, for example, shipments and deliveries. Bubenko argues that time should be introduced through recording the quantities-on-hand at times of change. This method guarantees that the database includes a finite number of time point references. Moreover, it points out an essential conflict between the finite representation in the database, and the user's perception of a continuity of changes over time. Some resolution to this conflict is essential to any successful database model.

The underlying perception of time as fluid and progressing into the future illustrates the point that handling time requires special care, and cannot be added in an *ad hoc* manner to a traditional data model. For instance, the relation QUANTITY-ON-HAND (article, quantity, time) is not a relation in the conventional sense, since if we define its meaning to be 'quantity at time t' where t is a variable, its extension potentially includes an infinite number of tuples. However, assuming that there is an underlying finite representation of the database, time views could be defined using the tools of relational algebra or relational calculus.

The latter point is one of the major issues dealt with by Clifford (1982). This work provides the theoretical background for the incorporation of a temporal component into the relational model. It shows how the semantics of such an extended relational model can be mirrored by the semantics of a formal temporal logic.

In particular, Clifford identified three major principles that must be addressed in any practical implementation of an historical database system. The first of these was the notion of a "completed relation," which motivated the intuitive concept of the database as a collection of three-dimensional cubes of facts. The comprehension principle, a three-dimensional analog of the "closed-world" assumption, stated that for all practical purposes it must be assumed that the database has complete information about objects *and* the time period which it is modeling. Finally, the need for explicit continuity assumptions was identified. These functions are associated with time-varying attributes in the model, and define a mapping from a partial specificaion of their value (for example, from a sampling at a small number of points) to a complete specification of their value over some time interval. While implementation was not directly addressed in this work, these principles need to be addressed in any serious implementation effort.

Some research efforts have tried to form a theoretical basis for a design proposal. For instance, Ben-Zvi (1982) introduces the time relational model as a basis for a new architecture which incorporates comprehensive time processing capabilities into the relational model. In this model, a traditional relation is extended into a time-relation which contains the tuples' history, i.e., all the values that each tuple has acquired over time.

A time view operator, TV, is then defined as the operation that extracts from a time-relation, a regular (flat) relation which corresponds to the state of affairs at a specified point in time (as seen from any specified point in time). Every operation on data (e.g., SELECT, PROJECT, or JOIN) is preceded by the TV operator, and therefore retains its standard relational meaning. The model proposes a unified view of present and past data, and thus maintains time-independence and time transparency, i.e., the user may operate on this model without paying special attention to time.

Although no actual implementation is reported, some aspects of a design are discussed. For instance, a major issue is whether to add a time-relational front-end interface to an existing DBMS, or to design and build a time-relational DBMS from scratch. The proposed structure includes rather elaborate indexing to allow efficient retrieval of current and historical tuples. Interestingly, the proposal includes a facility to support future data by automatically replacing current data with future data at the appropriate time.

A clear limitation of this time-relational model is that it does not include any operators that act *directly* upon the time-relations, and therefore none of the problems associated with such operators are actually addressed. For example, retrieval of an explicitly temporal object, such as the history of an entity, is not possible in this model since the only temporal view supported is a slice at a single point in time. The definition of richer temporal views is nevertheless the major concern of Ariav (1983 (b)), and Clifford 1983(b), where the data models developed include the data cube as a basic data construct (rather than relying on the "traditional" relation), and a set of operations and contraints on such constructs are outlines.

Almost all the attempts to construct a temporal data model, or implement a system based on one, have related to a three dimensional representation of temporal relations, mainly as a mental image. The essence of the model developed by Ariav (1983 (b)) was to make this pervasive cubic view a primary and tangible data structure. The dimensions of the cube are objects, attributes, and time, each of which can be manipulated directly by the user—the temporal aspect of data is not factored out when the user accesses it. This ubiquitous *cubic* view of temporally oriented data serves as the external (users') view of historical data in the DBMS architecture.

The operations in this model are the basic relational ones i.e., selection (manipulating the object dimension of the cube), and projection (manipulating the attribute dimension)), albeit modified to operate on, and produce cubes. An attempt was made to preserve the original definition of this operation such that is applied to flat relational extensions (rather than cubic ones), results are compatible with standard relational operations. An additional operation introduced as part of the model is time-selection, which manipulates (e.g., restricts) the temporal dimension of the cube.

The model was then used as the basis for the implementation design of a temporally oriented database. In particular, it underlies the design of an extension of SQL to incorporate temporal elements in a way that does not penalize users who are not interested in accessing the historical data. (Recent work by Snodgrass (1984) reports on a similar extension to the query language QUEL.) The query language is further expressed in a set of navigational operations, and ultimately translated into a DBTG schema. The same model is then used in the design of an integrated graphic user interface that pictorially introduced the time dimension into query responses.

Both Ben-Zvi (1982) and Ariav (1983(b)) developed some limited theoretical basis for their database implementation design, but did not validate their architectural proposals through a corresponding implementation (or prototyping) effort. Nevertheless, we should point to the existence of *ad hoc* implementation efforts, undertaken without any explicit theoretical basis (Wiederhold, 1975; Ariav, 1981), and of implemented AI-oriented system (Findler, 1971; Kahn 1975).

Issues in Temporally Oriented Data Modeling and Management

In the preceding sections some effort is made to address the fundamental organizing concept of time. Even though they attack the common problem from widely differing angles, these temporally sensitive data models seem to suggest some recurrent issues and desired properties for historical database systems. These themes suggest immediately relevant guidelines to the design of corresponding information systems, and we outline them in this section.

First and foremost among the observations made while developing these models is that time aspects of data management are indeed universal, and their treatment can be generalized across applications and application areas. The latter finding legitimized the attempt to incorporate time handling features into generalized DBMS's. If limitation on processing and storage capabilities do not rule it out, then, the ideal system should be able to retain histories. Moreover, the system should be structured in such a way that users with *no* temporal data needs are not unduly penalized by this added system dimension.

Another major recurrent theme is the centrality of the recording of instantaneous changes in attribute values, the *event*. The notion of an event as a fundamental modeling concept enjoys a broad consensus. It is quite imperative that a temporally oriented information system incorporate this notion into its data model.

It seems that an ideal information system should be capable of relaxing the assumption of temporal correspondence between the states of the database and the modeled environment. This raises the need to deal with more than one dimension of time within the same system. Specifically, the information system should be cognizant of at least two dimensions: its own recording history, and the extrinsic time of the facts it contains. Other temporal dimensions are conceivable, and this issue bears much additional research.

There are apparent differences with regard to the place of time in the hierarchy of constructs, namely "above" or "underneath" other objects. For instance, while the infological approach embeds every fact in a time context (time precedes the objects), TERM adopts the view of object history. We feel that time in the model should ideally be *orthogonal* to the other constructs, and thus simultaneously support both views. This orthogonality is the premise for the cube, as discussed below.

There are also apparent differences with respect to the criteria or mechanisms for determining the existence of an object. We tend to opt for an application-dependent interpretation of events, rather than using a designated special universal attribute. For instance, the firing of an employee, which would typically turn off the corresponding existence condition, is not a termination since the employee may still exist in terms of retirement benefits, or be eligible for retroactive salary adjustment.

Another fundamental problem tackled by almost all the temporal models related to the apparent tension between the finite recording of facts in the database, and the provision of a dense view of the data along the time dimension. Specifically, a user can derive for each moment during the database history a corresponding view of the database as it existed at that time. This gives rise to the notion of completion of relations or the specification of functions for temporal interpolation of values (Clifford, 1983(b)).

Last, but certainly not least, is the issue of completeness of temporally oriented data models. Without some generally accepted objective quality criteria, there is no real way to compare models and guarantee appropriateness. Current efforts rely heavily on established relational completeness as a basis, with some *ad hoc* extension. This indeed seems to be a safe departure point, but much is still to be understood about temporal modeling before a standard can be instituted.

An Architecture for Temporally Oriented DBMS

In general, the study of time in database systems has so far done very little to *integrate* the results of various efforts; isolated treatments, theoretical or implementation oriented, are the norm. Unlike implementation attempts of the past, this proposed framework emerges out of conceptual considerations to incorporate data modeling views, and it therefore provides sounder guidelines for implementation and a meaningful basis for evaluation.

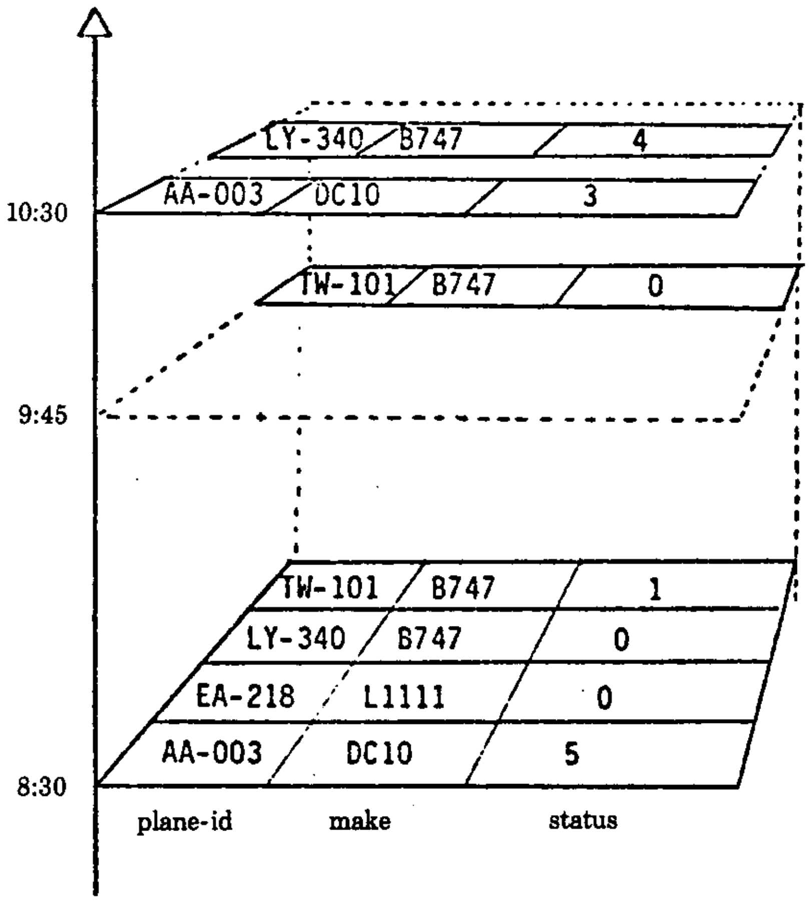

Figure 1

The schematic structure of a TODB system

The proposed approach is based on a three layered structure (see Figure 1), where the complexity of dealing with time is resolved by adding an external shell around the layers of traditional DBMS. Specifically, this outermost layer provides users of the system with an inherently temporal representation of data. The two inner layers deal respectively with the formulation of database queries to satisfy the external representations, and mapping to the internal representation (i.e., elementary data structures, files and access methods).

The external data model that underlies this TODB is based on the observation that the temporal lifespan of the objects described in a database can be represented as a three dimensional object (a cube), utilizing this pervasive spatial metaphor for time. The middle layer is a decomposition of the database cubes into traditional, flat relations. Finally, the innermost layer of the structure partitions these flat relations into static (non-time-varying) and dynamic (time-varying) components. The time-varying component consists of all of the attributes whose values are organized by the system along a temporal dimension. A simple scenario is depicted in Figure 2.

Deciding which time(s) will be used as the temporal dimension(s) of the cube (e.g., event time or recording time) is, of course, a fundamental issue addressed elsewhere (Ariav, 1983(b)). Given the centrality of three-dimensional objects in temporal models, we choose to focus here on the properties and mechanics of such data constructs, and defer the other question for future research. The jump from two to three dimensions is itself complicated, and while the formalism of an n-dimensional theory is probably not much more involved, the intuition for language constructs and metaphors at the user interface are not immediately forthcoming.

Existing "time varying" or "static" databases provide only a cut through an object's temporal path at each point, while the temporally oriented (or historical) database captures its temporal path in its entirety. The latter type of database, therefore, can be viewed as a collection of data spaces or cubes of data. This three-dimensional

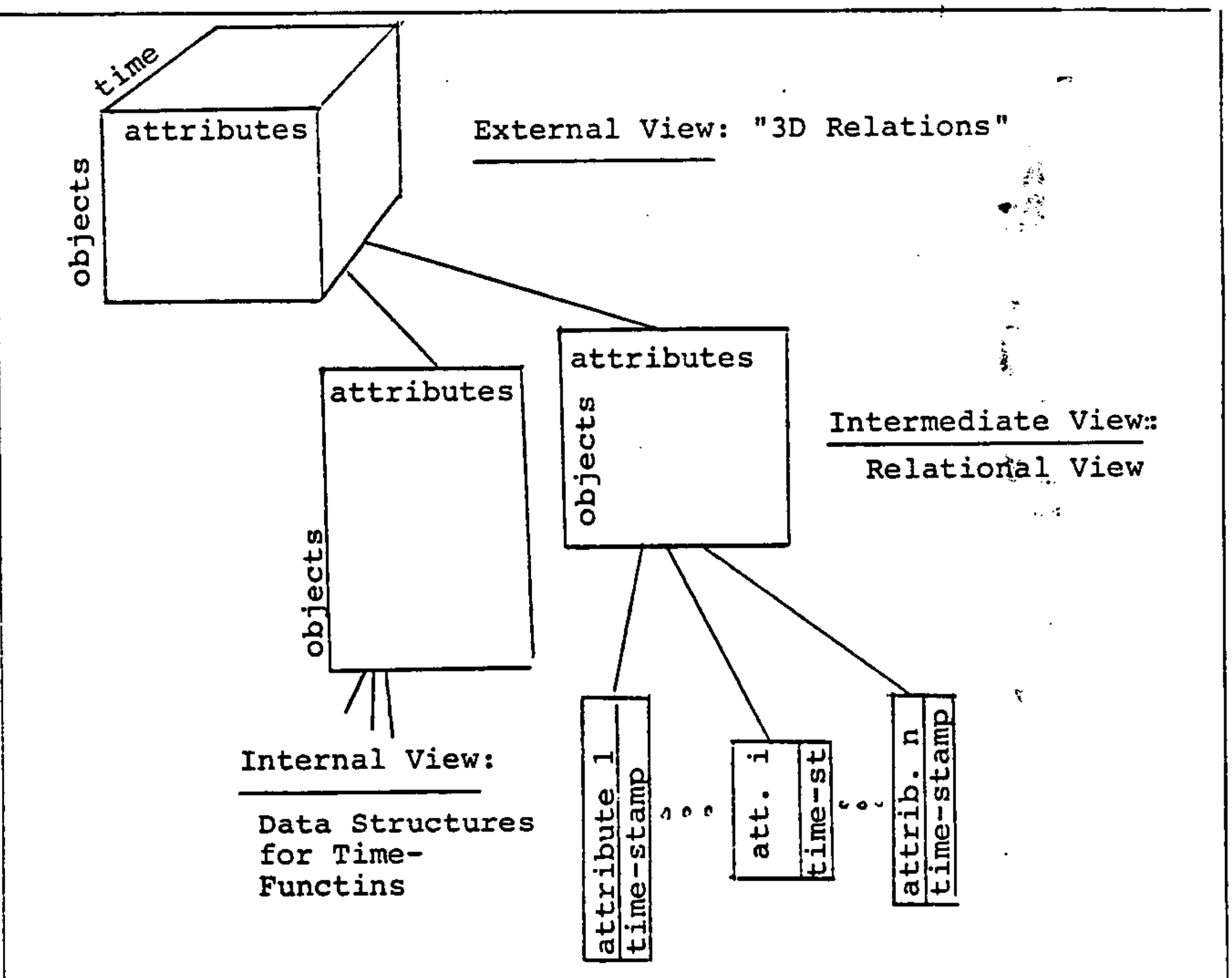

Figure 2
The temporal path of the AIRPLANE database

view of historical data has some psychological support (Aaronson, 1972), and is so pervasive in the related database literature (Wiederhold, 1975; Clifford, 1982; Ben-Zvi, 1982; Ariav, 1983(b)) that we believe it to be appropriate for the external users' view of the database.

The state of the relation at any point in time is determined by slicing the cube horizontally at a depth corresponding to the specified time, and observing the values of data that prevail for each of the objects represented in the relation.

What can a user do with the whole additional dimension that the historical database provides? At the elementary level, a user should be able to move forward and backward in time to view the database at various points along the time dimension which defines the cube. But merely enabling the user to zero in along the time dimension is not sufficient. More elaborate operations should allow the user to query the database and form *views* (i.e., new cubes extracted and rearranged from old ones) by operating on any combination of dimensions. In other words, users should be able to select attributes, and/or objects and/or periods of time. Even more advanced operations should allow the user to join views over possibly different points in time and thereby create new cubes, and to utilize the ordering relation on time ("earlier than") to form complex time-related queries.

The external shell of the system is therefore meant to allow its users to retain that comfortable spatial metaphor for temporal relationships while browsing through the database. Users' access to the data is through a set of specification operations which refer explicitly to the dimensions of the cube (i.e., specification of attribute, object, and time context), and are in a form compatible with common relational query syntax.

Any relational view of an historical database such as we have outlined above must involve some mapping from the three-dimensional view seen by the users to some internal view based upon the two-dimensional relations of the underlying database model. Because there are many ways to encode three-dimensional structures into two, it is not immediately obvious how best to define the mapping from the internal to the external view of these objects. The guide to choosing such a mapping should naturally be based upon a consideration of how best to provide users with a consistent, coherent, and flexible view of their temporally oriented data, and at the same time a view which can be implemented relatively efficiently. We have examined a number of possible mappings, and the one we deemed most reasonable is outlined below. For illustrative purposes we consider the single historical relation in Figure 2.

This approach considers attributes (as opposed to entire relations) to be the level at which control over the temporal dimension is exercised. In this view, an attribute such as STATUS has as its domain not readiness levels (such as 1, 4, or 5), but rather *functions* from moments in time to readiness levels. The STATUS of AA-003, for example, is not 3 in this view, but rather the function:

```
 8:30—> 5
10:30—> 3
11:30—> 1
```

Some representation of these functions, then, is what must comprise the internal view of the historical database. In other words, in this approach the three-dimensional relation is decomposed, not into individual *relations* indexed by time, but rather into individual *attributes* indexed by time. For greater flexibility, in fact, we allow some attributes to be simple, or non-time varying (for example, the attribute GENDER might be considered to fall into this category.) Now functions being themselves nothing more than relations constrained by a functional dependency (STATE—> value), each time-varying attribute can therefore be represented as a separate binary relation with STATE as the key and its domain as the other, non-key attribute. All that remains is to provide a mechanism for "linking" objects with the relations that represent their time-varying attributes. One simple mechanism for providing this linkage is depicted in Figure 3.

We found this internal view, which includes time as a property of an attribute rather than of a tuple, or a relation, or even a database, provides the most flexibility and allows for a fuller treatment of time. Clifford (1983(b)) presents a formal definition of an historical relational algebra based upon this view, and discusses the kinds of temporal properties that can be associated with each attribute, including (1) its own time domain, (2) an interpolation function to determine the value of the attribute at non-stored moments in time, and (3) its own periodicity or allowed rate of change.

Conclusion and Further Research Issues

The above integrated framework views the implementation of databases that address the concept of time, as a system composed of layers of abstraction; from the outermost, user view of a three-dimensional database, to the internal level of implemented time-series functions. What we have presented here has mainly been an exploration of both the conceptual modeling tools best suited for providing users with an historical view of their data, and the appropriate design strategies for effectively implementing such a view. There are naturally a number of significant research issues that need to be addressed.

We have focused in this paper on what are, conceptually, the upper levels of the database. These still remain to be implemented in terms of data structures and physical storage. Ariav (1984) has begun an effort to build a system based on this architectural framework and to use it as a vehicle to explore a variety of issues, including storage strategies and associated algorithms, performance issues, and user interfaces. Ways to exploit properties of new technologies like optical mass storage will also be further explored.

As far as actual implementation is concerned, perhaps the most overriding consideration is whether to start from scratch and build a complete DBMS tailor-made to provide an historical data perspective, or to build an historical front-end to some existing DBMS. We are exploring the feasibility of both approaches, and expect to build at least a prototype system in each of these modes. We nevertheless feel that the framework we have presented above outlines the overall architecture that an implemented system should eventually have.

Although the three-dimensional, cubic view appears to be a natural choice for the user interface to historical databases, there is no real experimental evidence in the domain of information systems to support this claim. Further research is necessary to explore and contrast this with other views that might create more appropriate user metaphors and provide effective communication between the user's and the system's views of the temporal information.

Another interesting area for further study is the problem of query optimization in the context of historical databases. How can we make use of the new structures of the historical relational model, its new operators, and the additional information carried by each attribute to perform more intelligent query analysis and transformations?

Our implementation framework also partitions the overall problem into distinct and complementary problems

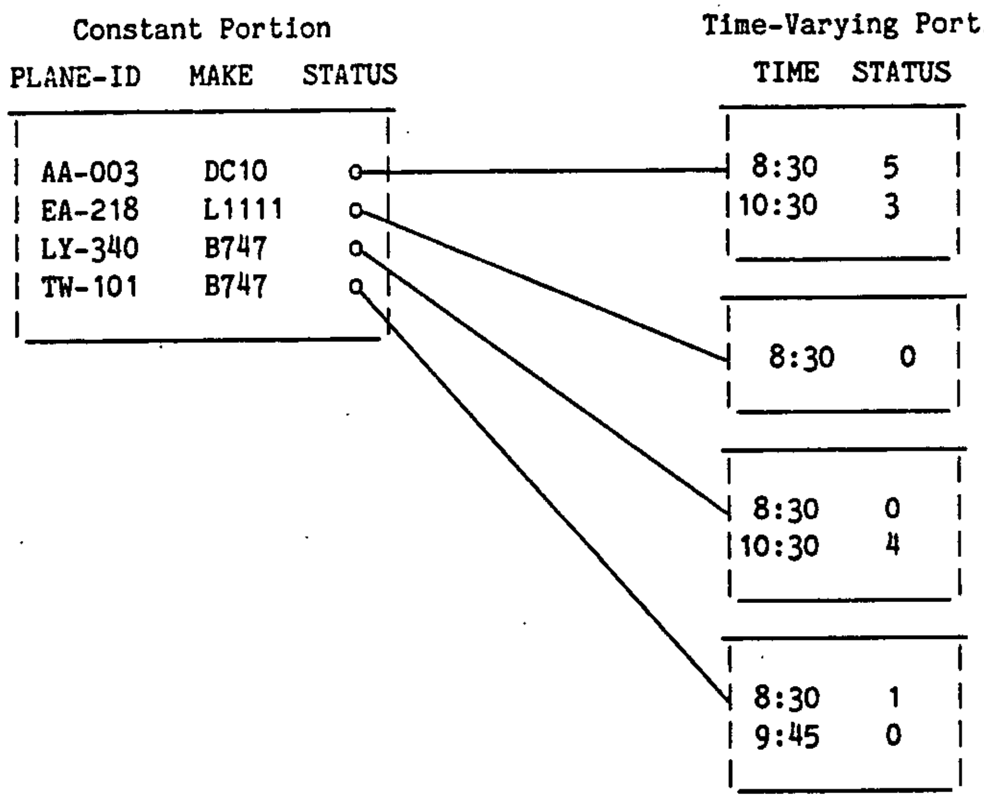

Figure 3

The underlying data structures

of a more manageable scale. Regardless of the implementation approach taken, the mappings between the layers of abstraction (e.g., from three-dimensional cubes to ordinary relations, and from the latter to actual data structures) must be appropriately defined, and algorithms developed that implement them efficiently. A concurrent effort is underway to augment the traditional relational algebra with enough temporal operators to provide a natural definition of the notion of temporal completeness with respect to a query language. The formal definitions of these operators must then be translated into correct algorithms as a basis for implementation.

Another problematic research issue within the context of time and databases is the impact of time in the evolution of the *structure* of the database itself. The schemata of databases change as the information processing needs of an enterprise change, and different perspectives become appropriate. This raises the issue of database restructuring (as dealt with by Sockut (1979), and Navathe (1980) and the need to capture such changes and coordinate them with changes in the factual content of the database. The temporally oriented model carries the potential of responding to the documented need "... to manage the data in a manner which is insensitive to such changes" (Kent, 1978, pp. 26-27). The problems involved in temporally oriented data models and their implementation in an actual DBMS are indeed difficult to solve, and our understanding of them as a research community is still preliminary. Nevertheless, the rewards of such a system will be correspondingly substantial. Time affects almost every facet of database practice and theory, and the explicit recognition of this component in a temporally oriented DBMS introduces new and interesting opportunities in the domains of database functionality, usability, recovery, and concurrency control.

REFERENCES

Aaronson, B.S. "Time, Time Stance, and Existence," in *The Study of Time*, Fraser *et al.* (eds.), Springer Verlag, Berlin, W. Germany, 1972, pp. 293-311.

Ackoff, R.L. *Creating The Corporate Future*, John Wiley and Sons, New York, New York, 1981.

Ariav, G., and Morgan, H.L. *MDM: Handling the Time Dimension in Generalized DBMS.* Technical Report DS-WP 81-05-06, Decision Sciences Department, University of Pennsylvania, May, 1981.

Ariav, G., Clifford, J., and Jarke, M. "Time and Databases," in *ACM-SIGMOD International Conference on Management of Data*, May, 1983, pp. 243-245.

Ariav, G. *Preserving the Time Dimension in Information Systems.* Technical Report DS-WP 83-12-06, Decision Sciences Department, University of Pennsylvania, December, 1983.

Ariav, G., Clifford, J., and Shiftan, J. *A Framework for Implementing Temporally Oriented Databases.* Technical Report, Department of Computer Applications and Information Systems, New York University February, 1984.

Ben-Zvi, J. *The Time Relational Model,* PhD Thesis, Department of Computer Science, University of California, Los Angeles, 1982.

Bolour, A., Anderson, T.L., Deketser, L.L., and Wong, H.K.T. "The Role of Time in Information Processing: A Survey," *ACM SIGMOD Record* (Spring) 1982, pp. 28-48.

Bebenko, J.A. "The Temporal Dimension in Information Modeling," in *Architecture and Models in Data Base Management Systems,* G.M. Nijssen (ed.) North Holland Publishing Company, Amsterdam, The Netherlands, 1977, pp. 93-118.

Bubenko, J.A. "Information Modeling in the Context of System Development," in *Information Processing 80,* S.H. Lavington (ed.), North-Holland/IFIP, 1980, pp. 395-411.

Chen, P.P.S. "The Entity-Relationship Model: Towards a Unified View of Data," *ACM Transactions on Database Systems* Volume 1, Number 1, March 1976, pp. 9-36.

Chi, C.S. "Advances in Computer Mass Storage Technology," *Computer* Volume 15, Number 5, May 1982, pp. 60-74.

Clifford, J. *A Logical Framework for the Temporal Semantics and Natural-Language Querying of Historical Databases,* PhD Thesis, Department of Computer Science, SUNY at Stony Brook, December, 1982.

Clifford, J., and Warren D.S. "Formal Semantics for Time in Databases," *ACM Transactions on Database Systems,* Volume 6, Number 2, June 1983.

Clifford, J. *Towards an Algebra of Historical Relational Databases,* Technical Report, Computer Applications and Information Systems, New York University, December, 1983.

Copeland, G. "What if Mass Storage Were Free?" *Computer,* Volume 15, Number 7, July 1982, pp. 27-35.

Findler, N. and Chen, D. "On The Problem of Time Retrievel, Temporal Relations, Causality, and Coexistance," in *Proceedings of the Second International Joint Conference on Artificial Intelligence,* Imperial College, September, 1971.

Kahn, K.M. *Mechanization of Temporal Knowledge,* Technical Report MIT-MAC- TR-155, Massachusetts Institute of Technology, Cambridge, Massachusetts, April, 1975.

Kent, W. *Data and Reality,* North Holland Publishing Co., Amsterdam, The Netherlands, 1978.

Klopprogge, M.R. "Term: An Approach to Include the Time Dimension in the Entity-Relationship Model," in *Entity-Relationship Approach to Information Modeling and Analysis,* P.P.S. Chen (ed.), ER Institute, 1981, pp. 477-512.

Langefors B. *Theoretical Analysis of Information Systems,* Student Litterature and Auerbach, Lund, Sweden, 1973.

Langefors B., and Sundgren B. *Information Systems Architecture,* Petrocelli/Charter, New York, New York, 1975.

Mays, E., Lanka, S., Joshi, A.K., and Weber, B.L. "Natural Language Interaction with Dynamic Knowledge Bases: Monitoring as Responses," in *Proceedings of the Seventh International Joint Conference on Artificial Intelligence,* Vancouver, British Columbia, Canada, August, 1981, pp. 134-149.

Navathe, S.B. "Schema Analysis for Database Restructuring," *ACM Transaction on Database Systems,* Volume 5, Number 2, June, 1980, pp. 157-184.

Reiter, R. "On Closed World Databases," in *Logic and Databases,* H. Gallaire, and J. Minker, (eds.), Plenum Press, New York, New York, 1978, pp. 55-76.

Snodgrass, R. "The Temporal Query Language TQuel," in *Proceedings of the 3rd ACM SIGMOD Symposium on Principles of Database Systems,* Waterloo, Ontario, Canada, April, 1984.

Sockut, G.H., and Goldberg, R. "Database Reorganization—Principles and Practice," *ACM Computing Surveys,* Volume 11, Number 4, December, 1979, pp. 371-396.

Tsichritzis, D.C., and Lochovsky, F.H. *Data Models.* Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1982.

Wiederhold, G., Fries, J.F., and Weyl, S. "Structured Organization of Clinical Databases," in *Proceedings of the NCC,* AFIPS Press, Montvale, New Jersey, 1975, pp. 479-485.