ABSTRACT
This paper describes a new architecture (based on the relational data model) for the handling of time-varying data which is well suited for medical applications. This model also provides a heterogeneous distributed capability such that current and historical data may be stored on a network of different computing facilities. We have prototyped such a temporal data management system under the UNIX operating system using the C programming language which interfaces with the commercial relational database management system - Ingres.

INTRODUCTION
Time is increasingly being perceived as an important aspect in medical applications and yet is not fully supported in currently existing database or file-handling systems. Most existing database systems maintain only data or information about up-to-date current-views of the real world. Old data in database systems is normally overwritten by a new incoming data. Hence information about the past is lost. Generally there are two main methods of capturing the historical data: either by adding additional time field(s) to the database schema or by using a temporal system. The former method provides simple ad hoc time-varying values to be associated together these can be handled by the existing database systems. However, one of the serious limitations of this approach is that temporal (or historical) data may not be manipulated automatically by these systems; often end-users have to interpret the temporal results themselves. The latter method attempts to automate the manipulation of the temporal data by the system and hence improve the query processing capabilities. This paper addresses the latter approach. (Readers who are interested in the taxonomy and bibliography of temporal/historical databases can refer to [1,2] and [3] respectively).

K.-P. Adlassnig et al. (eds.), Medical Informatics Europe 1991
© Springer-Verlag Berlin Heidelberg 1991
We describe a new architecture (based on the relational data model) particularly suited for handling of
time-varying data in medical domains. The proposed architecture for the temporal model overcomes
several weaknesses of currently developed historical models (more detail can be found in [4]). This model
also provides a heterogeneous distributed capability such that current and historical data may be stored in
a network of different computing facilities.

One of the main features, apart from automated processing/archiving temporal queries/data, of our model
is the ability to distinguish between the corrective and progressive updates. The progressive update
implies that a more recent value supercedes its older value whereas the corrective update indicates that a
value can be either incorrectly entered into the system or an external domain related error. Consider a
patient attending a clinic who was diagnosed as having an eczema and treated with steroids. A few days
later his condition was worse. Upon careful examination it was confirmed that he should have been
diagnosed as having tinea, and be treated with antifungal agents all the time. Most temporal systems (e.g.
[5,6,7,8]), which provide the progressive update mechanism only, would fail to record such common
occurrences in the medical realm correctly. We argued that semantically it is important to distinguish
between the exact status of updating data values (see next section).

OVERVIEW OF SYSTEM CONCEPT

Our temporal model is based on the relational approach and is similar to the multi-database approach for
distributed databases [9]. The main difference between our temporal system and existing DataBase
Management Systems (DBMSs) is that time related attributes can be stored, processed and manipulated
automatically by the temporal system itself. Time related (or varying) attributes are represented in the same
way as the other allowable data types. However each time related attribute value is associated with a three-
field element: status-kind, logical-time (time when an event occurs), and physical-time (time when an
event is processed by the computing facility). The status-kind field provides information which depicts the
reason(s) for associating with its value. Once a value is stored in the computing device(s) it can not be
erased. Therefore we use the surrogate key (a key generated by the system) concept for ensuring the
uniqueness of tuples in the temporal database.

Consider the patient-treatment scenario again. (For simplicity, let us ignore the physical time and consider
only the logical time with no intervals) The first entry for the patient, say, Mr. Brown, may look like: