Modeling Temporal Consistency in Data Warehouses

Robert M. Bruckner[1], Beate List[1], Josef Schiefer[2], A M. Tjoa[1]

[1]Institute of Software Technology
Vienna University of Technology
Favoritenstr. 9-11 /188, A-1040 Vienna, Austria
{bruckner, list, tjoa}@ifs.tuwien.ac.at

[2]IBM Watson Research Center
30 Saw Mill River Rd.
Hawthorne, NY 10532, USA
josef.schiefer@us.ibm.com

Abstract

Real-world changes are generally discovered delayed by computer systems. The typical update patterns for traditional data warehouses on an overnight or even weekly basis enlarge this propagation delay until the information is available to knowledge workers.

The main contribution of the paper is the identification of two different temporal characterizations of the information appearing in a data warehouse: one is the classical description of the time instant when a given fact occurred, the other represents the instant when the information has been entered into the system. We present an approach for modeling conceptual time consistency problems and introduce a data model that deals with timely delays and supports knowledge workers to determine what the situation was in the past, knowing only the information available at a given instant of time.

1 Introduction

The observation of real-world events by computer systems is characterized by a delay. In the applied context of information systems it is determined by external and internal factors (e.g. update patterns, processing speed). This so-called *propagation delay* is the time interval it takes for a monitoring system to realize an occurred state change. In contrast to operational systems (designed to meet well-specified response time requirements), the focus of data warehouses (DWHs) [6] is generally the strategic analysis of data integrated from heterogeneous systems. Late-arriving data that should have been loaded into the DWH weeks or months before complicate this situation [10]. Hence, keeping data current and consistent in that context is not an easy task.

Until recently, *timeliness* requirements [4] (describing the relative availability of data to support a given process within the timetable required to perform the process) were restricted to mid-term or long-term. W. H. Inmon, known as the founder of data warehousing, cites time variance [6] as one of four silent characteristics of a DWH. Timeliness can be viewed as an aspect of data quality, which has a strong influence on the delay until a system realizes a certain state of the data. While a complete, real-time enterprise DWH might still be the panacea; there are some approaches to enable DWHs to react "just-in-time" [16] to changing customer needs and financial concerns.

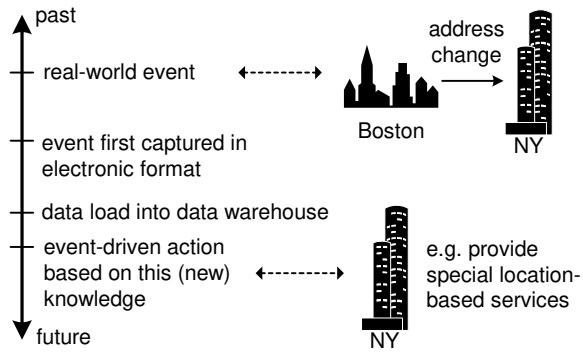

Figure 1. Delayed discovery of real-world changes.

Figure 1 demonstrates that typical update patterns for traditional DWHs on a weekly or even monthly basis enlarge propagation delays until the information is available to knowledge workers. Any significant delay in the recognition of events may result in a number of further considerations needing to be taken into account:

- **Data integration**. Aggregates have to be updated, because the new records will change counts and totals of the prior history.
- **Analytical processing**. Historical analysis results can no longer be repeated, if additional information regarding that time period is integrated delayed (numbers and summaries will change unexpectedly from the user's perspective).

We present a schema model that copes with delays and enables a timely consistent representation of information. This enhances analytical processing by considering that information validity of data is typically restricted to time periods, because of frequent updates or late-arriving data.

The remainder of this paper is organized as follows. Section 2 considers research issues and related work.

Sections 3 and 4 classify common temporal data structures in information systems and introduce the concept of temporal consistency described from different viewpoints (conceptual and logical). Section 5 evaluates the model and finally we give a conclusion.

2 Research issue and related work

The notion of time is fundamental to our existence and an important aspect of real-world phenomena. We can reflect on past events and on possible future events, and thus reason about events in the domain of time. In many models, time is an independent variable that determines the sequence of *states* of a system. There are two different temporal characterizations of the information appearing in a DWH: one is the classical description of the time instant when a given fact occurred; the other represents the instant when the information is *actually knowable* to the system. This distinction, implicit and usually not critical in on-line transactional processing (OLTP) applications, has a particular importance for DWHs, where it can be useful to determine what was the situation in the past, knowing only the information available at a given time.

Temporal databases provide support for past, current, or even future data and allow sophisticated queries over time to be stated [17]. Research in temporal databases [5] has grown immensely in recent years. In particular, transaction time and valid time have been proposed [7] and investigated in detail [8], [15].

In the field of *data warehousing*, Bliujute et al. in [1] concentrated on the shortcomings of star schemas in the context of slowly changing dimensions [9] and concluded that state-oriented warehouses allow easier analytical processing and even better query performance than observed in regular events warehouses. Our formal approach to managing temporal consistency (described in section 4) is state-oriented, too.

Pedersen and Jensen [11] describe features that entire DWH data models should have (including a requirement to handle changes in data over time) and evaluate previously proposed models. In the discipline of *temporal data warehouses* a lot of research was done in the context of temporal view maintenance, e.g. [13].

An interesting practice approach is described in [4], where *timeliness* is viewed as the time from when a fact is *first captured* in an electronic format and when it is actually *knowable* by a knowledge worker who needs it. Late-arriving facts and dimension records [10] can complicate this situation, because they are changing counts and totals for prior history. Some industries, like health care, have to deal with huge numbers of late-arriving records [3].

The nature of delays in active temporal databases is discussed in [12], concluding that *temporal faithfulness* has to be provided. Applying this concept to data warehousing ensures that information is analytically processed in a consistent way.

3 Temporal data structures

Timestamps allow the maintenance of temporal data. When considering temporal data for DWHs we need to understand how time is reflected in a database, how this relates to the structure of the data, and how a state change affects existing data. There are a number of approaches:

- **Transient data**. The key characteristics of transient data is that alterations to and deletions of existing records physically destroy the previous data content. This type of data is typically found in operational environments (e.g. order-entry systems).
- **Periodic data**. Once a record is added to a database, it is never physically deleted, nor is its content ever modified. Rather, new records are always added to reflect updates or even deletions. Periodic data thus contains a complete record of the changes that have occurred in the data. DWHs are periodic in nature.
- **Semi-periodic data**. This kind of data is typically found in the real-time data of operational systems where previous states are important (bank account systems, insurance premiums systems, etc.). However, in almost all operational systems, the duration for which persistent data are held is relatively short, due to performance and/or storage constraints. Therefore, this kind of data may be termed semi-periodic data.
- **Snapshot data**. Snapshot data are a stable view of data as they exist at some point in time. They are a special kind of periodic data. Snapshots usually represent the data at some time in the past, and a series of snapshots can provide a view of the history of an organization.

The standard approach to storing periodic data (typically found in DWHs) is to use time stamped status and event records. There are, however, a variety of schemes to maximize the efficiency of timestamps [2], [8], [15].

The *single timestamp* approach storing only a *start time* when a record became valid, is well applicable to event data, but faces serious deficiencies in the context of DWHs, where in general state information is stored. There are two relatively common types of queries used in DWH environments, which explain the problem:

1. A query that needs to access current data. In a single timestamp scheme, the only way to identify current records is to find the latest timestamp of the periodic set, which is an inefficient process.
2. A query that builds a view of the data at a particular time in the past. In order to support this kind of query, the period of validity of each record must be known, to

compare it with the required time. With a single timestamp approach, the end of the period of validity can only be found from the next record in the periodic sequence. In general this is also an expensive process.

In order to address the first problem, a second timestamp (called *end time*) can be added to each fact. It identifies the end of the period of validity. This causes performance improvements in retrieving data. However, it is not sufficient to fully solve the second problem, because the period of validity can change over time due to new information integrated into the DWH later. A temporally consistent view (similar to snapshot data) during analytical processing requires the storage of both, the old and the new (maybe *overlapping*) validity period. Therefore, we enhance DWH data models with following time dimensions (described in detail in section 4.2):

- **valid time dimension** (validity of knowledge) as motivated above.
- **revelation time dimension** (transaction time). It describes the point in time, when a piece of information was realized by at least one source system.
- **load timestamp.** This represents the point in time when the new piece of information was integrated.

4 Temporal consistency

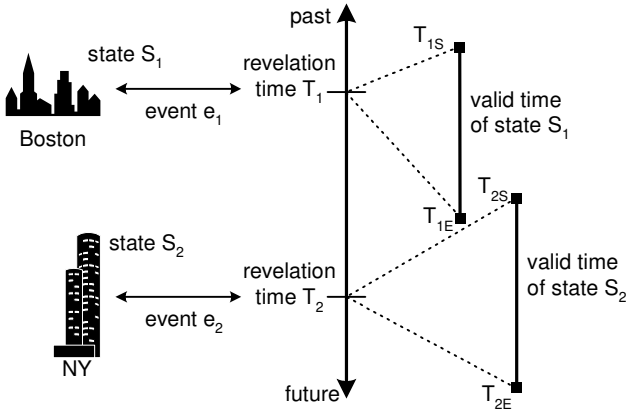

Figure 2. Overlapping validity periods.

The continuum of real time can be modeled by a *directed timeline* consisting of an infinite set {T} of *instants* (time points on an underlying time axis [7]). A section of the timeline is called a *duration*. An *event* takes place at an instant of time, and therefore does not have duration. A *time interval* is determined by the duration between two corresponding (start - end) instants.

Figure 2 describes a situation, where a computer system observes state S_1 at T_1 indicating that a specified person (Mr. Smith) lives in Boston. It knows that Mr. Smith stays there from T_{1S} to T_{1E}. The next state (S_2) knowable to the computer system regarding Mr. Smith is the new address in New York at T_2. Additional data reveals that he already lived in New York since T_{2S}.

In order to determine what was the situation before instant T_2, it must be feasible to process only those states known before T_2. By modeling this situation temporally consistent, it will *always* be possible to find out, that the system did *not* know where Mr. Smith lived after T_{1E} until the instant T_2, when the new piece of information (state S_2) was integrated into the DWH.

4.1 Conceptual model

In this section we will present a conceptual model that *generalizes* the example from Figure 2, which illustrated the usefulness of overlapping valid times. A temporally consistent representation of information requires a reliable view on historical data at any point in time *independent* from propagation delays. Therefore we define:

A *knowledge state* (*KS*) is determined by a specified instant T. It considers all information (knowledge) that was observed, captured, and integrated until the instant T. An ordered relation of two instants $T_1 < T_2$ implies that $KS(T_1) \leq KS(T_2)$. In other words, moving forward in time causes the knowledge state to grow.

In general an analysis focuses on a time interval containing at least one *instant of interest* (*II*): $II_{interest} = [II_{start}, II_{end}]$ (e.g. July 1st, 2001). The KS and II are two *orthogonal* time dimensions and therefore independent from each other regarding analysis capabilities.

In general a stored state S_X is determined by an instant T_X (*revelation time*) and a corresponding *valid time* interval indicated by $[T_{XS}, T_{XE}]$. A data model enables *temporal consistency* if a set of nine conditions listed in Table 1 is satisfied for *any* combination of stored datasets (S_1, S_2) regarding the same subject.

Table 1. Conceptual model for temporal consistency

Knowledge State (*KS*)	Instant(s) of interest (*II*)	Retrieved state
$\mathbf{KS} < T_1$	Any **II**	(not defined)[1]
$T_1 \leq \mathbf{KS} < T_2$	$\mathbf{II} < T_{1S}$	(not defined)[1]
	$T_{1S} \leq \mathbf{II} \leq T_{1E}$	$\mathbf{S_1}$
	$\mathbf{II} > T_{1E}$	(not defined)[1]
$\mathbf{KS} \geq T_2$	$\mathbf{II} < T_{1S}$	(not defined)[1]
	$T_{1S} \leq \mathbf{II} < \min(T_{1E}, T_{2S})$	$\mathbf{S_1}$
	$T_{1E} < \mathbf{II} < T_{2S}$ [2]	(not defined)[1]
	$\min(T_{1E}, T_{2S}) \leq \mathbf{II} \leq T_{2E}$	$\mathbf{S_2}$
	$\mathbf{II} > T_{2E}$	(not defined)[1]

[1] "not defined" means neither S_1 nor S_2.

[2] This case is obsolete, if the corresponding valid times $[T_{1S}, T_{1E}]$ and $[T_{2S}, T_{2E}]$ do not overlap.

Table 1 describes the timely correct state that will be retrieved during analytical processing. The retrieved state can be viewed as the output of a function considering the applied KS and the specified II. The reading examples below exactly describe the contributions of this conceptual model to conventional temporal models restricted to non-overlapping valid times.

- At any point in time an analysis based on a KS between T_1 and T_2 ($T_1 \leq KS < T_2$) is able to figure out that state S_1 was valid till T_{1E} (third entry in Table 1 above), and S_2 was not yet knowable to the system (forth entry).
- Moving forward in time (by applying a knowledge state of $KS \geq T_2$) will reveal that e.g. the state S_1 was only valid till T_{2S} (sixth entry in Table 1), under the precondition of *overlapping* valid times for S_1 and S_2.

Any combination of II with KS during analytical processing retrieves that state, which is knowable to the system at the specified KS. The model has full generality, because the number of the involved overlapping states is independent from the conditions. Example: If there are two overlapping states (S_1, S_2) considered in an analysis, only one of the nine types of overlapping can occur.

4.2 Modeling temporal consistency for DWHs

The identified propagation delays and temporal order issues in a DWH environment are represented as follows:

Valid time. This property is always related to a state (dataset) at an instant T, and describes the time interval of validity of this information - knowable at instant T.

Revelation time. The revelation time describes the point in time, when a piece of information was realized by at least one source system. This concept is tightly related to the notion of transaction time [15] in temporal databases. However, in the context of DWHs we call this property "revelation time" to clarify that 1) transaction results were already recorded in the source systems and 2) analytical processing is proposed to reveal interesting relations among facts.

Load timestamp. The load timestamp is a time value associated with some integrated dataset in a DWH. Thus, in the presence of delays, it is the load timestamp that represents the point in time to which automatic decision making or compensating actions should refer in active DWH environments.

The *conceptual model* is associated with the mentioned (orthogonal) types of time information as follows:

- The *valid time* is related to the *instants of interest (II)* during analytical processing.
- Typically the *knowledge state (KS)* relates to the *revelation time*, to get a timely accurate picture of the entity of interest. However, it is possible to relate to the *load timestamp*, to observe and control active mechanisms in the context of active DWHs.

Logical model:

When designing a temporal data model, an important and central aspect is the choice of appropriate timestamps of the database facts. Time intervals are used as an abbreviation for sets of time points for practical reasons. In order to model past and future instants two symbolic instants are introduced: $-\infty$ ("since ever") and $+\infty$ ("until changed"). Analytical processing, which is known from traditional DWHs, is effectively supported by reusing the commonly available time dimension for *load timestamps*.

Both, the *revelation* and *valid time* of facts will be integrated as separated logical time dimensions modeled by time intervals or by single timestamps. We already discussed this issue in Section 3. It depends on the applied environment in which the approach will be better suited for [2]. However, deciding on time intervals, the physical realization can be done by a *bitemporal* model (Table 2) combining the features of a transaction-time and a valid-time database. Thus, it represents reality more accurately and allows for retroactive as well as postactive updates.

The proposed logical model allows a designer to mix temporal and non-temporal aspects (different strategies for every fact star). Temporal consistency for *aggregates* is handled the same way. The integration of new facts concerning information about previous data loads causes the re-computation of the affected aggregates. The "old" aggregates can either be deleted or stored time consistently by adjusting the valid time from "until changed" to the re-computation timestamp.

Data staging:

Whereas data staging in traditional DWHs establishes a *delivery order* for new information, managing temporal consistency requires the (stronger) *temporal order*, which takes into account already stored facts and aggregates.

The integration of new data enhancing the knowledge state regarding a particular subject is done by *retroactive updates*. Existing data structures (in particular the end date for revelation time) of the involved datasets only have to be modified if valid times overlap. It is important to note that both the valid time of the old state provided by the DWH and the new one from data staging (provided by a temporal source systems) are *not* changed - only the end date of the revelation time will be changed (if a bitemporal model is used).

Table 2. Temporal order for overlapping valid times.

location	Valid time	revelation time	load timestamp
Boston	2001-02-01 – 2001-04-01	2001-02-05 – until changed	2001-02-06

location	Valid time	revelation time	load timestamp
Boston	2001-02-01 – 2001-04-01	2001-02-05 – **2001-04-14**	2001-02-06
NY	2001-03-15 – 2001-06-01	**2001-04-15** – until changed	2001-04-16

Table 2 shows the physical datasets according to the situation exemplified in Figure 2. This kind of temporal integration method slows down data staging (only in the case of overlapping valid times - typed in bold face in Table 2), but simplifies analytical processing dramatically.

5 Evaluation

The hypercube-based multidimensional model, and the star-schema based (extended-) relational model have emerged as candidate data models for DWHs. However, these models do not adequately address issues related to history data and temporal consistency, which are certainly core issues in data warehousing. In order to model the history of an organization as accurately as possible, DWHs have to cope with propagation delays. If a DWH has to "tie to the books" (cash flows, statistics, strategic planning, etc.), it is not possible to change aggregates (e.g. an old monthly sales total), even if the new information indicates that the old sales total was incorrect. Traditional DWHs ignoring the revelation time of late-arriving records will invalidate cash flows and statistics.

The proposed model enables explicit hierarchies in the time dimensions [11] to aid the user in navigation. It allows multiple hierarchies in the time dimension based on the load timestamp, e.g. days could roll up into weeks or months. To our knowledge it is the first DWH model that handles the change in data over time systematically by adding validity periods and allowing overlapping valid times regarding the same subject. This is actually an enhancement to the proposed three types of slowly changing dimensions [9], in particular for type two.

The usability of the proposed approach was evaluated by the application to an industry project (described in [3]) in the domain of accident insurance. There we found out that the introduction of the *knowledge state* viewpoint is an enhancement to analytical processing. The focus of our approach to managing temporal consistency enables knowledge workers to establish and control active behavior for DWH environments.

Managing temporal consistency of stored facts does not guarantee a timely correct view of the modeled real world. But it ensures that every piece of information captured by an organization's operational source system is integrated in a timely correct manner into an entire DWH.

Important future research directions in this field will be the maintenance of DWHs over dynamic information systems (data updates, schema changes, dynamic sources).

6 Conclusion

The advantages provided by built-in temporal consistency support in data warehouses include higher-fidelity data modeling, more efficient capturing of an organizations history, as well as analyzing the sequence of changes to that history.

The presented approach enables a timely consistent view for analysis purposes considering that the validity of detail data (or aggregates) is typically restricted to time periods, because of capturing delays and late-arriving records. Ignoring this temporal issue leads to impoverished expressive power and questionable query semantics in many real-life scenarios.

7 References

[1] R. Bliujute et al. Systematic Change Management in Dimensional Data Warehousing. Technical Report TR-23, TimeCenter, January 1998.

[2] M.H. Böhlen et al. Point- versus Interval-Based Temporal Data Models. In Proc. of 14th ICDE, IEEE Computer Society Press, pp. 192-201, Orlando, Florida, USA, 1998.

[3] R.M. Bruckner and A M. Tjoa. Managing Time Consistency for Active Data Warehouse Environments. To appear: Proc. Intl. Conf. DaWaK 2001, Munich, Germany.

[4] L.P. English. Improving Data Warehouse and Business Information Quality. John Wiley & Sons, New York, 1999.

[5] O. Etzion, S. Jajodia, and S. Sripada, (editors). Temporal Databases: Research and Practice, LNCS 1399, 1998.

[6] W.H. Inmon. Building the Data Warehouse. Second Edition, John Wiley & Sons, New York, 1996.

[7] C.S. Jensen and C.E. Dyreson, (editors). The Consensus Glossary of Temporal Database Concepts. In [5], pp. 367-405, 1998.

[8] C.S. Jensen and R.T. Snodgrass. Temporal Data Management. In IEEE Transactions on Knowledge and Data Engineering, Vol. 11(1): pp. 36-44, 1999.

[9] R. Kimball. The Data Warehouse Toolkit. Jon Wiley & Sons, New York, 1996.

[10] R. Kimball. Backward in Time. In Intelligent Enterprise Magazine, Vol. 3(15), September 2000.

[11] T.B. Pedersen and C.S. Jensen. Multidimensional Data Modeling for Complex Data. In Proc. of 15th ICDE, IEEE Computer Society, pp. 336-345, Sydney, Australia, 1999.

[12] J.F. Roddick and M. Schrefl. Towards an Accommodation of Delay in Temporal Active Databases. In Proc. of the 11th Australasien Database Conference (ADC2000), IEEE Computer Society, pp. 115-119, Canberra, Australia, 2000.

[13] E.A. Rundensteiner et al. Maintaining Data Warehouses over Changing Information Sources. Communications of the ACM, Vol. 43(6): pp. 57-62, 2000.

[14] M. Schrefl and T. Thalhammer. On Making Data Warehouses Active. In Proc. of 2nd Intl. Conf. DaWaK 2000, Springer, LNCS 1874, pp. 34-46, London, UK, 2000.

[15] K. Torp, C.S. Jensen, and R.T. Snodgrass. Effective Timestamping in Databases. The VLDB Journal, Vol. 8, Issue 3-4, pp. 267-288, 2000.

[16] P. Westerman. Data Warehousing: Using the Wal-Mart Model. Morgan Kaufmann Publ., San Francisco, 2000.

[17] C. Zaniolo et al. Advanced Database Systems. Morgan Kaufmann Publishers, San Francisco, 1997.