Managing Time Consistency for Active Data Warehouse Environments

Robert M. Bruckner, A M. Tjoa

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

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. Typically, traditional data warehouses focus on summarized data (at some level) rather than detail data.
For active data warehouse environments, also detailed data about individual entities are required for checking the data conditions and triggering actions. Hence, keeping data current and consistent in that context is not an easy task.
In this paper we present an approach for modeling conceptual time consistency problems and introduce a data model that deals with timely delays. It supports knowledge workers, to find out, why (or why not) an active system responded to a certain state of the data. Therefore the model enables analytical processing of detail data (enhanced by valid time) based on a knowledge state at a specified instant of time. All states that were not yet knowable to the system at that point in time are consistently ignored.

1. Introduction

The observation of real-world events by computer systems is characterized by a delay. This so-called *propagation delay* is the time interval it takes for a monitoring system to realize an occurred state change. While operational systems are among other things designed to meet well-specified response time requirements, the focus of data warehouses (DWHs) [8] is generally the strategic analysis of data integrated from heterogeneous systems. The data integration process is very complex and covers the acquisition, extraction, transformation (staging area), and loading of the data into the DWH. Traditionally there is no real-time connection between a DWH and its data sources, because the write-once read-many decision support characteristics would conflict with the continuous update workload of operational systems and cause poor analysis concurrency. Data loading is done during frequent update windows (e.g. once a week, every night), when the analysis capabilities of DWHs are not affected.

Consequently, up till recently, *timeliness* requirements [7] (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 [8] 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" [17] to changing customer needs, supply chain demands, and financial concerns.

New concepts, like *active data warehousing* [14], try to combine active (information) mechanisms based on ECA (event-condition-action) rules [1] with the integrated analysis power of DWH environments to extend (passive) systems with reactive capabilities: An active data warehouse (*ADWH*) is *event driven*, *reacts* in a timeframe appropriate to the business needs, and makes *tactical* decisions or causes operational actions rather than waiting for periodic reports. Therefore the design of an ADWH has to consider technical aspects (scalability, high availability, "just-in-time" data loading, mixed workload, etc.) as well as the integration of active mechanisms, which have to deal with three kinds of propagation delays in DWH environments:
1) Real-world changes are usually discovered and captured delayed by computer systems. 2) The typical update patterns for traditional DWHs on a weekly or even monthly basis enlarge these propagation delays. 3) Knowledge workers require consolidated and aggregated information for analysis purposes. However, in the case of *traditional DWHs* any significant delay in the recognition of real-world events may result in a number of additional considerations needing to be taken into account:

- *data integration.* "Old" aggregates have to be updated.
- *analytical processing.* Historical analysis results are not repeatable any longer, if additional information regarding that time period is integrated delayed (measures and summaries will change unexpectedly from the user's perspective).

The complexity of analysis and data access required places the typical ADWH query outside the scope of most transaction-processing systems and addresses a type of automated decision-making that has rarely been accomplished by passive DWHs [14].

Consequently, keeping data current and consistent in that context is not easy. Although higher refresh frequencies shrink the propagation delays, time consistency becomes even more important, because business rules can be formulated according to the ECA structure of triggers in ADWHs. Delays in the execution of such rules can cause unexpected and undesired side-effects [11].

We will concentrate on the development and examination of a schema model that copes with delays and allows a timely consistent representation of information. This distinction has a particular interest in DWHs, where it can be useful to consider information validity of detail data (or even aggregates), that is typically restricted to time periods, because of frequent updates. It is important to note that the aim of our approach is not necessarily to shrink delays. Rather, it is a pragmatic one to ensure that the side-effects of propagation delays are reduced in ADWHs. The main contribution of this paper is the introduction of overlapping valid time intervals.

The remainder of this paper is organized as follows. Section 2 considers research issues and related work. Time consistency is introduced in Section 3 and described from different viewpoints (conceptual, logical, and physical). Section 4 describes a case study followed by an assessment. Finally, we give a conclusion in Section 5.

2. Research Issue and Related Work

The importance of data quality for organizational success has been underestimated for a long time. Quality management in information processing is not nearly as established as it is in manufacturing [2]. Since DWHs store gigabytes up to terabytes of data, a manual quality control is not feasible at all. In data quality research, MIT's *Total Data Quality Management* (TDQM) [16] project considers *timeliness* as a so-called quality dimension of the *contextual* data quality.

An interesting practice approach is described in [7], 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. For data captured in a source database, uploaded to a central DWH over the weekend, the propagation delay can be several days. This is definitely unacceptable for implementations tightly integrated into business processes (like ADWHs), which are networked with other organizations.

Temporal databases provide support for past, current, or even future data and allow sophisticated queries over time to be stated [6, 19]. Research in temporal databases has grown immensely in recent years. In particular, transaction time and valid time have been proposed [9] and investigated [10, 15].

In the field of data warehousing [3] concentrated on the shortcomings of star schemas in the context of slowly changing dimensions. The conclusion is that state-oriented warehouses allow easier analytical processing and even better query performance than observed in regular events warehouses. Our formal approach to managing time consistency (described below) is state-oriented, too. Temporal data warehouses (in particular temporal view self-maintenance) are addressed by [18].

Recent research in *active databases* [5] aims at extending the power of active rules (for responding to changes to the database and to external events) to trigger on complex patterns of temporal events. An important issue is the accommodation of *delays*, which is investigated for *temporal active databases* in [11]. This paper concentrates on delays in the quite young discipline of *active data warehouses* [14].

3. Time Consistency

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. 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 [9]). A section of the timeline is called a *duration* or *time period*. An *event* takes place at an instant of time, and therefore does not have duration. If two events occur at an identical instant, then the two events are said to occur *simultaneously*. A *time interval* is determined by the duration between two corresponding (start - end) instants.

Figure 1 below describes a situation, where a computer system observes the state S_1 at T_1 that a specified person (Mr. Smith) lives in Boston. It knows (or assumes – that does not matter for the model) that Mr. Smith stays there from T_{1S} to T_{1E}. The next state (S_2) observed by the computer system regarding Mr. Smith is the new address in New York at T_2. Furthermore, additionally captured information reveals that Mr. Smith already lived in New York since T_{2S}. By modeling this situation timely 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. Based on this knowledge an ADWH can determine the "right" strategy e.g. for *customer relationship management* (CRM): An "information package for new citizens of NY" is appropriate at T_{2S}, but may be unsuitable at the instant T_2, when the real-world event was actually knowable to the ADWH.

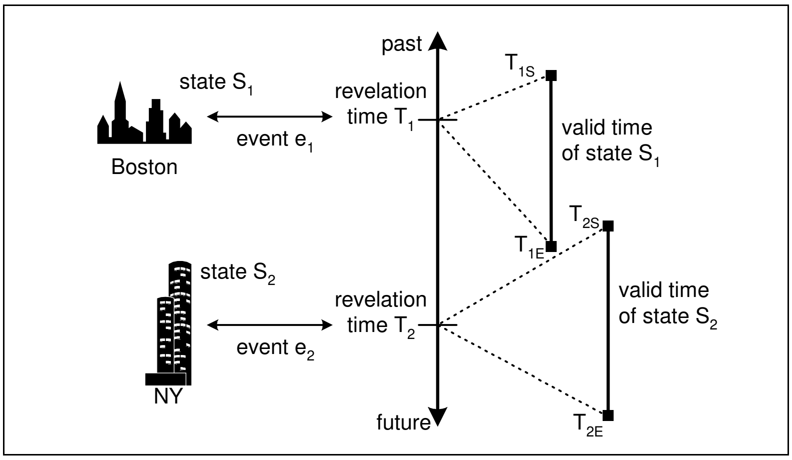

Fig. 1. Sequencing the changes to capture the history regarding an entity of interest.

3.1. A Conceptual Model for Time Consistency

A timely consistent representation of information requires a reliable view on historical data at any point in time independent from propagation delays.

A *knowledge state* (*KS*) is determined by a specified instant T. It considers all information (knowledge) that was observed, captured, and integrated until 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. May 1st, 2001). The KS and II are two *orthogonal* time dimensions and therefore independent from each other regarding analysis capabilities.

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

Any combination of II with KS during analytical processing has to retrieve a correct state, which is actually knowable to the system at the specified KS. The *retrieved state* regarding an entity of interest will be S_1, or S_2, or "*not defined*" (if it is neither S_1 nor S_2). Table 1 (which is related to Figure 1) describes the timely correct state that should 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 conceptual model is able to handle past and future validity periods (e.g. instant(T_1) > instant(T_{1E}), instant(T_1) < instant(T_{1S})), which is important 1) to deal with propagation delays regarding timeliness, and 2) to support planning. The reading examples below Table 1 exactly describe the major contributions of this conceptual model to conventional temporal models restricted to non-overlapping valid times.

Table 1. Conceptual model for time consistency.

Knowledge State (*KS*)	Instant(s) of interest (*II*)	Retrieved state[1]
KS $< T_1$	Any **II**	(not defined)[2]
$T_1 \leq$ **KS** $< T_2$	**II** $< T_{1S}$	(not defined)
	$T_{1S} \leq$ **II** $\leq T_{1E}$	$\mathbf{S_1}$
	II $> T_{1E}$	(not defined)
KS $\geq T_2$	**II** $< T_{1S}$	(not defined)
	$T_{1S} \leq$ **II** $< \min(T_{1E}, T_{2S})$	$\mathbf{S_1}$
	$T_{1E} <$ **II** $< T_{2S}$ [3]	(not defined)
	$\min(T_{1E}, T_{2S}) \leq$ **II** $\leq T_{2E}$	$\mathbf{S_2}$
	II $> T_{2E}$	(not defined)

Reading examples. 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 1) the state S_1 was valid till T_{1E} (third entry in Table 1 above), and 2) S_2 was not yet knowable to the system (fourth 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 assumption of overlapping valid times for S_1 and S_2.

3.2. Modeling Temporal Consistency for Data Warehouses

The issue of modeling time consistency for centralized DWH environments is challenging due to the following reasons:
1. DWHs have to deal with an additional propagation delay regarding electronically captured data by their source systems (described in Section 1).
2. The integration of distributed source systems (e.g. different time zones, delays in data processing) is necessary during data staging.

In our opinion, the most efficient approach is to separate these two issues as described above by extending the logical and physical data model to handle the former one. The latter step, which is complicated due to temporal order issues (see Section 3.3), is done during data staging and therefore does not affect analytical processing.

We propose an enhanced time dimension model for DWHs to accomplish time consistency, because simply adding timestamps during data loading on an overnight or even weekly basis is not precise enough to model the instant, when a new fact was first captured electronically and therefore knowable to an organization. The identified propagation delays in a ADWH implementation are represented as follows:
- **valid time** (validity of knowledge). This property is always related to a base event at an instant T (stored as dataset), and describes the time interval of validity (knowable at instant T) of that piece of information.
- **revelation time** (*transaction 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 [9, 15] in temporal

[1] The retrieved state is timely correct regarding the electronically captured input data.
[2] "not defined" means neither S_1 nor S_2.
[3] This case is obsolete, if the corresponding valid times $[T_{1S}, T_{1E}]$ and $[T_{2S}, T_{2E}]$ do not overlap.

databases. However, in the context of ADWHs we call this property "revelation time" to clarify that 1) transaction results were already recorded in the sources 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 environment and determines the instant, when a fact is actually knowable to *active* mechanisms. 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 ADWH 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 enable knowledge workers to observe and control active mechanisms in an ADWH.

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 [4] are used as an abbreviation for sets of points for practical reasons. A 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. Simply returning such temporary values to users during analytical processing *without* integration into the data model may lead to confusion.

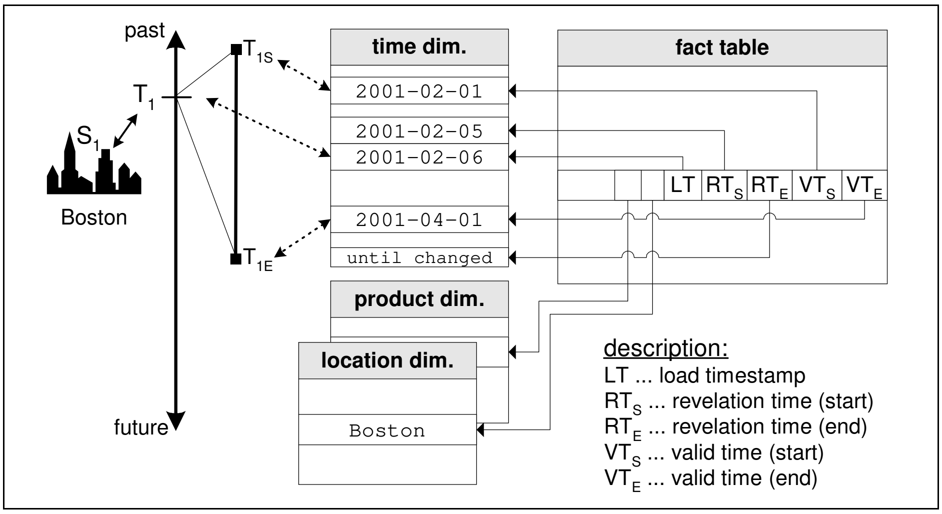

Fig. 2. Representation of state S_1. This piece of information was revealed on February 5th with a valid time from February 1st to April 1st and loaded into the ADWH on February 6th.

In order to model past and future instants two symbolic instants are introduced: $-\infty$ ("since ever") and $+\infty$ ("until changed"). We will reuse the time dimension commonly available in all warehouse models to integrate the *load timestamps* of facts (as described above). Thus analytical processing known from traditional DWHs is effectively supported. Both, the *revelation* and *valid time* of facts will be integrated as separated logical time dimensions modeled by time intervals. The physical implementation is done by a bitemporal model (exemplified in Figure 2) combining the features of a transaction-time and a valid-time database. Therefore it represents reality more accurately and allows for retroactive as well as postactive changes.

The proposed approach is suitable to on-line analytical processing (OLAP), supporting the creation of cubes well: e.g. analyzing the average propagation delays (differences between valid and revelation times) by drilling down to the distributed operational systems located in different geographical regions.

Time consistency for *aggregates* is handled the same way as for facts. 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.

Physical model:
The temporal logical model presented above permits the designer to mix temporal and nontemporal aspects. In particular it is possible to apply different strategies to every *fact star* in a DWH. Actually implementing a logical data model supporting time consistency is influenced by following issues:

- It is necessary to provide a *physical representation* for symbolic instants ($+\infty$, $-\infty$), e.g. by pre-defined date constants.
- Indexes are even more important in temporal relations due to their size. Furthermore, the focus of temporal analytical processing on *coalescing* (merging value-equivalent tuples with intervals that overlap or meet) and *temporal joins* [10] requires appropriate indexes [13].
- In temporal queries, conjunctions of inequality predicates (which are harder to optimize) appear more frequently. Consequently, improved cost models for the optimization of temporal operators have to be applied.

3.3. Data Staging

While some source systems may store partial history, many nontemporal sources store only the current state of their data. Even if DWH storage is not a concern, it is helpful to identify those sources/data where delays possibly occur that have side-effects on automatic decision making in ADWHs and thus should be modeled in a time consistent manner. Whereas data staging in traditional DWHs establishes a *delivery order* for new information, managing time consistency requires the (stronger) *temporal order*, which takes into account already stored facts and aggregates.

During data staging we have to consider three issues in order to get a single version of truth regarding time during data staging: 1) different time zones of distributed sources, 2) differences in observing the same event by different operational systems; e.g. a CRM system records an order cancellation on May 2nd, whereas the billing system observes it on May 3rd, and 3) propagation delays, as described in Section 1.

The former two issues are addressed by establishing a global time base known from the field of distributed or real-time systems. The latter one is managed by the proposed approach to time consistency modeling: 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 ADWH only have to be modified if valid times overlap. It is important to note that both the valid times of the old state provided by ADWH and the new one from data staging (provided by temporal source systems) are *not* changed.

Table 2 shows the physical datasets according to the situation exemplified in Figure 1 and Figure 2. This kind of temporal integration method slows down data staging in the case of overlapping valid times, but simplifies analytical processing dramatically. As mentioned earlier this method is applicable to aggregates, too.

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

4. Experimental Results

The usability of the proposed approach was evaluated by the application to an industry project. We found out that the introduction of the knowledge state viewpoint is actually an enhancement to analytical processing. Furthermore, *planning* and *monitoring* including *what-if scenarios* (i.e. different versions of plans) are supported at various aggregation levels, because valid times are suitable to past, present and future states. The focus of our approach to managing time consistency enables knowledge workers to establish and control active behavior for ADWH environments.

Case study: PLUS project

The PLUS project was an industry project in the domain of accident insurance implemented at a big Austrian insurance authority starting in 1998 and continued till 2000. One focus of the project was the management of time consistency and the introduction of *active behavior* to automate routine decision tasks. For example, initiate and control the stopping of accident pension payments due to pre-specified reasons (successful rehabilitation, death, etc.). From a technical viewpoint, the project implemented a *data mart* (~ 5 GB) on an Oracle database tightly integrated with a workflow management system and the organization's billing system. The data model contained fact stars using the proposed temporal model, as well as non-temporal fact stars. The source systems were partly enhanced to provide additional temporal information to better deal with delays. Legacy data was integrated using valid times reconstructed from the sequence of states available regarding the same subject.

A very important analytical application for public insurance authorities is the periodic generation of many *timely consistent* statistical reports classifying and investigating changes of accident pensions from various viewpoints (due to official statistics for the European Statistical System). The approach to manage time consistency reduced the necessary effort to generate and maintain statistics dramatically. Generating such statistics without temporal support is difficult, because every report has to consistently contain aggregated figures concerning previous months or years.

Another driving force to establish this kind of temporal data marts is the limitation of traditional DWHs regarding e.g. cash flows, which differ from reality, if significant changes are captured delayed. For example, a person has an accident on June 29th and thus will receive a pension. Due to typical delays (pension request, data processing, weekends, etc.) the decision is made on July 5th, when both the billing and statistics for June is already completed. Traditional DWHs ignoring the revelation time and thus storing June 29th, will invalidate cash flows and statistics (as already mentioned).

Assessment

Managing time consistency according to the presented *state-oriented* approach allows us to achieve consistent analysis results at any point in time for a specified knowledge state. In particular, the approach

+ is based on a formal model with strong (but simple) guidelines. The implementation uses well-understood concepts, e.g. bitemporal tables [3, 15].
+ enhances the time dimension and thus is suitable to various application domains.
+ improves data quality during data staging by considering stored data structures.
+ can be introduced at any point in time. If temporal information is not available for legacy data, regular valid and revelation time intervals ("until changed") will be used to "upgrade" the old data structures.
+ enhances analytical capabilities of DWHs (knowledge state, instants of interest).
+ does not restrict the entire data warehouse to be temporal, but rather permits the designer to mix temporal and nontemporal aspects.
– covers only time relevant aspects of data quality management in DWHs.
– needs to consider performance issues: Updates to time attributes of existing fact-table rows are necessary when bulk loading the ADWH.

In contrast, an *event-oriented* DWH (each row in the fact table represents the *change* occurred according to the previous event) is easier to feed with new data. However, analytical processing is slowed down, because queries asking questions about states of some object in some period of time will result in the comparison of timestamps from different rows.

5. Conclusion and Further Research

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 time consistency, which are certainly core issues in data warehousing, particularly for ADWHs.

The advantages provided by built-in time 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.

Overall, we have shown how to provide knowledge workers with simple, consistent, and efficient support for ADWH environments. The presented approach enables a time consistent view for analysis purposes considering that the validity of detail data (or aggregates) is typically restricted to time periods, because of frequent updates and capturing delays. Ignoring this temporal issues leads to impoverished expressive power and questionable query semantics in many real-life scenarios.

Important future research directions in this field will be the maintenance of DWHs over dynamic information systems (data updates, schema changes, dynamic sources) [12], and enhancements to the active behavior (e.g. *automatic* compensation actions for significant changes) in the field of ADWH.

References

1. The ACT-NET Consortium: *The Active Database Management System Manifesto: A Rulebase of ADBMS Features*. In ACM SIGMOD Record, Vol.25(3): pp. 414-471, 1996.
2. Ballou, D.P., Tayi, G.K.: *Enhancing Data Quality in Data Warehouse Environments.* Communications of the ACM, Vol. 42(1): pp. 73-78, 1999.
3. Bliujute, R., Saltenis, S., Slivinskas, G., Jensen, C.S.: *Systematic Change Management in Dimensional Data Warehousing*. Technical Report TR-23, TIMECENTER, January 1998.
4. Böhlen, M.H., Busatto, R., Jensen, C.S.: *Point- Versus Interval-Based Temporal Data Models*. Proc. of 14th Intl. Conf. ICDE, IEEE Comp. Society Press, pp. 192-201, Orlando, USA, 1998.
5. Dittrich, K.R., Gatziu, St.: *Aktive Datenbanksysteme*. Dpunkt. Verlag, Heidelberg, 2000.
6. Dyreson, C.E., Evans, W.S., Hong, L., Snodgrass, R.T.: *Efficiently Supporting Time Granularities*. In IEEE Trans. on Knowledge and Data Engineer., Vol. 12(4): pp. 568-587, 2000.
7. English, L.P.: *Improving Data Warehouse and Business Information Quality.* John Wiley and Sons, New York, 1999.
8. Inmon, W.H.: *Building the Data Warehouse*. John Wiley and Sons, New York, 1996.
9. Jensen, C.S., Dyreson, C.E., (eds): *The Consensus Glossary of Temporal Database Concepts*. In *Temporal Databases: Research and Practice*, Springer, LNCS 1399, pp. 367-405, 1998.
10. Jensen, C.S., Snodgrass, R.T.: *Temporal Data Management*. In IEEE Transactions on Knowledge and Data Engineering, Vol. 11(1): pp. 36-44, 1999.
11. Roddick, J.F., Schrefl, M.: *Towards an Accommodation of Delay in Temporal Active Databases*. Proceedings of the 11th Australasien Database Conference (ADC2000), IEEE Computer Society, pp. 115-119, Canberra, Australia, 2000.
12. Rundensteiner, E.A., Koeller, A., Zhang, X.: *Maintaining Data Warehouses over Changing Information Sources*. Communications of the ACM, Vol. 43(6): pp. 57-62, 2000.
13. Salzberg, B., Tsotras, V.J.: *Comparison of access methods for time-evolving data*. In ACM Computing Surveys, Vol. 31(2): pp. 158-221, 1999.
14. Schrefl, M., Thalhammer, T.: *On Making Data Warehouses Active*. Proceedings of the 2nd International Conference DaWaK, Springer, LNCS 1874, pp. 34-46, London, UK, 2000.
15. Torp, K., Jensen, C.S., Snodgrass, R.T.: *Effective Timestamping in Databases*. The VLDB Journal, Vol. 8, Issue 3-4, pp. 267-288, 2000.
16. Wang, R.Y.: *Total Data Quality Management*. Communications of the ACM, Vol. 41(2): pp. 58-65, 1998.
17. Westerman, P.: *Data Warehousing: Using the Wal-Mart Model*. Morgan Kaufmann Publishers, San Francisco, 2001.
18. Yang, J., Widom, J.: *Temporal View Self-Maintenance*. Proceedings of the 8th Intl. Conf. EDBT2000, Springer, LNCS 1777, pp. 395-412, Konstanz, Germany, 2000.
19. Zaniolo, C., Ceri, S., Faloutsos, C., Snodgrass, R.T., Subrahmanian, V.S., Zicari, R.: *Advanced Database Systems*. Morgan Kaufmann Publishers, San Francisco, 1997.