# Data Freshness Evaluation in Different Application Scenarios[1]

Verónika Peralta, Mokrane Bouzeghoub

Laboratoire PRISM, Université de Versailles
45, avenue des Etats-Unis
78035, Versailles cedex, FRANCE
{Veronika.Peralta, Mokrane.Bouzeghoub}@prism.uvsq.fr

**Abstract.** Data freshness has been identified as one of the most important data quality attributes in information systems. This importance increases especially in the context of systems that integrate a large set of autonomous data sources. In this paper we describe a quality evaluation framework which allows evaluation of data freshness in different architectural contexts. We also show how this quality factor may impact the reconfiguration of a data integration system to fulfill user expectations.

## 1 Introduction

Data freshness has been identified as one of the most important attributes of data quality for data consumers (Shin 2003) (Wang et al. 1996). Specifically, the increasing need to access to information which is available in several data sources introduces the problem of choosing between alternative data providers and of combining data having different freshness values (Naumann et al. 1999). This paper deals with data freshness evaluation in the context of a Data Integration System (DIS) that integrates data from different independent data sources and provides the users a uniform access to this data.

Data freshness represents a family of quality factors among which currency and timeliness are representative examples: *currency* describes how *stale* is data with respect to the sources and *timeliness* describes how *old* is data. In (Bouzeghoub et al. 2004) we analyze these factors and several metrics proposed to measure them. In (Peralta et al. 2004), we proposed a framework for analyzing and evaluating data freshness based on a calculation dag which abstracts a workflow of integration activities. After a brief recall of this framework, this paper shows how it can practically be used in different application scenarios and how the data integration system can be improved in order to fulfill user requirements in terms of data freshness.

The rest of the document is organized as follows: Section 2 briefly describes the data quality evaluation framework and discusses how to use it through different application scenarios. Section 3 focuses on the possible improvement actions to put on the DIS workflow to achieve user requirements. Finally, section 4 concludes with our general remarks.

[1] *This research was partially supported by the French Ministry of Research and New Technolologies under the ACI program devoted to Data Masses (ACI-MD), project #MD-33.*

# 2 Data Freshness Evaluation

In this section we describe the evaluation approach. We firstly recall the quality evaluation framework. Then, we give an intuitive idea of the freshness calculation strategy and we describe a base evaluation algorithm, discussing its instantiation to different application scenarios.

## 2.1 The Data Quality Evaluation Framework

Our quality framework models the DIS processes and properties and evaluates the freshness of the data returned to the user. The DIS is modeled as a workflow in which activities perform different tasks that extract, transform and convey data from sources to end-users. Similarly, the quality evaluation framework is represented by a *labeled calculation dag (LCDag)* which is isomorphic to the DIS workflow and which describes all necessary metadata to evaluate data freshness. Formally, a LCDag is a dag $G = <V, E, P, L_P>$ defined as follows: The nodes in V are of three types: *source nodes* (with no input edges), *target nodes* (with no output edges) and *activity nodes* (with both input and output edges), which respectively describe meta attributes on data sources, user queries and DIS activities. The edges in E represent that a node is calculated from another (data flows in the sense of the arrow). P is a set of properties describing DIS features and quality measures, and $L_P$ is a partial labeling function that assigns a property value to a node or edge of the dag. Figure 1 shows different examples of LCDags which will be discussed in section 2.3.

## 2.2 Freshness Evaluation Approach

The freshness of the data delivered to the user depends on the following properties:

- *Processing cost*: It is the amount of time, in the worst case, that an activity needs for reading input data, executing and building result data.
- *Synchronization delay*: It is the amount of time passed between the executions of two consecutive activities.
- *Actual freshness*: It is a measure of the freshness of data in a source.
- *Expected Freshness*: It is the desired data freshness specified by the user. It measures the extent to which the freshness of the data is appropriate for the task on hand.

Our base algorithm takes into account such properties and evaluates the freshness reached at each node of the calculation dag, using the following rules:

- For a source node A:
  Freshness(A) = getActualFreshness(A)
- For a non-source node A, and the set of all its predecessors P:
  Freshness(A) = combine {Freshness(B) +getSyncDelay(B,A) /B $\in$ P} + getProcCost(A)

For source nodes, data freshness is the source actual freshness. For the other nodes, the freshness of output data is calculated as the freshness of input data plus the synchronization delay plus the processing cost. When a node has several predecessors, the input freshness value is derived using a specific function; e.g. the maximum value among input values.

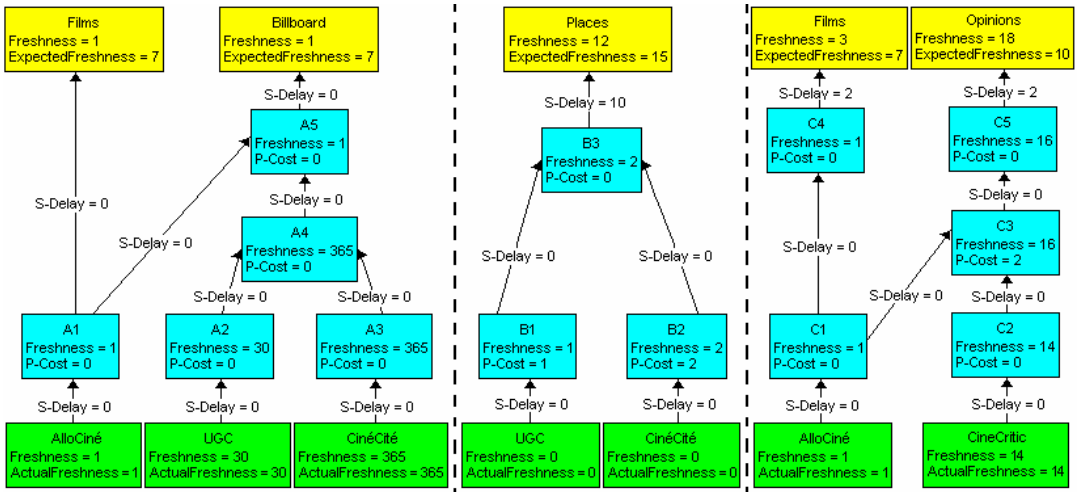


FIG. 1 – *Labeled calculation dags.*

## 2.3 Examples of Application Scenarios

Consider the three DIS of figure 1, which deal with information about cinemas and films:

- **$DIS_1$**: A mediation system that answers queries about films and the cinemas where they are in billboard. Typical queries are "Where can I see a film?" or "Which films are in billboard now?"
- **$DIS_2$**: A web portal that caches information about cinemas and the availability of places for their performances. Typical queries are "Where are available places to see a film?" or "How many places are available in a cinema?"
- **$DIS_3$**: A data warehousing system that stores statistic information about films, the number of persons that watch each film and their opinions. Typical questions are "Which films have the best ranking this week?" or "Which film should I watch?"

Users of $DIS_1$ and $DIS_3$ are concerned with *timeliness* but users of $DIS_2$ are concerned with *currency*. $DIS_1$ extracts film information from AlloCiné (via wrapper $A_1$) and cinema information from UGC and CinéCité (via wrappers $A_2$ and $A_3$). Activity $A_4$ merges the information from both cinema sites and activity $A_5$ joins film and cinema information. $DIS_2$ extracts place information from UGC and CinéCité. Activity $B_3$ is the cache core, that receives user requests and asks the sources when the cache needs refreshment (invoking wrappers $B_1$ and $B_2$). $DIS_3$ extracts film audience statistics from AlloCiné (via wrapper $C_1$) and spectator's opinions from CineCritic (via wrapper $C_2$). Activity $C_3$ reconciles data from both wrappers and activities $C_4$ and $C_5$ perform aggregations and calculate statistic data.

In the LCDags of figure 1, source nodes are labeled with their *actual freshness*, target nodes are labeled with *expected freshness*, activity nodes are labeled with *processing costs* (P-Cost) and edges are labeled with *synchronization delays* (S-Delay). Values are expressed in days for $DIS_1$ and $DIS_3$ but in minutes for $DIS_2$. Note that the "zeros" represent negligible values.

The relevance of the used properties depends on each particular scenario. A first remark is that freshness values should not be considered in the absolute but compared to freshness expectations. For example, users of $DIS_1$ may tolerate data freshness of "7 days", making processing costs and synchronization delays ("some minutes") negligible; while users of $DIS_2$ require "extremely fresh" data, making activity costs relevant. In addition, in the scenarios where the focus is data currency, source actual freshness is not relevant. For example, in $DIS_2$, it does not matter "how old is data in the sources"; the focus is in retrieving the same data that is stored in the sources.

Another aspect is how to calculate source actual freshness, processing costs and synchronization delays. Depending on the scenario, different DIS properties may influence their calculation. For example, in $DIS_2$ the processing cost of the wrappers is dominated by the cost of communicating with the sources. In $DIS_3$ and $DIS_2$ the materialization/caching of data introduces important synchronization delays, so the refreshment policies and frequencies are important properties to take into account. In virtual systems as $DIS_1$, these properties have no sense.

# 3 Data Freshness Enforcement

Data freshness provided by the DIS should be compared to expected freshness to check whether user requirements are satisfied or not. If freshness expectations are not achieved, one or both of the following actions can be initiated: (i) improve the design of DIS; (ii) negotiate with data providers or users to relax their constraints. In this section we discuss these ideas.

Observe that for each node, a path can exist from a source for which we add all synchronization delays and processing costs to the source actual freshness and we obtain the freshness of the node. For example, the freshness of activity $C_5$ can be calculated adding source actual freshness, processing costs and synchronizations delays in the path [CineCritic,$C_2$,$C_3$,$C_5$]. This path is called the *critical path* and represents the bottleneck for the freshness calculation.

The freshness of the data delivered to the user may be improved optimizing the design and implementation of the activities in order to reduce their processing cost or synchronizing the activities in order to reduce the delay between them. Sometimes, the changes can be concentrated in the critical path, other times a complete reengineering of the whole system is necessary. Optimization actions may include: optimizing activities implementation (algorithms, software or even hardware), improving synchronization policies (appropriate execution frequencies, parallelism) and redefining materialization strategy (refresh frequencies).

A direct application of the described evaluation approach is the selection between alternative implementations of the DIS. Data freshness can be estimated for several processes allowing the user/designer to choose the process with the best quality. For example, even improving activities design and synchronization, the freshness expectations of the *Opinions* query cannot be achieved because of the actual freshness of the *CineCritic* source. Considering an alternative process that queries other sources can be a solution.

Analogously, we can propagate freshness expectations from queries to sources (subtracting processing costs and synchronization delays). The propagated freshness expectations can help the DIS designer to know the freshness that he must ask the source provider for. A direct application of this strategy is the selection between alternative data sources to achieve freshness expectations. For example, propagating down freshness

expectations for the *Opinions* query we obtain a bound (6 days) for the actual freshness of the source providing user's opinions. This avoids considering sources as *CineCritic* that have greater actual values.

If the design of the DIS cannot be improved, an alternative is negotiating with users to relax their freshness expectations, based on the actual freshness estimated by our framework. Another alternative is negotiating with source data providers to relax source constraints. Sometimes the system hardware can be powered to support more frequent accesses to the sources. Other times, this alternative implies demanding and eventually paying for a better service, for example, receiving data with a lower actual freshness.

# 4 Conclusion

In this paper we addressed the problem of evaluating data freshness in a data integration system. We presented a quality evaluation framework and its practical use for evaluating data freshness in different application scenarios. The framework was implemented in a quality auditing tool that can be instantiated for evaluating data freshness in a concrete scenario. The tool allows identifying the critical path, changing property values in order to test alternative configurations and re-executing the evaluation algorithms to see the effects of the changes. In this sense, the tool brings an aggregate value to the auditing functionalities.

We are now working in confronting the evaluation results with user quality profiles. Future work will be concentrated on other quality factors and their mutual correlations.

# References


Bouzeghoub M., Peralta V. (2004), A Framework for Analysis of Data Freshness, in Proc. of the Int. Workshop on Information Quality in Information Systems (IQIS'2004), collocated with SIGMOD'2004, France, 2004.

Naumann F., Leser U. (1999), Quality-driven Integration of Heterogeneous Information Systems, Proc. of the 25th Int. Conf. on Very Large Databases (VLDB'99), Scotland, 1999.

Peralta V., Ruggia, R.; Kedad, Z.; Bouzeghoub M. (2004), A Framework for Data Quality Evaluation in a Data Integration System, Proc. of the 19º Simposio Brasileiro de Banco de Dados (SBBD'2004), Brazil, 2004.

Shin B. (2003), An exploratory Investigation of System Success Factors in Data Warehousing, Journal of the Association for Information Systems, Vol. 4(2003): 141-170, 2003.

Wang R., Strong D. (1996), Beyond accuracy: What data quality means to data consumers, Journal on Management of Information Systems, Vol. 12 (4):5-34, 1996.