Engineering Privacy Requirements in Business Intelligence Applications

Annamaria Chiasera, Fabio Casati, Florian Daniel, and Yannis Velegrakis

Department of Information Engineering and Computer Science
University of Trento, Italy
{chiasera,casati,daniel,velgias}@disi.unitn.it

Abstract. In this paper we discuss the problem of engineering privacy requirements for business intelligence applications, i.e., of eliciting, modeling, testing, and auditing privacy requirements imposed by the source data owner on the business intelligence applications that use these data to compute reports for analysts. We describe the peculiar challenges of this problem, propose and evaluate different solutions for eliciting and modeling such requirements, and make the case in particular for what we experienced as being the most promising and realistic approach: eliciting and modeling privacy requirements on the reports themselves, rather than on the source or as part of the data warehouse.

Keywords: privacy, business intelligence, outsourcing, compliance, provenance, reports, meta-reports.

1 Introduction

With the rapid increase in the amount of people's data that is gathered and exchanged electronically, the problem of information privacy is rapidly gaining attention. Dozens of public and private organizations now hold bits and pieces of our personal information, subject to a variety of more or less explicit privacy agreements and government laws. At the same time, business intelligence (BI) applications are gaining popularity, consistently with the desire of officials of public and private companies to monitor, analyze, understand, and eventually improve business processes and better serve customers and citizens. BI applications typically extract data from multiple data sources, clean them to ensure data quality and consistency to the possible extent, transform them, and then generate various kinds of reports used by managers and officials to analyze the performed processes.

From a privacy perspective, this scenario poses interesting and very concrete research challenges. The first is that data sources used by BI application often reside in different systems, different departments, even in different companies. This implies that data in the sources of the BI applications is subject to different constraints, both because it was collected under different privacy agreements with the citizens in the first place, but also because the different institutions may further regulate the use of the information they obtained.

The second (and biggest) issue is to define the privacy requirements the BI application must obey when processing the data provided by the source. Privacy laws and

W. Jonker and M. Petkovi\'{c} (Eds.): SDM 2008, LNCS 5159, pp. 219–228, 2008.

agreements are typically defined at a very high level and with a certain degree of "fuzzyness". However, the BI developers need to know which data can be extracted from the source databases, whether these data can be used to clean/refine data from other providers (e.g., entity resolution), which report users can view the data, whether data can be shown in aggregate form, at which level of aggregation, and so on. This degree of precision is needed to know how to *develop* and *test* the BI application and also how to *audit* and to resolve possible *disputes*.

Current privacy policy languages like P3P 16 and access control languages like IBM's EPAL or OASIS' XACML [3] allow one to express privacy requirements in terms of the authorized purposes for the use of the data. Purpose-based access control mechanisms as proposed with P-RBAC [10] extend standard RBAC approaches with the notion of purpose, condition for data usage, and obligations. Privacy policy languages and purpose-based access control languages are of general applicability and can be used in different contexts where data are released to third parties. However, their generality makes it hard to express actionable privacy requirements that are directly "testable" and "verifiable" along the BI data lifecycle [11].

In this paper we study the problem of engineering privacy requirements in BI applications. Specifically, we study different ways in which precise, testable and auditable requirements can be agreed upon with the source owners and then modeled as part of the BI application. We explore different and possibly complementary options, including ways to define privacy requirements via privacy metadata coupled with the source data, or coupled with the warehouse data, or coupled with reports, and we conclude that defining engineering requirements directly on reports is viable alternative to conventional approaches.

2 Privacy in Outsourced Business Intelligence Environments

Our research originated from projects developed in our area with the local governments, hospitals, and social agencies, where BI reports are needed by those institutions to have a comprehensive view of provided treatments, to evaluate the quality of delivered processes, and to compute reimbursements. Figure 1 illustrates such a scenario. The arrows in the figure illustrate information and data flow.

Any information provided by or related to a patient is typically considered sensitive personal information and as such, any retain, processing, or presentation should respect the privacy of the patient. Privacy restrictions are provided at multiple levels. First, they may be provided by the patients themselves. As patients visit a health-care center, they sign a consent agreement specifying how their personal information can be treated by the health-care institution. These restrictions accompany the provided data and are illustrated in Figure 1 as *privacy level agreements* (PLAs) or privacy requirements (we will use these terms interchangeably, as the PLA constitute requirements, from a privacy perspective, for the BI developer). The provided information is enhanced at the health-care location (the actual *data provider*, from the perspective of the BI provider who delivers the BI solution) by additional data on the treatment offered. Both kinds of data are provided to the BI applications, and PLAs between the institutions are defined for both. In addition, policies on usage and retention of patient data may also be regulated by local and national laws [22, 23].

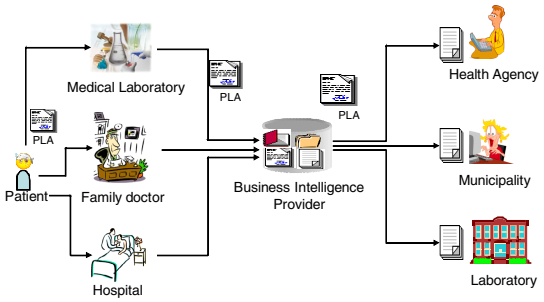

Fig. 1. A privacy-aware business intelligence outsourcing scenario

The *BI provider* extracts, integrates and transforms data that is then loaded on a data warehouse, from which reports are extracted and delivered to the BI users (e.g. reports combining data on usage of prescription drugs and their costs to identify differences in usages and prices, and what causes such differences). The BI provider needs to guarantee that the data it stores, the transformations it performs on that data, and the content of the reports delivered to the users (the co-called *information consumers*) it generates are all complying with the PLAs. It is thus important, and also in the interest of the BI provider, that PLAs are precise, that the BI solution can be tested against them and that it can be audited by third-party auditing agencies.

Most common and in particular such *outsourced BI* scenarios raise several important privacy-related challenges: i) precisely eliciting privacy requirements, ii) integrating privacy requirements from multiple data sources, iii) making requirements analysis robust to changes in the reports, and iv) enforcing and auditing privacy.

Eliciting sufficiently precise PLAs. This refers to the identification of privacy constraints the source whishes to impose by discussing privacy issues with data sources and customers. With the term "sufficiently precise" here we mean formal or semi-formal description of the PLAs that are unambiguous (so that developers know the implementation requirements), that are testable, and that are auditable.

PLA integration. This challenge is related to the integration of multiple privacy requirements from different sources and checking for their compliance in data transformations and reporting.

Robustness of the requirements. While ETL and data warehouse tend to be relatively stable, BI reports are in constant evolution. It is very common to add new reports or modify existing ones, especially in the period after the initial deployment.

Enforcing and auditing privacy. Once requirements (expressed as PLAs) are collected, we have to face the problem of how to implement a solution that i) enforces them and ii) supports monitoring and auditing to detect violations.

In this paper we focus on the elicitation and the precise modeling of robust requirements. These are cornerstone problems since the other challenges can be addressed only after requirements have been engineered. We experienced this to be one of the hardest aspects in any practical BI application we have developed.

3 Privacy Requirements Engineering at the Data Source

A typical way to specify privacy is by defining constraints on the data at the moment that the data is provided, i.e., having the privacy constraints defined at the source level. There are multiple ways one can achieve this goal. One is to leverage UML or some language among those used extensively in the requirements engineering community, for instance, i* [18]. These languages are expressive enough, but hard to use, and due to the fact that they have been to a large extend ignored by modeling techniques, their integration into a data management solutions is not an easy task. An alternative option is to model the PLAs in terms of meta-data that accompany the data and controls its access and use. The advantage of this approach is the metadata can be easily defined and can accompany the data throughout transformations [17]. The meta-data can be part of the data model, typically as data annotations [19][20].

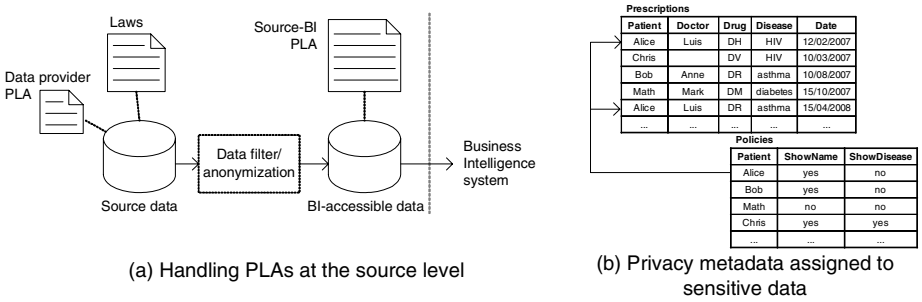

Fig. 2. PLAs at the data source level

A solution we are currently developing [21] is the use of intensional associations between data and metadata. The idea is that metadata is stored in completely different tables from the data. Since no modification is required on the data, no disruption of the existing systems functionality occurs. Furthermore, the privacy requirements data can be of any level of structural richness and complexity since it is not accessed by the existing data source applications. The association between the data and metadata is specified in the form of generic queries that serve as intentional descriptions. For instance, the aforementioned privacy restriction may be implemented by a query that selects among all the patients in the database those that have been diagnosed HIV-positive. The advantage is that if a new HIV patient is inserted in the database, for instance, his/her data is automatically associated to the aforementioned privacy restriction without any need for additional modification.

A different approach is the exploitation of the notion of views. In particular, to disallow access to the base tables but define views on top of them with different permissions and operators in each one. The use of views has the additional advantage that it can combine information that is distributed across different tables, thus defining privacy restrictions on the integrated information would have never been possible by defining restrictions on the individual base tables [5].

Alternatives or complements to the use of views as an access control mechanism include automatic *query rewriting* techniques, such as those found in commercial databases like Oracle Virtual Private Database (VPD) or in the Hippocratic Database (HDB) [2]. Apart from controlling the access to the data, the data delivered to BI providers may additionally undergo a *data anonymization* procedure that eliminates sensitive data that could be used to drill down from the provided data to the data of an actual individual (Figure 2(a)). Known anonymization techniques are those based on *k-anonymity* [12] or *l-diversity* [9].

An important issue that needs to be decided when privacy requirements are defined at the source level is how the privacy requirements are used. On one hand, the data source can restrict access to its data one when that access does not violate the privacy restrictions. In other words, the source is responsible for ensuring the PLA compliance. On the other hand, the source can expose to the BI provided all its data, but provide along with the data the PLA. In that case the BI provider will be responsible for the privacy enforcement, but the source will have no control over it. The choice depends on the level of trust to the BI provider. However, experience with real scenarios has shown that the decision is typically based on the IT skills at the data source, with smaller organizations always going for the first option.

A different challenge when defining privacy at the source level is to decide the level of privacy that is needed and to explain to the source owner the privacy requirements. Typically, the managers in charge of privacy are unaware of the details and the meaning of the data in the tables, something that is very often true even for the IT personnel. Furthermore, the schema may be too complex and may make difficult to understand which requirements to model exactly, and how. Furthermore, there is always the risk of "over-engineering" the requirements, i.e., while the source may have a large and complex database, the BI provider may only need a part of that information and for a limited use.

4 Privacy Requirements Engineering at the Warehouse

Instead of defining the privacy requirement at the source level, an alternative is to do so at the warehouse level, i.e., in terms of the data warehouse (DWH) schema and ETL operations. As shown in Figure 3(a), the implementation of PLAs at the DWH/ETL level occurs via meta-data in the DWH and annotations on the ETL procedures that feed the data warehouse. Typically, but not necessarily, before loading the actual warehouse and in order to reduce the complexity of ETL, data is extracted from the data sources and stored in a so-called *staging area* .

Metadata can also be used here to allow the specification of privacy restrictions over tables, rows, or fields, joins or aggregations. Techniques for modeling fine-grained authorizations in *data cubes* can also be used [14]. Restrictions on data disambiguation, correction, and cleaning procedures, can be expressed as annotations to the ETL flows, or to high level views of such flows. Figure 3(b) illustrates how PLAs associated with the ETL procedures can restrict the operations that are allowed on the source tables.

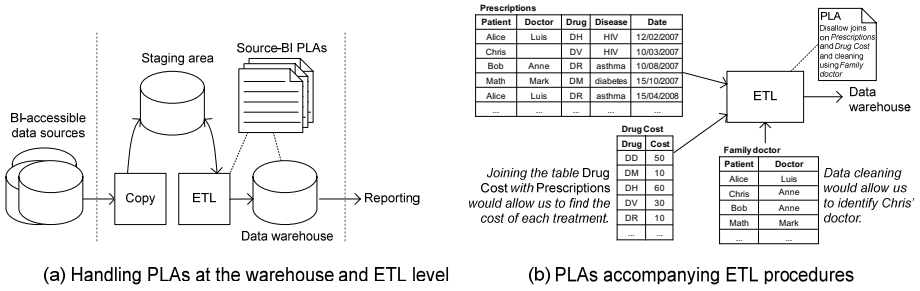

Fig. 3. PLAs at the DWH and ETL level

Specifying privacy at the DW level has certain advantages as opposed to specyfing it at the source. First, the schema is typically easier to understand, and second, the risk of over-engineering is reduced as the source owner can clearly see which data is used and in which form is stored. By having the restrictions directly on the warehouse data and the ETL, the source owners are guaranteed that the data warehouse schema, though which only their data can be exposed to others, is guaranteed to preserve the privacy. Furthermore, restrictions can be posed also on the ETL processes performed by the warehouse. This allows the BI provider to explain all the ways in which the user's information is used and explicitly ask for permission. The malicious approach of hiding these issues relying on the owners' lack of knowledge of the possible uses of the information does not pay in the long run.

Related work has focused mostly on privacy-preserving data integration [6][7], whose usage, in the case of ETL, for instance, may indeed be part of the requirements. Also, *data perturbation* may be used to modify the data in input, adding noise in such a way that the statistical distribution and the patterns of the input data are preserved and the quality of aggregate reports or mined results is not compromised, even if derived from altered data [13]. *Cryptographic* techniques can be used to scramble the data, again without compromising the possibility of computing aggregates or mining data [13].

Both the task of eliciting privacy requirements with the source owners and later testing PLAs once they have been agreed upon can be supported by *provenance* or *lineage* techniques, that capture the origins of data [17] and facilitate privacy and compliance management. Specifically, provenance traces the necessary meta-data required in compliance checking (i.e., auditing) to understand the data transformations. Widom focuses on lineage and uncertainty in Trio [15] and on non-annotation-based lineage for ETL transformations in [8]. The work described in [1] instead proposes an annotation-based approach to provenance in which elements at the sources (tables, rows, fields) are referred to by means of unique identifiers and provenance annotations (*where-provenance*) is propagated along transformations like copy, insert, and update, commonly used in *curated databases* (databases maintained via a large amount of manual labor). The previous works are not specific to the problem of privacy metadata, but nonetheless they provide techniques that could easily be adapted to our outsourced BI context.

A limitation of defining PLAs at the data warehouse level is that one needs to expose the data warehouse schema to the source owners. More than a confidentiality problem, the challenge here is that the data warehouse is the result of significant data processing and it may be difficult to present and explain to owners the meaning of the various terms – as in all integration problems one of the key challenges is understanding what the various fields mean. Furthermore, we notice that the problems discussed for modeling PLAs at the source level (e.g., complexity and over-engineering) are reduced, but yet not eliminated.

5 Privacy Requirements Engineering on Reports

As discussed, collecting privacy requirements in form of privacy metadata associated with either the source data or the warehouse data and ETL procedures demands significant expertise from the source owners. Engineering privacy requirements directly on the actual reports, instead, hides implementation details and allows the source owners to see exactly which information is shown to which user. It is therefore much easier for them to discuss and define PLAs as annotations on the reports themselves (Figure 4 (a)), typically in terms of which reports are allowed. We have experienced that fact that an interactive discussion of final reports with the data source owners enhances the mutual understanding and enables the BI provider to elicit a complete, precise, and easily testable and auditable set of privacy requirements. Testability is particularly important as source owners, auditors, and BI providers can (relatively) easily detect if development and executions are not compliant with the PLAs. Furthermore, there is no risk of over-engineering, i.e., only the PLAs that are actually needed are specified.

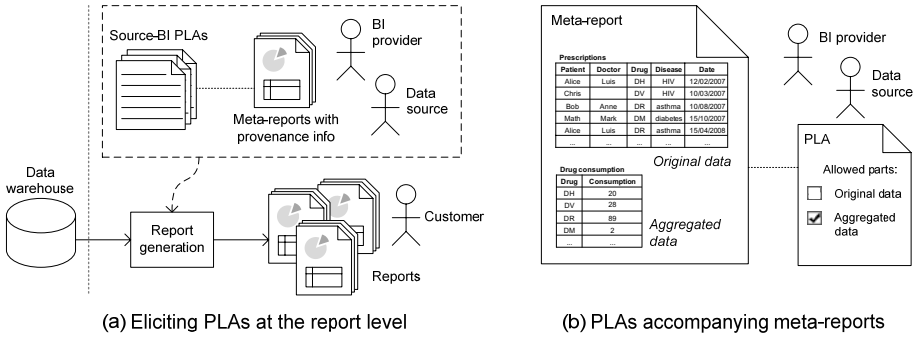

Fig. 4. PLAs at the report level

Defining privacy on the reports does not make us exempt from defining PLAs also based on how data is used during transformation. In addition, it is important to show on the reports where each report data item comes from, and what happens when the same data element can be obtained from multiple sources. The interaction between the BI provider and the data source can be assisted by a privacy requirements elicitation tool with a simple graphical user interface (GUI), which enables the BI provider to explain the provenance of each data element and the transformations/integrations it

goes through. Privacy requirements will then be collected and formalized directly in the tool by annotating reports and provenance schemes. An intuition of how meta-reports can be annotated is given in Figure 4(b). In general, annotations can include i) who can access a certain attribute, ii) what are the aggregation requirements on a table (how many base elements should be present before the aggregation), iii) anonymization requirements on an attribute, iv) join permissions/prohibitions, that is, the permission or prohibition to join information from multiple data sources (even belonging to the same owner) and v) integration permission, that is, the permission to use information to clean/resolve data from other owners. These requirements can be again expressed in intensional form, and in fact sometimes it is necessary to do so as they are instance specific. For example, a PLA may express that in a patient-related column, medical examinations results can be shown only for patients that are not HIV positive. HIV can be a separate column in the same report that is used only for purposes of defining PLAs, even if it is not made visible to users.

On the negative side, this approach has two main limitations: the first is that we need to share with the source owners the reports we deliver to users. The second is that the evolutionary nature of the reports themselves makes PLAs less stable. This is due to the fact that collected requirements are defined on each specific report, thus losing their validity with the evolution of the report. Furthermore, interactions and agreements with source owners are needed each time a new report is defined. This can be a significant limitation as the number of reports in a BI application is very high (having dozens or even hundreds of reports is common even in relatively small applications) and as some BI solutions also give users the ability to create new reports.

To overcome these drawbacks, we use *PLA meta-reports* in the discussion with the data sources, instead of concrete instances of individual reports. Meta-reports represent tables or views over the data warehouse that contain data that can be used to define reports. As depicted in Figure 4(a), we envision that the BI provider will discuss such meta-reports directly with the source owners. Meta-reports are also a subset of the actual reports. The idea is that they constitute an intermediate step between the complexity and stability of the data warehouse, and the simplicity and volatility of the final reports (Figure 5). Meta-reports typically contain wide tables that contain the same information used to populate the final reports. Notice that the meta-reports are intended to facilitate PLA definitions. In general they are not expected to be materialized or to be used as intermediate steps in the generation of the actual reports.

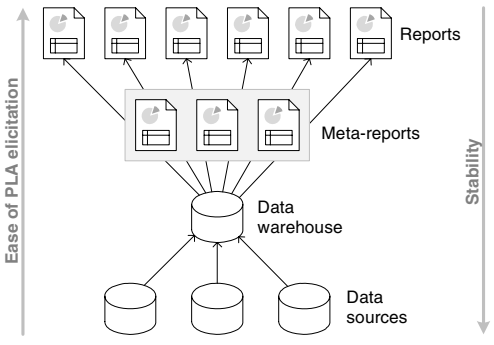

Fig. 5. PLA definition at different levels of abstraction

Once meta-reports are approved by the data sources they will be used not only as a reference for the implementation of privacy requirements compliant ETL procedures but also as a set of test cases on which the design of the cleaning and reporting activities could be tested before they are actually put in operation on the real data.

Each time a new report is created or an existing one is modified, PLAs on the meta-reports are used to determine if the new report is privacy-compliant. This can be often done easily as the reports can, at least conceptually, be expressed as a subset or view over a meta-report.

One of the main challenges in the development of meta-reports for the elicitation of privacy requirements is the identification and implementation of a minimal yet exhaustive set of meta-reports that is able to provide for the necessary flexibility to cope with a continuously changing set of final reports without requiring a new elicitation of requirements. It is further crucial to identify an adequate level of granularity for each of the meta-reports, so as to be able to elicit requirements that are precise enough to derive compliant reports from them, but still immediately understandable by the data source, in order to prevent misunderstandings. In other words, the design challenge here is how many meta-reports to define and how close they should be to the complexity of the data warehouse or the simplicity of the reports. At one extreme, the data warehouse can be viewed as a particularly complex case of meta-reports or universe, just like reporting tools allow the report universe to be the data warehouse itself. In fact, we can take this argument even further and observe that there is a continuum from the PLAs defined on the sources, data warehouse, meta-reports, and reports, going at increasing levels of simplicity and volatility of the PLA definitions.

6 Discussion and Comparisons of the Proposed Solutions

In this work we discussed various approaches to privacy requirements engineering. We described the problem of PLAs in outsourced BI applications, describing why the problem is important from a business perspective and challenging from a research point of view. We also emphasized the differences of this problem with respect to traditional privacy or access control problems. We believe this is still a research void, as we are not aware of systems in the BI arena where privacy policies are tested before they are put in operation in the system. Errors in capturing the intentions of the source owners and data providers with the definition and implementation of the privacy requirements are discovered only when the system is released and it is too late to avoid the disclosure of sensitive data. The problem is more and more pressing, though very often it is underestimated even by data owners, until the various issues and their complexity are made explicit to them.

The work is in its infancy, with a number of fascinating research challenges waiting to be addressed. These challenges range from defining languages and models for annotations and PLAs for BI applications, to identifying ways to support the generation of meta-reports, to defining methodologies for interacting with the source owners in order to quickly converge to a set of PLAs, to even methods for translating PLAs into internal data structures that can be used for automated privacy management support at design time or runtime.

References

1. Vansummeren, S., Cheney, J.: Recording Provenance for SQL Queries and Updates. IEEE Data Eng. Bull 30(4), 29–37 (2007)
2. Agrawal, R., Grandison, T., Johnson, C., Kiernan, J.: Enabling the 21st century health care information technology revolution. Commun. ACM 50(2), 34–42 (2007)
3. Anderson, H.: A comparison of two privacy policy languages: EPAL and XACML. In: SWS 2006, pp. 53–60. ACM Press, New York (2006)
4. Antón, I., Bertino, E., Li, N., Yu, T.: A roadmap for comprehensive online privacy policy management. Commun. ACM 50(7), 109–116 (2007)
5. Bertino, E., Sandhu, R.: Database security-concepts, approaches, and challenges. IEEE Transactions on Dependable and Secure Computing 02(1), 2–19 (2005)
6. Bhowmick, S.S., Gruenwald, L., Iwaihara, M., Chatvichienchai, S.: PRIVATE-IYE: A framework for privacy preserving data integration. In: ICDEW 2006, p. 91. IEEE, Los Alamitos (2006)
7. Clifton, C., Kantarcioğlu, M., Doan, A., Schadow, G., Vaidya, J., Elmagarmid, A., Suciu, D.: Privacy-preserving data integration and sharing. In: DMKD 2004, pp. 19–26. ACM Press, New York (2004)
8. Cui, Y., Widom, J.: Lineage tracing for general data warehouse transformations. The VLDB Journal 12(1), 471–480 (2003)
9. Machanavajjhala, J., Gehrke, D.K., Venkitasubramaniam, M.: l-diversity: Privacy beyond k-anonymity. ACM Trans. Knowl. Discov. Data 1(1), 1556–4681 (2007)
10. Ni, Q., Trombetta, A., Bertino, E., Lobo, J.: Privacy-aware role based access control. In: SACMAT 2007, pp. 41–50. ACM Press, New York (2007)
11. Rizzi, S., Abelló, A., Lechtenbörger, J., Trujillo, J.: Research in data warehouse modeling and design: dead or alive? In: DOLAP 2006, pp. 3–10. ACM Press, New York (2006)
12. Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression. Int. J. Uncertain. Fuzziness Knowl. -Based Syst. 10(5), 571–588 (2002)
13. Verykios, V.S., Bertino, E., Fovino, I.N., Provenza, L.P., Saygin, Y., Theodoridis, Y.: State-of-the-art in privacy preserving data mining. SIGMOD Rec. 33(1), 50–57 (2004)
14. Wang, L., Jajodia, S., Wijesekera, D.: Securing OLAP data cubes against privacy breaches. In: IEEE Symposium on Security and Privacy, pp. 161–175. IEEE, Los Alamitos (2004)
15. Widom, J.: Trio: A system for integrated management of data, accuracy, and lineage. In: CIDR 2005, pp. 262–276 (2005)
16. Wenning, R., Schunter, M. (eds.): The Platform for Privacy Preferences 1.1 (P3P1.1) Specification. W3C Working Group Note (November 2006), `http://www.w3.org/TR/P3P11/`
17. Tan, W.: Research Problems in Data Provenance. IEEE Data Engineering Bulletin 27(4), 45–52 (2004)
18. Dehousse, S., Liu, L., Faulkner, S., Kolp, M., Mouratidis, H.: Modeling Delegation through an i*-based Approach. In: IAT 2006, pp. 393–397 (2006)
19. Chiticariu, L., Tan, W.C., Vijayvargiya, G.: DBNotes: a post-it system for relational databases based on provenance. In: SIGMOD 2005, pp. 942–944 (2005)
20. Geerts, F., Kementsietsidis, A., Milano, D.: iMONDRIAN: A Visual Tool to Annotate and Query Scientific Databases. In: Ioannidis, Y., Scholl, M.H., Schmidt, J.W., Matthes, F., Hatzopoulos, M., Böhm, K., Kemper, A., Grust, T., Böhm, C. (eds.) EDBT 2006. LNCS, vol. 3896, pp. 1168–1171. Springer, Heidelberg (2006)
21. Srivastava, D., Velegrakis, Y.: Intensional associations between data and metadata. In: SIGMOD 2007, pp. 401–412 (2007)
22. Italian's Data Protection Code, DL n. 196/30 (June 2003)
23. European Directive 1995/46/EC, OJ L 281, p. 31 of 23.11.1995