# Bottom up approach to manage data privacy policy through the front end filter paradigm

# Bottom up approach to manage data privacy policy through the front end filter paradigm.

Gerardo Canfora, Elisa Costante, Igino Pennino, Corrado Aaron Visaggio

*Research Centre on Software Technology*
*University of Sannio – 82100 Benevento*

## Abstract

An increasing number of business services for private companies and citizens are accomplished trough the web and mobile devices. Such a scenario is characterized by high dynamism and untrustworthiness, as a large number of applications exchange different kinds of data. This poses an urgent need for effective means in preserving data privacy. This paper proposes an approach, inspired to the front-end trust filter paradigm, to manage data privacy in a very flexible way. Preliminary experimentation suggests that the solution could be a promising path to follow for web-based transactions which will be very widespread in the next future.

**Keywords:** data privacy, front end trust filter.

## Introduction

The number and the complexity of processes which are accomplished throughout the web are increasing. Confidential data are more exposed to be collected lawlessly by humans, devices or software. The actors involved are often autonomous systems with a high degree of dynamism [15]; negotiations are performed among multiple actors, and cross the boundaries of a single organization [10]. As a consequence, privacy of personal and confidential data is exposed to several threats [13].

Different technologies have been ideated in order to face this problem, such as: anonymization [16], fine grain access control (FGAC) [2], data randomization and perturbation [9].

These solutions show some limitations when applied in contexts characterized by high dynamism and a few opportunities to control data exchange: they are scarcely scalable, they cannot be used in untrustworthy transactions, or they propose too invasive data access mechanisms, which hinder flexibility. This discussion is largely achieved in the related work section.

The realization and the adaptation of a data privacy policy is a process of transformation, which spans from the definition of strategies to properly protect data up to the design of a supporting technology which implements the established policies. Such process includes three main stages. At a first stage, a data privacy policy is described in natural language in a document which contains the rules to disclose sensitive data. At a second stage, the general policy must be refined in specific strategies, in order to understand which kinds of actions could be performed on certain categories of data by some categories of users, and under which conditions. Finally, the established strategies need to be implemented with a suitable technology ensuring that accesses to the data repository happen accordingly with the strategy.

This paper proposes a three-layered approach which aims at facilitating the management of data privacy in such a scenario.

The main purpose is to provide the data manager with the capabilities of:

1. translating the privacy policies, expressed in a natural language, into low-level protection rules, directly defined on database fields;
2. providing the database with an adaptive protection, which is able to change accordingly to: (i) the current state of the database, and (ii) to the knowledge that the user acquires by aggregating the information achieved throughout the submitted queries over time.

The paper proceeds as follows: in the next section related work is discussed; in the following section the solution is presented. Thus, the results of the experimentation are provided; and, finally, conclusions are drawn.

## Related Work

Different technologies have been proposed to preserve data privacy, but some of them, which could be properly adopted in many contexts, could be scarcely effective in highly dynamic systems.

The W3C Consortium developed P3P [17]. It provides a method that permits a Web Site to codify within an XML file the purposes for which data are collected. It is based on confronting privacy preferences between information provider and requester. P3P is used by different web browsers and lets web site to express the privacy policy with a standard structure: the server according to this structure can choose if deliver data or not. P3P synthesizes the purposes, treatment modes and retention period for data, but it does not guarantee that data are used accordingly to the declared policies. Consequently it may be successful in trusted environments.

Researchers of IBM proposed the model of Hippocratic database[1]: it supports the management of information sharing with third parties. It establishes ten rules for exchanging data; relying on these rules, queries are re-written, data are obfuscated and cryptography is in place, when needed. Hippocratic databases use metadata for designing an automatic model for privacy policy enhancement, named Privacy Metadata Schema. This technique degrades performances, as at each steps purposes and user authorization must be checked at each transaction. Memory occupation is a further matter, as the metadata could grow up fast.

The fine grain access control (FGAC)[2], is a mechanism designed for a complete integration with the overall system infrastructure. Constructs which implement this method must: (i) assure that access strategies are hidden to users; (ii) minimize the complexity of policies; and (iii) guarantee the access to tables' rows, columns, or fields. Traditional implementation of FGAC use static views. This kind of solution could be used only when constraints on data are few.

Further solutions, like EPAL [3] and the one proposed in [14], allow actors of a transaction to exchange services and information within a trusted context. The trust is verified throughout the exchange of credentials or the verifications of permissions to perform a certain action.

Anonymization techniques let organizations to retain sensible information, by changing values of specific table's fields. The underlying idea is to make data undistinguishable, as happens in the k-anonymity algorithm [16], throughout the perturbation of values within records. Another techniques require to make data less specific, as happens in the generalization [5]. This technique affects seriously data quality and may leave the released data set in vulnerable states.

Further mechanisms of data randomization and perturbation [9] hinder the retrieval of information at individual level. These techniques are difficult to implement, as they are based on complex mathematics, and however are invasive both for data and applications.

Cryptography is the most widespread technique for securing data exchange [8], even if it shows some limitations: high costs for governing distribution of keys, and low performances in complex and multi-users transactions.

## Definitions

For a better understanding of this work it is necessary to give the following definitions:

- A **privacy policy** defines the sensitive data whose access must be denied; it is captured as a set of purposes.
- A **protection rule (pr)** defines if the result set can be disclosed (**Legal rule**) or not (**Illegal rule**). For example, let's consider the following rules:
    a. **NO SELECT** Fiscal_Code, Surname **FROM** Person;
    b. **SELECT** Age, Zip_Code **FROM** Person.

The rule (a) is not legal and establishes that the couple *Fiscal_Code – Surname* cannot be disclosed. Otherwise it doesn't explicitly deny the access to the single attributes. Vice versa the rule (b) makes attributes *Age* and *Zip_Code* of the table *Person* accessible whether in pairs or singularly.

The **state of the database** is time dependent and it is defined by the informative content of the database. It can be modified by means of insert, delete and alter operations. Depending on the database state, the privacy policy could be enforced or made less restrictive, as vulnerabilities and threats to privacy preservation could rise or disappear.

### Approach

Two complementary approaches could be followed in order to meet the goal:

- Top-down, that derives a set of protection rules by the privacy strategy.
- Bottom-up, that allows the rules definition from the analysis of vulnerability and the aggregation inference.

The system acts like a filter between the user applications interrogating the database to protect and the database itself, captures the submitted queries, compares them with the protection rules and decide if they are to allow or to block. The *Top-down* approach suffers a major weakness: the rules can be eluded, by exploiting specific vulnerabilities of the database or, more simply, taking advantage of the flexibility of SQL that allows to write a single query in a lot of ways. Moreover, the growth of the user's knowledge can entail the generation of new protection rules. The goal of the *bottom-up* approach is to solve these problems.

## *Query Filtering*

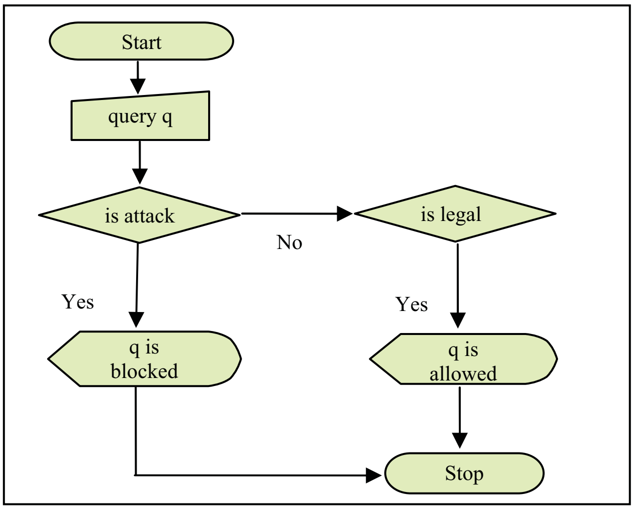


Figure 1 – **Filtering algorithm**

The goal of the filtering is to establish if a query is:

- *Legal* (to allow), when it doesn't disclose sensitive information;
- *Illegal* (to block) when it tries to access to protected data.

To make this possible it is necessary to evaluate if the handed out query (q, from here on) matches with a protection rule (pr, from here on).

As showed in figure 1, the filtering process can be divided in three steps:

1. Query submission;
2. Search for a pr, belonging to the **illegal catalog,** which matches the q; if a correspondence is found, the query is blocked, otherwise it is proceeded with step 3;
3. Search for a matching of the q with a pr belonging to the **legal catalog**; if a correspondence occurs, the query is forwarded to the database.

In order to recognize the correspondence between a query and a rule, the algorithm *Result Matching* has been formulated. Such a comparison is based on the interrogation result rather than on the syntax used to write it. By this way it's guaranteed that more queries expressed in different ways and disclosing the same data, are considered equivalent and thus blocked, as well. When a user submits a query, the system evaluates if at least one rule that involves the same tables of the query exists; and than forwards the found rules and the query to the database, capture the result set of the rules and the query , and, finally, compare them.

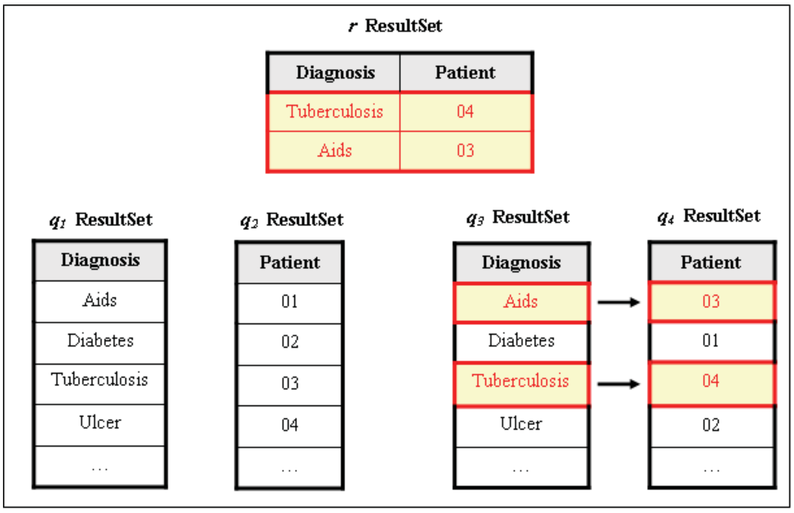
## *Analysis of Acquired knowledge*

If there is no matching between the query and the set of rules, the system must establish if the obtained result set can be disclosed on the base of the information already released. In order to make this decision, the system must estimate if the aggregation of the information that the user has acquired through the previous queries and the information released with the last query violate the established privacy policy. As a matter of fact, a sensitive information can be often composed by more information with a less sensibility degree. For instance, let's consider the following illegal rule, which denies the spreading of the information about which patients are affected by Aids or Tuberculosis:

- **NO SELECT** Diagnosis, Patient **FROM** Illness **WHERE** Diagnosis = 'Aids' **OR** Diagnosis = 'Tuberculosis'; (*r*)

and the submission of this two different queries combinations:

- { (*$q_1$*) **SELECT** Diagnosis **FROM** Illness; (*$q_2$*) **SELECT** Patient **FROM** Illness; }
- { (*$q_3$*) **SELECT** Diagnosis **FROM** Illness **ORDER BY** Diagnosis; (*$q_4$*) **SELECT** Patient **FROM** Illness **ORDER BY** Diagnosis; }



*r* **ResultSet**

| Diagnosis | Patient |
|---|---|
| Tuberculosis | 04 |
| Aids | 03 |

*$q_1$* **ResultSet**

| Diagnosis |
|---|
| Aids |
| Diabetes |
| Tuberculosis |
| Ulcer |
| … |

*$q_2$* **ResultSet**

| Patient |
|---|
| 01 |
| 02 |
| 03 |
| 04 |
| … |

*$q_3$* **ResultSet** → *$q_4$* **ResultSet**

| Diagnosis | Patient |
|---|---|
| Aids | 03 |
| Diabetes | 01 |
| Tuberculosis | 04 |
| Ulcer | 02 |
| … | … |

Figure 2 – **Possible Resultset Aggregation**

As showed in figure 2, the combination of *$q_3$* and *$q_4$* is more dangerous than the combination of *$q_1$* e *$q_2$*. The latter, in fact, allows to match the patient's id to his illness, because the result sets are ordered by the same criteria. Conversely, *$q_1$* and *$q_2$* do not expose any sorting rationales.

### *Log & Quarantine*

The knowledge given by $q_4$ is harmful only if the information released by $q_3$ has been already obtained and vice versa. That means that it is not compulsory to block both queries to avoid the violation of the rule *r*, but it is enough to block only the last submitted one. It is necessary to track the history of user's interrogations over time in order to get a complete picture of the overall knowledge acquired by the user. However, all the queries forwarded to the database will be logged in a file together with the corresponding information about their success achieved or missed.

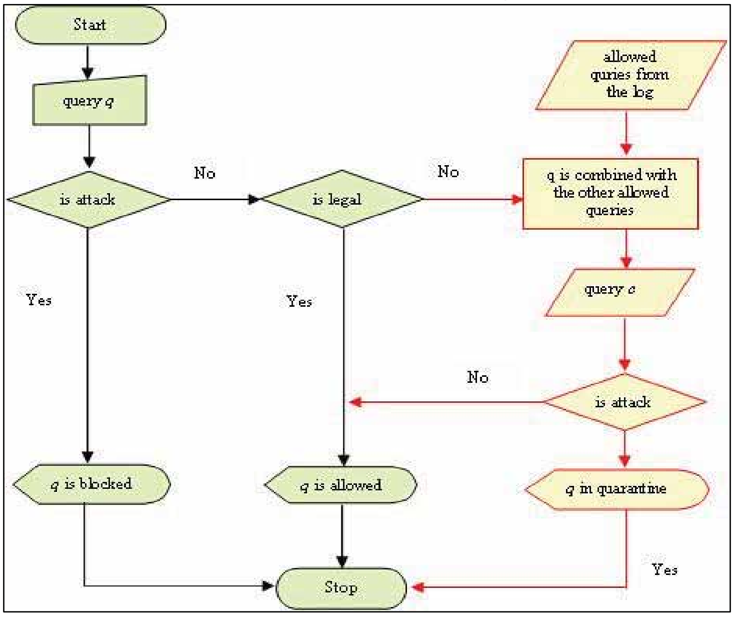


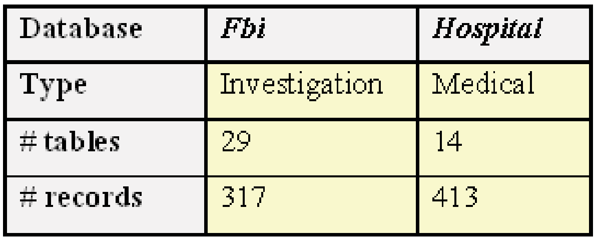
Figure 3 – **Complete filtering algorithm**

When a query is submitted, if it does not match with any rule, i.e. it does not belong to the illegal catalog, the system evaluates if it can disclose sensitive information and alerts the administrator. To do this, the system combines the current query with the previous allowed ones (described in the log file), formulating a new query that represents the aggregation. If this query is not matched with an illegal rule, the current user query is allowed, otherwise it is suspended in a *quarantine* status. The whole filtering algorithm is described in figure 3.

The administrator can decide for each single suspended query if it is to block or to allow in the future, generating a new protection rule. An "in-vitro" experimentation has been carried out, in order to validate the approach, whose outcomes are encouraging and stimulated new directions for future research: the next steps consists of realizing a system for modeling the data domain from a privacy preservation perspective and a system to capture the knowledge acquired by each user over time, in order to limit exploits based on the inference.

## Experimentation

The experimentation aims at evaluating the approach's effectiveness, in order to estimate the robustness of the data protection offered, as the semantic flexibility of SQL could let cheating the adopted mechanisms to preserve data privacy; moreover, the experimentation is headed to estimate the performances degradation of the system, in terms of response time, while the catalogued rules' set grows up.



| Database | *Fbi* | *Hospital* |
|---|---|---|
| Type | Investigation | Medical |
| # tables | 29 | 14 |
| # records | 317 | 413 |

Figure 4- **Experimental Vitro**

The figure 4 shows the databases used as experimental vitro.

In order to test the effectiveness of the *Result Matching* algorithm, an experiment has been realized, which consisted of evaluating the percentage of blocked queries –which is expected to be the 100%- within a set of forwarded queries to the target database.

For each database have been formulated:

- 4 rules on 1 attribute of 1 table;
- 4 rules on 2 attributes of 1 table;
- 4 rules on 4 attributes of 1 table;
- 4 rules on 2 attributes of 2 tables.

For each rule, 4 equivalent queries have been written and their effects have been observed. The matching algorithm proved to be well-built and particularly effective to face up to SQL flexibility. As a matter of fact the algorithm successfully achieved blocking the overall set of queries.

The second part of the experiment helped analyze how the performances of the solution changed, in terms of response time, with correspondence to an increasing of both the rules' number and of the *Resultset* size.

It is important to recall that to make possible the result matching it is necessary to submit to the database both the query to analyze and the rules' set against which the query is confronted, as not all the rules in the catalogue are involved when filtering a query.

In order to carry out a more consistent experiment, rules that involve the same tables of the query have been formulated and catalogued.

The following queries, with a growing number of attributes (and so *Resultset* size), is analyzed:

- Query1: **SELECT** fiscal_code **FROM** person
- Query2: **SELECT** fiscal_code, name **FROM** person
- Query3: **SELECT** fiscal_code, name, surname **FROM** person
- Query4: **SELECT** fiscal_code, name, surname, birth_place **FROM** person
- Query5: **SELECT** fiscal_code, name, surname, birth_place, nationality **FROM** person

All the protection rules refer to the table PERSON, that has the following schema:

- **PERSON**(<u>fiscal_code</u>, name, surname, sex, birth_place, nationality)

For each query, the response times have been measured with correspondence to a catalogue with, respectively, 10 (5 legal and 5 illegal), 50 (25 legal and 25 illegal) and 80 (40 legal and 40 illegal) rules. Consider that all the queries were allowed with exception of Query4 and Query5 that were blocked when the catalogues containing 50 and 80 protection rules were used.

The following graphs show the obtained results. It's possible to observe that Query1, Query2 and Query3 have the same trend, that is: the response time increases with the growth of the catalogued rules' number, because they produced the same outcomes, namely they are allowed at all.

As expected, the performances seem to decrease proportionally with the growth of the catalogue's size, but the proportional factor could be not equal to one. In fact,

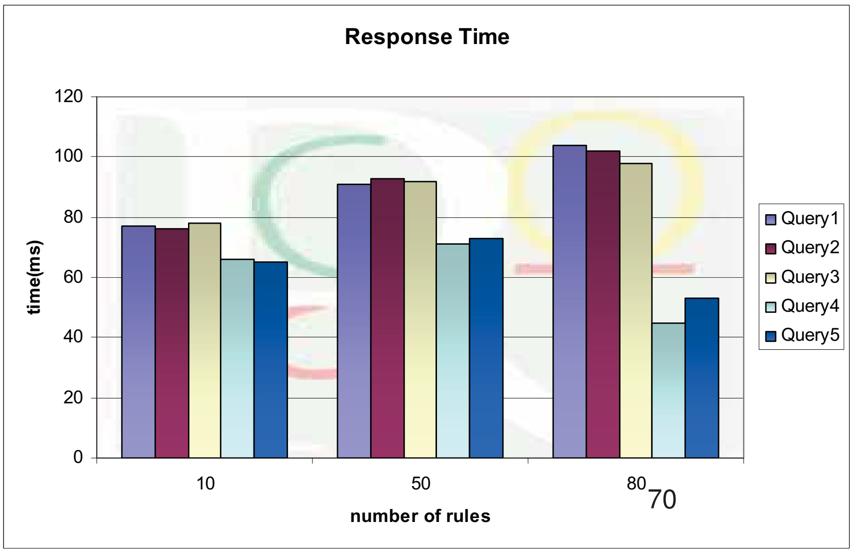


Figure 4 – Response time corrispong to an increasng of the rules' set

corresponding to a 500%growth of the rule's number, it was recorder a 25% increment of the response time. Moreover, corresponding to an 800% growth of the rules' number, it was observed a 40% increment of the response time.

Concerning Query4 and Query5, it's possible to observe a different behavior: the response times are smaller then the previous ones, because they match with an illegal rule. This means that there is a fewer number of comparisons to accomplish.

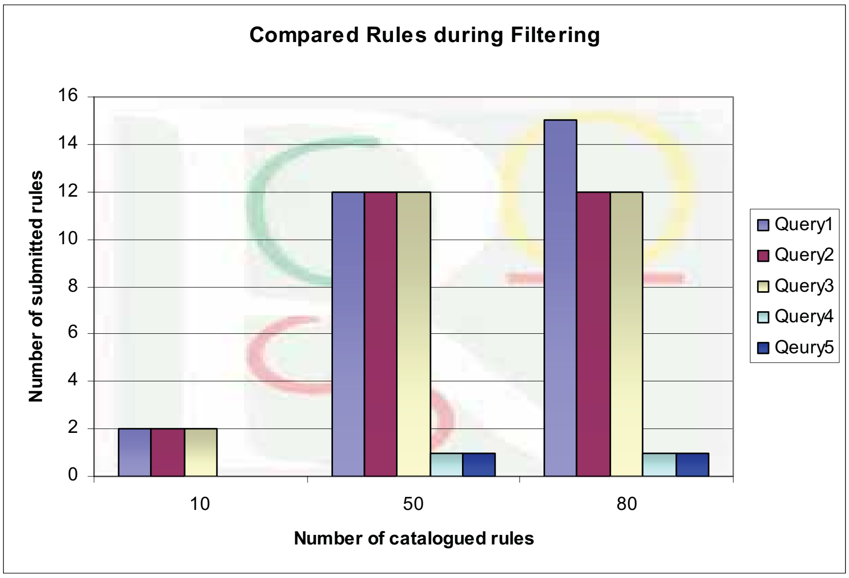


Figure 5 – Effectivly compared rules

As well as observed in figure 5, when the catalogue counted 80 rules, only 15 were actually used for comparison, which means that in the worst case less then 20% rules out of the catalogue size are effectively considered in the analysis.

## Conclusion

With the growing migration of services towards the net, privacy should be managed within environments characterized by high dynamism: multiple applications are able to access different data sources, without having in place trust-based mechanisms.

As such scenarios foresee a high scalability and a loose control, existing solutions for data privacy management could be unfeasible, too costly or scarcely successful.

This work introduces a novel approach to data privacy, inspired to the paradigm of front end trust filtering. According to this approach the data privacy is managed in a way which aims at reducing control on transactions exchanging data set, while keeping a high level of robustness in preserving data privacy.

The proposed solution implements a bottom-up approach, which relies on the comparison of the result set produced by the forwarded query and the one containing the information which should be banned, accordingly to the established privacy policy.

Furthermore, this solution helps discover new queries which could menace the privacy of data, but are not included in the catalogue' rules, throughout the quarantine management policy.

A preliminary experimentation was carried out in order to prove the effectiveness and the efficacy of the approach. It emerged that the system is able to successfully face the semantic flexibility of the SQL, and the degradation of performances with the growing of rules' number is limited to the 20% for the worst case.

As future work we are planning a larger experimentation in order to detect further weakness points of the solution and identify improvement opportunities.

## References

[1]. Agrawal R., Kiernan,J., Srikant R., and Xu Y., 2002, Hippocratic databases. In *VLDB*, the 28th Int'l Conference on Very Large Database.

[2]. Agrawal R., Bird P., Grandison T., Kiernan J., Logan S., Rjaibt W., 2005 Extending Relational Database Systems to Automatically Enforce Privacy Policies. In *ICDE'05 Int'l Conference on Data Engineering*, IEEE Computer Society.

[3]. Ashley P., Hada S., Karjoth G., Powers C., Schunter M., 2003. Enterprise Privacy Authorization Language (EPAL 1.1). *IBM Reserach Repor*t. (available at: http://www.zurich.ibm.com/security/enterprice-privacy/epal – last access on 19.02.07)

[4]. Bayardo R.J., and Srikant R., 2003. Technology Solutions for Protecting Privacy. In *Computer*. IEEE Computer Society.

[5]. Fung C.M:, Wang K., and Yu S.P., 2005. Top-Down Specialization for information and Privacy Preservation. In *ICDE'05, 21st International Conference on Data Engineering*. IEEE Computer Society.

[6]. Langheinrich M.,2005. Personal privacy in ubiquitous computing –Tools and System Support. *PhD. Dissertation, ETH Zurich*.

[7]. Machanavajjhala A., Gehrke J., and Kifer D., 2006. l-Diversity: Privacy Beyond k-Anonymity. In *ICDE'06 22nd Int'l Conference on Data Engineering* . IEEE Computer Society.

[8]. Maurer U., 2004. The role of Cryptography in Database Security. In *SIGMOD, int'l conference on Management of Data*. ACM.

[9]. Muralidhar, K., Parsa, R., and Sarathy R. 1999. A General Additive Data Perturbation Method for Database Security. In *Management Science, Vol. 45, No. 10*.

[10]. Northrop L., 2006. Ultra-Large-Scale System. The software Challenge of the Future. *SEI Carnegie Mellon University Report* (available at http://www.sei.cmu.edu/uls/ – last access on 19.02.07).

[11]. Oberholzer H.J.G., and Olivier M.S., 2005, Privacy Contracts as an Extension of Privacy Policy. In *ICDE'05*, *21st Int'l Conference on Data Engineering*. IEEE Computer Society.

[12]. Pfleeger C.R., and Pfleeger S.L., 2002. *Security in Computing*. Prentice Hall.

[13]. Sackman S., Struker J., and Accorsi R., 2006. Personalization in Privacy-Aware Highly dynamic Systems. *Communications of the ACM, Vol. 49 No.9*.ACM.

[14]. Squicciarini A., Bertino E., Ferrari E., Ray I., 2006 Achieving Privacy in Trust Negotiations with an Ontology-Based Approach. In *IEEE Transactions on Dependable and Secure Computing,* IEEE CS.

[15]. Subirana B., and Bain M., 2006. Legal Programming. In *Communications of the ACM, Vol. 49 No.9.* ACM.

[16]. Sweeney L., 2002. k-Anonymity: A model for Protecting Privacy. In *International Journal on Uncertainty, Fuzziness and Knowledge Based Systems, 10*..

[17]. Platform for Privacy Preferences (P3P) Project, W3C, http://www.w3.org/P3P/ (last access on January 2007).