Security of collaborative processes in large data sets applications

Ion IVAN, Cristian CIUREA, Sorin PAVEL, Mihai DOINEA
Informatics Economics Department
Academy of Economic Studies, Bucharest, Romania
ionivan@ase.ro, cristian.ciurea@ie.ase.ro, pavelsorin@gmail.com, mihai.doinea@ie.ase.ro

Abstract: *Collaborative processes oriented on large data sets are presented. Security requirements are defined. Metrics for collaborative processes, for working with large data sets and for minimization of the effects generated by vulnerabilities resulted from the use of distributed applications are proposed. Specific risk factors are identified for the interactions developed for collaborative processes. For each one of the identified risks, solutions are found for keeping an acceptable level of security. Architecture is proposed for optimization of collaborative processes security level. The behaviour of large data sets applications, in conjecture with the security framework, is analyzed for collaborative banking system flows.*
Keywords: *collaborative, security, large data sets, metrics, optimization, modelling.*

1. Collaborative processes

A collaborative process means all the actions performed by several agents in a distributed environment, in order to achieve a common goal. The collaborative process is defined as a cybernetic system, containing inputs, processes and outputs. This representation of the collaborative process is shown in figure 1:

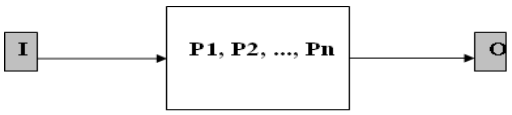

Figure 1. Collaborative process representation, Ciurea (2010)

Collaborative processes require the existence of such activities that need to be automated to streamline the workflow within an organization. In collaborative processes within a bank, any change in the workflow must be found in the corresponding rules and procedures.
Daily transactions taking place in a bank conduct to generate very large data sets. For a transfer between two accounts opened within the same bank, in the banking information system appears two entries, namely one for debiting the payer account and the other for crediting the beneficiary account.
Table 1 shows a very large set of transactions, the first two entries representing the transfer between own accounts.

Table 1. Very large set of transactions

No.	Recording date	Processing date	Transaction code	Details	Amount
1	30-07-2010	30-07-2010	74	transfer	1000.00-
2	30-07-2010	30-07-2010	89	transfer	1000.00
3	15-09-2010	16-09-2010	55	electronic payment	345.67
...	...	...	...	...	...
7000	19-09-2010	19-09-2010	89	tax payment	420.00

Electronic payment services offered by the bank to the customers are collaborative processes because they involve automatic execution of millions of transactions per minute. The great corporate customers of the banks transmit daily files of thousands of payments to be processed automatically. The business process regarding the acquisition of electronic payment service by a customer of a bank is a collaborative process. In describing the business process are used two participants who transmit messages between them, respectively the customer and the bank. Within the bank, there are several departments involved, namely the agency/branch, technical assistance, server administration and customer intervention.
The process of buying the electronic payment service of a bank consists of the following activities:

A_1 – requesting for purchase an electronic payment application;
A_2 – signing and stamping the contract;
A_3 – receiving the request;
A_4 – delivering the contract;
A_5 – completing the annex for customer enrolment;
A_6 – sending the annex by email;
A_7 – receiving the annex;
A_8 – enrolling the customer on the server;
A_9 – scheduling the intervention for installation;
A_{10} – notifying the date for installing the application;
A_{11} – receiving the notification;
A_{12} – sending installation details;
A_{13} – receiving installation details;
A_{14} – confirming availability;
A_{15} – receiving confirmation;
A_{16} – supplying installation details;
A_{17} – providing communication file;
A_{18} – receiving intervention details;
A_{19} – preparing the installation;
A_{20} – installation and training;
A_{21} – preparing the intervention report;
A_{22} – sending the report to the bank;
A_{23} – receiving the report;
A_{24} – completing the installation;
A_{25} – using the application.

Figure 2 shows the BPMN diagram for the acquisition process of the electronic payment service:

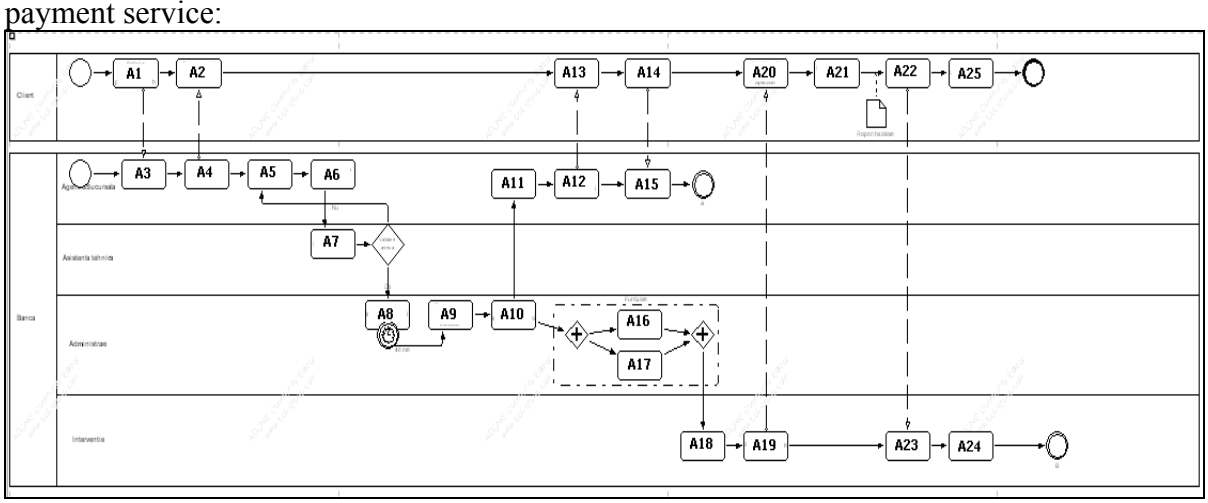

Figure 2. BPMN Acquisition Diagram

Table 2 shows the dependency matrix of 15 activities of the business process regarding the acquisition of the electronic payment service:

Table 2. Dependency matrix between activities

	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	A_{14}	A_{15}
A_1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
A_2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
A_3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
A_4	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
A_5	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0
A_6	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
A_7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
A_8	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
A_9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
A_{10}	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
A_{11}	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
A_{12}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
A_{13}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
A_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
A_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The m_{ij} element of the dependency matrix has the value 1 if the activity A_j is dependent on the activity A_i and the value 0 if the activity on the column j is not dependent on the activity on lines i.

If the dependency matrix for the selected business process has less than 30% non-zero items, this matrix is considered a sparse matrix and is represented using two structure vectors, one that contains the line positions and the other the positions of columns where are the values 1.

To determine whether the dependencies matrix between activities is a sparse matrix, is calculated the load degree, GI, as the ratio between the number of non-zero elements and the total number of matrix elements:

$$GI = \frac{NEN}{NTE},$$

where:

NEN – the number of non-zero elements;

NTE – the total number of matrix elements.

In this case, the dependency matrix between activities presented in table 2 has a loading degree of:

$$GI = \frac{NEN}{NTE} = \frac{18}{225} = 0.08.$$

As $GI = 8\% => GI < 30\%$ meaning that, the dependencies matrix between activities, is considered sparse matrix.

Changing the dependencies between activities leads to changes in matrix elements, which assure a dynamic characteristic of the matrix. According to the representation of figure 2, activity A20 - application installation and usage training is directly dependent on the A14 - confirm availability. If the bank made the intervention to the customer without the customer to confirm his availability at time of intervention, then the dependence between the two activities disappears. Changes in dependencies

between activities conduct to changes of elements inside the dependency matrix namely influencing the sparse character of the matrix.
For the selected business process, there are a number of activities that need to be automated to streamline the workflow, namely schedule the intervention for installation and notify the date for installing the application. Programming the intervention for installation is done by checking a list of interventions and selection by the server administrator of convenient dates and times. Automate this activity involves developing an application to keep track of installations and to automatically schedule the customer at a specific date, immediately after he was enrolled on the server. Also, the application automatically forward an email to notify the person in charge from the agency which has sent the customer enrolment annex.
In a bank are taking place daily around 2 million transactions consisting in different types of operations like:

- transfers between existing accounts;
- opening new accounts;
- realization or liquidation of deposits;
- according loans;
- foreign exchanges;
- payments to state budget;
- payments to customs;
- direct payments to suppliers;
- other operations.

Table 3 shows an analysis of the types of operations performed in one day inside a commercial bank.

Table 3. Analysis of types of banking operations

No.	Transaction type	Percentage of total operations	Commission
1	transfers between existing accounts	35%	NO
2	opening new accounts	5%	NO
3	realization or liquidation of deposits	10%	NO
4	according loans	3%	YES
5	foreign exchanges	7%	NO
6	payments to state budget	5%	YES
7	payments to customs	2%	YES
8	direct payments to suppliers	23%	YES
9	other operations	10%	YES

As shown in table 3, the bank charge fees for certain types of operations. Thus, the bank profit results from according loans, payments to state budget, payments to customs, direct payments to suppliers and other operations. Although intrabanking transfers between existing accounts, have a significant share in total of transactions, the bank not obtain profit from them. Most risky operations are those with great share, such as transfers between existing accounts and direct payments to suppliers.
These transactions require the existence of an advanced database management system and an integrated computer system. Electronic transactions that take place in a bank are saved in databases and are never deleted. Each bank has well tuned procedures for

backup and disaster recovery, to avoid the loss of database records, even for natural disasters events.

For the daily transactions conducted in a bank, the characteristics target the workstation, the client account, the beneficiary account and the transaction value. Transactions are sorted by value. It is determined the frequencies of the traded amount values and the highest frequencies are chosen. It is verified if the transactions were made from the same workstation. The persons who operated the trading are determined. The traded amount beneficiaries are determined. From the analysis, abnormal situations arise, that is all transactions were made to a certain destination, or all transactions were made from the same workstation. An explanation of these abnormal situations is sought to prevent any fraud attempts.

If an incorrect transaction on a client account is made, meaning that it has made a payment to another beneficiary than the correct one or it transferred a wrong amount of money, then the payment reversal is carried out and a new transaction account is registered. Once registered a transaction on an account, it is no longer clear. The payment reversal requires crediting the customer's account with the equivalent payment, a situation which leads to two entries in the database, the one related to the payment and other related to the payment reversal. This working method has advantages such as keeping track of all transactions carried out on an account. Information regarding banking transactions are increasing their value as aging. Banks realized this opportunity and charges for the availability of the old customer transactions for an account. If a customer requires a proof of one payment from his account, and the payment was made three years ago, the bank offers to customer the account statement for the required payment day, for an extra fee.

Table 4 presents a matrix of transactions, which contains on the lines the types of operations and on the columns the time moments at which these operations took place.

Tabel 4. Banking transaction matrix

Operation / Period	$\mathbf{Y_1}$	$\mathbf{Y_2}$	**...**	$\mathbf{Y_j}$	**...**	$\mathbf{Y_m}$
$\mathbf{X_1}$	f_{11}	f_{12}	...	f_{1j}	...	f_{1m}
$\mathbf{X_2}$	f_{21}	f_{22}	...	f_{2j}	...	f_{2m}
...	...	...	...	...	...	...
$\mathbf{X_i}$	f_{i1}	f_{i2}	...	$\mathbf{f_{ij}}$	...	f_{im}
...	...	...	...	...	...	...
$\mathbf{X_n}$	f_{n1}	f_{n2}	...	f_{nj}	...	f_{nm}

The frequency f_{ij} of occurrence of a certain type of operation at a certain time moment is determined, so that at every time the bank to know the number of operations carried out and detailed on each type.

There is an amount limit of 50.000 EUR for foreign transfers which are reported to National Bank. For all transactions exceeding this limit, customers must submit to the bank a statement, containing the payment details and the beneficiary identification elements. Customers, who wish to avoid filing such statements, as they do not want to reveal the destination of money, will perform several operations with the maximum amount of 49.999 EUR. All external transfers between 49.000 and 49.999 EUR amounts are tracked by the bank in order to identify any attempts of fraud or money laundering.

Identification of customers who make such transfers is made by sorting the transactions from a certain period of time by name and amount. Table 5 presents a

situation of operations with amounts between 49.000 and 49.999 EUR conducted during one week.

Table 5. Operations with high risk degree

No.	Customer name	Date	Amount	Beneficiary
1	A	13-09-2010	49.999 EUR	W
2	A	14-09-2010	49.999 EUR	W
3	A	15-09-2010	49.999 EUR	Z
4	B	16-09-2010	44.000 EUR	Q
...	…	...	...	…
1000	C	17-09-2010	41.200 EUR	T

From table 5 is shown that the customer A has made in two consecutive days transfers of 49.999 EUR to the beneficiary W, which cause the bank to carry out investigations regarding the destination of money and purpose.

When a customer requests preferential discounts and commissions for conducting operations, the bank analyses the history of the customer transactions and the monthly transactions volume. With complete databases related to all transactions conducted by all customers, the bank examines the customer's money turnover and decides whether to grant preferential discounts and commissions.

Databases with transactions performed in a bank contains information about the user who performed the operation, the channel through was done, from which workstation, in which date and which hour. These databases are updated in real time and are consulted by the Banking Security Department to discover any fraud attempts. If you find that, from a workstation, an operator makes a lot of transactions compared to other operators, or amounts transferred are very high, then it is done thorough research regarding these operations.

2. Very large data sets

Banking transactions represent one of the modern virtual frameworks that involve many simultaneous users, very large data – $10^7 \div 10^{10}$ sets – and applications for data management, Rao (2008). Due to the large quantities of data sets to be processed, applications acquire specific properties and functionalities.

Very large data collections used in banking represent, but are not limited to: databases, collections of text files/XML files/multimedia, data warehouse, or any combination thereof. Administration, Ivan (2008), requires specialized tools to harmonize the specific hardware and software aspects of large data sets.

The size or volume of database is the quantitative expression of corporate data. In the practice of handling databases and files, or in human-computer interaction, data volume is understood as:

- length of a database file or the aggregate length of a collection of files in number of articles/records, or in physical space occupied expressed in bytes;
- number of documents placed in a file or database;
- number of transactions;
- processing time.

Each dimension expression is limited and reduces the ability to operate if it is used by itself. When volume data is expressed as a number of articles or records, information is missing about the basic element, namely the structure of the article or record.

Creating a very large database involves both introducing new data sets and getting data from other existing databases. Thus, the database creation is done from many sources that are logically or physically dispersed.
Let it be m large databases, DB_1, DB_2,... DB_m created by the same software product, containing banking transaction data from *m* territorially disjoint places. A computer application is built to achieve a virtual database *VDB* by concatenation of the basic data extracted from databases DB_1, DB_2, ... DB_m.
Essential information is placed in the local database in a single file, containing the keys of records from that collection. The virtual database joins the essential information and, without the physical copy of data, it will be part of the new large database, as shown in figure 3.

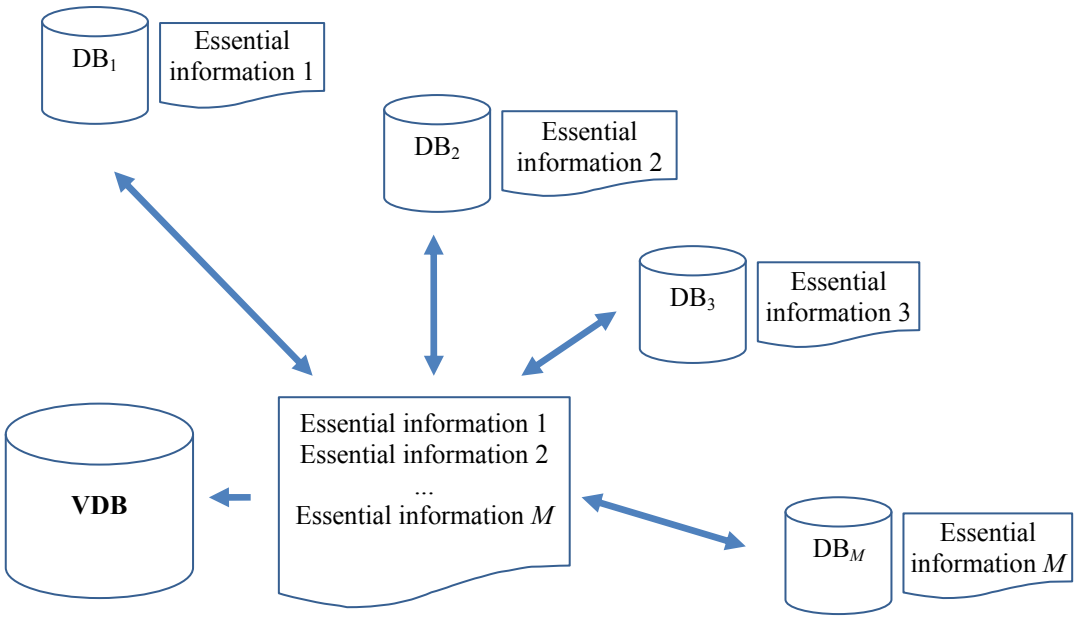

Figure 3. Creating VDB from several points

The homogeneity of the *VDB* created from several points is a sensitive aspect. If the compound *DB* is heterogeneous in structure, then the *VDB* is more of an aggregation of data rather than a database. To ensure homogeneity of structure, uniformization processes are adopted.
The homogenization to maximum, by addition, requires that each DB_i is brought to the same number of structural components by insertion. Let c_i be the number of fields in database DB_i. DB_j is chosen as the benchmark, where:

$$c_j = \max_{1 \le i \le m}\{c_i\}.$$

Each DB_i with $i \neq j$ has a field "deficit" quantified by d_i, where:

$$d_i = c_j - c_i, i = \overline{1,m}, i \neq j.$$

For each DBi where $d_i \neq 0$, di fields are inserted with values recorded from reality, if data is available or with NULL values.
The homogenization to minimum, by deletion, requires that each DB_i is brought to the same number of structural components by erasing other fields. This time DB_k is chosen as the benchmark, where:

$$c_k = \min_{1 \le i \le m}\{c_i\}.$$

Each DB_k with $i \neq k$ has a field "surplus" quantified by si, where:

$$s_i = c_k - c_i, i = \overline{1,m}, i \neq k.$$

For each DB_t where $s_t \neq 0$, all the fields that are not included in the intersection $DB_k \cap DB_t$ are deleted. From the two homogenization processes, the former has the advantage of keeping data in the DB, and even add some other. The disadvantages are counted by significant time loss during processing due to inserting missing values, and physical space occupied by the added data.
The latter's advantage is that it occupies minimum physical space and spends less processing time. The disadvantage is the loss of data that would affect usability.
Concatenation of data sets, Ivan (2009), resolves and optimizes the data aggregation at the record field's level. By linking multiple data sets, a single set of data is obtained representing the selected components. The difference between aggregation or selection and concatenation operation resides in creation of a single, representative set, while selection forms a new collection of sets.
In carrying out any banking collaborative project, moments occur when activities and results no longer fall within the parameters of the designed model. The reasons leading to such situations are some unexpected events that deviate the project from the planned course and therefore require special approaches. Hence, the design, development and implementation of banking projects should include treatment of uncertainty about the future.
Risk is a measure of probability of occurrence and severity of effects of future events. It treats the matter from two perspectives:

- how likely are future events;
- how important are the consequences if it occur.

In software projects, Ivan and Vintila (2009) the risks diversify because of different components that enter into the development of applications: stuff, technology, equipment, methodologies, etc. Complexity of the topic makes the handling of risks to be integrated as part of project management, as risk management – MR.
The place of MR, Pavel (2009), within the very large datasets oriented application development cycle is distributed along the processes and steps, as shown in figure 4. Discussion of MR occurs when dealing with use cases, is structured into the design stage, is refined within the coding and testing phases and is finalized during the launch and use phase.

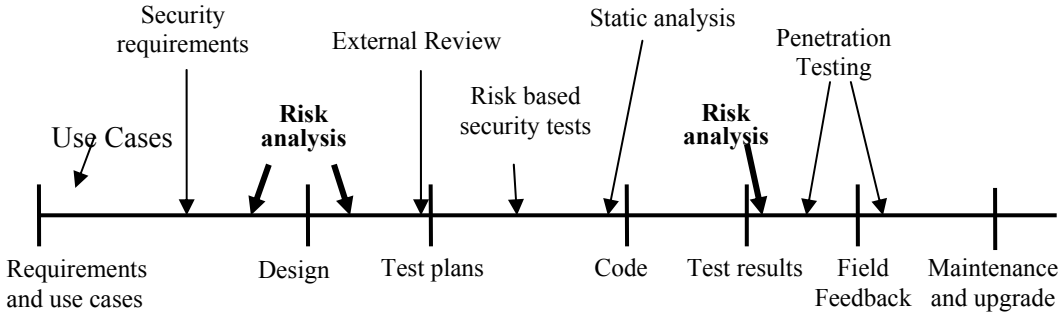

Figure 4. Risk analysis in the development cycle of VLDOA

The multitude ways of approaching risks in large datasets oriented projects, requires classification having different criteria. Depending on the category to which it belongs, the risk is treated or enhanced, depending on its position within the project, risk affects the development plan.
Depending on their size, risks are divided into low, moderate and high risk. Within the very large datasets oriented applications from collaborative banking system, appear:

- small risk - technical malfunction of a machine running the banking software system; whereas system was designed to work collaboratively online, any technical failure of the client computers have a limited financial impact and a practically null one in the banking unit;
- moderate risk - functional requirements from banking processes are not expressed or explained, which affects the development cycle by replaying design, coding and testing; moderate risks delay the development process: lack of accurate data of credit documents, wrong value records of the payments received/made, incorrect calculation of the credit rates;
- high risk - project funding ceases due to changes in legislation or bank's policy: phishing attacks that expose collaborative system's security and gain access to customer accounts.

Depending on the areas of risk event in the very large datasets oriented project, risks affects areas of: planning, budget, operational, technical and programmatic - figure 5:

Figure 5. Very large data sets risks

- *planning risks* arise when planning and scheduling is not addressed properly, or they are forced to change;
- *budget risks* relate to poor financial planning, reflected in: wrong estimations, cost overruns, technology replacement, poor or inexistent monitoring of expenses, inadequate functioning of data sets storage instruments;
- *operational risks* refer to the process of project implementation and have human, system or external causes;
- *technical risks* lead to failure of functionality and performance;
- *coding risks* reflect the software quality, Pavel and Palaghita (2009) in terms of security, Ivan and Doinea (2009), Burtescu (2008), and protection against failures, Wua (2009).

Depending on the stages of the development cycle of a bank collaborative process, figure 4, during which risks can occur, are identified several types:

- *user requirements risks*: lack of coverage of all situations of using the banking system, incomplete treatment of security requirements;

- *design risks*: lack of understanding the non-functional cases and constraints related to programming language;
- *implementation risks:* failure to identify significant components, subsystems integration failure, lack of testing use cases;
- *launch risks*: failure of collaborative processes on host machines, negative feedback from users;
- *maintenance and upgrade risks*: the emergence of unsolvable bug, the occurrence of use cases for the implementation of which should be reload the entire development cycle.

These expectations and risk classifications don't avoid unexpected events, but encourage an informed handling of situations. Classifying risks determines first risk identification and then implementation of appropriate methods for treatment in their context.

The reason why the risks are included in the project management process is given by the impact that the specified events in the project. Between the two aspects of risk - probability and impact – the second one is most feared. It is classified as low risk, the event described by a high probability of occurring, but with negligible impact. On the other side, even with a low probability, an event with catastrophic impact is considered high risk. Experimental results relates to the effects of the events involved by the previously described risks. They are included in table 6.

Table 6. Hazardous events and their effects in VLDSO projects

Risk class	Event	Effects
Planning risks	Erroneous estimation of activity periods	Acquisition activities – of data sets, collection modelling etc. – takes longer than planned; delayed start time for other activities and hence delays in completion of the project; superficial treatment of the data sets quality due to shortage of time; time consumed in explaining additional features that were not present in the original requirements; poor quality of the manufacturing processes of datasets.
	Failure to identify complex functionalities and the time required for their development	
	Unexpected development of the project scope	
	Inadequate knowledge of current technologies	
	Incomplete specification of objectives for each phase of the project	
	Lack of understanding between customer and developer	
	Difficulties in implementing the various requirements on data sets;	
Budget Risks	Erroneous estimates of expenditure categories	Failure to cover financial needs arising from activities such as data acquisition, data sets modelling, data storage; impossibility of demonstrating the of settlement costs; occurrence of unforeseen expenditures that deplete the
	Activities cost overruns	
	Change to other, more expensive technologies	
	Poor or inexistent monitoring of expenditure	
	Malfunctions of instruments and	

	their need for change	financial resources and jeopardize the project.
Operational Risks	Failure of conflict management	The emergence of a hostile environment within the team, lack of concentration, lack of motivation, superficiality; poor quality in application implementation, data sets modelling; occurrence of redundancy in the module browser data sets; delays in the process because of the longer staff training for using data sets technology.
	Failure in allocation and monitoring responsibilities within the team	
	Insufficient human, material or immaterial resources	
	Lack of planning and allocation of resources in development stages	
	Inadequate preparation of staff in handling data sets	
	Lack of adequate communication between project team members	
Technical Risks	Permanent change of functional requirements	Occurrence of stress factors due to repeated changes; consumption of time using undeveloped technology; occurrence of failures due to inadequate integration of source modules.
	Lack of developed technology	
	Excessive complexity, which discourages the project implementation	
	Difficult integration of project modules	
Coding Risks	Inadequate modules documentation	Poor source code quality; neglect of data sets functionalities; errors of modelling sets, loss of data sets due to hardware; lack of continuity in programming style and functionality approach.
	Lack of programmers ability or skills	
	Lack of adequate modules testing	
	Emergence of hardware failures	
	Excessive architecture complexity	
	Lack of communication between developers	
	Repeated changes of members, or environmental technology development	

Whatever the causes of undesirable events, risks must be treated properly and eventually require finding ways to reduce the probability of occurrence and mitigate their impact on the very large datasets oriented project.

3. Distributed application security

Distributed applications that are working with large data sets and on which a high variety of users are operating are more vulnerable. This fact is due to a specific characteristic met to this kind of applications as user and data diversity found in the system at a certain moment of time *T*. This inconsistency could lead to unpredictable situations, difficult to anticipate, because of the lack of knowledge of all exogenous variables which are directly shaping the level of security of such distributed applications.

On a security system implemented on a collaborative system that is working with very large data sets, a probe attack is launched determining in this way the possible

vulnerabilities which could be exploited later in a new attack as described in Wang (2007).
Let S, be the level of security of a distributed application. This is influenced by a large category of factors such as: endogenous, exogenous and residual factors, F_E, F_e and F_R as presented in figure 6.

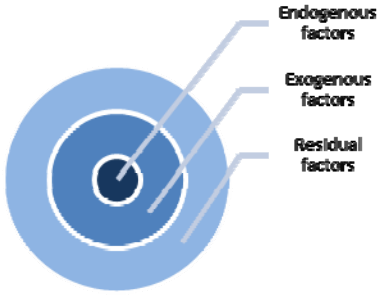

Figure 6. Types of influencing factors

The factors which can lower the level of security of a distributed application prior defined can be classified by their nature in:

- endogenous factors – they are part of the distributed system which in majority of cases can be avoided if on the life cycle of a distributed application, security is rigorously managed, Rehman (2009); in this category of factors the following can be mentioned: factors determined by the programming style which can influence the development, implementation and configuration stage of a distributed application; factors determined by the low performance of the distributed equipment; factors caused by a bad management of the process of analysing and planning the security level; this are the factors that implicitly minimize or maximize the rate of penetration in case of external attacks;
- exogenous factors – they are not dependent on the initial security distributed system configuration and are acting no matter of the existing level of security; factors that are outside the system and because of the openness characteristic of a distributed system they interact with the system's security components; factors caused by an inappropriate human utilization of the security equipment; unpredictable factors such as electricity falls or natural cataclysms;
- residual factors – are factors which are caused by the action of one or more aforementioned factors; this type of factors are inflicting damage as result of a dysfunction initially caused by others.

Influence factors are acting against the security system S at the following levels:

- data level – an important aspect of a distributed system is given by the raw data which is processed, the input components which should be interpreted and converted into results, output data based on the principle of a black box; the system distributed data, input or output, are vital for the good functioning of the entire system; lots of factors can influence through modify, delete and insert operations data with a severe impact on to the quality of entire distributed process;
- hardware level – at the hardware components level, of the equipments that are part of the distributed system, this factors can cause malfunctions, partial or total interruptions of the system functionality;
- controllers level – the link between data and hardware components of a distributed system is made by the distributed controller's level which in case of a dysfunction due to some factors won't be capable of manage correctly

their tasks leading to vulnerabilities on the entire level of the distributed system.

Supposing that a distributed system has a certain level of security, S_t, at the t time moment, and a set of factors F_t is influencing it with k_t, then the entire set of factors can be in one of the following situations:

a) they have no impact on the security of the distributed system when $k_t = 0$;

b) the impact on the security of the distributed system is between acceptable limits when $0 < k_t \leq S_t$;

c) the security system is overrun, the set of factors is able to exploit different vulnerabilities left untreated $k_t > S_t$;

The final situation c) is the unwanted one, in which the distributed system is vulnerable to different factors that might expose it and alter its integrity.

This set of factors is composed from threats that take advantage of vulnerabilities found in a distributed system, in this way having an internal part but also an external one, system independent, defined by a variety of outside threats and an attack rate which can't be controlled by any means from inside the system.

4. Security metrics

A security metric is a mathematical model described by a function with the following structure:

$$IS = f(F_1, F_2, \dots, F_n),$$

where:

F_i - variable or a set of n variables associated with some influence factors, $i = \overline{1, n}$;

IS - the result returned by the function.

Security metrics represent a system for assessing the quality and efficiency of the targets set by the security field. Quality metrics are analyzed from the perspective of some interpretations defined in Hinson (2010) according to which a metric should be:

- objective;
- with discrete values;
- not using absolute values in the measurement process;
- effective;
- complex security metrics should be analyzed and formed from other secondary metrics.

The metrics for applications oriented on very large data sets have to determine:

- the number of errors on application size with very large data sets, NED:

$$NED = \frac{NEA}{DAS},$$

where:

NEA – total number of errors identified in the application;
DAS – size application with very large data sets, measured in number of code lines, LOC.

The NED metric is $NED \in [0, +\infty)$. If $NEA = 0$ then $NED = 0$, being the best value which can be recorded, otherwise $\lim_{NEA \to \infty} NED = \infty$.

- the cost of testing application with very large data sets, CTS:

$$CTS = \sum_{i=1}^{N} CTC_i,$$

where:

N – number of application components;

CTC_i – test cost of component i.

- the number of security vulnerabilities, *NSV*, identified in the application with very large data sets:

$$NVS = \frac{NVDA}{DAS},$$

where:

NVDA – the number of vulnerabilities detected in the application;

DAS – size application with very large data sets, measured in number of code lines, *LOC* or *KLOC*.

The NVS metric is $NVS \in [0, +\infty)$. If $NVDA = 0$ then $NVS = 0$, being the best value which can be recorded, otherwise $\lim_{NVDA \to \infty} NVS = \infty$.

For the Collaborative Multicash Servicedesk application, used in Raiffeisen Bank, before the TDES encryption algorithm implementation, the number of security vulnerabilities per size of application was 0.7, but after the implementation of the encryption algorithm the NVS value decrease to 0.3.

- the attack rate, AR, upon the CMS application is:

$$AR = \frac{NIP}{TNIP} * 100,$$

where:

NIP – the number of IP addresses from which a different type of attack was launched;

TNIP – the total number of accessed IP addresses.

This metric was measured before and after the implementation of TDES encryption algorithm inside the Collaborative Multicash Servicedesk application. Before the moment when the application was secured, the attack rate was bigger and has diminished in time. In Table 1 are presented the AR values measured between October 2009 and August 2010:

Table 7. The measured values of attack rate indicator, Ciurea (2010b)

Month	Attack rate (AR) value
October 2009	70%
November 2009	68%
December 2009	65%
January 2010	61%
February 2010	60%
March 2010	54%
April 2010	45%
May 2010	32%
June 2010	26%
July 2010	12%
August 2010	3%

The data was automatically acquired from defects, times moments, errors, and based on their values were calculated the indicators for each metric.

- daily data volume of virtual database application with very large data sets obtained by the relationship, Ivan (2010):

$$VD_z = NP * \sum_{i=1}^{ND} NC_i ,$$

where:

NP – number of persons;

ND – number of documents;

NC_i – number of fields in document i.

The metric $VD \in [0, +\infty)$. If $NP = 0$ then $VD = 0$, else $VD > 0$.

- the data volume for a period of k days:

$$VD_{k\text{-}days} = k * VD_z,$$

where:

VD_z – daily data volume per day.

- the size in bits of the DBV for the application with very large data sets is:

$$DIM = N * \sum_{i=1}^{M} L_i ,$$

where:

L_i – length of field K_i:

M – number of characteristics;

N – number of records.

In terms of security of distributed systems that work with very large data sets, security metrics should be imposed to ensure an accurate and continuous operation of all processes performed in the distributed system.

Negative factors that influence final product quality must be removed so that when distributed system is implemented and configured, no vulnerabilities can be exploited with both qualitative and quantitative damage to developers and users.

As a result, everything deficient in the life cycle of a distributed system that works with large data sets, ranging from programming errors to inappropriate system configuration according to the environment, user requirements and security, should be avoided.

Risks of operating with very large sets refers to the difficulty of eliminating outliers or incorrect values, introduced by human operators or retrieved by acquisition from various equipment. The existence of such erroneous data affects parameter values calculated on the entire community.

5. Security optimization as statistical process

To increase performance level a collaborative banking system, specialized metrics to measure the quality characteristics levels for finding new collaborative situations are used. Collaborative processes influence the work with large data sets and their security by streamlining operations conducted with large data sets and increasing the security degree at the database level that stores these data sets.

Collaborative processes that take place within a collaborative banking system supposes the existence of a centralized database, in which operations are performed by inserting, modifying, archiving of records related to bank transactions. These operations are not performed directly on the centralized database, but they are added in a queue managed using intelligent agents. These agents analyze the type of operations and its priority and execute it on a centralized database. This provides an increased level of security through the introduction of intelligent agents in managing large data sets in the collaborative banking system.

Working with very large data sets influence collaborative processes and their security primarily by the size of the values that are available. Decisions from collaborative processes are professionally taken when very large data collections are mined and the necessary information for management operators is extracted. Working with very large data sets on the one hand leads collaborative processes in organizing and manipulating data, and on the other hand, these processes are those which provide a data stream that feeds the collections and increase their size.

Because of significant economic value of the collections and applications VLDS oriented, security must be treated separately, taking into account the dimensions and their specificities. Thus, security must cover the component (data set) and access to any amount included, the distributed collection and even the virtual database with the entire collection. Security is thus challenged to ensure safety on these three levels of working with very large data sets.

The security is influencing the collaboration and the work with large data sets in terms of constraints regarding the procedures and regulations that must be met in order to have a reliable result at the end of each collaborative process. This kind of influence is meant to lower the efficiency of each one of the components but this is acceptable and it is fair trade when thinking of the possible actions which a threat could inflict upon the application, generating costs that otherwise wouldn't be comparable with the time spent with the use of security aspects.

Let IC, IB and IS, be the variables of a new model, where:

IC – the level of collaboration;

IB – large data sets efficiency working level;

IS – security level.

Because collaborative processes which are working with large data sets are part of an open distributed system, they are exposed to internal or external factors, so, an aggregated level of efficiency between the collaborative processes, the procedures of working with large data sets and the security controls must be met.

Because of this, a number of variants for each one of the levels IC, IB and IS are defined. The optimization problem is defined as a maximization function for the performance level between the following factors: collaborative processes, the applications with large data sets and the security aspects:

$$max\ f(IC, IB, IS) = a_1 \cdot IC + a_2 \cdot IB + a_3 \cdot IS + a_4 \cdot (IC, IB) + a_5 \cdot (IC, IS) + a_6 \cdot (IB, IS)$$

where:

$a_1, a_2, a_3, a_4, a_5, a_6$ – components multipliers;

(IC,IB), (IC,IS) and (IB,IS) – the combination of each two factors implying the restrictions between them.

The algorithm through which the maximum performance level is wanted to be achieved is presented:

A. defining a number of independent variants *l, m* and *n* for each one of the levels IC, IB and IS found in the maximization function;
B. creating the sets for each function component, $\overline{\mathcal{A}_{IC}}$, $\overline{\mathcal{A}_{IB}}$ and $\overline{\mathcal{A}_{IS}}$;
C. choosing the good variants based on the restrictions of any type and the interactions between the components, creating the set of admissible solutions $\mathcal{X}$, defined as $\mathcal{X} = \overline{\mathcal{A}_{IC}} \cup \overline{\mathcal{A}_{IB}} \cup \overline{\mathcal{A}_{IS}}$;
D. determining the Cartesian product $X = \{(w_i, x_j, y_k) | w_i \in \overline{\mathcal{A}_{IC}}, x_j \in \overline{\mathcal{A}_{IB}}, y_k \in \overline{\mathcal{A}_{IS}}\}$, with $i = \overline{1, Card(\overline{\mathcal{A}_{IC}})}$, $j = \overline{1, Card(\overline{\mathcal{A}_{IB}})}$, $k = \overline{1, Card(\overline{\mathcal{A}_{IS}})}$;

E. for each one of the pairs from the Cartesian product, the level of the objective function is calculated, storing the values $(\overline{w_l}, \overline{x_j}, \overline{y_k})$, which maximize the optimization function $f(IC, IB, IS)$.

The logical schema for this algorithm is represented is depicted below:

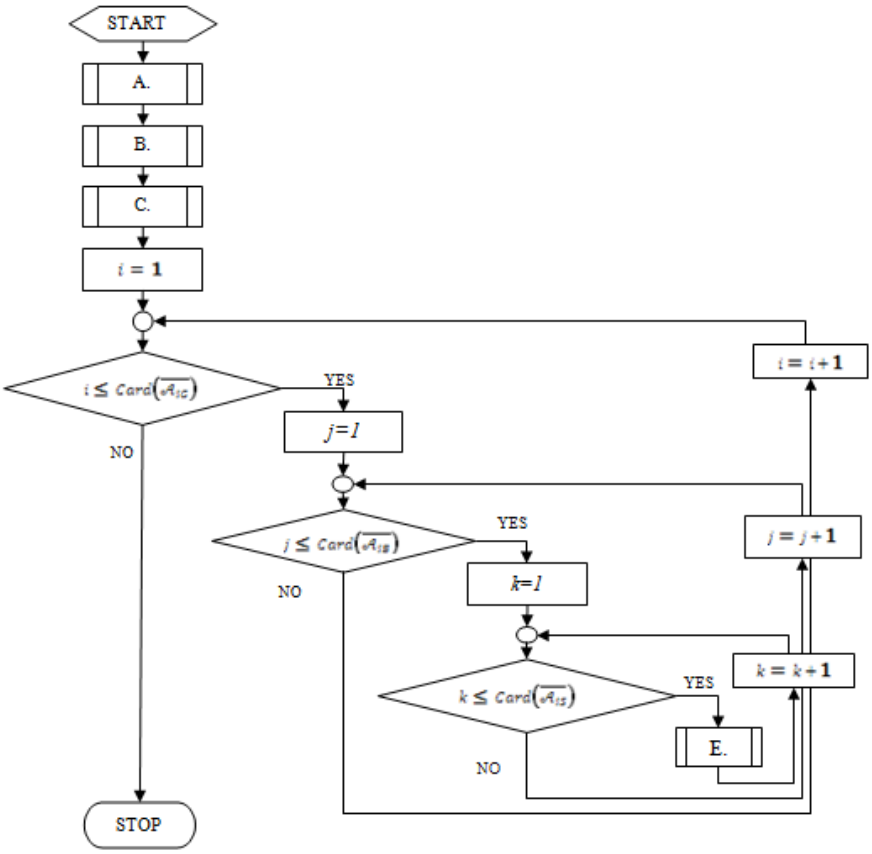

Figure 7. Logical schema for performance maximization

For the performance maximization of a collaborative application which is working with large data sets are known the following input data:

- the security level wanted to be achieved – $\overline{L_S}$;
- the efficiency level of working with large data sets – $\overline{L_B}$;
- the collaboration level of the application – $\overline{L_C}$;

The aforementioned maximization algorithm can be improved by minimizing the number of generated solutions to a partial set $\bar{X}$ with $\bar{X} \subseteq X$.

If the numbers are high enough than a value prior determined, then the sets $\overline{A_{IC}}$, $\overline{A_{IB}}$ and $\overline{A_{IS}}$ should be sorted to achieve the solution in a shorter time.

$$\overline{A_{IC}} = \{w_l | w_{l-1} < w_l < w_{l+1}\}, i = \overline{1, l}$$
$$\overline{A_{IB}} = \{x_j | x_{j-1} < x_j < x_{j+1}\}, j = \overline{1, m}$$
$$\overline{A_{IS}} = \{y_k | y_{k-1} < y_k < y_{k+1}\}, k = \overline{1, n}$$

The optimization is given by the fact that only several combinations will be generated for calculating the values of the maximization function, the algorithm achieving a full stop when the following three conditions will be met:

$$L_C \succ \overline{L_C} + \delta;$$
$$L_B \succ \overline{L_B} + \delta;$$
$$L_S \succ \overline{L_S} + \delta;$$

where:

L_C, L_B, L_S – the values determined on the run;

$\lim_{\substack{\varepsilon \to 0 \\ \varepsilon > 0}} (\delta - \varepsilon) = 0$;

$\succ$ – the operator for comparing the levels of security, collaboration and efficiency in working with large data sets.

Having this said, the number of total checked combinations for the algorithm will be much lower than the one resulting from a complete run, the algorithm stopping when the three levels will be almost equal with the ones established in the beginning.

For calculating the average number of iterations for the algorithm in achieving the desired solution, also the number of (IC, IB, IS) combinations, the statistical procedure of calculating the confidence interval for a random variable X is used.

After a number *n* of algorithm runs the values $X_1, X_2, \ldots, X_n$ are recorded, where:

X_i – the number of algorithm iterations before optimal value has been achieved;

Because of the fact that the conditions of normal repartition are met N(m, δ) [ ], the confidence interval for $\bar{X}$ is given by:

$$\bar{X} \in \left[\bar{X} - t_{1-\frac{\varepsilon}{2};n-1} \cdot \frac{s}{\sqrt{n}}; \bar{X} + t_{1-\frac{\varepsilon}{2};n-1} \cdot \frac{s}{\sqrt{n}}\right],$$

where:

$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{n}$ – the average value of iteration number made for entire set used to measure the values of X;

$t_{1-\frac{\varepsilon}{2};n-1}$ – the Student tabled value for n-1 degree of freedom and a confidence degree of *1-å*;

$s = \sqrt{\frac{1}{n-1} \cdot \sum_{i=1}^{n} (X_j - \bar{X})^2}$ – dispersion of X variable;

n – the selection volume.

For a numeric example, let's consider that $Card(\mathcal{A}_{IC}) = Card(\mathcal{A}_{IB}) = Card(\mathcal{A}_{IS}) = 20$, that is a total number of possible combinations of IC, IB and IS components equal to $Card(X) = Card(\mathcal{A}_{IC}) \cdot Card(\mathcal{A}_{IB}) \cdot Card(\mathcal{A}_{IS}) = 8000$. This fact indicates that, if the maximization function that is not optimized is used, the average number of iteration made for each set is equal to 8000. The improvement given by the optimized interpretation is highlighted in the numeric particularities that this function takes into account when it comes to finding the first combination that generated a level of the f function better than the input level, against the fix number of 8000 iterations.

6. Conclusions

The collaborative processes that are working with large data sets are vulnerable, first of all, because of the collaborative environment and all what is implied by it, and second, because of the huge amount of data processed, hard to managed in optimal and secure conditions because of the long processing time, when parts of the data are exposed to several risks due to two reasons: possible collaborative processes' deficiencies gathered from previous steps of system implementation and because a large variety of threats in all the fields of informatics are ready to take advantage of collaborative processes and used data.

The modality of implementing security for applications that are using very large data sets for collaborative processing determines directly the quality characteristics. For improving these characteristics, the optimization of security implementation procedures by increasing security complexity model and treating as much

vulnerabilities it is possible has a quantifiable effect in terms of increasing the processing quality and the obtained results.
From this perspective, having a good level of security for the collaborative processes and data is a must for assuring a high level of quality for the processes and confidence in the interpretation of final results for using them in the decision processes.

Acknowledgements

This article is a result of the project „Doctoral Programme and PhD Students in the education research and innovation triangle". This project is co funded by European Social Fund through The Sectorial Operational Program for Human Resources Development 2007-2013, coordinated by The Bucharest Academy of Economic Studies (project no. 7832, "Doctoral Programme and PhD Students in the education research and innovation triangle, DOCECI").

References

Ciurea, C. (2010). The Informatics Audit – A Collaborative Process. INFOREC Publishing House. Informatica Economică Journal, Vol. 14, No. 1, ISSN 1453-1305

Ciurea, C. (2010b). Implementing an Encryption Algorithm in Collaborative Multicash Servicedesk Application. Open Source Science Journal, Vol. 2, No. 3, ISSN 2066–740X.

Adhikari, A., Rao, P.R. (2008). Efficient clustering of databases induced by local patterns. Decision Support Systems, Vol. 44, No. 4, pp. 925-943

Ivan, I., Dumitrascu, E. (2008). Stable Structures for Distributed Applications. Informatica Economica Journal, Vol. 12, No. 1

Ivan, I., Ciurea, C. (2009). Using Very Large Volume Data Sets for Collaborative Study. Informatica Economica Journal, Vol. 13, No. 1

Ivan, I., Vintila, B., Palaghita, D., Pavel, S., Doinea, M. (2009). Risk Estimation Models In Distributed Informatics Applications. Globalization and Higher Education in Economics and Business Administration GEBA, Iaşi, România

Pavel, S. (2009). Software Development Life Cycle, Open Source Engineering Component. Open Source Science Journal, Vol. 1, No. 2, 2009, pp. 111-126

Pavel, S., Palaghita, D. (2009). Large Data Volumes Quality Analysis. International Simposium of Young Researches (The VIIth Edition), Chişinău, Moldova Republic

Ivan, I., Doinea, M., Vinturis, S., Pavel, S. (2009). Distributed application security management. 9th International Conference on Informatics in Economy IE2009, Academy of Economic Studies, Bucharest, Romania

Burtescu, E. (2008). The Network's data Security Risk Analysis. Informatica Economică, Vol. 12, No. 4(48)

Wua, J., Manivannan, D. Thuraisinghamb, B. (2009). Necessary and sufficient conditions for transaction-consistent global checkpoints in a distributed database system. Information Sciences, Vol. 179, No. 20, pp. 3659-3672

Shmatikov, V., Wang, M.H. (2007). Security Against Probe-Response Attacks in Collaborative Intrusion Detection. Proceedings of the 2007 workshop on Large scale attack defense, pg. 129-136, ISBN 978-1-59593-785-8

Rehman, S., Mustafa, K. (2009). Research on software design level security vulnerabilities. ACM SIGSOFT Software Engineering Notes, Vol. 34, No.6, pg. 1-5, ISSN 0163-5948

Ivan, I., Popescu, M., Sinioros, P., Simion, F. (1997). Metrici software. Inforec Publishing House, Bucharest

Hinson, G. (2006). Seven myths about information security metrics. ISSA Journal**,** July, Available at: http://www.noticebored.com/html/metrics.html

Ivan, I., Ciurea, C., Pavel, S. (2010). Very Large Data Volumes Analysis of Collaborative Systems with Finite Number of States. Journal of Applied Quantitative Methods, Vol. 5, No. 1, ISSN 1842–4562