RISKS IN DEVELOPMENT OF VERY LARGE DATA SETS ORIENTED APPLICATIONS

Sorin Pavel[1]

Abstract

Very large data sets are defined.
The features of the online operation with large data sets are described.
A risk classification and analysis is made in working with large data sets oriented software.
Experimental results are presented.

Keywords: *large data sets, risk, online operation*

1. Introduction

The objective of this paper is to identify and characterize the risks present in developing and optimizing online applications oriented on very large collections of datasets. The need to study the proposed domain is given by:

- computerization of modern society and widely spread of software products that work with large collections of data in any field of activity: economic, social and cultural;
- promulgation of new IT laws that encourage the formation of large collections of data;
- disproportionate costs of processing raw very large data sets in contrast with processing already prepared data;
- effects arising from errors in existing very large sized data sets;
- significant effort to correct such errors in very large data sets.

All these arguments are reasons for undertaking a specific effort as a project. The project of online operation of very large data sets is a unique process consisting of many coordinated and controlled activities, with start and end limits, which ensures objectives achievement in required down time, cost and resources. Identifying and treating risk guarantees efficient project management in obtaining final products. Inclusion of risk management is necessary not only because of the very large scale data collection but also because of the significant resources involved.

2. Large data sets

Computerization of modern society, citizen-oriented software distribution and promulgation of new IT laws has led to applications that work with large data sets:

- telecommunications operators record each call or message within the network for a period of six months;

[1] Sorin Pavel, Academy of Economic Studies, Bucharest

- internet and e-mail providers record accessed sites for each IP address in its administration, together with the exact date of access and data about each email message;
- government keep track of different payments for millions of people;
- national providers of utilities – gas, electricity etc. – process hundreds of millions of annual consumer bills;
- online search engines integrate content management of billions of sites.

Situations mentioned above involve many simultaneous users, using very large data – $10^{7} \div 10^{10}$ sets – and applications for data management [ADRA08]. Due to the large quantities of data sets to be processed, applications acquire specific properties and functionalities.

Data collections represent, but are not limited to: databases, collections of text files/XML files/multimedia, data warehouse, or any combination thereof. Administration [IVDU08] requires specialized tools to harmonize the specific hardware and software aspects of large data sets.

The size or volume of database is 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 operationability 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 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, logically or physically dispersed.

Let it be M large databases, DB_1, DB_2,... DB_M created by the same software product, containing data from 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 1.

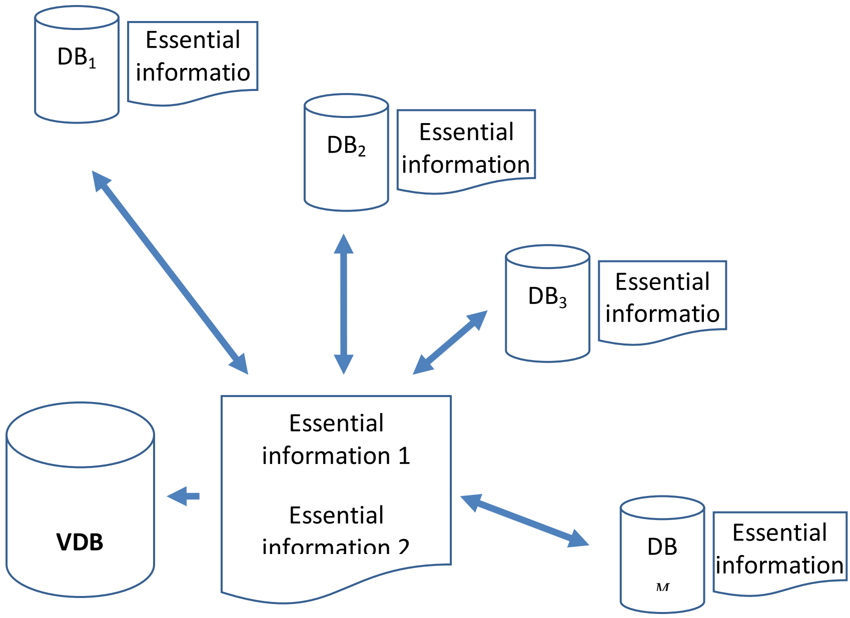

Figure 1 - Creating VDB from several points

Concatenation of data sets [IVCI09] resolves and optimizes the data aggregation at the record fields 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.

Concatenation must be defined in the application context: let BD_i be a database previously defined, with H_i records noted $S_1^i, S_2^i, ..., S_{H_i}^i$ each of them with h_i fields, referred by expression $S_k^i.field_j$. Concatenation of sets means:

- for text fields, adding at the end of the content the new data sets and obtaining a single text string by joining together:

  $S_k^i.field_j = S_k^i.field_j \oplus S_t^i.field_j, \;\; t \in MC$

  where:

  $field_j$ – linking field which is processed;

  $\oplus$ – operation of adding at the end;

  MC – the set to be concatenated – contains data sets in order of concatenation;
- for numeric fields, concatenation is an user-defined operation on integer values, often adding or multiplying, but not exceeding the data type restrictions:

  $S_k^i.field_j = S_k^i.field_j \otimes S_t^i.field_j, \;\; t \in MC$

  where:

  $\otimes$ – user-defined operation.

The concatenation process must grant the opportunity for the user to modify each field because many types of data within a set must be treated according to the need of those who use them.

3. Online operation with very large data sets

Very large data sets oriented applications interact with a wide and varied range of users. Therefore, its level of generality must be accepted by most users while the level of specialization must assure problem solution. At the same time, the application must always remain independent of users. Thinking, planning and designing [PAVE09a] the application in this way involves the following features:

- presence or absence of users does not affect application operationability; the software product is available both for a large number of simultaneous users and in the situation of no user, without changing application functionability;
- user behavior does not affect the application structure; the degree to which the user understands or not, and use the application correctly or not, should only influence the quality of input respectively output data sets, not the operating process or the internal structure of the application;
- input does not affect processing flow; quality of data entered by the user only has repercussions on the solution offered; the application does not enter technological impasse due to lack of input parts or their incorrectness but continues processing and issues the output data, with warning of possible errors.

Using application and processing data should be transparent, so that the user is aware of its problem solving stage, knowing what data enters the process, what resources are available [CETM08] and what processes are run at a time.

The application transparency is ensured by:

- visual selection of data entered by the user and labeling them as successfully retrieved;
- real-time messages about processing status;
- user guidance throughout the process of solving by indicating the current status, assessing the steps, the current step and/or the remaining steps to the solution;
- releasing a log-type file at the end, containing textual summary of the most relevant actions, together with related messages and solutions offered;
- granting possibility at any time to interrupt processing, to cancel the processing effects and to return to its original state.

The purpose of transparency is entrusting full control to the user which master the application actions so that no other processes interferes. During this time, although the user is given control, the application implements a scheme to maintain the integrity of data collection [KAME09]

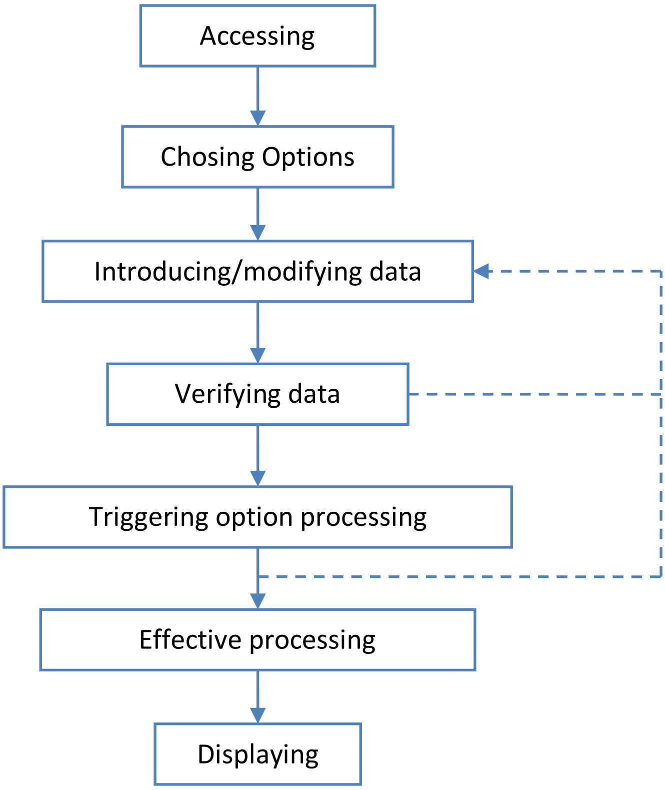

Figure 2 – Operation processes on very large data sets

Operating processes [BORO08] must allow users to visualize all input data prior to processing. The user verifies and is able to change, edit, detail and comment before submitting. Also, after loading the data, the application provides a short period of time for the user to return and edit entries and then retransmit. The sets are saved for future forwarding. The order of operating processes is shown in Figure 2.

In case of multiple identical processes: identification data, account data, periodic processing options etc. the application contains interfaces with user-defined forms. This way, the client avoids placing the same data every time, when processes are similar. Payment orders within e-banking applications use the same work paradigm. Once completed, the orders are saved as a template and a set of preformed orders is build, which are used for various payments from the same account holder.

Also, when submitting online orders, the identification data of the recipients – delivery address, phone, e-mail – are stored with the first order and then made available for automatic completion to subsequent orders. Given the above principle, data is still editable with every command, allowing the client to change details of a particular command. Firstly, these features must be tested extensively [RDAB09] to eliminate the appearance of undesirable outcomes.

4. Risks in using very large data sets

In carrying out any 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 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, [IVPP09] 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 [PAVE09b] within the very large datasets oriented application development cycle is distributed along the processes and steps, as shown in Figure 3. 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.

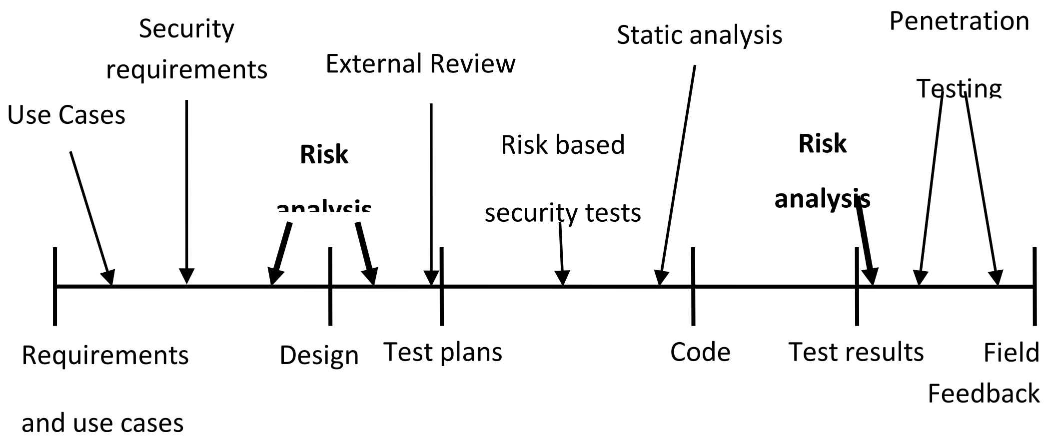

Figure 3 – Risk analysis in the development cycle of VLDOA

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, appear:

- *small risk* - loss of profit after application implementation and its launch to the public; the risk is low because the likelihood of this situation is poor due to feasibility study taken and its impact on the project itself is absent because it was already finished;
- *moderate risk* - functional requirements are not expressed or explained, which affects the development cycle by replaying design, coding and testing; moderate risks delay the development process;
- *high risk* - project funding ceases due to changes in legislation or client's intentions; without adequate funding, a large project is likely to break and then be canceled.

Depending on the areas of risk event in the very large datasets oriented project, risks affect areas of: planning, budget, operational, technical and programmatic - Figure 4:

Figure 4 – Risks of Very Large Datasets Oriented project

- *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 [PAPA09] in terms of security [IVDP09], [BURT08], and protection against failures [WMTH09];

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.

5. Experimental Results

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 occuring, 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 1.

Table 1 – 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 modeling 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 modeling, data storage; impossibility of demonstrating the of settlement costs; occurrence of unforeseen expenditures that deplete the financial resources and jeopardize the project.
	Activities cost overruns	
	Change to other, more expensive technologies	
	Poor or inexistent monitoring of expenditure	
	Malfunctions of instruments and their need for change	
Operational Risks	Failure of conflict management	The emergence of a hostile environment within the team, lack of concentration, lack of
	Failure in allocation and monitoring responsibilities within the team	

	Insufficient human, material or immaterial resources	motivation, superficiality; poor quality in application implementation, data sets modeling; occurrence of redundancy in the module browser data sets; delays in the process because of the longer staff training for using data sets technology.
	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 modeling 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.

6. Conclusions

Designing techniques for risk treatment begins with a full description of them as the first and most important step to avoid unintended consequences of events is knowing the issues involved. Details to be known are:

- sources of risks;
- favorable environment of production;
- probability of occurrence;
- event manifestation;
- impacts of production;
- costs or financial impact quantification.

After analyzing the details of context and risk behavior, the treatment focuses on source of production or its impact on the project.

Risks related to the development process are minimized by:

- using a project development/implementation methodology and explaining the different stages and expected results;
- avoiding new technologies not enough developed;
- maintaining clear and simple project objectives;
- organizing team meetings;
- detailed documentation of the changes;
- customer involvement in project design;
- ongoing monitoring of risks and activities.

Compliance with mitigation methods of undesirable events not only improves the situations of risk appearance, but brings further clarification in the development process, and provides alternatives to address the extreme cases.

REFERENCES

[ADRA08]	Animesh ADHIKARI, P.R. RAO – *Efficient clustering of databases induced by local patterns*, Decision Support Systems, Vol. 44, No. 4, March 2008, pp. 925-943
[BORO08]	Esther BOROWSKI, Hans-J. LENZ – *Design Workflow System to Improve Data Quality Using Oracle Warehouse*, Journal of Applied Quantitative Methods, Vol. 3, No. 3, September 2008
[BURT08]	Emil BURTESCU – *The Network's data Security Risk Analysis*, Informatica Economică, Vol. XII, No. 4(48), 2008, INFOREC Publishing House
[CETM08]	Dan CHEN, Roland EWALD, Georgios K. THEODOROPOULOS, Robert MINSON, Ton OGUARA, Michael LEES, Brian LOGAN and Adelinde M. UHRMACHER – Data access in distributed simulations of multi-agent systems, Journal of Systems and Software, Vol. 81, No. 12, December 2008, pp. 2345-2360
[ICPA10]	Ion IVAN, Cristian CIUREA, Sorin PAVEL – *Very Large Data Volumes Analysis of Collaborative Systems with Finite Number of States*, Journal of Applied Quantitative Methods, Vol 5, No. 1, March 2010
[IVCI09]	Ion IVAN, Cristian CIUREA – *Using Very Large Volume Data Sets for Collaborative Study*, Informatica Economica Journal, Vol. 13, No. 1/2009, INFOREC Publishing House
[IVDP09]	Ion IVAN, Mihai DOINEA, Sorin VINTURIS, Sorin PAVEL – *Distributed application security management*, 9th International Conference on Informatics in Economy IE2009, Academy of Economic Studies, Bucharest, Romania, May 2009
[IVDU08]	Ion IVAN, Eugen DUMITRAŞCU – *Stable Structures for Distributed Applications*, Informatica Economica, Vol. XII, No. 1(45), 2008
[IVPP09]	Ion IVAN, Bogdan VINTILĂ, Dragoș PALAGHIȚĂ, Sorin PAVEL, Mihai DOINEA – *Risk Estimation Models In Distributed Informatics Applications*, Globalization and Higher Education in Economics and

	Business Administration GEBA 2009, Iași, România, Octombrie 2009
[KAME09]	Ibrahim KAMEL – *A schema for protecting the integrity of databases*, Computers & Security, Vol. 28, No. 7, October 2009, pp. 698-709
[PAPA09]	Sorin PAVEL, Dragos PALAGHITA – *Large Data Volumes Quality Analysis*, International Simposium of Young Researches (The VIIth Edition), Chișinău, Moldova Republic, April 2009
[PAVE09a]	Sorin PAVEL – *Large datasets oriented software architecture*, Annual Conference of PhD Students in Economic Sciences, Academy of Economic Studies, Bucharest, Romania, May 2009
[PAVE09b]	Sorin PAVEL – *Software Development Life Cycle, Open Source Engineering Component*, Open Source Science Journal, Vol. 1, No. 2, 2009, pp. 111-126
[RDAB09]	Lihua RAN, Curtis DYRESON, Anneliese ANDREWS, Renée BRYCE, Christopher MALLERY – *Building test cases and oracles to automate the testing of web database applications*, Information and Software Technology, Vol. 51, No. 2, February 2009, pp. 460-477
[WMTH09]	Jiang WUA, D. MANIVANNAN, Bhavani THURAISINGHAMB – *Necessary and sufficient conditions for transaction-consistent global checkpoints in a distributed database system*, Information Sciences, Vol. 179, No. 20, September 2009, pp. 3659-3672