RISK MANAGEMENT IN CITIZEN ORIENTED INNOVATIVE SOFTWARE DEVELOPMENT PROJECTS

Emanuel Herteliu[1]
Mihai Liviu Despa[2]

ABSTRACT

This paper tackles the subject of risk management in the specific context of citizen oriented innovative software development projects. Research hypotheses are enunciated. Research standards and methodology are defined. Using data collected from specialized scientific literature, the risk management issue is highlighted as an important aspect of fighting software development projects failure. Methods for identifying risk in innovative software development projects are submitted for debate. Risk patterns in citizen oriented applications are identified. Risk assessing tools are depicted and illustrated using a real life software development project's data. Measuring risk is also performed in the context of an actual software development project. Risk monitoring procedures are submitted for analysis. Research hypotheses are validated. Research conclusions are formulated based on the facts, information, data, theories and results showcased throughout the article.

Keywords: software development, risk management, quality metrics, innovation

1. IDENTIFYING RISK IN INNOVATIVE SOFTWARE DEVELOPMENT PROJECTS

Risk represents an uncertain event or condition that, if materialized, has a positive or negative effect on project objectives [1]. Traditional project management approaches risk in a reactive mode, solving problems when the risk materializes. In innovation oriented projects risk must be addressed proactively by treating the factors that negatively impact the project [2].

The risk identification process should be carried out throughout the entire lifespan of the project. In order to identify the risks associated with a software development project the project manager must analyze several sources. First research hypothesis is that the project manager can identify specific risks by analyzing previous projects, consulting with the project owner and with the project team. Hypothesis will be validated by testing assumptions on an actual software development project. The project will be referred throughout this paper using the acronym ALFA.

Previous experiences are scenarios where the project manager was involved in implementing similar software development projects. The project manager's experience is a cheap and easily accessible source for identifying risks. The process of closing a project should include a data and information collecting activity that generates valuable know-how for future projects. Based on previous experience the project manager can create a checklist that particular and basic risks encountered in other similar software development projects.

[1] University of Economic Studies, Bucharest, emanuel.herteliu@gmail.com
[2] University of Economic Studies, Bucharest, mihai.despa@yahoo.com

The checklist should be updated following each project. Table 1 depicts a checklist used in identifying risk in the ALFA project.

ITEMS	*YES*	*NO*	*Does not apply*
Are there sufficient resources available to implement the project in optimum conditions?	X		
Have the project team members enough experience and training in order to perform their duties efficiently?	X		
Are there any solutions available for the scenario where the team is forced to change its structure or reduce its size?		X	
Is the project team properly sized?	X		
Do project activities follow a logical succession?	X		
Is the time resource optimally used?	X		
Is there buffer time allocated for high complexity activities?		X	
Does budget structure include all types of expenditures necessary to accomplish the project objectives?	X		
Are there any backup alternatives if suppliers fail to fulfill their contractual obligations?			X
Is the national and international legislation in conflict with any of the project's activities?		X	
Are the project objectives in line with EU standards?			X
Are the project's objectives eligible for any EU programs?		X	
Are the team members satisfied with the working conditions and the salary level?	X		
Has the market been studied in order to identify opportunities and ways of exploiting them?			X

Table 1 – ALFA project checklist

According to the checklist shown in Table 1 there are considerable risks at project team level. If one of the team members leaves the project no quick solutions for its replacement are available. There is also the risk that estimates regarding activities' duration are flawed and the checklist highlights that there were no buffer time periods setup to handle contingency situations.

The project owner is a source of risk but also a valuable asset in identifying risk. The project owner is the one that defines specifications and also the one who knows in detail

the non-technical aspects of the project. The software development project is often related to the project owner's area of activity. The ALFA project is aimed at developing an application designed for managing events. Project owner was a professional event planner and highlighted a significant number of risks. The project owner stated the event planning filed is very dynamic as clients tend to ask have exotic and maverick requests very often. This aspect translates in the possibility of changing requirements for the application very often.

The project team should be involved primarily in identifying technical risks. Technical solutions implemented in the development process represent risk sources. Table 2 depicts a questionnaire used for identifying risks. The target group for the questionnaire is the project team. In the ALFA project, the project owner requested a live video stream solution to be implemented. The project team highlighted the fact that the server that was hosting the application had an Internet connection whose bandwidth allowed for no more than 10 simultaneous video stream connections. As a result the project manager requested that the project owner will commit to provide Gigabyte Internet connection. The tools that were used for identifying risks are: interviews, questionnaires and brainstorming sessions. Table 2 presents the questionnaire template used in the ALFA project to identify risks.

QUESTION	*YES*	*To a great extent*	*Not sure*	*To a small extent*	*NO*
Given the current state of the project do you consider that the project's objectives are achievable?		X			
Do you consider that the project manager performs its tasks efficiently and accurately?	X				
Do you consider that within the project team there are conflicts, tensions or misunderstandings?					X
Is the project's budget in danger of being exceeded?			X		
Are the project's activities properly sized and budgeted?			X		
Have necessary measures been taken to ensure proper security for the software application?		X			
Is your wok valued within the team?			X		

Table 2 – Risk identification questionnaire

Analyzing responses presented in questionnaire from Table 2 shows that project scope and objectives were not clearly transmitted to the project's team. Also the respondent is not sure whether his efforts are appreciated within the project team. Direct interviews with project team members are also used in order to identify risks. During interviews project team members suggested that the project environment is an element that must be studied to

identify the restrictions imposed to the application being developed. Within the ALFA project the video stream and audio recording are elements requested by the project owner. The current legal framework does not allow the interception and recording of audio without people's consent. As a result *audio recording* was removed from the specifications list. Risks identified for the ALFA project are listed in Table 3.

No.	*RISK*	*SOURCE*
1	*If the owner of the project changes requirements during the project development process a cost increase will occur and deadlines will be endangered.*	Previous projects;
2	*If requirements, objectives and project scope are not properly explained to project team members, than the end result of the project will not match the project owner's expectations.*	Project team; Previous projects;
3	*If the time required to complete a tasks was underestimated, then this leads to postponing other tasks which in the end will lead to not meeting the deadline.*	Previous projects;
4	*If client data stored on the developer's servers gets compromised or corrupted than developer will face real credibility issue.*	Project team;
5	*If project costs are underestimated than the desired resource quality in terms of human resources, equipment or information will not be met.*	Project team; Previous projects;
6	*If project team members are not motivated then productivity decreases and delays occur in developing and testing functionality.*	Previous projects;
7	*If the APIs and libraries that are used for implementing the application don't have proper documentation then integration will take longer than expected and the application deadline will be jeopardized.*	Project team;
8	*If the project owner does not give timely feedback then delays will occur and the programming process will be delayed.*	Previous projects;
9	*If one of the team members leaves the project because he doesn't feel that his work is appreciated then the project deadline will be jeopardized.*	Previous projects; Project team;
10	*If the platform that mediates the collaboration is unavailable then developers can't work on the project and delays will occur.*	Previous projects;

Table 3 – Risks identified for the ALFA project

For the ALFA project 10 risks were identified, 7 of which are defined on the basis of previous experiences. Another important aspect is the fact 4 of the risks were pinpointed by multiple sources. Risks that have been pinpointed by multiple sources are more likely to occur.

2. RISK PATTERNS IN CITIZEN ORIENTED APPLICATIONS

Managing risk is an ongoing process in software development projects and depends on the type of the developed software solution. Certain risks are analyzed for online software solutions where data is sent and received from server to client involving the use of secure protocols but for desktop applications this kind of risks are not analyzed. The same approach is used for the risks within e-commerce applications with digital payments implemented. Identifying and handling risks is done at the level of project management and at the level of the solution [5]. Citizen oriented applications are informatics applications developed for helping final users solving problems in their day by day life. The goal in developing citizen oriented informatics applications is to increase the comfort and satisfaction of the final user minimizing the necessary time to obtain the results. It is aimed at minimizing the time and effort of using the application to be done keeping a high rate of efficiency and accuracy for the final results. The characteristics of citizen oriented software solutions are:

- applications that run online always with a high level of availability from various classic and mobile devices;
- IT skills required in order to be able to use the application are the basic ones in browsing Internet and using simple and intuitive graphical interfaces;
- the use of application is usually free or with very small fees in order to provide access to functionalities to a wide category of users;
- logical and easy to follow steps for obtaining the desired results; this is done so as users are able to use the application with a minimum effort;
- authentication is required for transactions that need to secure user's information and on the points where the user needs to have an identity on the application in order to interact with other users.

For citizen oriented projects, risk is analyzed as impact on the development process and impact on the solution being developed. Impact is quantified in terms of grades. Table 4 presents the grades of impact on development process.

IMPACT	*DESCRIPTION*	*RESULT*
Low	The occurred risk situation leads to the impossibility to move to the next activity in the current set of activities in the ongoing stage of the development process.	Small delays of the activities. The cost of handling this type of risk is very low.

Deep	The occurred risk situation leads to the impossibility to carry on with the current set of activities of the ongoing stage of the development process.	Noticeable delays in the project's deadlines. The cost of resolving deep impact risk situations is noticeable for the project budget.
Critical	The occurred risk situation leads to the impossibility to move to the next stage of the development process. Reiteration of the current stage is necessary.	The critical impact risk situations are causing high delays for deadlines from weeks to months with high costs depending on project's complexity.

Table 4 – Risk impact grades for citizen oriented projects

The impact of risk situations on the project is analyzed using standardized degrees as presented in Table 4. The same approach is used as presented in Table 5 for analyzing risk impact on the developed application.

IMPACT	*DESCRIPTION*	*RESULT*
Low	The low impact risk situation causes the unavailability of few features of the application. The feedback between the application's support team is important for this type of impact.	The user's satisfaction in using the application decreases to a great extent.
Deep	The deep impact risk situation causes the unavailability of some of the important features of the application being unavailable for the current user or a small amount of the total users.	Deep impact risk has the result of a lower credibility for the software solution.
Critical	The critical impact risk situation leads to the application being unavailable for all the users. If there is an older version of the application the servers are switched to it in order to maintain the activity while the problem is being solved.	Unsolved critical risk for a long period of time has the result of users choosing similar software solutions to solve their problem.

Table 5 – Risk impact grades for citizen oriented applications

Preventing risk situations with a deep and critical impact to occur is a must for applications which are using payment methods and are running processes with a very low response time. The development of projects for citizen oriented applications is a process that follows a pattern of stages which succeed in a specific order. The output of each stage represents input for the next one thus risk management is very important because solving risk situations implies additional costs. It is difficult and resource consuming to make changes to work that has already been completes during a stage of the development process. Risk occurrence has a negative effect on the overall project. Controlling risks is based on the impact on the project because some risk situations may cause the project to stop at the current stage. Reiteration of activities specific to a stage is often necessary because of

mismanaged risks and it implies additional costs and delays. Risk factors for citizen oriented projects are identified for each of the project's development stage. The factors together with their impact on the developed solution and on the developing process are presented in Table 6.

STAGE	*ACTIVITY*	*RISK FACTORS*	*IMPACT ON PROCESS MANAGEMENT*	*IMPACT ON THE FINAL SOLUTION*
Problem defining	Defining the problem to be solved with the application.	Not covering all aspects of the problem	-	Deep
	Defining the solutions of the problem.	Efficiency/Complexity balance	-	Deep
	Defining problem solving steps.	Skipping steps	-	Deep
		Redundant steps	-	Deep
Target group establishing	Establishing the number of users.	Estimating fewer users	-	Critical
	Establishing users diversity based on: age, location, computer skills.	Wrong language provided	-	Low
		Not handling representativeness	-	Deep
Feasibility analyze	Analyzing the efficiency of the solution	Too few feasibility tests	-	Deep
	Compared analyze with similar released solutions	Choosing to few solutions	-	Deep
Planning	Defining specifications.	Confusing specifications	-	Critical
		Changing specifications after this stage	-	Critical
		Overestimating	-	Deep

Resource planning	Underestimating		Deep
Establishing the designing team	Too few members	Deep	Deep
Establishing the coding team	Too few members	Deep	Deep
Establishing the planning team	Too few members	Deep	Deep
Establishing the testing team	Too few members	Deep	Deep
Establishing deadlines	Changing deadlines	Critical	-
	Assigning too short periods of time	Deep	-
	Assigning too long periods of time	Deep	-
Choosing programming language	Choosing the wrong technologies	-	Critical
Choosing database technology	Not compatible with chosen programming language	-	Critical
Buying software licenses	Too many licenses	Low	-
	Too few licenses	Deep	-
	Wrong providers	Critical	Deep
Buying hardware equipment	Wrong technologies	Critical	Deep

Designing	Establishing solution's internal architecture	Not designing according to the type of solution	-	Critical
	Designing data structures and models	Wrong data structures	-	Deep
		Wrong data models	-	Deep
	Designing modules	The module coupling rate too high	-	Deep
	Designing user graphical interfaces	Redundant components	-	Deep
	Establishing database structure	Wrong links between tables	-	Deep
		No indexes	-	Low
Coding	Writing I/O operations classes	Wrong peripheral devices	-	Deep
	Writing database operations classes	Wrong database framework	-	Deep
	Writing business classes	Too many lines of code per class	-	Deep
Testing	Designing test scenarios	Wrong scenarios	-	Deep
	Choosing real test users	Not respecting representativeness	-	Deep
	Modules testing	Wrong test data	-	Deep
	Graphical user interfaces testing	Not testing all components	-	Deep
Releasing	Releasing first version	Not enough testing	-	Deep
	Managing user's feedback	Wrong feedback mechanisms	-	Low

Maintaining	Manage user accounts databases	Wrong role assigning	-	Low
	Manage servers	No mirroring servers	-	Critical

Table 6 – Citizen oriented risk factors

The risk influence factors from Table 6 are presented based on impact on the developing management process and on the final solution. The factors are used for handling and preventing risks according to their impact.

3. MEASURING AND ASSESSING RISK

Within the field of software development project, success rate is low. Risks involved in a software development project are diverse, complex, and difficult to identify and treat. Table 7 presents data regarding the evolution of statistics in terms of success rate of software development projects.

YEAR	*SUCCESSFUL*	*CHALLENGED*	*FAILED*
1994	16%	53%	31%
1996	27%	33%	40%
1998	26%	46%	28%
2000	28%	49%	23%
2004	29%	53%	18%
2006	35%	46%	19%
2009	32%	44%	24%

Table 7 – Software development projects statistics [3]

Successfully completed projects are those that have been completed on time, without exceeding the budget and include all the required functionalities. Failed projects are those that were never completed or that were never launched into the production environment. Challenged projects are those that have exceeded deadlines, exceeded budget or include less functionality than was intended. Table 7 depicts an increase in software development projects completed successfully as success rate doubled from 1994 to 2009, from 16% to 32%. Although the increase is significant the successful projects rate is still alarmingly low. In 2009 only 32% of software development projects were successful, 44% were challenged and 24% failed completely. In [4] statistics show an increase for the year 2010 in terms of software projects implemented successfully to 37%. In 2012 software development project success rate continued to rise and hit 39%. Figure 1 presents software development projects statistics for 2012.

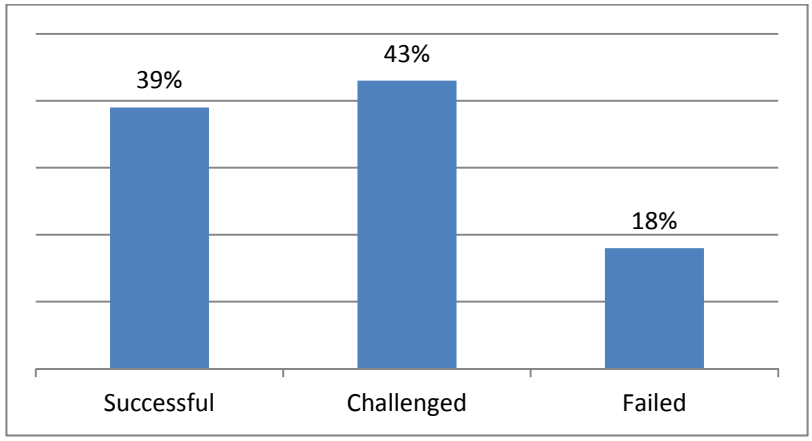

Fig. 1 - Software development projects statistics for 2012 [4]

In 2012, 39% of software development projects were successfully carried out, 43% of projects were completed with problems, 18% of the projects resulted in failure [4]. This is a proof that risk management in software development projects has still significant gaps that need to be addressed. Second research hypostasis is that in software development projects risks are categorized according to their origin and their technical nature.

Technical nature aspect determines technical risks or non-technical risks subcategories. Technical risks are derived from the programming process, technologies, tools, hardware and software components. Non-technical risks are proprietary to the management process and involve planning, motivation and control issues. Technical risks and non-technical risks don not overlap.

Origin aspect determines internal and external risks subcategories. Internal risks concern the project team, project manager, project owner and all actions in which the previous mentioned entities are involved. External risks involve all exogenous elements which have an impact on the project. Internal and external risks do not overlap. Table 8 presents the classification of risks identified in the ALFA project depending on subcategories of the *technical nature* category.

No.	*RISK*	*TECHNICAL NATURE*
1	*If the owner of the project changes requirements during the project development process a cost increase will occur and deadlines will be endangered.*	non- technical
2	*If requirements, objectives and project scope are not properly explained to project team members, than the end result of the project will not match the project owner's expectations.*	non- technical

3	*If the time required to complete a tasks was underestimated, then this leads to postponing other tasks which in the end will lead to not meeting the deadline.*	non- technical
4	*If client data stored on the developer's servers gets compromised or corrupted than developer will face real credibility issue.*	technical
5	*If project costs are underestimated than the desired resource quality in terms of human resources, equipment or information will not be met.*	non- technical
6	*If project team members are not motivated then productivity decreases and delays occur in developing and testing functionality.*	non- technical
7	*If the APIs and libraries that are used for implementing the application don't have proper documentation then integration will take longer than expected and the application deadline will be jeopardized.*	technical
8	*If the project owner does not give timely feedback then delays will occur and the programming process will be delayed.*	non- technical
9	*If one of the team members leaves the project because he doesn't feel that his work is appreciated then the project deadline will be jeopardized.*	non- technical
10	*If the platform that mediates the collaboration is unavailable then developers can't work on the project and delays will occur.*	technical

Table 8 –Risk classification in the ALFA project by *technical nature*

The data presented in Table 8 points out that most of ALFA project risks falls into non-technical subcategory. 70% of identified risks are or of non-technical nature. Table 9 presents the classification of risks identified in the ALFA project depending on subcategories of the *origin* category.

No.	*RISK*	*ORIGIN*
1	*If the owner of the project changes requirements during the project development process a cost increase will occur and deadlines will be endangered.*	internal
2	*If requirements, objectives and project scope are not properly explained to project team members, than the end result of the project will not match the project owner's expectations.*	internal
3	*If the time required to complete a tasks was underestimated, then this leads to postponing other tasks which in the end will lead to not meeting the deadline.*	internal

4	*If client data stored on the developer's servers gets compromised or corrupted than developer will face real credibility issue.*	internal
5	*If project costs are underestimated than the desired resource quality in terms of human resources, equipment or information will not be met.*	internal
6	*If project team members are not motivated then productivity decreases and delays occur in developing and testing functionality.*	internal
7	*If the APIs and libraries that are used for implementing the application don't have proper documentation then integration will take longer than expected and the application deadline will be jeopardized.*	external
8	*If the project owner does not give timely feedback then delays will occur and the programming process will be delayed.*	internal
9	*If one of the team members leaves the project because he doesn't feel that his work is appreciated then the project deadline will be jeopardized.*	internal
10	*If the platform that mediates the collaboration is unavailable then developers can't work on the project and delays will occur.*	external

Table 9 –Risk classification in the ALFA project origin

80% of identified risks are of internal origin. The *technical nature* and *origin* of risks are analyzed together in order to obtain a more accurate risk information. Figure 2 presents a risk breakdown according to risk category.

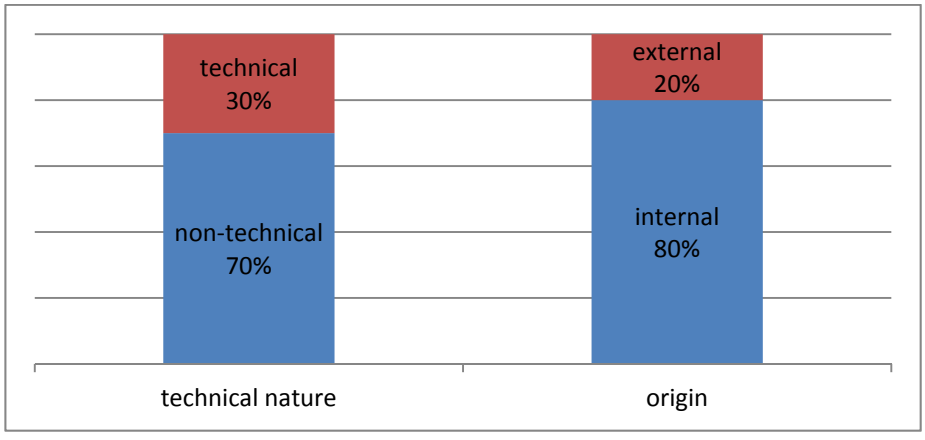

Fig. 2 – Risk breakdown in the ALFA project

Figure 2 it is shows that non-technical risks and internal risks are predominant in the ALFA project. This is information is valuable for the project manager as it pinpoints the nature of risks the project is facing. Non-technical risks are risks that the project manager can tackle himself as they do not require technical skills. Internal risks are risks that the project manager together with the project team can tackle as they have direct access to the source of the risk. Mitigating the risks implies decreasing the probability of occurrence or the overall impact. On internal risks both the probability of occurrence and the impact can be decreased by implementing treatment measures. On external risks only the overall impact can be minimized as occurrence is of external origin and cannot be influenced. All risks identified for the ALFA project have to be evaluated and classified according to their impact on the project development. Table 10 depicts a grading method for evaluating and classifying risks.

GRADING	*DESCRIPTION*	
	Impact	Occurrence
Low	Financial losses are very low, less than 1% from the project's budget.	Unlikely to occur.
Medium	Financial losses are considerable, between 1% and 5% from the project's budget.	Medium likelihood of occurrence.
High	Financial losses are high, more than 5% from the project's budget.	High likelihood of occurrence.

Table 10 – Risk grading method

In order to easily evaluate risks an intuitive grading mechanism is used, thus risks are categorized as *low*, *medium* and *high*. The attributes or risks are impact on the project and probability of occurrence. The impact is evaluated as financial loss caused to the project. Low impact risks cause a financial loss of less than 1% from the project's budget. Medium impact risks cause a financial loss between 1% and 5% from the project's budget. High impact risks cause a financial larger than 5% from the project's budget.

In terms of likelihood of occurrence risks that are graded as *low* are unlikely to occur and have a probability of occurrence below 0,1. Risks that are graded as *medium* have a probability of occurrence between 0,1 and 0,3. Risks that are graded as *high* have a probability of occurrence greater than 0,4. In order to simplify the grading of risks actual probability are not used as they tend to confuse personnel involved in the risks identification process. Instead of probabilities phrases like *Unlikely to occur, Medium likelihood of occurrence* and *High likelihood of occurrence* are used. Team members and other personnel involved in identifying and handling risks can relate easier with phrases that they use on a daily basis than with probabilities. If required for build a risk budget, probabilities can be determined by specialized risk handling personnel. Using the risk grading method presented in Table 10 risks identified for the ALFA project were evaluated. The result of the evaluation process is depicted in Table 11.

No.	*RISK*	*GRADING*
1	*If the owner of the project changes requirements during the project development process a cost increase will occur and deadlines will be endangered.*	Medium
2	*If requirements, objectives and project scope are not properly explained to project team members, than the end result of the project will not match the project owner's expectations.*	Medium
3	*If the time required to complete a tasks was underestimated, then this leads to postponing other tasks which in the end will lead to not meeting the deadline.*	Medium
4	*If client data stored on the developer's servers gets compromised or corrupted than developer will face real credibility issue.*	Low
5	*If project costs are underestimated than the desired resource quality in terms of human resources, equipment or information will not be met.*	Low
6	*If project team members are not motivated then productivity decreases and delays occur in developing and testing functionality.*	Medium
7	*If the APIs and libraries that are used for implementing the application don't have proper documentation then integration will take longer than expected and the application deadline will be jeopardized.*	Low
8	*If the project owner does not give timely feedback then delays will occur and the programming process will be delayed.*	Low
9	*If one of the team members leaves the project because he doesn't feel that his work is appreciated then the project deadline will be jeopardized.*	High
10	*If the platform that mediates the collaboration is unavailable then developers can't work on the project and delays will occur.*	Low

Table 11 – Risk grading for the ALFA project

According to data in Table 11, 50% of risks identified for the ALFA project were graded as *low*, 40% of risks were graded as *medium* and 10% of risks were graded as *high*. The risk graded as *high* is the fact that one of the developers might leave the project team. The risk is an internal, non-technical risk that was highlighted by two sources: project team and previous experiences. Risks have to be handled according to the grading system that was implemented. Risks graded as *high* are the first on the handling list, followed by *medium* and *low* risks. For the ALFA project only the *high* and *medium* risks were handled. The cost of handling risks is also important as it has to be correlated with the risks grading and it should not exceed the risks impact.

4. MONITORING AND HANDLING RISK

The nature of risks within software projects demands an activity of handling and monitoring the risk factors. Developing a mechanism for preventing, monitoring and controlling is imperious necessary [6]. The approach of this activity presents a high level of efficiency based on the experience of team members gained by handling risk situations in similar software projects. In both innovation based software projects and in citizen oriented software projects the activities for preventing, monitoring and controlling are realized based on risks characteristic to the type of the developed application. Risk monitoring implies building a database, harvesting and recording data about every identified risk situation. This database is used for multiple software projects, projects in which the developing team has been involved. Data in this database is relevant for the similar types of projects. The recorded data for the monitoring activity is:

- **project identifier** represents the codification for the name of project on which the monitoring activity realized; for an efficient centralization the codification is realized using positive integer numbers or generating GUID identifiers;
- **project type** is based on the developed application; for the undergoing developing project relevant information is the one provided from the same type of applications; the criteria the types are based on is: authentication mode, online running mode, offline running mode, electronic payments supporting, interactivity level;
- **risk identifier** is chosen similarly to the way of choosing project's identifier;
- **occurrence moment** is recorded in the format of date and time relevant to the date and time when the event qualified as risk situation has occurred;
- **influence factors** are recorded as events and actions which has influenced the materialization of risk situation; data about influence factors is used for preventing risks thus an highly accurate description is necessary;
- **successfully/unsuccessfully solving** represents a grade given as a result of using the control mechanism for solving the occurred risk situation; the unsuccessful in solving the negative influences of the risk situation may lead to the impossibility to carry on with the project; based on the given grade the impact on the project is calculate
- **risk solving time** is calculated as the time interval between risk situation's occurrence and the granting of the grade for successfully/unsuccessfully solving;
- **risk solving costs** are calculated based on the estimated use of resources for solving the risk situation; resources are: the time spent by team members in handling the risk, hardware resources, software licenses.

Risk monitoring activity is an ongoing process during prevention and control activities. During this stage data recorded in the database is processed and used for estimations on the identified risks. This is the reason why a big number of records is necessary and the database is continuously and progressively updated. A way of increasing the recorded volume of data is collaboration with software developing teams involved in similar software projects and information exchange about risk factors. The process of harvesting data about risk occurrences is presented in Figure 3.

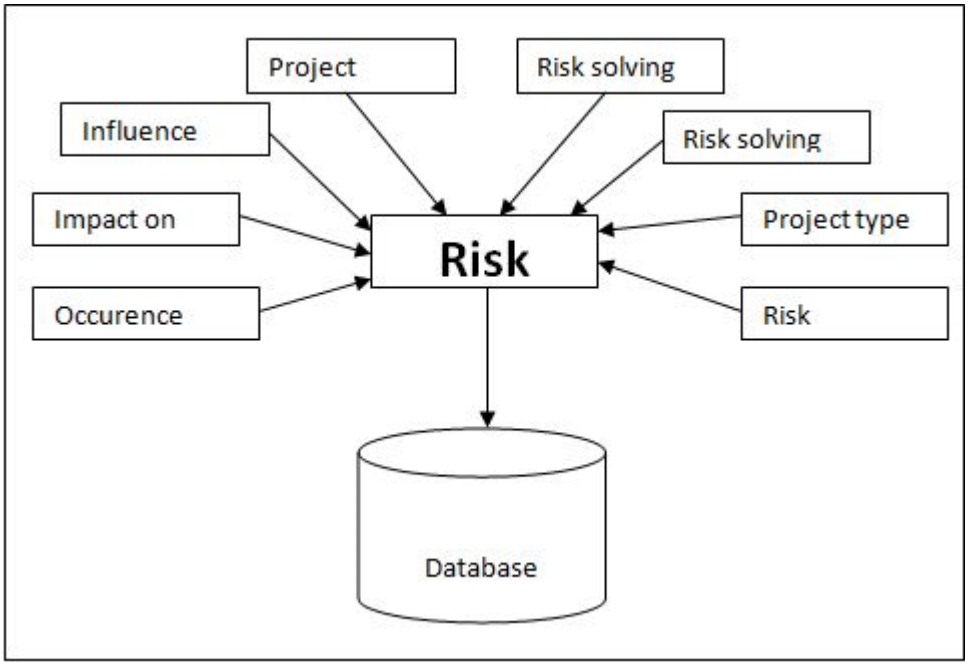

Fig. 3 – Harvesting data about risk occurrences

Centralized and standardized harvested data is stored into risk monitoring database and used through the developing stages of the project. Handling risks for ongoing projects implies preventing and controlling mechanisms. Preventing mechanisms are used on each stage of the project based on identified risk factors in order to prevent their materialization. It is important to prevent the risk during the ongoing stage because returning to previous stages to resolve and control risk problems it is very difficult and implies high costs. One change done going steps back in the developing stages has influence over other components of the project. Sometimes reiteration of all activities of the previous developing step is necessary.

5. CONCLUSIONS

Proper risk handling increases the success rate of software development projects. The project manager can identify specific risks by analyzing previous projects, consulting with the project owner and with the project team. In order to identify risks checklists, questioners, direct interviews and brainstorming sessions are used. In citizen oriented applications risks are defined as *low, critical* and *deep.* The development of projects for citizen oriented applications is a process that follows a pattern of stages which succeed in a specific order. Each stage of the development process has specific risks. Software development project risks have to be categorized according their technical nature and source. Technical nature has two risk subcategories: technical and non-technical. According to source risks belong to either internal or external subcategory. Risks are graded in terms of impact on the projects budget and in terms of likelihood of occurrence. The grading system for risks includes *low*, *medium* and *high* risks. Risks have to be handled according to the impact they have on the project's budget and on their probability of occurrence. Risks graded as *high* are the first on the handling list, followed by *medium* and *low* risks. The

nature of risks within software projects demands an ongoing activity of handling and monitoring the risk factors.

6. REFERENCES

[1] Project Management Institute - *A Guide to the Project Management Body of Knowledge*, Publisher: Project Management Institute; 5 edition, 2013, pg. 589, ISBN-13: 978-1935589679

[2] J. DHLAMINI, I. NHAMU, A. KAIHEPA - *Intelligent risk management tools for software development*, Proceedings of the 2009 Annual Conference of the Southern African Computer Lecturers' Association, 29 Jun. – 1 Jul. 2009, Mpekweni Beach Resort, Eastern Cape, South Africa, Publisher: ACM, 2011 ISBN 978-1-60558-683-0, pg. 33-40

[3] J. L. EVELEENS, C. VERHOEF - *The Rise and Fall of the Chaos Report Figures*, IEEE Software, vol. 27, no. 1, pg. 30-36, 2010, ISSN : 0740-7459

[4] STANDISH GROUP - *Chaos Manifesto 2013 Think Big, Act Small*, available at: http://versionone.com/assets/img/files/ChaosManifesto2013.pdf

[5] Q. KHAN, S. GHAYYUR – *Software Risks and mitigation in global software development,* Journal of Theoretical and Applied Information Technology, Vol. 22, No. 1, 2010, pp. 58-69, ISSN 1992-8645

[6] C. BRANDAS, O. DIRAGA, N. BIBU – *Study on Risk Approaches in Software Development Projects*, Informatica Economica Vol. 16, Issue 3, 2012, pp. 148-157, ISSN 1453-1305