Competitive Risk Identification Method for Distributed Teams

Yegor Bugayenko

TechnoPark Corp.
568 9th Street South 202
Naples, Florida 34102
egor@technoparkcorp.com

Abstract. The described method is a risk identification scenario for software development projects, where the project team is multi-lingual and distributed, time for the risk identification meeting is limited, meetings are recurrent and an amount of risks required is bigger than a hundred. The meeting is conducted as an online chat game, where participants compete for each risk source, inventing the most severe risk. The winner gets a bonus, while the meeting facilitator gets a big list of raw risks.

Keywords. Risk Management, Distributed Team, Risk Identification, Distributed Meetings

1 Introduction

One of the most important "process area" in project management is risk management, which includes risk planning, risk identification, qualitative and quantitative analysis, risk response planning and risk monitoring [1]. The risk identification process provides the material for risk analysis and risk response planning. A raw list of risks, that shall include hundreds of them [2, pp. 61–102], can be generated by different methods, including brainstorming [3], historical records, checklists and templates [4], risk charting, objectives-based, scenario-based, taxonomy-based [5], conduct a "pre-mortem" [6], Affinity Diagrams [7, pp. 135–141], Delphi technique [8, 9], expert interviews, Nominal Group Technique [10, 11], and others.

Every method has its own advantages and drawbacks [12, 16, 13, 14, 15]. However, any of them when applied to a project with the following constraints, will fail to produce a required result: *a*) project team is multi-lingual and distributed (online text chats only), *b*) risk identification meetings must take less than one hour, *c*) meetings are held regularly (every iteration), and *d*) each meeting shall produce at least a hundrend risks.

Existing methods will fail in these circumstances because of (most common causes): *a*) Inattention; *b*) Language barriers; *c*) Unavoidable personal criticism; *d*) Weariness after repeating meetings; and *e*) Untrained risk identifiers.

A good solution to the outlined problems could be a method that will reduce the amount of efforts required for risk identification, at the same time increasing personnel engagement and motivation.

2 Method

The purpose of this method is to increase the effectiveness of risk identification meetings in software development projects, reduce the time and effort required for the meetings and make the meetings possible to be held in online text chat form.

The meeting facilitator prepares and presents to the meeting participants the list of five key project objectives and a list of risk sources (up to dozen). The objectives go horizontally, while risk sources vertically. The matrix becomes a field for the Risk Game.

The rules of the Risk Game are:

1. Facilitator announces the next row (risk source);
2. Everyone invents risks for the given risk source;
3. Facilitator chooses the winner for the row;
4. Matrix gets the name of the winner in a corresponding cell;
5. The person who has the most cells wins the game.

The meeting recorder maintans the list of all invented risks. This list is passed to the facilitator by the end of the meeting. The result of the Risk Game is a long list of raw unsorted risks, which will be used by the project management for quantitative and qualitative risk analysis.

The proposed method is more effective than other existing methods when meeting time is limited (less than one hour) and a big list of risks is required (more than a hundred). The method is more effective in such circumstances because it quickly involves everybody in the process, converts individual cricism into a fair competition and stimulates group thinking.

3 Practical Example

Figure 1 illustrates a workflow of the method, starting with the definition of project objectives, risk sources and a list of meeting participants (`101`). Project objectives are the most important "targets" that the project has to achieve in order to be successfully closed. Risk sources are facts that may become root causes of risks. Sources are something that already happened, while risks are events that may happen and have negative (or positive) effects for at least one project objectives.

It's desired to have a limited number of project objectives (up to 5) and a limited amount of risk sources (less than a dozen). Bigger numbers will make the risk identification meeting too long and not so effective, mostly because participants won't stay focused.

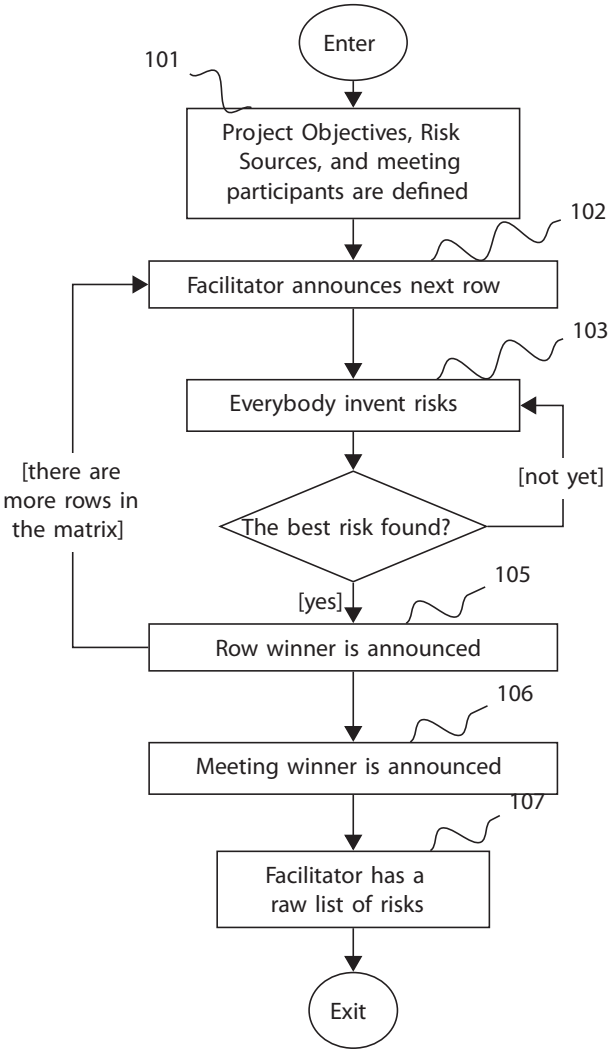

Fig. 1. Method flow chart that illustrates interconnections between key activities of the described method. The activities are performed during one risk identification meetings taking totally 40-60 minutes.

The facilitator asks all meeting participants to draw a risk matrix (example is on Figure 2), where horizontally they place project objectives and vertically risk sources. Each meeting participant has such a matrix locally (we assume that the meeting is help in a virtual online chat environment).

Then the meeting goes in iterative manner, from risk source to risk source, down the risk matrix. For each next row in a matrix the facilitator announces a risk source (`102`) and asks meeting participants to invent and propose possible risks. The risks identified should have the declared source and should affect one or more project objectives. Meeting participants invent risks (`103`) and send them in more or less universal format to the chat.

The faciliator waits until the best risk is found, according to his/her own understanding and criteria. As soon as such a risk is found, facilitator announces the winner of the row (`105`) and goes to the next row.

At the end of the meeting the facilitator announces the winner of the meeting (`106`) – the person who won the most of rows. The most important result of the meeting is the raw list of risks (`107`), which will be long enough for any project size.

A full-scale identification meeting with a team of 6-10 participants shall take 40-60 minutes, if the facilitator has some experience of using the method. The meeting goes in a very aggressive and competitive manner, challenging all participants and getting the maximum of their creativity. Even better meeting performance could be achieved by means of a monetized award to the meeting winner.

Figure 2 is a sample risk matrix, used during a risk identification meeting. Horizontally it has a list of project objectives and vertically a list of risk sources. Cells of the matrix are placeholders for risk, identified during the meeting.

In total, there were 70 risks identified with this matrix. It is important to note that the amount of risks identified does not say anything about how "risky" the project is. The only thing it indicates is the quality of risk identification outcome.

It is known that any software product has an unlimited amount of defects [17, pp. 9–20]. A similar statement is applicable to risk identification: "Any project has an unlimited amount of risks". The task for the project manager is to identify the most critical of them.

Using the proposed method risk identification may be a very iterative and repetitive process. If project manager feels that there is not enough risks identified in the risk list, he/she can organize additional risk identification meetings, make some changes to the Risk Game Matrix and make some changes to the meeting team. A new meeting will produce new useful raw list of risks.

4 Conclusion and Future Research

The described method was invented and implemented in TechnoPark Corp. in June 2008. Since that time the method was applied to seven commercial projects. In total, 40+ risk identification meetings using this method have been already

	Project achieves all baselined objectives in budget	Proof-of-concept prototype is deployed to users on 10-Aug	Web site performance of 500 pages per second	Test code coverage measured by LoC is over 80%	Over 150 critical defects discovered and fixed
Lack of technical experience in used technologies (J2EE and Web Services)	4	2	3	4	1
Budget shortage (underestimated project)	2	3	1	2	3
Vague end-user requirements	3	4	1	1	2
Distributed and multi-lingual team	1	3	4	2	6
Deployment platform is provided by customer and is not ready yet	2	3	7	2	4
Half of project team members are new workers to Project Manager	3	4	0	0	1

Fig. 2. Risk Game Matrix is a key artifact used during the Risk Game. Risk Sources are listed vertically and Project Objectives are listed horizontally. Cells of the matrix include numbers of risks suggested by meeting participants and recorded by the meeting facilitator.

conducted. In comparison with previous projects a number of advantages were received:

- We enabled an effective distributed and multi-lingual risk discussions and identification;
- Risk management is not a boring bureaucracy any longer, but is a challenging game. All team members stay focused on risks and their pro-active identification;
- We significantly reduce project expenses due to much more throrough risk identification;
- We keep risk identification meeting protocols in text files accessible for all project participants, including the customer. Thus, we optimize communication and avoid loss of information.

In the next years we are going to collect more numeric results afer the method application and give more formal proof of its effectiveness. We are also thinking about inventing a similar method for risk response planning.

References

1. Project Management Institute, Project Management Body of Knowledge (PMBOK) Guide v.3, PMI Press, 3rd edition (2004)
2. Mulcahy, R.: Risk Management, Tricks of the Trade for Project Managers, RMC Publications, Inc., USA (2003)
3. Osborn, A.F.: Applied imagination: Principles and procedures of creative problem solving, 3rd Revised Edition, Charles Scribners Sons., New York, NY (1963)
4. National Cyber Security Division of the U.S. Department of Homeland Security, Common vulnerabilities and exposures, Technical report, The MITRE Corporation, USA (2008)
5. Carr, M.J., Kondra, S.L., Monarch, I., Ulrich, F.C., Walker, C.F.: Taxonomy-based risk identification, Technical Report CMU/SEI-93-TR-006, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA (1993)
6. Mitchell, J.E.R., Deborah, J., Pennington, N.: Back to the future: Temporal perspective in the explanation of events, Journal of Behavioral Decision Making, vol. 2, pp. 25–39 (1989)
7. Britz, G.C.,, Emerling, D.W., Hare, L.B., Hoerl, R.W., Janis, S.J., Shade, J.E.: Improving Performance Through Statistical Thinking, American Society for Quality, USA (2000)
8. Linstone, H.A., Turoff, M.: The Delphi Method: Techniques and Applications, New Jersey's Science & Technology University, NJ, USA (1975)
9. Schmidt, R., Lyytinen, K., Keil, M., Cule, P.: Identifying software project risks: An international delphi study, Journal of Management of Information Systems, 17(4), pp. 5–36 (2001)
10. Delbecq, A.L., VandeVen, A.H.: A group process model for problem identification and program planning, Journal Of Applied Behavioral Science VII, July/August, pp. 466–491 (1971)
11. Delbecq, A.L, VandeVen, A.H., Gustafson, D.H.: Group Techniques for Program Planners, Scott Foresman and Company, Glenview, Illinois (1975)
12. Freimut, B., Hartkopf, S., Kaiser, P., Kontio, J., Kobitzsch, W.: An industrial case study of implementing software risk management, In ESEC/FSE-9: Proceedings of the 8th European software engineering conference held jointly with 9th ACM SIGSOFT international symposium on Foundations of software engineering, pp. 277–287, New York, NY, USA (2001)
13. Keil, M., Cule, P.E., Lyytinen, K., Schmidt, R.C.: A framework for identifying software project risks, Communications of the ACM, 41(11), pp. 76–83 (1998)
14. Wallace, L., Keil, M.: Software project risks and their effect on outcomes, Communications of the ACM, 47(4), pp. 68–73 (2004)
15. Fairley, R.E.: Software risk management, IEEE Software, 22(3), pp. 101 (2005)
16. Charette, R.N.: Software engineering risk analysis and management, McGraw-Hill, Inc., New York, NY, USA (1989)
17. Myers, G.J.: The Art of Software Testing, 2nd Edition, John Wiley & Sons, Inc., NJ, USA (2004)