Annals of DAAAM for 2010 & Proceedings of the 21st International DAAAM Symposium, Volume 21, No. 1, ISSN 1726-9679
ISBN 978-3-901509-73-5, Editor B. Katalinic, Published by DAAAM International, Vienna, Austria, EU, 2010
Make Harmony Between Technology and Nature, and Your Mind will Fly Free as a Bird
Annals of DAAAM International

TURNING NEARSHORING INTO A SUCCESS MANAGING TECHNICAL BACKGROUND DIFFERENCES

MOLDOVEANU, A[lin] D[ragos] B[ogdan]; ASAVEI, V[ictor]; MOLDOVEANU, F[lorica]; MORAR, A[nca] - A[ndreea]; EGNER, A[lexandru] I[onut] & BOIANGIU, C[ostin] A[nton]

Abstract: *We present here the results of a 3 year long nearshoring experiment, conducted in a semi-controlled academic environment. The experiment involved teams of 10-15 people from 6 European universities and proved that the biggest issue in nearshoring lies in the difference between the 2 involved teams in terms of technical background differences.*
Key words: *nearshoring, offshoring, outsourcing, background*

1. INTRODUCTION

As types of outsourcing, offshoring and nearshoring are important means of reducing software costs and sometimes also improving quality. However promising, they must be perfectly understood as they come with intrinsic risks. Software life cycle and team knowledge must be carefully chosen and adapted.

This article presents an academic nearshoring experiment involving teams from 6 universities of UE countries and the resulted lessons and conclusions.

2. IT OFFSHORING / NEARSHORING

2.1 Offshoring vs. Nearshoring

Outsourcing is one of the modern successful approaches to reduce costs (and sometimes also improve quality) in software development and also in other hi-tech areas.

While "outsourcing" appeared during the 1980s and there are a lot of success stories and specialized structures that promote outsourcing, it still comes with some particularities that need to be considered carefully to eliminate risks (Wiener, 2007; Karle & Schoenthaler).

Offshoring means outsourcing to a company from another country. Offshoring appeared from economical reasons, due to the lower cost of software development in some countries from Asia, like India and China. Many countries from USA, Canada and some from west Europe have obtained huge benefit on cost savings by offshoring in such countries due to wage difference of cost of hiring and training combined with high quality skills.

Nearshoring is a particular case of offshoring – to providers in nearby countries. Numerous companies from West-Europe offshore their IT processes and services to companies from Central and Eastern Europe (Deutsche Bank Research, 2006).

Compared with offshoring, nearshoring offers benefits:

- Geographic proximity, enabling easy deployment of project team members, during the project's critical phases;
- Cultural closeness, an identical time zone and the service provider's knowledge of the local and English languages facilitates the communication between the offshoring team and the provider's team, project management and control;
- Shorter development times

So basically nearshoring reduces the risk of offshoring, since the outsourcing country is closer geographically and in cultural aspects but still lower in cost than the local country.

Compared with traditional offshoring locations CEE wages are higher but communication more efficient (Deutsche Bank Research, 2006). The region is recommended for more sophisticated services. For simpler IT services the traditional offshore locations, such as India, should be preferred.

2.2 Issues With Off and Near Shoring

The economic significance of offshoring is debated. Dislocation of technology in the provider's country reduces employment in the offshoring company's country. But, offshoring / nearshoring are a form of distributed software development, which has some valuable benefits. Thus, besides the cost reducing, the strategic reasons for a company to decide for distributed software development can be to access the global resource pools and to speed up the time-to-market.

The offshoring, and generally, the distributed software development, is now a challenge. Studies reported many failures of outsourced projects (Wiener, 2007; Karle & Schoenthaler). The main reasons of these failures seem to be:

- The lack of adequate software development methodologies;
- Insufficient standardized methods and tools for software specification, architectural design and test cases specification (and automatic generation);
- Ignorance of the risks of offshoring;
- Inadequate project management;
- Insufficient control of the resulted software product quality.

Many IT specialists analyze such advanced issues about special methodologies/tools needed for offshoring.

However, our opinion is that most of the failures in offshoring projects have simpler reasons, mainly residing in cultural and technical background differences between the offshoring company and the offshoring provider – which we try to pinpoint and tackle in this paper.

3. OUR NEARSHORING EXPERIMENT

Our experiment simulated nearshoring activities in an academic controlled environment. We wanted to observe directly the issues with nearshoring activities, trace and identify their primary reasons.

The experiment was hosted by the European Intensive Program "Nearshoring: the next step in Offshoring" (Moldoveanu & Moldoveanu, 2008), involving:

- Hogeschool van Amsterdam (HVA), Netherland
- University POLITEHNICA Bucharest, Romania,
- Mid Sweden University, Sweden
- EVTEK Institute of Technology, Finland
- Politechnica Krakowska, Poland
- Technical University of Ostrava, Czeck Republic

Each university was involved with a team of 10 students, supervised by 2-3 professors.

The experiment evolved through a period of 3 years.

During each year a couple of IT projects were proposed and the teams were paired: one team played the role of the company that uses outsourcing services (teams C –contractor) and other team was the outsourcing service provider (teams SC – subcontractor). There were 3 distinct phases in each year:

- Creation of specifications for IT projects, by each team C.

- Implementation following specifications, by each team SC.
- Meeting with all the participants, presenting the executable programs and discussing the results.

Each year we incorporated conclusions of previous years in the methodology and the instructions given to student teams.

The professors' supervisor role was minimal – just organizational stuff and observing the activity. Teams were supposed to be autonomous to emulate a real environment. They were free to use any development methodologies they considered fit to their expertise and the scope of the projects.

The resulted environment, though slightly controlled and fully observable, was free and natural enough to emulate a real environment, and we can extrapolate the results to industry.

4. RESULTS OF THE EXPERIMENT

4.1 Observed Aspects

Each year, we tried to observe the following aspects:

- the ability of each team to define User Requirements, System Requirements and Software Requirements;
- the methods used by teams from different countries;
- the adequacy of easy method to nearshoring;
- if the specification were well understood;
- how much of the specified functional and non-functional requirements were actually implemented;
- if students building the system followed the design and testing recommendations included in the specification;
- how each team pair has communicated.

We observed these aspects both direct and through questionnaires conducted at the end of each year.

4.2 Main Issues

As a rough summarization, the main issues observed during the nearshoring experiment felt into the following categories:

- Huge technical background differences between the contractor and subcontractor teams, especially related to Analysis & Specification activities and deliverables.
- Usage of various software development methodologies by the contractor and subcontractor teams. For example, during the 1st year of the experiment, the Dutch team acting as subcontractor used most of the time to build an incomplete version of the system, because they were used to the DSDM method, that doesn't require building a complete product in the first iteration.
- Inability to check the actual understanding of the specification – led to waste of time during implementation and even to implementation of wrong functionalities.
- Communication issues

Some of these were gradually corrected to some degree, by adjusting the rules each year to avoid the issues.

Our conclusion was that, apart from advanced issues that experts try to handle, like advanced methodologies and tools, seems that a huge part of the issues in nearshoring can come from different technical background of the 2 involved parties.

4.3 Background Differences and Nearshoring

Different teams have different backgrounds and perspectives on using even standardized software development methods. This is even more likely when the teams are from different countries or from universities with different targets. For example, the majority of the participating students have got, before the project, programming and database courses, but only one Software Engineering course. But Dutch team have got initial courses in Java, UML, RDBMS and ERD, and they have done projects using Prince2 and DSDM.

Cultural background seems to have minimal or no influence in European nearshoring. There is enough convergence in the various European cultures and habits to provide a solid foundation for any nearshoring IT project. More important in this case are the educational background and work habits.

5. CONCLUSION

Even limited as amplitude, our study was relevant enough to highlight deep reasons for typical offshoring/nearshoring issues and even to suggest ways to prevent them.

Future research should aim at experimenting and measuring the effectiveness of these ideas we conclude below.

In the offshoring/nearshoring perspective, it is important to exist a preliminary phase, when the two teams can present and discuss the methods, the documents templates and the notations which will be used. This is necessary because there are not standards that define the content and the format of the requirements documents, not enough formalized and spread software development methods.

It is important to choose the appropriate methods, taking into account the specifics of the project, the specifics of the nearshoring and the level of specialization the teams involved. For example, the DSDM seems to be inappropriate in the nearshoring context, as we mentioned before.

Apparently at least, classical software development methodologies like waterfall or the iterative and incremental life cycle, both supported by detailed specifications, seem to give better results than the more modern but complicated approaches (Betz & Mäkiö, 2007) – probably due to the fact that they are easier to understand and provide clear deliverables and checkpoints for both parties involved in a nearshoring.

Strong communication between the involved teams is very important. Reports and daily online meetings are very useful, assuring the detection of misunderstandings and the possibility of immediate correction. In particular, is very important to check upfront then repeatedly if both parties have a common understanding of the methods and expected deliverables.

Any company involved in the offshoring should see it also as a learning process. Following the above rules has proven, in the 3rd year of our experiment, to allow the success of most the proposed IT projects, which were medium-sized projects. Big scale projects would also benefit from this approach, of course, combined with more advanced tools for collaboration and integrated software analysis, design and testing.

6. REFERENCES

Betz, S. & Mäkiö J. (2007), "Amplification of the COCOMO II regarding Offshore Software Projects", Proceeding of the Workshop on Offshoring of Software Development-Methods and Tools for Risk Management at the second International Conference on Global Software Engineering 2007, ISBN: 978-3-86644-203-0.

Deutsche Bank Research (2006), "Offshoring to new shores - Nearshoring to Central and Eastern Europe", *Available from:*http://www.dbresearch.com/PROD/DBR_INTERNET_EN-PROD/PROD0000000000201757.pdf *Accessed:* 2010-06-13

Karle, T. & Schoenthaler, F., "Prevention of Failure Situations in Offshore Software Projects", *Available from:* http://www.promatis.de/fileadmin/user_upload/documents/Failure_Situations_Offshoring.pdf *Accessed:* 2010-06-13

Moldoveanu, F.; Moldoveanu A. (2008), The influence of technical background differences in nearshoring projects – Conclusions of the IP "Nearshoring: the next step in Offshoring",*Available from:* http://ip-nearshoring.cs.vsb.cz/index.php?option=com_content&view=category&layout=blog&id=9&Itemid=13 *Accessed:* 2010-06-13

Wiener, M. (2007), "Successful Offshore Software Development", ICGSE 2007, *Available from:* http://www.outshore.org/LinkClick.aspx?fileticket=061-GMmDrlI%3D&tabid=58&mid=387 *Accessed:* 010-06-13