Contextual Factors Affecting the Software Development Process – An Initial View

Alexandre Bern
Lappeenranta University of Technology,
Lappeenranta, Finland
bern@lut.fi

Satya Jaya Aparna Pasi
Lappeenranta University of Technology,
Lappeenranta, Finland
pasi@lut.fi

Uolevi Nikula
Lappeenranta University of Technology,
Lappeenranta, Finland
unikula@lut.fi

Kari Smolander
Lappeenranta University of Technology,
Lappeenranta, Finland
ksmoland@lut.fi

ABSTRACT

This paper presents the preliminary results of a research project where the context of software development is observed. The goal of this study is to observe, extract, and categorize the contextual factors that have an effect on the software development process. The study uses theme-based interviews as its data collection method and approaches the research question qualitatively. Various managers, developers, and analysts in four organizations were interviewed and the factors were extracted by interpreting them from the interview transcripts. Altogether nine contextual categories were identified. The categories are described and the need for future research is discussed.

Keywords

Software development, development context, critical success factors, contingency theory, process improvement.

INTRODUCTION

The failures in software development projects have been the topic of numerous studies to date (McFarlan, 1982; The Standish Group, 1994) and many risk identification frameworks have been suggested to alleviate these problems (Barki and Rivard, 1993; Boehm, 1991; Keil, Cule, Lyytinen and Schmidt, 1998; Lyytinen, Mathiassen and Ropponen, 1996; Schmidt, Lyytinen, Keil and Cule, 2001). However, these frameworks are usually focused on the project management level, and provide little help in deciding what kind of software development practices would fit each organizational and project setting. In particular, how different contextual factors affect the software development processes has not received much attention so far.

In the literature the fit between a software development process and its context has been approached mainly from the contingency theory point of view. One of the central assumptions of the contingency theory is that the better the fit there is between the studied contingency variables (e.g., process, developers, and tools), the better the performance is (Hardgrave, Wilson and Eastman, 1999). This approach has been used, for example, to develop a model to help in choosing information system prototyping strategies (Hardgrave et al., 1999), software project coordination strategies (Andres and Zmud, 2001), and to manage software project risks (Barki, Rivard and Talbot, 2001). However, the contingency approach has also raised a lot of objections mainly due to fact that in any given study the contingency variables only account for a small percentage of the variance in organizational performance (Weill and Olson, 1989).

An increasing amount of research has focused on the critical factors of systems development. The critical success factors have been defined as "the few key areas where *things must go right* for the business to flourish" (Rockart, 1979, p. 217), and both quantitative and qualitative studies have been conducted to identify them, for example, in the context of software process improvement (Rainer and Hall, 2003; Trienekens, Kusters, van Genuchten and Aerts, 2007), project management (Phan, Vogel and Nunamaker, 1995; Rose, Pedersen, Hosbond and Kraemmergaard, 2007), and other detailed topics in software and information systems development (e.g., Layman, Williams, Damian and Bures, 2006; Sumner and Ryan, 1994). Butler and Fitzgerald (1999) provides rare insight on the identification of critical success factors in one organization, and Sambamurthy and Kirsch (2000) have developed an integrative framework for the information systems development process based on the existing literature.

Other approaches that try to link the development process with the domain specific aspects include domain engineering and situational method engineering. Domain engineering has been defined as "the systematic process collecting, organizing, and storing past experience in building systems in a particular domain" (Czarnecki,

2002). Even if the reuse of existing artifacts is an intuitive concept in software engineering, Glass and Vessey (1998) report that little real results from the domain specific approaches has been achieved. After the publication of their article, research efforts underlining the differences between different domains (e.g., Baskerville, Ramesh, Levine, Pries-Heje and Slaughter, 2003; Ramesh, Pries-Heje and Baskerville, 2002) and the benefits of domain specific approaches (Port and Dincel, 2001) have been reported. The situational method engineering focuses on the fact that each project situation is actually unique. The approach has been defined as "the engineering discipline to design, construct and adapt methods, techniques and tools for the development of information systems" (Brinkkemper, 1996), and the idea is to construct each method separately for each project based on the prevailing situation utilizing the existing method fragments.

The issue of fit between context and the process, or form, was originally raised by Alexander (1964). In this early book the *form* was defined as the part of world over which you have control, and the *context* as everything that you cannot control but that puts demands on the form. In this paper we will approach the software development process in the same way – we report our initial results of a study where we seek to identify contextual factors that affect the software development process. The goal of our study is to identify the contextual factors empirically in actual work context in multiple companies, integrate the company specific findings to a common framework depicting the relationships between the identified factors, and finally use the developed framework to plan process improvement actions in the studied companies. However, as this paper reports research in progress, we will only present the initial view on the categories and factors affecting the software development process.

The rest of the paper is structured as follows. The next section describes the research process and it is followed by a section that presents the categories of the contextual factors and discusses shortly about the effects of the factors. The complete list of identified categories and associated factors are presented in Appendix A. The final section concludes the paper and discusses the completion plans of the study and possible implications of the results.

RESEARCH PROCESS

This study explores the context of software development in software development organizations. The study is a part of a larger project aiming at process improvement in the participating companies with action research approach. To establish a solid foundation for the action phase, the project was started with a problem analysis phase to identify central problems in the companies. Since the study involved only few companies, quantitative methods were deemed unsuitable for the phase, and qualitative methods were chosen instead. Baskerville and Pries-Heje (1999) propose the use of grounded theory (Glaser and Strauss, 1967; Strauss and Corbin, 1990) in the theory formulation part of an action research study. Since grounded theory has also been found suitable for the study of software and information systems (e.g. Baskerville et al., 2003; Coleman and O'Connor, 2007; Hansen and Kautz, 2005; Kirsch and Haney, 2006; Paré and Elam, 1997; Seaman, 1999), and especially suited to analyze interview data (Calloway and Knapp, 1995), we decided to base this study on the grounded theory approach.

The study used theme-based interviews as its main data collection method. When designing the themes for the study, we formed a picture of the research topic (Figure 1). Software development processes are used by development projects in a company context, and the company operates in a business environment. We also put some examples of expected factors in the picture so that we could use the picture to guide the interview theme and question development. The picture and its factor examples were not, however, used in the analysis. Instead, we progressed in data analysis as inductively as possible.

In total, we used 23 interviews from 4 organizations as the data material for this study (see Table 1). Common to all the organizations were that the major part of their business is software development. All interviews were audio recorded and transcribed to text. The total amount of recordings adds up to 26 hours and the number of transcribed pages is in total about 600 pages.

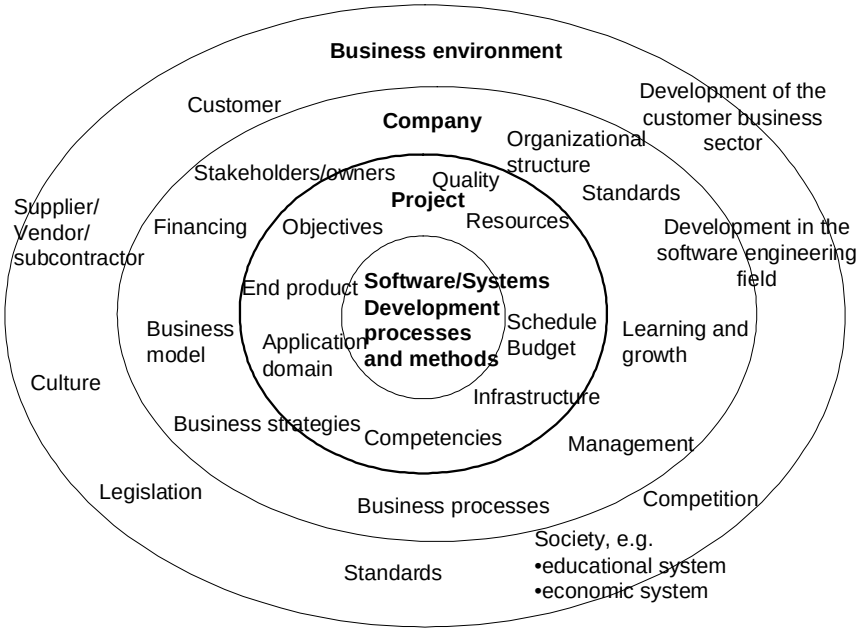

Figure 1. The research topics of interest

Org.	Offered products and services	Number of employees	Interviews
A	Operative systems for industry	800	7 (2 upper managers, 3 project managers, 2 developer)
B	Automation systems for industry	100	6 (4 upper managers, 2 project managers, 1 developer)
C	Software development services, subcontracting	200	5 (2 upper managers, 1 project manager, 2 developers)
D	Systems for a specific kind of resource planning	60	5 (3 upper managers, 2 developers)

Table 1. Observed organizations

The data analysis is still at the time of writing an ongoing process. The analysis started with open coding (Strauss and Corbin, 1990), where three researchers went through the data and identified sections that represent a contextual phenomenon. These identified sections were conceptualized and recorded as codes. Later, all the authors of this paper held a workshop, where all the identified codes were classified in more broad categories and the categories were given a descriptive name, as shown in the next section. Currently we are moving to axial coding (Strauss and Corbin, 1990), where the relationships between the categories are in focus.

INITIAL RESULTS

In the analysis, the researchers looked for any kind of contextual factor that had an effect on software development process. By a contextual factor, we mean any kind of affecting phenomenon that exists in the environment of the software development process. Even if theories classifying the environments of systems development have been proposed (e.g. Lyytinen, 1987), we wanted to use inductive reasoning and observe empirically in real industrial settings how the contextual factors occur and how do they relate to each other.

As an example of an observation, we observed that the resource management approach of the company has an influence on the development process. An organization may e.g. use employees in multiple projects at a time or overuse their personnel resources and therefore cause problems with managing the development projects in a predictable manner. Thus, the phenomenon of resource management approach is observed to have an influence on software development process. As its side-effect the chosen resource management approach may hinder predictable project management, which then may result on work overload, poor software quality or shifted deadlines.

The observed contextual factors were given more descriptive names. As the final step, the factors were grouped to form nine *categories*, and the factors were given *definitions* to describe the nature of their influence on software development process.

The categories, their associated factors and definitions are presented in the following two subsections: the first gives the overall picture; whereas in the second, the defined categories and their properties along with definitions are described in more detail.

The Overall Picture

The nine categories of the contextual factors that were observed to have an influence on software development in the four software organizations are presented in Figure 2. There is also one line in the figure showing that although customers can be classified as parts of the business environment, their role in the data and observations was so strong that they should be regarded as a separate category.

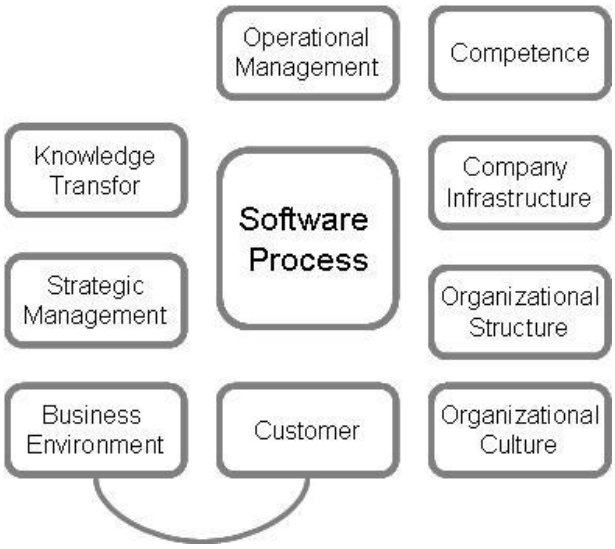

Figure 2: Categories of contextual factors

There are probably much more relations between the categories. However, this shows the status of our current observations and identifying the relations between the categories is currently under analysis.

Description of the categories of the contextual factors

So far 23 interviews from four different companies were used for observations. The interviewees were from three levels of the organization structure: top managers, middle managers and developers. In this section, we present the defined categories and show supporting evidence for them from the data; the factors in each category and their definitions are presented in full in Appendix A.

In the **Customer** category four contextual factors were identified: requirements clarity, level of involvement, customer orientation, and level of customer management formalization. Drawbacks caused by the customer were observed in all the companies. Customer is perhaps the

most important component of the business environment and hence it has a lot of influence on the entire process of developing software. In one of the interviewed companies, a developer commented on the customer as follows: "Too familiar customer who wants to operate in the same way in which we have been working with him for the past 20 years."

In the **Competence** category four factors were identified: level of software engineering skills, working experience, level of business knowledge, and level of application domain knowledge. Problems with using rare skills and acquiring sufficient knowledge of e.g. technology, customer's application and business domains were observed as troublesome areas.

In the **Organizational Culture** category four factors were observed: attitude to process improvement, attitude to quality, work time policy, and personal effect on working practices. For example, the quality attitude was an important issue in all the interviews of one company which is well represented by a developer who noted the following: "...I don't send any incomplete work products to testing since it is clear, that it only causes trouble to you later on." One tester had a similar attitude as he mentioned that he does not sign off an artifact from testing unless he has confidence that it follows the quality expected in the company.

In the **Organizational Structure** category two factors were identified: number of roles, and consistent understanding of software engineering practices. Organizational structure deals mostly with the organization of the company. It was observed in some companies that there are multiple roles for an individual within one project and also he participates in multiple projects. Knowledge of the software engineering practices within the organization and their consistency in that was also a problem area in some companies. For example, in one organization the conceptions about the maturity of practices differed between developers and managers and this caused trouble in determining the right way to improve the practices.

In the **Business Environment** category six factors were identified: level of competition, level of trust, attitude to subcontracting, cost level, language barrier, and physical distance. The business environment category is related to the operating environment of the company. It determines in a general level how much resources and costs can be allocated to development projects and how the development can be distributed across organizations and geographical locations.

In the **Operational Management** category four factors were identified: resource management approach, schedule management approach, project management approach, and company quality assurance. Operational management is the category where practices that involve managing resources, schedules, and projects are dealt. The ways in which development schedules were planned differed from one company to another. We observed that especially the resource management was an area of difficulty for many of the organizations. This was apparently related to the organization structure category, because many of the problems were caused by the approach of having multiple roles in multiple projects for employees. This had also negative effects on schedules and quality.

In the **Strategic Management** category two factors were identified: business strategy, and awareness of the strategy throughout the organization. Strategic management is a category that deals with the business strategy of the company and the way this strategy influences the software development practices. Awareness of such a strategy throughout the organization was also observed as a factor. In one of the interviewed company, a developer said: "It [the business strategy] can be seen in tightening schedules… Savings are made everywhere… They don't dare to recruit more people… This is reflected in overloading… People get tired… People start taking longer sick leaves… In this way it [the business strategy] has an influence".

In the **Knowledge Transfer** category two factors were identified, efficiency of communication, and level of documentation. In this category, the existence of documents like templates and the level of their use in projects and the availability of such documents to developers along with efficiency of communication between the project members and other co-operating organizations were observed.

In the **Company Infrastructure** category six factors were identified: level of methods and process formalization, selection of tools, amount of training available, level of change management formalization, availability of reusable artifacts, and availability of a common architecture. Company infrastructure is seen as another important category by the companies. The selection of methods, processes, and tools caused difficulties for organizations. The importance of following defined processes was quite regularly mentioned in interviews. In one company the presence of a well defined process with separate phases and steps between the phases was described by all the interviewees. This defined process was supported by a workflow management tool where each task was assigned, and tracked by the system. In another organization, there were no signs of formally defined processes at all. The variation in the company infrastructure was high in all areas between the organizations observed.

DISCUSSION AND CONCLUSIONS

The observations made in the interviewed companies led to the set of nine categories of contextual factors that influenced software development practices in the studied organizations. These categories were formed by classifying 34 individual factors that were observed in the interviews. The full list of categories and their factors is shown in Appendix A.

The limitations of the present study include the fact that it is still in the middle of the analysis phase. In addition, the set of contextual factors is probably not yet complete due to the limited number of cases. However, we have already identified the initial set of key categories and their associated contextual factors from the conducted interviews. The factors need still more work to form a dense and substantiated theory, and a comparative study on these categories and properties based on existing literature needs to be done in order to saturate the theory. We have already confirmed that the identified factors have similarities with the factors discussed in other research reports, for example the nine critical success factors that were found in the study of Butler and Fitzgerald (1999).

The study will have implications both for research and practice even when the study has focused so far on the identification of the categories and properties. In the theoretical side the present work will eventually establish an empirically defined set of factors that affect software development processes as observed in multiple companies. For example, Boehm and Turner (2003) suggest that five factors should be considered when finding a right balance between agility and discipline of a software development process: size, criticality, dynamism, personnel, and culture. However, it is not clear where these factors came from and how they were identified. Butler and Fitzgerald (1999), on the other hand, provide a rare example of actual identification of factors affecting development process, but their work is limited to one company only. This limitation has been considered as the main problem in generalizing the results of the studies identifying process related events with the grounded theory approach (Sambamurthy and Kirsch, 2000). The wider relevance of the factors can be verified later by triangulating them with other, preferably quantitative, studies. In this initial stage of the study we have looked at some other studies with factors affecting the software development process, but the studies have been rather substantive theories (Glaser and Strauss, 1967) in limited domains (e.g., Phan et al., 1995; Rainer and Hall, 2003; Sumner and Ryan, 1994; Trienekens et al., 2007) than formal theories focusing on the topic of interest as a whole.

On the practical side the study is expected to provide a framework that can be used to help in planning process improvement actions in industry. Our current study continues to work on the relationships between the factors to establish company specific understanding of the contextual factors and their interplay. The company specific results will further be combined as an integrated framework, and the insights gained in this process will be used to plan process improvement actions in the studied companies. After using the framework in multiple companies, we expect the results of this study to be valuable for other companies also in their efforts to improve their software and systems development processes.

REFERENCES

1. Alexander, C. (1964) *Notes on the Synthesis of Form*, Harvard University Press, Cambridge, Massachusetts.
2. Andres, H. P. and Zmud, R. W. (2001) A Contingency Approach to Software Project Coordination, *Journal of Management Information Systems*, **18,** 41.
3. Barki, H. and Rivard, S. (1993) Toward an assessment of software development risk, *Journal of Management Information Systems*, **10,** 203.
4. Barki, H., Rivard, S. and Talbot, J. (2001) An Integrative Contingency Model of Software Project Risk Management, *Journal of Management Information Systems*, **17,** 37-69.
5. Baskerville, R. and Pries-Heje, J. (1999) Grounded action research: a method for understanding IT in practice, *Accounting, Management and Information Technologies*, **9,** 1-23.
6. Baskerville, R. L., Ramesh, B., Levine, L., Pries-Heje, J. and Slaughter, S. (2003) Is Internet-Speed Software Development Different, *IEEE Software*, **20,** 70-77.
7. Boehm, B. and Turner, R. (2003) Balancing Agility and Discipline: A Guide for the Perplexed, Addison-Wesley, Boston.
8. Boehm, B. W. (1991) Software risk management: principles and practices, *IEEE Software*, **8,** 32-41.
9. Brinkkemper, S. (1996) Method Engineering: Engineering of Information Systems Development Methods and Tools, *Information and Software Technology*, **38,** 275-280.
10. Butler, T. and Fitzgerald, B. (1999) Unpacking the systems development process: an empirical application of the CSF concept in a research context, *The Journal of Strategic Information Systems*, **8,** 351-371.
11. Calloway, L. J. and Knapp, C. A. (1995) In *First Americas Conference on Information Systems* Association for Information Systems, Pittsburgh, Pennsylvania USA.
12. Coleman, G. and O'Connor, R. (2007) Using grounded theory to understand software process improvement: A study of Irish software product companies (In Press, Corrected Proof), *Information and Software Technology***,** 624.
13. Czarnecki, K. (2002) In *Encyclopedia of Software Engineering*, Vol. 1 (Ed, Marciniak, J. J.) John Wiley & Sons, New York, pp. 433-444.
14. Glaser, B. and Strauss, A. L. (1967) The Discovery of Grounded Theory: Strategies for Qualitative Research, Aldine, Chigago.

15. Glass, R. L. and Vessey, I. (1998) In *Thirty-First Hawaii International Conference on System Sciences*, Vol. 3 IEEE Computer Society, Hawaii, pp. 187-196.
16. Hansen, B. H. and Kautz, K. (2005) In 38th Annual Hawaii International Conference on System Sciences (HICSS '05), pp. 264b-264b.
17. Hardgrave, B. C., Wilson, R. L. and Eastman, K. (1999) Toward a Contingency Model for Selecting an Information System Prototyping Strategy, *Journal of Management Information Systems*, **16,** 113-136.
18. Keil, M., Cule, P. E., Lyytinen, K. and Schmidt, R. C. (1998) A framework for identifying software project risks, *Communications of the ACM*, **41,** 76-83.
19. Kirsch, L. J. and Haney, M. H. (2006) Requirements determination for common systems: turning a global vision into a local reality, *The Journal of Strategic Information Systems*, **15,** 79-104.
20. Layman, L., Williams, L., Damian, D. and Bures, H. (2006) Essential communication practices for Extreme Programming in a global software development team, *Information and Software Technology*, **48,** 781-794.
21. Lyytinen, K. (1987) In *Critical Issues in Information Systems Research* (Eds, Boland, R. J. J. and Hirschheim, R. A.) John Wiley & Sons, pp. 3-41.
22. Lyytinen, K., Mathiassen, L. and Ropponen, J. (1996) A framework for software risk management, *Journal of Information Technology*, **11,** 275.
23. McFarlan, W. (1982) Portfolio Approach to Information Systems, *Journal of Systems Management*, **33,** 12-19.
24. Paré, G. and Elam, J. J. (1997) In The IFIP TC8 Working Conference on Information Systems and Qualitative Research Philadelphia, Pennsylvania, United States.
25. Phan, D. D., Vogel, D. R. and Nunamaker, J. F. (1995) Empirical studies in software development projects: Field survey and OS/400 study, *Information & Management*, **28,** 271-280.
26. Port, D. and Dincel, E. (2001) Experiences Using Domain Specific Techniques within Multimedia Software Engineering, *Annals of Software Engineering*, **12,** 11-45.
27. Rainer, A. and Hall, T. (2003) A quantitative and qualitative analysis of factors affecting software processes, *Journal of Systems and Software*, **66,** 7-21.
28. Ramesh, B., Pries-Heje, J. and Baskerville, R. L. (2002) Internet Software Engineering: A Different Class of Processes, *Annals of Software Engineering*, **14,** 169-195.
29. Rockart, J. F. (1979) In *The Rise of Managerial Computing: The Best of the Center for Information Systems Research* (Eds, Rockart, J. F. and Bullen, C. V.) Sloan School of Management, MIT, Cambridge, pp. 207–234.
30. Rose, J., Pedersen, K., Hosbond, J. H. and Kraemmergaard, P. (2007) Management competences, not tools and techniques: A grounded examination of software project management at WM-data, *Information and Software Technology*, **49,** 605-624.
31. Sambamurthy, V. and Kirsch, L. J. (2000) An Integrative Framework of the Information Systems Development Process, *Decision Sciences*, **31,** 391-411.
32. Schmidt, R., Lyytinen, K., Keil, M. and Cule, P. (2001) Identifying Software Project Risks: An International Delphi Study, *Journal of Management Information Systems*, **17,** 5-36.
33. Seaman, C. B. (1999) Qualitative methods in empirical studies of software engineering, *IEEE Transactions on Software Engineering*, **25,** 557-572.
34. Strauss, A. L. and Corbin, J. (1990) Basics of Qualitative Research: Grounded Theory Procedures and Applications, Sage Publications, Newbury Park, CA.
35. Sumner, M. and Ryan, T. (1994) The impact of CASE: Can it achieve critical success factors?, *Journal of Systems Management*, **45,** 16.
36. The Standish Group (1994) The Standish Group.
37. Trienekens, J., Kusters, R., van Genuchten, M. and Aerts, H. (2007) Targets, drivers and metrics in software process improvement: Results of a survey in a multinational organization, *Software Quality Journal*, **15,** 135-153.
38. Weill, P. and Olson, M. H. (1989) An Assessment of the Contingency Theory of Management Information Systems, *Journal of Management Information Systems*, **6,** 59.

APPENDIX A: CATEGORIES AND ASSOCIATED FACTORS WITH DEFINITIONS

Category 1: Customer	
Requirements Clarity	Level of clarity in the requirement specification or definition documents that are specified by the customer.
Level of Involvement	Level of involvement and thus influence of the customer.
Customer Orientation	Level of orientation towards customer meaning how strong weight of the word customer has.
Level of Customer Management Formalization	Does the company have established customer management process and how strongly does the company follow it?
Category 2: Competence	
Level of Software Engineering Skills	Amount of expertise present in certain areas of software engineering.
Working Experience	Level of knowledge and ease in using the software development tools.
Level of Business Knowledge	Level of customer's business knowledge.
Level of Application Domain Knowledge	Level of customer's application domain knowledge.
Category 3: Organizational Culture	
Attitude to Process Improvement	What is the general attitude towards process improvement in the company?
Attitude to Quality	What is the general attitude towards quality in the company?
Work Time Policy	What is the company policy regarding working hours and overtime work?
Personal Effect on Working Practices	How much does the company dictate and control the way individuals do their work?
Category 4: Organizational Structure	
Number of Roles	Number of roles in a project that are specific for an individual.
Consistent Understanding of Software Engineering Practices	How consistent is the understanding of software engineering practices exist through the management levels of the company?
Category 5: Business Environment	
Level of Competition	Level of competition of the company. The competition might initiate a business objective reflecting on software development practices.
Level of Trust	Level of trust that one organization has on another organization.
Attitude to Subcontracting	What are the attitudes towards external subcontracting in the company?
Cost Level	What kind of costs do different development alternatives have, e.g. developing the software in-house or contracting it to a local, near-shore, or offshore company?
Language Barrier	Degree of language barrier between the co-working organizations.

Physical Distance	How far is the subcontractor / vendor / partner / supplier located physically, e.g. in the same building, same city, same country, or the other side of the world?
Category 6: Operational Management	
Resource Management Approach	What kind of practices are used in resource management and how efficient are they?
Schedule Management Approach	What kind of practices are used in schedule management and how efficient are they?
Project Management Approach	What kind of practices are used in project management and how efficient are they?
Company Quality Assurance	How much attention does the company pay on the quality of its work and work products?
Category 7: Strategic Management	
Business Strategy	What is the level and way of influence of business strategy on the software development practices?
Awareness of the Strategy Throughout the Organization	Does everyone involved in the development know the business strategy and its objectives?
Category 8: Knowledge Transfer	
Efficiency of Communication	How efficient is the communication inside the organization, i.e. between the project members, and between the co-working organizations.
Level of Documentation	What kind of documentation is available in the company e.g. does a requirements document or technical specification exists and is it available for the developers and testers?
Category 9: Company Infrastructure	
Level of Methods & Process Formalization	Does the company have established software development methods and processes and how tightly does the company follow them?
Selection of Tools	How strongly is the selected tool reflected on software development practices?
Amount of Training Available	Amount and quality of training available for the project members.
Level of Change Management Formalization	Is there established change management process and how tightly does the company follow them?
Availability of Reusable Artifacts	Does the company have reusable artifacts, such as templates, code modules, etc., available?
Availability of Common Architecture	Does the company have common architecture and how often does it use the architecture in own projects.