Contract Design and Uncertainty in Software Development Projects

Cornelia Gaebert,

Research Group on Strategic Information Management,
European Research Center for Information Systems
c/o University of Muenster, Leonardo Campus 11, D-48149 Muenster, Germany
Cornelia.Gaebert@{uni-muenster.de,indal.de}

Abstract. Recent research has described software development projects in terms of the economic principal agent theory. These models mostly describe the customer as the principal, whereas the supplier is the agent. Our study shows, that regarding gaps in software requirement specifications, the supplier is in a principal situation, and the customer plays the role of an agent. Specifications are incomplete due to systematical reasons. Therefore, the customer must work on closing the gaps during the design and development phase of the project. From this, behavioral uncertainties arise. An empirical study supports our theoretical argument. We discuss consequences from these findings and derive suggestions for practitioners in software development projects.

Keywords: Incomplete Information, Principal-Agent Approach, Credible Contract, Software Development Project, Failure Reasons

1 Introduction

Research on the contractual structure of software development projects, namely on outsourcing projects, has shown, that the relationship between the customer and the supplier can be described in terms of the economic principal agent theory [1, 2, 3]. The aim of these works is to use the findings of the economic theory for describing the contract structure of a software project in terms of the theory by understanding the behavior of the customer and the supplier as rational actors, and to make predictions for risk and success of a project with respect to the contractual situation.

Until now, most researchers describe the customer as the principal, and the supplier as the agent. This seems to be obvious, because the supplier works on behalf of the customer, having the needed information.

The aim of this paper is to show, that in a software development project there is also the contrary situation, where the customer is an agent, and the supplier is a principal. The root cause is that in the moment of signing the contract, the requirement specification is nearly never complete, and for systematical reasons, it cannot be complete. In this paper, we call this deficit requirement gaps.

We will develop our argument in section 2 of this paper. In section 3, we will show some empirical findings, supporting our argument. Finally, in section 4, we will derive some consequences and conclusions and we will give a brief outlook for further research.

2 Principal and Agent in Software Development Projects

Researchers in the field of software project management focus mostly on the control of decisions and activities of the acting participants and stakeholders within the development organization [4, 5]. They often describe them as rational agents having goals and making decisions for the cooperation with other actors, with the purpose of achieving a maximum of benefit [6, 7]. However, as shown by Tollefsen [8], we also can consider organizations like companies or public authorities as rational agents having their own goals and making rational decisions for reaching these goals.

There is wide variety of cooperation structures between organizations in software projects. One software company may produce and deliver a system for many companies having the same requirements. In contrast, one company may engage more than one supplier in one project, or one supplier may engage some other companies or freelancers with a special knowledge within a project. Furthermore, a project may be processed completely within one company with its own development department for the implementation of information systems for several business departments. However, many software development projects are two-party projects like in the case of outsourcing [9]. For simplicity, the focus of this paper is on this kind of projects.

Regarding these software development projects, at the organizational level we can define two kinds of actors: First, there are organizations acting as customers; and second are the organizations acting as suppliers. The customer has business goals resulting in requirements for a software system, described in a requirement specification document. The supplier has the ability to develop an information system that meets these requirements. Therefore, the customer and the supplier sign a contract to carry out a software development project.

The supplier and the customer mostly agree on fixed-price contracts or contracts with a price ceiling (upper limit) [2, 10]. In these cases, the income of the supplier is nearly independent of its actual effort. Furthermore, if during the project new information regarding the requirements arise, the supplier will have additional expenses. In this paper we focus on projects under fixed-price conditions.

At first glance it seems to be obvious, that the supplier acts as an agent on behalf of the customer. The supplier has all the information needed for the development of the software system desired by the customer. The behavior of the supplier is not fully observable during the development phase, therefore, the customer is in a role called the principal by the economic principal agent theory. In recent years, researchers have shown that some parts of the structure of a software development project can be described by means of the principal agent theory, interpreting the customer as the principal, and the supplier as the agent.

Elitzur et al. [11] describe for outsourcing projects a double sided moral hazard problem, where also the customer is in an agent situation and the supplier works as a principal. The precondition stated by these authors is, that the outsourcing market is dominated by a few outsourcing companies. Therefore, these suppliers are in a strong position. As our empirical study shows (see section 3), in most cases of software development projects the customer is in a dominant role. We focus our work on this kind of projects.

In this section, we will develop the argument that in nearly all software development projects also the customer acts as an agent, and the supplier acts as a principal. To put it briefly, due to the fact, that requirement specifications are incomplete in most cases, the customer must work on closing gaps within the specifications. For the supplier, the efforts of the customer are not completely observable.

2.1 The Necessity of Incompleteness of Requirement Specifications

In an ideal world, the requirement specification is complete, unambiguous, and clear. In such a perfect world, the supplier has calculated all efforts for the implementation of the requirements before signing the contract. Based on the specification, the designers and developers will implement the needed system. No communication and no interaction between the parties will be necessary during the project.

Unfortunately, requirements are not complete and unambiguous. As shown in research literature [12, 13], and as stated by all experts in our empirical survey (see section 3), there are gaps in the requirements specifications. These gaps are one of the main reason for project failure. The Standish Group in 1995 [14] cited “incomplete requirements” as the number-one reason for failure, El Emam and Koru in 2008 [15] cited “too many requirements and scope changes.” Researchers and practitioners have exerted a lot of effort in developing methods for producing better specifications without gaps, misunderstandings, and unclear descriptions.

Nevertheless, there are systematic causes for the gaps in requirement specifications. Software requirement specifications contain knowledge in a strict sense only about the past and the present. For instance, the customer knows problems that exist with the currently used system, the present market situation, and business cases. About the future, there are only assumptions. In particular, how the new system will change the business processes is not a matter of fact, but a matter of expectation and anticipation. During the technical design process and the impementation phase of the project, questions regarding these assuptions arise, and the parties must decide on details not specified within the requirement specification document. In addition, the requirement engineer can only document consciously available knowledge, and to some extent subconsciously available knowledge. However, in all business processes, relevant conditions and information exist that no one knows about [16].

The customer has knowledge primarily regarding the business for which the software system is needed. In contrast, the supplier has knowledge regarding technical issues, like the properties of used frameworks and development techniques. Furthermore, on the supplier’s side, experiences from other projects regarding user acceptance and performance problems exist. This knowledge is also relevant for the development of a software system, but in the moment of documenting the requirements it is not avaialable.

Consequently, we have to accept the fact, that requirement specifications will contain gaps also in the future, and even if research in requirement engineering finds new and better methods, it will be a very long way to have complete requirement specifications in practice.

2.2 The Principal Agent Situation Regarding Incomplete Specifications

The choice of contract design depends on the expected uncertainty. Uncertainty occurs when customer and supplier close gaps in the requirement specification. The closing of gaps itself already represents an expense that supplier and the consumer have to pay during the course of the project. This task requires efforts from both sides in order to achieve an optimal result. The supplier has to deliver input such as technical information and the customer shall give business process details and professional information. Thus, asymmetrically distributed information [17] requires the closing of gaps in the specification during the development project. Interaction-related quality- and accountability problems are the consequence; interaction-induced behavioral uncertainty grows up [18]. Evidently, there is a great potential for conflicts.

Behavioral uncertainty is obviously, on both sides. Customers and suppliers have to quantify objectively all their supplies and efforts in order to measure performances and thereby reduce uncertainty [19]. Costs arise from procurement and measurement of information. These cost results from measurement itself, from loss through measurement errors or inaccuracies as well as exceeded costs through mutual opportunistically benefits from measurement errors and inaccuracies [20]. Therefore, prohibitively expensive measurement costs avoid the elimination of uncertainty. The contract has to deliver a compensation system that serves the dual function of allocating risks and rewarding cooperation.

The principal-agent approach is a central theoretical approach inside the new institutional economics. Here, the agent has an information advantage over the principal. Therefore, the principal has to manage his uncertainty about agent's behavior [21]. The agent knows earlier and more precisely his own intention and his own weaknesses. He uses this for his own advantage. The principal-agent approach concerns itself with issues and consequences from this. By contract design ex-ante provided penalties and incentive schemes are supposed to reduce uncertainty and their consequences during the execution of the contract. Thus, the contract reduces the risk of exploitation for the worse informed principal. In the best case, the contract protects against a misallocated contractual partner, economical disadvantages, and welfare loss.

Spremann [17] distinguishes between three basic types of behavioral uncertainty resulting from information asymmetries. These are (1) uncertainties about agent's capabilities to be able to perform the promised service (hidden characteristics), (2) uncertainties about agent's fairness to be willing to perform the promised service (hidden intention) and (3) uncertainties about agent's effort to perform the promised service (hidden action).

The terms revelation, authority, and incentive describe the possible designs of institutional hedging uncertainty. Revelation protects the principal against uncertainties about agent's capabilities (1). Authority protects the principal against uncertainties about agent's fairness (2). Incentive protects the principal against uncertainties about agent's effort (3).

For closing the gaps within the requirement specification, during system design and development the customer must answer questions, deliver business information, evaluate the suggestions of the supplier, and make decisions regarding software design issues. For the supplier, the processes within the organization of the customer are at least partly hidden. The supplier requests an information, and receives an answer, maybe as a document written by experts not known within the project. Obviously, the supplier is in a principal situation, whereas the customer is the agent working on behalf of the supplier. We will discuss now the cases known from the principal agent theory regarding behavioral uncertainty.

(1) *Hidden characteristics*. The supplier does not know if the customer is able to deliver the required information. Maybe the business information needed is not available from the business experts, or they have not the ability to understand the problem raising during the system design. Furthermore, it is possible, that the management of the customer is not able to organize the information generation and delivery in parallel to the everyday business.

For recognizing problems resulting from hidden characteristics, the principal agent theory describes signaling and screening. From this, we suggest for contract negotiation, that the customer shall state clearly the qualification of the experts available during the project for answering questions of the supplier (signaling). Furthermore, the customer shall describe the procedures of acquiring the needed information and shall commit on amounts of work time available for clarification. Consequently, the supplier is able to calculate the risk left from hidden characteristics. On the other side, the supplier should implement methods for getting informations regarding the relevant qualifications on the customer side (screening).

The contract design facing the problems from hidden characteristics is *revelation*. For the software development project, this means, that the parties define within the contract clearly the obligations of the customer for delivering information and that the customer commits the availability of experts with the needed qualification.

(2) *Hidden intention*. Because incomplete and ambiguous software specifications are subject to interpretation, the customer has the option to reject project results, stating that these results do not fulfill the requirements according to the interpretation of the customer. In such cases, the customer has no interest to make the specifications clear. He will use gaps in specifications for saving costs, arguing that the supplier does not have implemented the system in a way he should. For hidden intentions of this kind, the customer may remain undefined namely non-functional requirements like performance and usability requirements. By testing the system, the customer may argue that the performance of the system does not meet the needs of the business, and that the usability is not state of the art. With this argument, he can try to reduce the price or to refuse the acceptance if he does not need the system anymore due to changes in the customer's business.

The principal agent theory suggests avoiding specific investments for reducing loss of investments on the principal's side. Obviously, for the supplier developing an individual software system it is nearly not possible to avoid specific investments. He cannot expect to be able to use the developed software within another project. Furthermore, the contract design suggested by the theory for handling hidden intention is authority. However, we can interpret the fixed price contract as an authoritarian contract, but with the customer as the authority, not the supplier. We must note that the case of hidden intention on the customer's side is highly risky for the supplier.

(3) *Hidden action*. If the contribution of the customer for filling requirement gaps and clarify open issues is not visible to the supplier, the latter cannot evaluate the real effort. Maybe, the experts do not deeply analyze the raising issues or do not search for relevant business facts but state own assumptions as objective facts. Furthermore, it is possible, that in the backstage of the project some stakeholders play their own game, trying to hinder the project's success.

According to the theory, the principal shall monitor the behavior of the agent to recognize hidden action. The best way to do this is to make the work of clarification of requirement issues in common teams. If this is not possible, we suggest the supplier to ask for detailed information regarding the question who was involved within the clarification process and to what extent.

The suggested contract design for preventing issues raising from hidden action is incentive. Nevertheless, it seems to be difficult for the supplier to agree incentive for the delivery of high quality specification clarifications. Furthermore, the effect of incentives for avoiding hidden actions on customer's side in a software development project is uncertain.

3 Empirical Support of the Theoretical Argument

We support our theoretical findings with an empirical survey. First, it is essential that the supplier gets a fixed price for realizing the software system, or that there is a cap on the effort-based price. If the customer would pay an effort-based price for all of the work done by the supplier, he could invest a lot of work needed for closing the gaps in requirement specifications. Furthermore, if the resulting software system would not meet the real requirements of the customer, the supplier could do any rework necessary for implementing the needed functionality. Second, are there gaps in the requirement specification delivered by the customer by signing the contract? Third, are there uncertainties on both sides, especially on supplier's side?

For this empirical part of our study, we conducted a two-step evaluation. First, we developed a questionnaire in the form of a standardized online survey as a special kind of standardized survey [22]. Next, we conducted personal interviews to deepen our understanding of the results from the questionnaire. The period of the evaluation was one year.

For the questionnaire, we chose the standardized online survey to give the respondents an opportunity to reflect and to question their own companies [23]. The format of the online survey itself was legitimate because the interviewees were an IT-savvy group. Open answers supplemented the closed questions to not be too restrictive and to gather the covered information [24]. In the following, we will analyze and interpret the results descriptively.

Experienced project participants on both sides (customer and supplier) were interviewed. The questionnaire had to take the management perspective into account. Because it is not possible to address trivially the population of all manufacturers and customers of custom software, and because questioning the population about any associated unacceptably high cost is not realistic, we chose a smaller population. Therefore, we could not achieve complete representativeness [23]. For practical reasons, we addressed the 45 members of a network of IT companies in Germany. Fifty additional addressees were available from other contacts. To expand the circle of respondents and to amplify the customer side, we used contacts in social networks such as Facebook (approximately 30), Xing (approximately 20), and Twitter (approximately 50). This ensured that the respondents had experience in different contexts of possible projects. Of the 200 addressees who were requested to participate in the survey, 29 actually completed the questionnaire (14 suppliers, 5 customers, 9 suppliers and customers (both), and 1 other).

A total of 48.3% of the respondents indicated that they belong to management and that they have responsibility for the contracts; 27.6% are project managers; 6.9% are employees at the working level; and 17.2% perform other activities, such as consulting. A total of 89.7% of the respondents had 10 or more years of experience with software development projects. The participants represented a broad range of sizes of projects with regard to the duration and number of employees.

For the exemplary and in-depth interviews, we conducted semi-structured expert interviews. We questioned, on the one side, a consultant with experience in software projects for approximately 15 years. He supports big companies in defining and organizing the contractual issues of software projects. On the other side, we spoke with a supplier with experience in software projects for approximately 20 years. He is an owner of a software development company with 10 programmers. Considering the sensitivity of failure research and the resulting difficulty in gaining access to project details, this methodology was most appropriate. The incomplete script of the semi-structured interview format left room for improvising questions [25]. The first interview lasted approximately 3 hours; the second lasted 1.5 hours. We made extensive notes during the interviews, which we evaluated afterward through a qualitative content analysis. Because we demanded appointed circumstances and facts, we avoided free interpretation problems [26].

3.1 Results from the Online Survey

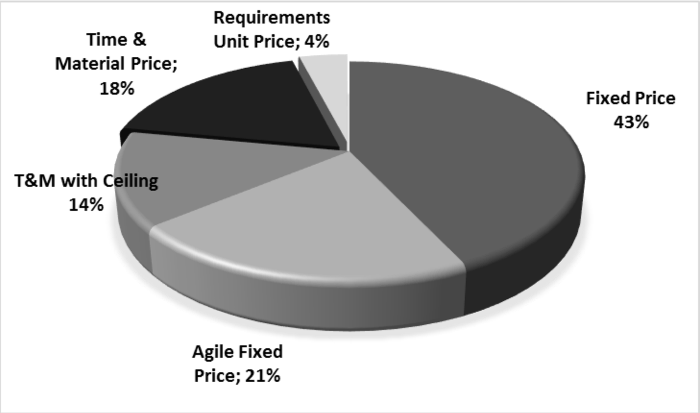

Fig. 1. Proportion of different types of contracts on software development projects.

The survey showed that the proportion of fixed-price contracts for software development projects is extremely high (Fig. 1). Taking into account that even the so-called agile fixed price, and time and material (T & M) price with ceiling ultimately determines the maximum total budget for the consumer, the proportion of this type of contract is a total of more than three quarters of the software development projects. A manager on the side of the supplier added in free text: “Even if it is charged at T & M, the expectation of the customer is the compliance with the budget / value of the order.”

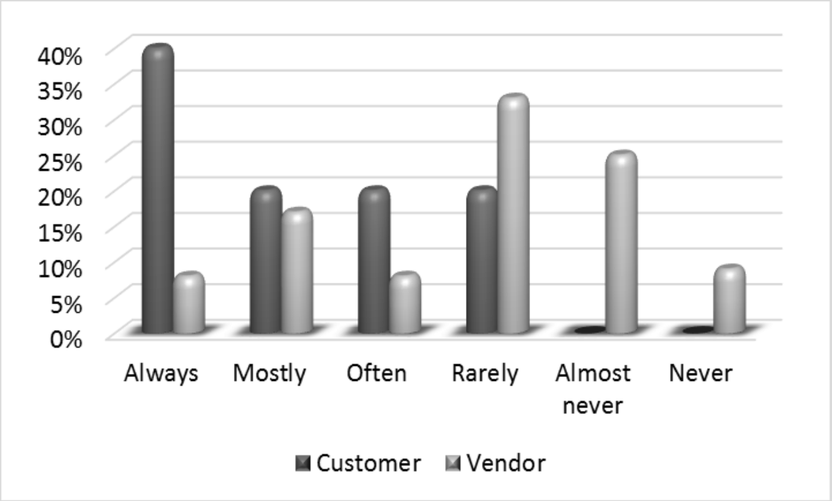

Fig. 2. Answer to the question "Do you determine the contract model?"

On the bottom line, the T & M price with ceiling and the agile fixed price mean the implementation of the requirements at fixed cost. Often the ceiling does not differ significantly from the calculated expense. An agile fixed price, however, allows one to the implementation of requirements when new requirements emerge. Then, these new requirements can replace earlier ones. However, such contractual subtleties relate only to new requirements. A third party (judge) can evaluate them. Nevertheless, this rarely helps in cases of closing the requirement gaps. Rather, closing gaps only makes unconscious knowledge aware. For the customer, it appeared typically obvious, whereas it was unknown to the supplier and vice versa. Filling the gaps makes it known explicitly.

The customers predominantly determine the contract model (Fig. 2). Although 80% of the customers indicate that they at least often determine the contract model, suppliers say quite the opposite. Two-thirds of them admit that they have little or no influence on the contract model. One comment from a project leader on the supplier side is: “I do not understand the question. The contract model is in all cases defined by the customer.” Thus, customers clearly choose the contract design.

Customers and suppliers have different views on emerging problems inside a fixed-price project, like when an imbalance occurs in terms of time, cost, and quality (Fig. 3).

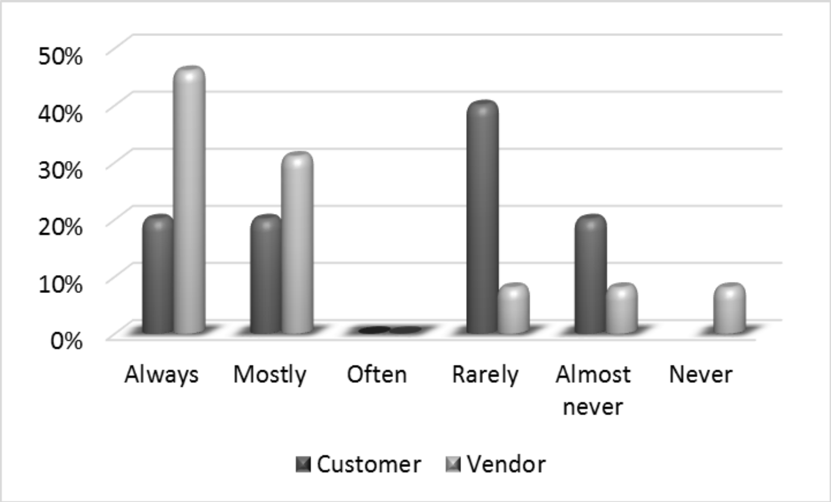

Fig. 3. Is an imbalance of time, cost, or quality in the project under fixed-price problematic?

Although 77% of the suppliers consider such a situation always or usually as problematic, 60% of the customers believe that this is rarely or almost never a problem for them.

Against this background, it is important to consider how the contract reflects gaps in the requirement specifications and how the signed contract supports the project itself. After all, such gaps lead to increased interaction. Most respondents stated for the vast number of projects (Fig. 4) that such gaps exist.

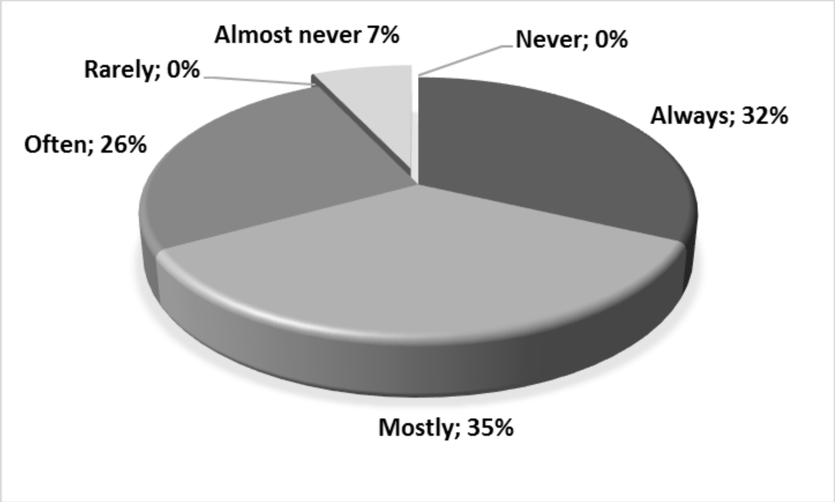

Fig. 4. Frequency of requirement gaps.

Almost a third of the respondents said that such gaps “always” happen; 93% say that this case occurs at least often. However, a fixed-price contract hardly takes this sufficiently into account. For suppliers to do this seems hardly to be possible, as the notes to the relevant questions show. They try to work with a kind of overhead calculation but requirement gaps "are rarely sufficiently taken into account."

However, contracts widely do not reflect this fact. On the question, whether contractors continuously update the contract during the project, 81% of participants responded that this rarely or never happens.

Customers and suppliers have a different perspective regarding whether demand gaps leading to unforeseen interaction would be renegotiated (Fig. 5). Although customers are of the opinion that this would always or at least often happen, 61% of the suppliers believe that there are never or almost never renegotiations.

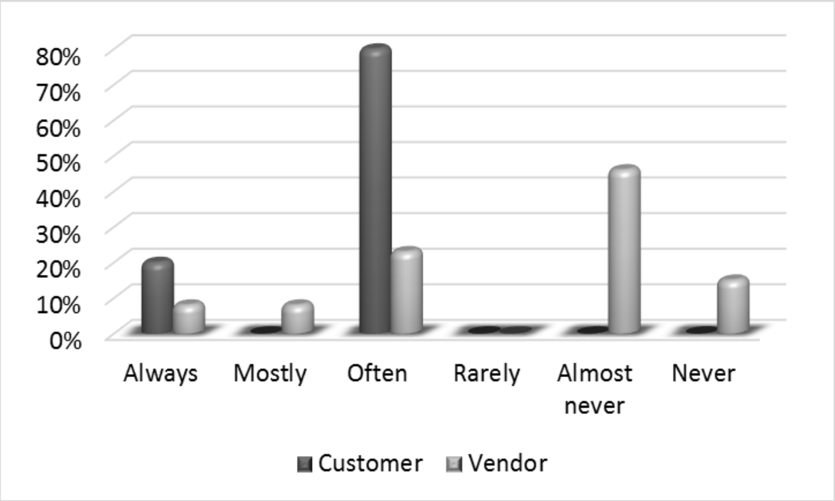

Fig. 5. Renegotiate customer and supplier requirement gaps.

Two-thirds of all respondents say that gaps in the requirement specifications always or almost always lead to unplanned discussions. The contract usually does not take into account the extra costs, which interactions trigger.

3.2 Results from Interviews

We documented the interviews in a structured way with references to each question and to the paragraph of the answer. In the following, we give a short overview of the results. In brackets, we note the reference to the minutes of the interviews. For example, (S Q3A2) references the supplier interview, question 3, answer paragraph 2.

Both interview partners said that the mostly preferred contract model is the fixed-price contract, especially if the requirements are documented and if they seem to be clear (C Q3A1, S Q5A2). This is because of the customer’s restriction in having a limited budget and that customers must calculate the expected benefits against the costs beforehand (C Q16A1, (S Q6A1). Nevertheless, because “it is very seldom that the requirements are specified in a formal way” (C Q10A1), it is almost impossible to calculate the real costs. In addition, the supplier stated: “The problem does not come from the fixed price itself, but from unclear, incomplete, or changing requirements. And the problem is that the customer is not willing to change the price if he changes the requirements” (S Q6A3).

The interviews supported the finding from the online survey, that the customers mostly dominate the contract design(S Q5A2, C Q3A1). Nevertheless, both interview partners gave hints, that obligations for a cooperating behavior of the customer are possible in practice (C Q14A4, S Q11A6).

Because the requirement specifications were so important, we asked our interview partners to explain the reasons for the gaps, the possibilities for dealing with these gaps, as well as the consequences. Both sides cited the reasons as being “special” or “exceptional use cases” that the experts were not aware of during the requirements analysis or were too difficult to model (C Q11A1; C Q11A4; S Q10A1). Furthermore, the facts were “obvious” (C Q11A3) or “self-evident” (S Q10A1) to the business experts, so they did not speak about them. Nonfunctional requirements were often unknown to the users (S Q10A1).

Both interview partners showed a high degree of uncertainty regarding the behavior, intentions, and skills of the other side. Customers try to get certainty beforehand from information like “descriptions of credential projects, facts about the know-how of their staff, information about the methods in designing and processing a software project” (C Q7A1). With “governance structures for the project” (C Q5A1) the customer hopes to “get at early phases of the project a good feeling of the progress and the quality of the vendor’s work” (C Q6A1). However, uncertainty remains high: “Nearly nobody can distinguish the clever, good one from the slow and poor one. And if the vendor mentions that there are unforeseeable problems, you don’t know if he is right or he is not professional enough for doing his job” (C Q16A1).

Regarding the same issue, the interview partner from the supplier side said, “a new management, problems in his market, new relevant law, and maybe, the customer does not need the software anymore or the costs will be higher than the effects. Then, maybe, the customer’s management tries to cancel the project” (S Q11A5).

On the customer side, the strategy is to handle all problems in a formal way and to avoid all discussions regarding efforts in narrowing the gaps in the requirements (C Q11A5; C Q15A1). In contrast, the supplier obviously has strategies of its own, knowing that the customer cannot see all that the supplier is doing (S Q12A1).

4 Conclusions and Further Research

As our empirical study shows, most contracts for software development projects are fixed price projects, mostly with a predefined price for the information system to be implemented, sometimes with a time & material price with ceiling or with an agile fixed price. Furthermore, the customer is in a dominant, authoritarian position, defining the contract design and the conditions of cooperation. If problems arise during the project, the consequences for the supplier can be hard, whereas the customer is often in a better position.

On the other hand, the empirical investigation shows that there are nearly in every project gaps in the requirement specifications. The customer must take part in closing these gaps during the development of the project. Therefore, the supplier is in a principal situation regarding the activities for closing requirement gaps, whereas the customer is the agent regarding the task of providing the needed information.

As we have shown during our theoretical argumentation, the supplier has some possibilities for dealing with the situation being the principal and the uncertainties raising from the information asymmetry. Regarding the problem of *hidden characteristics*, we suggest to ask the customer for details of the qualifications of the experts before signing the contract. Signals for qualifications can be experiences from other projects or knowledge in software requirement engineering.

Regarding the possibility of *hidden action*, we suggest the supplier to participate in the work of filling the gaps by doing the investigation needed in common teams. In this way, the supplier can monitor the work of the customer. If such a participation is not possible, the supplier should ask for details regarding the process of collecting and providing the information: Who was involved? Where are the sources of information?

As our empirical study shows, the customer is the dominant party in most projects. Therefore, for the supplier it is hard to deal with problems arising from *hidden intentions*. On one side, the contract situation hinders the supplier to implement authority. On the other side, in an individual software development project the supplier the supplier has no chance to avoid specific investments, because the software system to be implemented shall meet the specific requirements of this customer, and therefore the chance for using the system within another project are small.

Further research therefore shall focus on the problem of possible hidden intentions of the customer in software development projects. Furthermore, empirical studies shall investigate, which tools of signaling, screening and monitoring are possible in software development projects. With results from such investigations, suggestions for the contract design and for the project management process can be derived. Practitioners can avoid the problems from the principal agent situation described in this study by using these tools.

References

1. Keil, P.: Principal agent theory and its application to analyze outsourcing of software development. In: ACM SIGSOFT Software Engineering Notes, 30(4), pp. 1-5 (2005)
2. Lichtenstein, Y.: Puzzles in Software Development Contracting. In: Communications of the ACM, 47(2), pp. 61-65 (2004)
3. Basu, V., Lederer, A. L.: Agency theory and consultant management in enterprise resource planning systems implementation. In: ACM SIGMIS Database, 42(3), pp. 10-33 (2011)
4. Keil, M., Smith, H.J., Pawlowski, S., Jin, L.: 'Why Didn't Somebody Tell Me?': Climate, Information Asymmetry, and Bad News About Troubled Projects. In: SIGMIS Database, 35(2), pp. 65-84 (2004)
5. Rustagi, S., King, W.R., Kirsch, L.J.. Predictors of Formal Control Usage in IT Outsourcing Partnerships. In: Information Systems Research 19(2), pp. 126-143 (2008)

6. Yilmaz, M., O'Connor, R.V., Collins, J.: Improving Software Development Process through Economic Mechanism Design Communications. In: Computer and Information Science 99, pp. 177-188 (2010)
7. Cockburn, A.: The End of Software Engineering and the Start of Economic-Cooperative Gaming. In: ComSIS 1(1) (2004)
8. Tollefsen, D.: Organizations as true believers. In: Journal of social philosophy 33(3), pp. 395-410 (2002)
9. Chua, C.E.H., Lim, W.-K., Soh, C., Sia, S.K.: Client strategies in vendor transition: A threat balancing perspective. In: The Journal of Strategic Information Systems 21(1), pp. 72-83 (2012)
10. Oestereich, B.: Der agile Festpreis und andere Preis- und Vertragsmodelle. In: Objekt-Spektrum 01, pp. 29-33 (2006)
11. Elitzur, R., Gavious, A., Wensley, A.K.P.: Information systems outsourcing projects as a double moral hazard problem. In: Omega 40, pp. 379–389 (2012)
12. Liu, J.Y.-C., Chen, H.-G., Chen, C.C., Sheu, T.S.: Relationships among interpersonal conflict, requirements uncertainty, and software project performance. In: International Journal of Project Management 29, pp. 547–556 (2011)
13. McGee, S., Greer, D.: Towards an understanding of the causes and effects of software requirements change: two case studies. In: Requirement Engineering 17, pp. 133-155 (2012)
14. Standish Group: CHAOS Report, http://www.projectsmart.co.uk/docs/chaos-report.pdf, 21.06.2011. (1995)
15. El Emam, K., Koru, A.G.: A Replicated Survey of IT Software Project Failures. In: IEEE Software 25(5), pp. 84-90 (2008)
16. Kano, N., Seraku, N., Takahashi, F., Tsuji, S.: Attractive Quality and Must Be Quality, In: Quality - Journal of the Japanese Society for Quality Control, 14(2), pp. 39-44 (1984)
17. Spremann, K.: Asymmetrische Information. In: ZfB 60(5/6), pp. 561-586 (1990)
18. Kaas, K.P.: Informationsökonomik. In: Tietz, B; Köhler, R; Zentes, J (eds) Handwörterbuch des Marketing. 2., völlig neu gestaltete Auflage. Schäffer-Poeschel, Stuttgart. (1995)
19. Barzel, Y.: Measurement Cost and the Organisation of Markets. In: Journal of Law & Economics 25(1), pp. 27-48 (1982)
20. Alchian, A., Demsetz, H.: Production, Information Costs, and Economic Organization. In: American Economic Review 62(5), pp. 777-795 (1972)
21. Arrow, K.J.: Agency and the Market. In: Arrow, KJ, Intriligator, MD (eds) Handbook of Mathematical Economics 23(3). North-Holland Publ. Co., Amsterdam. (1986)
22. Klammer, B.: Empirische Sozialforschung. Eine Einführung für Kommunikationswissenschaftler und Journalisten. Utb, Konstanz, (2005)
23. Schnell, R., Hill, P., Esser, E.: Methoden der Sozialforschung. 9. Auflage, Oldenbourg Wissenschaftsverlag, München, (2011)
24. Mayer, H.: Interview und schriftliche Befragung. Entwicklung, Durchführung und Auswertung. Oldenbourg Wissenschaftsverlag, München, (2012)
25. Myers, M. D., Newman, M.: The qualitative interview in IS research: Examining the craft. In: Information and Organization 17(1), pp. 2-26 (2007)
26. Gläser, J., Laudel, G.: Experteninterviews und qualitative Inhaltsanalyse. 4.Auflage, VS Verlag, Wiesbaden, (2010)