Principal Agent Theory and its Application to Analyze Outsourcing of Software Development

Patrick Keil[1]

Technische Universität München, Institut für Informatik – I4
Boltzmannstr. 3, Germany – 85748 Garching
+49 (0)89 289 17386
keilp@in.tum.de

ABSTRACT

Much has been written on process models, project management or tool support to increase the return on investment in software through higher quality of the development process and the resulting software or system. Yet, we lack understanding in the underlying economic principles; e.g., an external firm paid to develop software for someone else tries to maximize their own profit instead of the contractor's. These divergences of interests result in projects that consume more time and money and meet fewer requirements than expected. In this paper, we try to fill the gap by providing an insight into the theory and presenting applicable suggestions how to diminish or avoid the problems that arise when selecting the 'best' contractor and during the project. Basic advises on the formulation of contracts can be derived.

Categories and Subject Descriptors

K.6.0 [**Management of Computing and Information Systems**]: General---economics; D.2.9 [**Software Engineering**]: Management---programming teams

General Terms

Management, Economics, Human Factors, Legal Aspects

Keywords

Principal Agent Theory; Outsourcing

1. INTRODUCTION

Outsourcing of some tasks and phases in the software development process as well as ASP, BPO and, currently popular, 'software offshoring' have increased in market volume rapidly during the last decade [9]. Manifold contributions analyzed quality measures, management challenges (e.g., see [8]) and many other different aspects of the complex relationships between a customer and the contractor. Yet, on the ground of all these topics, is an economic problem that is – thrillingly – common in all sorts of relations in which a customer's profit or payoff depends on the behavior of a contractor. In economic literature, this research area is called Principal Agent Theory.

Subject of this theory is the relationship between the customer (principal), who pays for services or goods, and the agent. The principal is limited in his ability to monitor and judge the contractor's input and output. This leads to mistrust and can only be avoided under high monitoring costs.

The problem is especially glaring for the software business due to missing metrics and measures for programmers' productivity and for software quality. The missing concreteness of software makes it harder to control effort invested and results reached. In the following, we consider a firm that outsources software development tasks to an external supplier.

2. OUTLINE OF THE THEORY

Foundation of the whole analysis are the assumptions, that in such a dependency

1. both parties have rational behavior and rational expectations and interact on basis of institutions like freedom of contract and private property.
2. the actions undertaken by the agent and the results of his activities have external effects on the principal's profit and success.
3. the agent has discretionary freedom due to incomplete and asymmetric information and monitoring costs. The agent's discretionary freedom leads ex ante to uncertainty (since the principal cannot rely on any motivation like loyalty or conscientiousness) and ex post to concrete disadvantages. The smaller the ability to control the agent's activity (i.e., the bigger the information asymmetry), the bigger is the principal's uncertainty.
4. a divergence of interests exists, i.e. the agent shows opportunistic behavior to maximize her own expected profit instead of acting in line with the goals of the principal. The three types of opportunistic behavior are hidden characteristics (the abilities and skills of the agent are not 'common knowledge'), hidden intention (agent has goals and interests not known by the principal) and hidden action (principal cannot fully control the principal's actions).

EDSER'05, May 15, 2005, St. Louis, Missouri, USA.

[1] The author wants to thank Tilman Seifert and three anonymous reviewers for their helpful comments.

3. ECONOMIC PERSPECTIVE OF THE PROBLEM: PARTIAL MARKET FAILURE

Aspects 2 to 4 lead to three basic types of coordination and motivation problems.[2] In this section, we explain these problems and sketch some practical approaches to avoid them or at least to weaken their impact by aligning the conflicting interests and by limiting the freedom of the agent which we call 'disciplining' [3].

3.1 Adverse Selection

Before the start of the interaction (ex ante), the principal cannot fully judge

- the agent's 'quality' indicated by productivity, soft skills, education, etc. (hidden characteristics),
- the plans of the agent if and how to maximize her own profit through consuming 'perks', shirking, gather valuable knowledge, etc. (hidden intentions).

This leads to quality uncertainty [14], which means that the principal takes the risk to pay a price higher than the agent's real market value. A price cap instead implicates the risk that the "high-value" agents don't apply. In his pioneering work, Akerlof [1] showed that adverse selection can lead to partial market failure, since the principals offer the average price which is too low for the high-quality applicants. Therefore, only low-quality firms offer their service, even if the minimum price the high-quality agents want is lower than the principal's maximum willingness to pay.[3]

In the following, we present three major ways to minimize the risk of selecting an underperforming partner.

3.1.1 Verification by Independent Authorities

If market research firms or other independent institutions collect information about the track record of development firms, the uncertainty of outsourcers can be massively reduced. Supplier evaluations and rankings are valuable only if they are collected independently and the reviews all follow the same methodology. To the author's knowledge, there is no such widely accepted information source. Some international corporations are building up internal supplier evaluation systems based on consistent methods and criteria. Such systems should be enhanced and used more broadly.

3.1.2 Screening and Self-Selection

It is costly for a principal to properly evaluate the quality of a service offered on her own. Therefore we suggest a screening mechanism to cut the contract or task in parts and chose a contractor for the first, small work package. The output is evaluated and the agent's contract will be prolonged or not. The principal benefits from this mechanism only if

- the quality of the output can be judged
- the quality of the output is not dependent on unidentifiable exogenous influences or if he can differentiate between external factors and the results which can be accounted to the agent's actions
- the costs of a new agent to gather the required knowledge are lower than the costs of continuing with the first agent who proved to deliver low quality. Of course, the risk to hire a low-quality agent exists again in the next round.

A more complex idea is the self-selection mechanism [15]. The less informed party offers a pricing scheme which forces the prospective agents to release true information about their respective quality. For instance, in the first phase the price paid could be below market levels and higher in the next phases in case that the evaluation of the service or the software delivered leads to a positive result. This minimizes the principal's risk.

Of course, there are limitations: Agents would only agree on such a scheme if the quality of the output depends directly and mainly on the quality of their work and if the 'quality' can be evaluated properly and based on detailed and contractually agreed criteria. Otherwise the risk of such a mechanism is too high for the high-quality agents [11].

3.1.3 Signalling

This mechanism to avoid adverse selection has been introduced by Spence [12] and is based on the idea that the agent chooses an action which credibly signals the private information. The principal still does not know the real characteristics, but can lower uncertainty over 'anchors' like indices (information and data such as track records, economic situation, number of employees etc.) or signals (guaranties given by the agent, special qualifications, certificates, etc.). Then, the optimal situation of a 'separating equilibrium' in which different types of agents get different prices is reachable and the market efficiency increases, compared to a pooling equilibrium in which all types of agents are paid equally.

But signals like certifications only help to decrease information asymmetries if the signaling-costs are higher for agents with lower 'quality'.[4] Take the example of process maturity or capability indicators like CMM [10] or Spice [13]: they only can serve as signals if there is a significant correlation between the level and the quality of the software produced[5] and if – at the same time – the costs for a firm that delivers low-quality software to reach a certain level are higher than for a high-quality software

---

[2] Pioneer work in this field has been done by Arrow [2].

[3] Akerlof's example was the market for used cars, called 'lemons'. Even if this seems to be a completely different problem, in both markets the quality of the 'goods' offered is not obvious for the buyers; therefore they regret to pay prices for high quality and only low-quality goods are traded.

[4] In his contribution, Spence [12] showed that academic education does not need to increase productivity. Even if students do not learn anything of value for their future jobs, a university degree serves as an 'ability signal' in the job market. It is sufficient that the 'signalling costs' (time and pain of learning) vary for different types of high-school graduates.

[5] This correlation seems to be accepted. [6] showed that a strong relationship between 'project performance' and a higher CMM level can only be observed when an organization reaches Level III. The focus – however – should not be project performance, but the customer's overall return of the investment in the project.

firm. The future acceptance of maturity models will be heavily dependent on how strongly they accomplish these conditions.

Other signal, i.e. credible signs for the agent's quality, are commitments to lower the price ex post if the software does not meet agreed requirements or to undertake corrective maintenance free of charge since only high-quality firms will accept such contracts.

3.2 Moral Hazard

Ex post, i.e. after having selected a contractor and having signed a contract, the principal can control the activities undertaken by the agent only incompletely or under positive information costs (hidden action). Moral risk also appears if the principal can fully monitor the actions, but has limited ability to judge them properly.

The underlying problem is the information advantage of the agent about the quantity and quality of input and of the output (hidden information). All this gives the agent an opportunity to utilize the discretionary freedom and to maximize his profit function (e.g. through 'fringe benefits', reduced effort, unnecessarily high budgets, etc.) which leads to inefficiencies since the decrease in the principal's payoffs is higher than the increase of the agent's.

3.2.1 Monitoring

Monitoring comprises all means of control through which the principal intends to decrease her information disadvantage. The principal needs to control the effort invested by the agent. These activities lead to monitoring-costs. Some principals may measure how many hours the development team invests; but that does not account for productivity or playing computer games during work time. Alternatively, fines can be defined for cases in which the principal recognizes opportunistic behavior. Often, this is not applicable due to difficulties in defining such a behavior (for playing computer games, this problem – of course – does not arise).

To conclude, the challenging questions are which criteria have the strongest influence and impact on the quality of the resulting software (since the latter is the only thing of interest) and whether the costs of monitoring these indicators does not exceed their value.

More fundamentally, monitoring (as well as most of the other aspects we mention) requires that the principal has a good understanding of software development in general and the respective project.

3.2.2 Restrictions and Contractual Agreements

The principal can abandon to measure the input invested by the agent and instead formulates rules that limit the agent's choices. Examples are coding guidelines, process definitions or project management rules. But this also bears costs, since the compliance with these claims needs to be controlled.

3.2.3 Incentive-compatible Contracts

In the last two paragraphs, we briefly sketched how difficult it is to measure the input invested by the agent. Therefore, we should concentrate on other mechanisms to assure that the output meets the principal's requirements.

The best way to align interests for reducing moral hazard is to introduce an incentive system which homogenizes the goals of principal and agent, i.e. the agent's payoff should be a function of the principal's profit. Then, it is optimal for the agent to act in the interests of the principal and follow the incentives incorporated in the contract. In practice, this is done via variable payments, e.g. a predefined bonus if the quality of the delivered software exceeds a given level. The challenge is to find an indicator or measure for the 'quality' of the project results, e.g. the software or system that has been delivered.[6] In the following, we want to show the use of such criteria and their respective problems to work out the pitfalls and challenges of such incentive schemes.

3.2.3.1 Indicators for the 'Success' of a Project

In the early days of the software industry, Nixdorf as a big German hardware and software producer paid external programmers 6.50 German Marks (which is around $ 3.50) per line of Assembler code written. At the beginning, some members of the team inserted lines stating 'nop' which stands for 'no operation' and does not do anything but wasting processor time [Ruh, personal communication]. This behavior is what we call 'gaming the system', i.e. agents maximize the value of these indicators, but – again – not the profit function of the principal.[7] Therefore we need to further investigate the field of metrics for software and systems and the 'degree of customer satisfaction' they reach.

3.2.3.2 Exogenous Factors

The indicators' ability to measure the result of the agent's work is not the only criterion for their value. Imagine we could measure the return on investment of projects. If the contractually agreed variable payment is dependent exclusively and directly on this ROI, we have a complete and perfect alignment of interest, since the profit maximization calculus of both parties is the same. Even though, such an indicator might be of little value because the project's ROI depends on many exogenous factors that cannot be influenced by the agent (and often, not even by the principal). Thus, there is a risk that even if the effort and productivity of the agent is exceptional, it will not be rewarded.[8] Then, risk-averse agents may be prevented from signing the contract even if they are high-performers and – for whatever reason – would not use the discretionary freedom.

3.2.3.3 Summary

From the analysis, we can draw the conclusion that metrics or indicators must be dependent on the agent's actions as strongly as

---

[6] Please note the difference between this analysis and our comments on monitoring: for the latter, we control the input; now we want to evaluate the output.

[7] Of course, there are many other arguments against the use of LoC as an indicator for 'output', e.g. the lack in comparability between programming languages ([5] states that the number of LoC per function point differs between 21 (Smalltalk) and 320 (Assembler)). Also [7] presents data on productivity and output in software teams. Pioneering work has been done by [4].

[8] This shows the difference between the 'moral risk' and 'objective risks' (which cannot be influenced by the agent).

possible to be useful for deriving the bonus (or deduction). At the same time, the appearance and impact of exogenous influences needs to be limited. This suggests that principals should invest more time in clearly defining the requirements, building up the infrastructure and defining project guidelines etc. before appointing an external firm because the agent then has a clear understanding of his responsibility for the success.

Also of importance is that, if we talk of 'quality' of software, we should keep in mind the value of the software during its entire lifecycle. Thus, aspects like maintainability, reusability, scalability etc. need to be considered to evaluate the long-term 'quality' of software. This seems to be an important field of research in which we lack theoretical and empirical knowledge.

3.2.4 Strengthen Loyalty

The deeper the emotional binding of the agent, the less opportunistic behavior will occur since values like loyalty, fairness or friendship are parameters in the agent's utility function that is maximized. Therefore, the principal should try to build up such an emotional pressure which prevents the agent from exploiting the chances for opportunistic behavior. Again (even if we have not explicitly mentioned this aspect before) this shows how important close cooperation and intense contact between the two parties are.

3.2.5 Verification by Independent Authorities

We discussed this aspect briefly in 3.1.1. But it is applicable here as well since observed opportunistic behavior reduces the reputation. If the reputation of a contractor is (made) public, exploiting discretionary freedom is not optimal since it lessens the chances for future projects.

3.3 Hold-up

Before, we discussed situations before and during a project. Now imagine the principal realizes that an agent maximizes her own profit instead of the goals of her customer or did so in a former project. Even then, in some cases it might be optimal for the principal to further employ the agent because of irreversible investments ('sunk costs') which make it more costly to appoint a new contractor compared to keeping this agent. This situation could have been planned by the agent before, what we would call hidden intention. In our application domain, hold-up can be seen in cases where one contractor sold tools, software or methods that can be used, maintained or provided by others only if money and time is invested for educating people and/or making them familiar with the system.

Most often, implicit knowledge (due to missing documentation) or exclusive skills of the agent lead to hold-up. We often hear that some project manager request one certain external developer since 'he is the only one who knows the system'. Some mechanisms to avoid such a dependency are outlined in this section.

3.3.1 Avoiding specific Investments

Dependence of the principal is given only if a change of the agent would create costs that are high enough to give the current contractor the opportunity to increase prices above market value. This suggests that the principal should insist on using standard tools and well-known technologies to guarantee that there are a sufficient number of other developers who could maintain the system. This argument undermines the high value of a proper, detailed documentation the agent should request from every contractor. Also coding standards and other guidelines are one step to assure maintainability of code and therefore partially avoid hold-up by the software firm that developed the system.

If innovative projects are outsourced and/or if certain technologies or programming languages etc. are used for the first time, more than one external firm should participate. Since there are no own competencies, this is the only way to avoid the dependence from one supplier and guarantee that in the future only a competition price will need to be paid.

Aber genau diese innovativen Projekte werden schwerpunktmäßig outgesourct, um sich fehlendes Know-how ins Haus zu holen (Quelle:). Beide Aspekte (Know-how aufbauen, hold-up vermeiden legen nahe, eigene MA viel stärker als normalerweise einzubinden. Bspw. sollen externe MA im Haus arbeiten oder in Projektgruppen interne einsetzen o.ä.

3.3.2 Increase the number of interactions and decrease their volume

An agent will not exploit a dependence of the principal if there is a risk to being not considered for other projects. Through repeated interaction, the agent can build up good reputation. The basic argument has been outlined in sections 3.1.1 and 3.2.5, where it has been explained for the time before or during a project.

3.3.3 Strengthen Loyalty

As outlined in section 3.2.4, an atmosphere of loyalty and partnership can help to avoid that the agent exploits potential hold-up situations. However, the question how this can be reached is not the focus of this work.

4. CONCLUSION

This paper describes the economic foundation of outsourcing relationships, helps to understand reality in software projects and gives some basic recommendations. Nonetheless, the software engineering is far from being a value-oriented discipline. Therefore, we need to analyze the mechanisms in more detail, analyze some case studies and adapt solutions from the economic theory to the real life of consulting and development. Especially answers on how to measure productivity of developers and the quality of project results would be helpful. Another important research area seems to be how to measure the value of a software system over its entire lifecycle; this is closely related to work on maintenance and total cost of ownership.

5. REFERENCES

[1] Akerlof, G.A. The Market for 'Lemons': Quality Uncertainty and the Market Mechanism. In *Quarterly Journal of Economics*, Vol. 89, 1970, 488-500.

[2] Arrow, K.J. *The Limits of Organization.* W.W. Norton & Co. Inc., 1974.

[3] Bea, F.X. and Göbel, E. *Organisation.* Lucius & Lucius, 1999.

[4] Boehm, B.W. *Software Engineering Economics*. Prentice Hall, 1981.

[5] Humphrey, W.S. *A Discipline for Software Engineering*. Addison-Wesley, 1994.

[6] Jiang, J.J., Klein, G., Hwang, H.-G., Huang, J. and Hung, S.-Y. An Exploration of the Relationship between Software Development Process Maturity and Project Performance. In *Information & Management*, Vol. 41, 2004, 279-288.

[7] Jones, C. *Applied Software Measurement*. McGraw-Hill, 1996.

[8] Krishna, S., Sahay, S. and Walsham, G. Managing Cross-Cultural Issues in Global Software Outsourcing. In *Communications of the ACM*, Vol. 47(4), 2004, 62-66.

[9] Lacity, M.C. and Wilkocks, L.P. *Global Information Technology Outsourcing*. Wiley, 2001.

[10] Paulk, M.C., Weber, C., Curtis, B. and Chrissis, M. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Addison-Wesley, 1995.

[11] Rothschild, M. and Stiglitz, J. Equilibrium in Competitive Insurance Markets: An Essay on the Economics of Imperfect Information. In *The Quarterly Journal of Economics*, Vol. 90(4), 1976, 630-49.

[12] Spence, M. Job Market Signaling. In *The Quarterly Journal of Economics*, Vol. 87, 1973, 355-374.

[13] *Spice - Software Process Improvement and Capability Determination*. Technical report, The SPICE Project, 1995.

[14] Stigler, G. The Economics of Information. In *Journal of Political Economy*, Vol. 69, 1961, 213-225.

[15] Wilson, C. A Model of Insurance Markets with Incomplete Information. In *Journal of Economic Theory*, Vol. 16, 1977, 167-207.