Information systems development methods and reducing information asymmetry: a way to decrease project escalation in outsourcing?

A Kataja
arttu.kataja@avarko.com

Tuure Tuunanen
University of Auckland, tuure@tuunanen.fi

Follow this and additional works at: http://aisel.aisnet.org/ecis2006

Recommended Citation
http://aisel.aisnet.org/ecis2006/126
INFORMATION SYSTEMS DEVELOPMENT METHODS AND REDUCING INFORMATION ASYMMETRY: A WAY TO DECREASE PROJECT ESCALATION IN OUTSOURCING?

Katja, Arttu, Helsinki School of Economics, Runeberginkatu 14-16, 00100 Helsinki, Finland, arttu.kataja@avarko.com

Tuunanen, Tuure, The University of Auckland Business School, Private Bag 92019, Auckland, New Zealand, tuure@tuunanen.fi

Abstract

The paper describes an on-going research project that uses agency theory and the concept of information asymmetry to explore possibilities in decreasing project escalation in outsourced information systems (IS) development projects. We analysed three IS development methods for this purpose. As results of the analysis we proposed a framework for studying the information asymmetries of IS development methods. The framework includes three features that are interrelated to each other, namely sharing agent behavior and outcome information and information sharing system. We used the framework to study standard contract terms for IS development in Finland (IT2000). Specifically, we wanted to see if the features of the proposed framework are visible in IT2000 contract. The results were promising in that aspect. Moreover, we found that IT2000 contract does not take well into account the more flexible agile IS development methods and can therefore lead into higher information asymmetries in IS development projects than would be typical for flexible development methods. Hence, we conclude that this area should be further investigated to understand how the framework could be used in IS development projects to decrease project escalation. This research will extend our understanding of the reasons behind project escalation in IS development projects.

Keywords: information asymmetry, agent theory, IS development methods, agile software development, Extreme programming, outsourcing, IT contracting.
1 INTRODUCTION

Studies show that every third information systems (IS) development project suffer from some kind of symptom of project escalation, e.g. (Keil, Mann et al. 2000), and thus development projects are likely to exceed budget constraints, go overtime, demand more resources than expected etc. Furthermore, IS development outsourcing has been a topic of lively discussion lately. Even though many success stories exist, there are many outsourced projects that Keil et al (2000) might describe as “runway” projects. One reason researchers have presented is that many outsourcing contracts have been very complex and contained multiple clauses aimed at reducing unexpected contingencies, possible cost increases, and opportunistic service provider behaviour (Lee, Huynh et al. 2003). Furthermore, agile IS development methods have gained much interest among academics and practitioners (Boehm and Turner 2003; Erickson, Lytinen et al. 2005), which do not often emphasize project management features present with more traditional IS development methods (Merisalo-Rantanen, Rossi et al. 2005).

The fact that projects are tempted to run wild is no new thing in IS literature. The literature is rich with suggestions for decreasing failure risks by concentrating attention to one risk source, such as requirements determination and analysis or managing development process more efficiently. Recently, researchers have also taken interest in understanding the more in-depth reasons that could possibly cause of the phenomena. Keil et al (2000) suggests that major factors which relate to escalation are project complexity and completion effect. Hence, they have presented that the use of avoidance theory could feasibly explain the reasons behind project escalation. However, their study touched the use of agency theory as well. They found that if information asymmetry was present, there was sevenfold increase in project escalation (Keil, Mann et al. 2000). Furthermore, researchers have suggested that reducing information asymmetry is an important topic when you consider feasibility of IS development methods to a particular development task (Peppers and Tuunanen 2005). However, we were not able to find literature that would consider using a particular IS development method to reduce information asymmetry within a project. Hence, we find interesting to compare potential benefits arriving from the use of agile development methods vs. more traditional IS development methods.

The paper approaches to investigate how agency theory, and more specifically information asymmetry, could be used to decrease project escalation in IS development projects. We will take IS development methods as the target of our investigation and focus into three general and well-known methods: waterfall, simple iterative and agile software development. The selection of the IS development methods includes two important polar extremes, namely traditional waterfall model and newer extreme programming model. The third method included, iterative process model represents a middle of the road approach of software development. We conceptually analyze these methods in order to estimate their support for providing aid for reducing information asymmetry. Finally, we use the findings of the analysis to examine how the most common Finnish IT outsourcing contract framework supports reducing information asymmetry.

The rest of the paper is structured as follows. First, we briefly review the selected IS development methods and their characteristics. This is followed by a review of agency theory and conceptual analysis of the two IS development methods. Then we use our findings to investigate how IT2000 contracts supports reducing information asymmetry. Finally, we discuss the results and conclude.

2 RELATED WORK

2.1 IS development methods

The history of IS development can be characterized as a venture to find solutions for raising the productivity of programmers, making systems less defective or developing systems by techniques that
pay more attention to the end-users and their needs. The story of process improvement in IS development methods began with the ‘code-and-fix’ approach, which can be perhaps described as a creative state of chaos (Merisalo-Rantanen, Rossi et al. 2005). This ad hoc way of working was blamed for containing many problems, like poorly understood requirements and problematical structure of coding and resulting in project escalation. We review briefly three exemplary IS development methods developed to resolve these issues. The main characteristics of the methods are described in Figure 1. The ‘waterfall model’ was developed to provide a systematic, sequential solution to IS development problems. It is often illustrated as a fixed sequence of consecutive steps of development phases (Royce 1970). While exact step contents and naming conventions vary according authors, the main idea remains simple: plan carefully first, then implement and at last test exhaustively (Sommerville 2001). The next step in the history of method development was a turn towards more flexible methods. This led to the development of iterative models in 1980s (Raccoon 1997). In below, we have illustrated a general iterative method. Basically, it is an incremental method where the project is divided into small waterfalls. In this example, the specification is frozen for a single increment, but the specifications of other increments may change (Mills 1980). Finally, the emergence of agile methods started in late 1990s with Extreme programming (XP) being one of the first and better known agile methodologies (Beck 1999; Erickson, Lytinen et al. 2005; Merisalo-Rantanen, Rossi et al. 2005). The driving forces for the new methods were the fast changing requirements and technology. In some areas of software development the change was considered so fast that the specification couldn’t be frozen even for a single, few months’ long iteration. Agile methods give typically only a little weight for writing documents and instead emphasizes working software and very flexible development process with a strong customer focus through out the whole development process (Beck 1999).

![Diagram showing Waterfall, Iterative, and Agile models](image_url)

*Figure 1: waterfall, iterative and agile models, modified from (Beck 1999)*

2.2 Agency theory and Information Asymmetry

Agency theory extended the research of risk sharing in the early 1970 by introducing so-called agency problem to the research (Ross 1973; Eisenhardt 1989). The theory describes a relationship between a principal who delegates work to an agent, who then performs that work. The central notion of the theory is so called agency problem which occurs when the principal and the agent have conflicting goals and it is expensive or difficult for the principal to determine whether the agent has behaved appropriately (Eisenhardt 1989).

Eisenhardt (1989) states in her review of agency theory that traditionally the agency problem has been solved by (a) reducing the information asymmetry and (b) creating reward based incentives for the agent. Her review concludes that information asymmetry can be reduced by establishing various information systems between the agent and the principle in for an example form of frequent progress reports. Initially, *Sharing agent behavior information* should reduce the agent’s temptation of moral hazard and other unwanted behavior as the risk of getting caught acting against principal’s interest raises (Eisenhardt 1989). The other widely used approach is to tie agent’s rewards to the project
outcomes, opposed to the standard behavior based rewards to the agent (Ekanayake 2004). The motivation of outcome based reward is to establish incentive for the agent to act according the principal’s best interest and transfer the principal’s goals to the agent (Ekanayake 2004). However, outcome based rewards transfer not only the principal’s goals but also principal’s risks to the agent (Ekanayake 2004), and thus this could be described as sharing project outcome information. The third and last way of reducing information asymmetry according to the agency theory is to establish a dedicated information sharing system between the agent and the vendor (Eisenhardt 1989). According Eisenhardt (1989), these information sharing systems are for example agreed upon ways of reporting the project status.

Lichtenstein (2004) has presented that agency theory could be applied to IS development contracting. According to him the client (principal) and the vendor (agent) are likely to have conflicting goals for the contracting project and their risk preference regarding to the project can be different. In addition of the self-evident contradiction between maximizing vendor profit and minimizing client cost, the goals regarding the ease of software maintenance, software robustness and long-term system effectiveness are similarly likely to be different to at least some extent. Furthermore, information asymmetry between the client and the vendor is likely to be very high, because a piece of modern software is a highly complex engineering artefact and most definitely client organization cannot understand all the technical details of it. Finally, Lichtenstein (2004) states that high information asymmetry makes the software contracting principal-agent relationship prone to moral hazard and adverse selection and that agent may, for example, reserve excess amount of time for even simple tasks and then over charge the client.

In context of software contracting fixed price projects can be seen as an outcome based reward for the vendor. When the project price is fixed, the vendor has a strong incentive to complete the project under budget. As agency theory predicts, using a fixed pricing scheme not only transfers the goals from the client to the vendor but also the risks (Lichtenstein 2004). On the other hand, the possibility of giving intellectual property rights (IPR) to the vendor is another way of base the reward on the project outcome. The release of IPR to the vendor can establish a strong shared incentive of software quality between the client and the vendor (Eisenhardt 1989).

3 CONCEPTUAL ANALYSIS OF IS DEVELOPMENT METHODS

We have chosen the reviewed three IS development methods as a target for our conceptual analysis, namely waterfall, general iterative and agile development methods. All of the above methods provide interesting built-in ways of reducing the information asymmetry between the principal and the agent. Table 1 summarises our analysis of the methods. It presents the selected IS development methods in the context of agency theory.

Sharing agent behavior information. According to agency theory, understanding how agent conducts his work is very important for principle (Eisenhardt 1989). Therefore it is interesting to analyse how the development methods share information about the vendor behaviour. We have a similar view with Boehm et al. (2003) that the interactions between the client and the vendor of plan driven models are conducted on demand and therefore are usually infrequent compared to the agile models, where interactions are conducted through dedicated onsite customers on daily basis. On waterfall model, the principal cannot really assess the agent’s behavior directly, because the interaction is based on demand, and if there is no problems to solve, the agent maybe be working quite a long times without any interactions with the principal (Boehm and Turner 2003). However, the outcomes of a single waterfall model step provide the principal an indirect way of assessing the agent’s behaviour. Agile development potentially gives the principal rather much information about the agent’s behavior assuming the principal is capable of understanding the software development activities on some level. The key factor is agile models’ emphasis on the informal and very frequent interactions between the principal and the agent as the client is supposed to dedicate full time employee to the development project as an onsite customer. From a principal’s point of view, literally
being full time where the actual work is done provides potentially outstanding means of understanding the agent’s behavior (Beck 1999).

<table>
<thead>
<tr>
<th>Ways of reducing information asymmetry</th>
<th>Sharing agent behavior information</th>
<th>Sharing project outcome information</th>
<th>Information sharing system</th>
<th>Over all relative information asymmetry</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>After each model step</td>
<td>After each iteration (frequency: months)</td>
<td>Working software once in few weeks, outcomes of model steps</td>
<td>High</td>
</tr>
<tr>
<td></td>
<td>Only after whole project</td>
<td>After each iteration (frequency: weeks)</td>
<td>Working software once in few weeks, onsite customer</td>
<td>Medium</td>
</tr>
<tr>
<td></td>
<td>Outcomes of model steps</td>
<td></td>
<td></td>
<td>Low</td>
</tr>
</tbody>
</table>

Table 1: IS development methods and principal-agent information asymmetries (Beck 1999; Sommerville 2001; Boehm and Turner 2003)

**Sharing project outcome information.** While agency theory emphasizes understanding the behavior of the agent, in the end the buyer of IS development project cares only about the concrete outcome of the project. Therefore reducing uncertainty of the project outcome is very valuable for the principal. Waterfall and agile models reduce the uncertainty of the project outcome very much differently assuming the most important project outcome to the principal is the working information systems. While waterfall delivers the working information systems only after the whole project is finished (Sommerville 2001), the agile models deliver pieces of fully working information systems after each short iteration (Beck 1999). Hence, information asymmetry of waterfall model is concluded to remain high because principal assess agent’s behavior only occasionally (Boehm and Turner 2003) and will see the project outcome, working information systems, only after the whole information systems is implemented (Sommerville 2001). Compared to the waterfall, iterative model lowers the information asymmetry in one significant way: a piece of working information systems is revealed to the principal after each iteration (Sommerville 2001) and therefore the information asymmetry is regarded to be medium in the presented framework. However, the lowest information asymmetry is found from agile development as principal can assess the agent’s behavior daily via on-site customer and the pieces of working information systems is presented to the principal more frequently as the iteration cycles are shorter (Beck 1999).

**Information sharing system** describes agreed upon ways of delivering project artefacts, for example documentation or software components to the project client. With waterfall we can state that built-in information sharing systems relate to individual phases of the method. Each phase delivers an artefact, beginning with requirements determination and ending with the working information system. The iterative methods add prototypes and other early versions of the information system to this. Hence, its capability of reducing information asymmetry is considered to be better. Finally, agile software development increases the iteration frequency with providing weekly or bi-weekly builds of the information system. Furthermore, it emphasises the informal information exchange between the principal and the agent as it is preferred that the customer has on-site representation.

**Over-all relative information asymmetries** summarizes the over all information asymmetries of the process models. It’s worth noting that this synthesis is by no means meant to be absolute, moreover it is relative in its nature. Information asymmetry of waterfall model is concluded to remain high because principal assess agent’s behavior only occasionally (Boehm and Turner 2003) and will see the project outcome, working information systems, only after the whole information systems is implemented (Sommerville 2001). Compared to the waterfall, iterative model lowers the information asymmetry in one significant way: a piece of working information systems is revealed to the principal after each iteration (Sommerville 2001) and therefore the information asymmetry is regarded to be
medium in the presented framework. However, the lowest information asymmetry is found from agile software development as principal can assess the agent’s behavior daily via on-site customer and the pieces of working information systems is presented to the principal more frequently as the iteration cycles are shorter (Beck 1999).

4 CASE STUDY OF IT2000 OUTSOURCING CONTRACT

In this section we briefly describe the Finnish industry standard contract terms for IS development (later IT2000) (Keskuskauppakamari 2000) and analyze it using the developed framework (Table 1). Our purpose is to examine how IT2000 contract supports information asymmetry when the main IS development method is an agile model. The contract framework has been derived from the traditional IS development perspective and it includes strong support for waterfall method. For example IT2000 framework strongly assumes upfront specification as the purpose of IT2000 EAP is essentially to implement a well-documented written specification. Correspondingly acceptance testing and warranty of IT2000 EAT contract refers only to the documented functional specification. Our analysis of IT2000 framework and the agile models showed deep contradiction between the two as the very core idea of agile development is to agree the functional definition during the implementation phase and focus on the working software instead of exhaustive documentation (Beck 1999).

IT2000 terms and conditions are written by collective of Finnish governmental officials, information technology industry associations and buyer representatives1. For the scope of this research, only the general terms and conditions (IT2000 YSE) and special terms and conditions for the deliveries of customised software (IT2000 EAT) apply. Table 2 presents the information asymmetry related features of IT2000 EAT contract in relation to our framework of information asymmetry of IS development methods (Table 1).

<table>
<thead>
<tr>
<th>Ways of reducing information asymmetry</th>
<th>IT2000 EAT features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sharing agent behavior information</td>
<td>Minimum requirement for project organization: a project steering group and a project manager</td>
</tr>
<tr>
<td>Sharing project outcome information</td>
<td>Vendor receives automatically intellectual property rights and therefore benefits directly from good quality software product</td>
</tr>
<tr>
<td>Information sharing system</td>
<td>Minimum requirements for reporting: progress report at least once a month</td>
</tr>
</tbody>
</table>

Table 2: information asymmetry related features of IT2000 EAT contracts (Eisenhardt 1989; Keskuskauppakamari 2000)

Sharing agent behavior information. Clear project organization is a basic requirement for a good agency problem solution. If the project organization is poor, the information flow between the principal and the agent is very likely to remain inadequate, and the unwanted agent behavior may therefore increase. IT2000 requires the parties to set up a project steering group to implement the agreement and to generally organize the co-operation. (IT2000 EAT, 4.1) The vendor is required to appoint a project manager who will report the status and progress of the project to the steering group. (IT2000 EAT, 4.2)

Sharing project outcome information. Our analysis showed that the aspect of the sharing project outcome information in the context of IT2000 is more focused on rewards the vendor can gain than sharing the information in more equal manner. Namely, the IT2000 framework provides the intellectual property rights to the vendor which can be seen as a form of outcome based reward

---

1 Keskuskauppakamari, Suomen Logistiikkayhdistys ry, Tietotekniikan Liitto ry and Tietotekniikan Palveluliitto TIPAL ry.
(Lichtenstein 2004). The rationality is that as the vendor gains the intellectual property rights, well-implemented project serves also the best interests of the vendor because the vendor can reuse the outcome with multiple other clients (Eisenhardt 1989). However, while good quality product may be in the best interests of both sides, some other agency problem related issues might raise because of the way the intellectual property rights are managed in the IT2000 framework. Vendor might set completely different goals for the product than the client has originally asked because of the future reuse and resell potentials. For an example, the vendor might start building a whole product platform within the project while the client only asked the simplest possible working solution.

Information-sharing system. In terms of agency theory, the required reports are an information system that provides the principal with information about the behavior of the agent. The IT2000 requires supplier to report on the progress of the project in writing at least once every month. If the project pricing scheme is time and material, the monthly report has to include information on the working time used (IT2000 EAT 6.1). In order to apply for small projects, the reporting system is kept light weight, and therefore the reporting system provides only relatively small influence to the agency problem. For bigger and riskier projects, the reporting system could be easily extended to cover more detailed and more frequent information flow from the vendor to the client.

5 DISCUSSION AND CONCLUSION

The paper explores the possibility of using agency theory and the concept of information asymmetry in order to aid to decrease project escalation in outsourced IS development projects. We analysed three IS development methods for this purpose. As a results of the analysis we proposed a framework (table 1) for examining the information asymmetries of IS development methods. The preliminary findings propose that this field has potential in understanding how information asymmetries, like Keil et al (2000) have suggested earlier.

Our framework includes three features that are interrelated to each other. These were results of a conceptual analysis of agency theory and IS development method literatures. Initially we see that sharing agent behavior information is a key for understanding the dynamics how the principal and the agent are interacting together. We share the view of Boehm et al. (2003) that the interactions between the client and the vendor differ based on the selected IS development method, i.e. plan driven models usually interact with the client less frequently compared to the agile methods. Secondly, we present that sharing project outcome information differs among IS development methods as they reduce the uncertainty of the project outcome in distinctly different ways. This is an outcome of the different view on producing iterations of IS artefacts (Beck 1999; Sommerville 2001). Finally, we propose that IS development methods differ in ways they provide information-sharing system in relation to information asymmetry. Our conceptual analysis of the selected methods shows that the relative information asymmetry is likely to remain high in waterfall projects, medium in iterative projects and low in agile projects.

Furthermore, we used the developed framework to examine one of the most common standard contract terms for IS development in Finland (IT2000) (Keskuskauppakamari 2000). Specifically, we wanted to examine if the proposed three items of our framework are visible in IT2000 contract that is often used in outsourcing projects in Finland and if the features share the view given by the agency theory literature. The results of the case study were encouraging. They support the use of our framework and further research in the area. Our analysis illustrate that sharing agent behavior information and information-sharing system were present as suggested by the literature. However, we found that even though agency theory recommends sharing project outcome information (Eisenhardt 1989; Ekanayake 2004) IS development methods and IT2000 contract take a contradicting view to the matter. It can be said that traditionally IS development methods safe guard principal’s interests, whereas IT2000 contract more focuses on agents’ rewards. What's more, we found that the IT2000 contract framework does not take well into account the more flexible IS agile development methods, like extreme programming. From perspective of agency theory, information system development project may be
forced to choose between committing to agile methodology or using IT2000 contract framework. Choosing IT2000 over agile development may potentially increase information asymmetries in IS development projects. This is worrying finding since researchers have claimed that this can cause even sevenfold increase in project escalation (Keil, Mann et al. 2000).

In the next phase of the study we will do an interpretive case study of an outsourced IS development project. With the case study we aim to elaborate how information sharing affects principal’s and agent’s behavior and how the information sharing system for the project is constructed. What features it includes and how it is supposed to support the development project. Furthermore, we find it interesting if we could extent the discussion to acquisition of package software. This would probably lead to further elaboration of the contextual factors affecting the information asymmetry ratings that we excluded from this study. The results are hoped to contribute extending our understanding how information asymmetries can be decreased in outsourced IS development projects and how using agency theory to could contribute in resolving these risks.

Acknowledges
Authors would like to thank Tarja Virma / Tietoalojen liitto ry for assisting this study.

References
14(1): 57-75.
Erickson, J., K. Lyytinen, et al. (2005). "Agile modeling, agile software development, and extreme
Terms and Conditions for IT Procurement, Talentum.
Communications of the ACM 46(5): 84-89.
Merisalo-Rantanen, H., M. Rossi, et al. (2005). "Is Extreme Programming really something new:
Review 63: 134-139.
Proceedings of IEEE WESTCON, Los Angeles, USA.