Collaboration drivers and breakdowns in large scale software customization

Samuli Pekkola
Tampere University, Finland
samuli.pekkola@tuni.fi

Matti Rossi
Aalto University, Finland
matti.rossi@aalto.fi

Kari Smolander
LUT University, Finland
kari.smolander@lut.fi

Abstract

Software is not developed in a vacuum. Development teams and organizations must react to various incidents, such as changes in personnel, development practices, project objectives, and business environment, and all kinds of delays occurring during large-scale software development projects. These incidents challenge contemporary communication, collaboration and development practices. They also require an ability to adapt to the new situation. We argue that the problems in reacting to those incidents and the inability in implementing corrective actions are the main reasons for software project failures. We identify four drivers of collaboration. They are named as contract, individual person, group of collaborators, and development process, each being appropriate in different points of time. Yet their emphasis and appropriateness vary over time. This emphasizes the developers' ability to transit between the drivers of collaboration.

Keywords: information systems development, incidents, collaboration, project management

1. Introduction

Contemporary software development takes place in networks of people. Programmers, systems designers, and other actors collaborate within and across teams and organizations, often in global settings. To succeed, they need to communicate, coordinate, and interact. Software development is increasingly social.

Single software components and programs are developed independently and integrated to serve different groups of users and their needs as comprehensively as possible. For example, enterprise resource planning (ERP) systems are composed of different "modules" or "components" for different tasks, such as finance, supply chain, stocks, and production management. Individual components and programs form larger aggregates, which, due to their size, are often complex and time-consuming to develop. As the components are integrated under the same user interface and database, they look like a single system to their users, but in reality, they are built as assemblages from parts of larger infrastructural building blocks and smaller sub systems, or tied through hooks and glue code to platforms. The development of such large-scale systems requires software development methods and practices that handle both the scale and extensive length of the project.

Large-scale software development is seldom straightforward and easy. Not only the technical project objectives but also non-technical circumstances, such as the number of actors, tightness of schedules, and size and length of the project make the development process complex. The projects are plagued by unexpected incidents, caused by the changes in the personnel, development practices or project objectives, delays, or external events, such as mergers and acquisitions or significant changes in the company's business environment. In many cases changes to the platforms that the software relies on to run can alter the course of development. The success and outcome of long projects is dependent on the ability to prepare, react, and recover from these incidents.

Unexpected incidents initiate intentional or unintentional changes to the project and its practices. These changes easily disrupt software development as they cause uncertainties and discontinuities, possibly affecting the project objectives or customer commitment. However, there is little research about the project disruptions over time and their consequences [1]. Most studies focus on short-term changes and survival strategies [2, 3], not on prolonged projects and evolving long-term relationships between the stakeholders. Large volume of software engineering research focuses on software processes, requirements, or communication [1, 4], not on organizational effects on the projects. These issues are emphasized in large projects where changes are likely to happen. Although similar large projects are studied for example in the construction context, the abstractness and intangibility of software development makes it difficult to apply the

URI: https://hdl.handle.net/10125/64425
978-0-9981331-3-3

findings from other fields in the more intangible software context.

As an example of large-scale software development, many ERP projects still fail [5, 6] despite having been studied extensively over the past thirty years. The studies have mostly focused on success and failure factors, short-term activities, and technical issues. We take an alternative view. We focus on collaboration practices between the software developers and their customers in a large-scale ERP development project, during which different incidents take place over extended period of time. We try to understand *why the project failed despite several seemingly successful attempts to recover from different incidents and crisis*.

The paper is organized as follows. First, we will present our research methods and the case description. Second, we will summarize our findings and reflect them to the literature. Finally, they are discussed in a broader context so that conclusions can be made.

2. Methodology

This case study focuses on an ERP system renewal project, taking place 2007-2014. We conducted 16 qualitative interviews in the large manufacturer company (12 interviewees) and the smallish ERP vendor (4 interviewees). The interviewees were from customer IT (Corporate CIO, ICT manager, and three technical persons), customer business (five managers and two end-users), and vendor organization (CEO and three developers), each having participated in the ERP system renewal within the last few years. The interviewees were selected by using snowball sampling, i.e. each person recommended new interviewees [7]. We asked each interviewee to tell their story about the project, its progress and challenges from his/her perspective. These stories, lasting about 49 minutes on average, were audio recorded.

By inductively and iteratively analyzing the qualitative stories [8, 9] it became evident that there were four different drivers of collaboration, each dominating in different points of time. This analysis was initially done by the first author, and later collectively discussed among all authors. Our findings will be discussed next.

3. Case study

The customer organization manufactures consumer products in Northern Europe. They have 4500 employees, with annual worldwide sales of 1400M€ (2016). Over the years, the company has grown by mergers and acquisitions to more than 3000 sales offices in 62 countries. In 2008, they identified a need to renew their sales process and its systems support, namely ERP. The customer organization (hereafter a customer) had a joint history with their ERP provider, dating back to 1997. The vendor had developed and designed a tailored ERP that had been in use ever since. Selecting a renewal partner was thus based on a fact that “they knew our processes already” (Customer CIO).

Our case story begins in 2008. The old ERP system did not meet the customer’s needs and expectations, and it was perceived as technologically outdated. The customer and vendor mutually agreed on a renewal project (hereafter the project). A contract was signed. It defined the project objectives, used technologies, modes of project collaboration – and the use of the old ERP as a basis for requirements specification. This approach turned out to be quite problematic. Customer IT support articulated that “the specification that is done with the vendor can be just email conversations” while the vendor had similar experiences: “Customer has been using the old version, and when the specifications were fixed, many things have been left unspecified. Of course, it has been assumed, by default, that they will be the same as earlier' (Vendor Senior designer). No one was pleased with the situation. It is worth noting here that this is similar to the idea that one would use one platform APIs and services as requirements for a new platform. This can work in a pure transition from one highly standardized platform to another, but not in a renewal or major upgrade, or in a radical process change.

Nevertheless, the contract, largely a standard agreement between the parties, was signed. It was written more like a memorandum of understanding rather than a binding and limiting agreement. This loose approach was perceived appropriate because of the existing long-term partnership and practices. The vendor started to develop a new system.

Then the global economy declined. The project was halted for two years, until June 2010. But the company growth did not stop. Its internationalization continued with acquiring more sales offices around the world. This meant a change in needs and requirements for the new ERP. The old ERP lost its role as an example for necessary functionality. To convey new needs and requirements to the vendor, the customer assigned a project manager to work closely with the vendor. This changed the relationship between the organizations profoundly. Instead of entering a time-consuming re-negotiation about the contract, they continued the project with a mutual understanding of what was the target. The project was driven by the will to collaborate, not by the formal contract.

Over the next two years, the project manager developed a strong vision of what is ultimately needed. Yet he perceived that “strategic and managerial planning was challenging because the vendor was not

able to present a roadmap… (Phasing) was done as hand-to-mouth (for the practical purpose in the IS product development) …" Despite these difficulties, he developed a vision and communicated it to the vendor but did not check its congruence with the customer's strategy and vision well enough. The project collaboration was not driven by the organizations, but by the project manager personally.

The project manager was formally employed by the customer. However, he was practically working in both organizations, understanding their needs, capabilities, and limitations. In February 2012, he was even appointed to the vendor's internal board of management. Optimally this dual role could have resulted in a perfect outcome. Practically, it turned out to be a chaos.

The project manager, being intensively tied with the project and with the vendor, had lost his touch to business development activities in the customer. The customer's needs had significantly diverged and changed over time since more and more sales offices were acquired and the business processes had evolved. The old ERP, the old vision, and even the new vision based on those, had become inappropriate. This was highlighted in December 2012, at the time of the first roll-out, when the customer CIO postponed the roll-outs because of immature ERP. The system was not yet ready for use. The project manager resigned in January 2013.

A new project manager was hired from another branch of industry, the highly regulated pharmacy. He emphasized his experiences, old customs and software development practices which he believed would help in overcoming the problems. He thus started to steer the project towards strict, defined and implemented development processes and practices.

However, earlier software development practices were nothing but defined. The customer IT manager concretized this in terms of tracking the ideas: "Now we have the practical problem that the ideas are forgotten. We have a traditional challenge: (how to manage a design process with changes), how to get a centralized tracking of the ideas for advancing with these properly (until the design and work tasks are incorporated into the product development)". Meanwhile different actors collaborated without clear instructions and methods, with persons they had earlier been collaborating, answering the needs of those who request them the most. This created contradictions when the formal project manager tried to enforce the strictness of the process. The others just continued their old ways of working and undefined collaboration. The relationship between the customer and the vendor was consequently driven by both the process and the customary collaboration practices. This contradiction resulted in another chaos. After four months, a crisis meeting was set to clean the air.

The customer's CIO made an intervention, and personally tried to steer the project back on track. As he brags about having "tricks that all the other [vendor's] tasks will stop if we face that kind of [major] problem", he defined the practices, gave a carrot to the vendor to deliver the ERP on time, and commanded the next roll-out in early May (in three weeks) on five test sites. The roll-out was successful in a sense that the system was installed. However, it was still far from perfect as more than 40 severe faults were identified. This resulted in a situation, where "[the customer was] doing a lot of testing (on behalf of) [the vendor]. We are identifying the defects that they should already have found (in their testing environment)" (Customer CIO).

The vendor continued fixing the bugs and developing new features with their outsourcing partner, and with the project manager acting as a boundary spanner between the customer and the vendor. The progress was minimal. The bugs remained, new features were continuously expected and expectations failed, and new roll-outs were postponed. Although the situation annoyed both the customer and the vendor, there was mutual understanding. In the words of the vendor customer support: "There are still lots of customer wishes about what they want. [They express] that they want this and that, but we've gone a bit backwards, and the customer understands it and agrees that we should focus on fixing these".

In early November, the CIO made another intervention, and summoned another meeting with the vendor. This time the contract was re-negotiated. It explicitly defined what is needed and when – in early January. As the timeline was very strict, the project objectives were mitigated. This made the project tight but realistic, which all agreed. The managers assumed the contract would steer the project to a happy end.

But it didn't. The contract neither defined the collaboration practices unambiguously nor were they clearly communicated. For example, Vendor Customer support suggested that they "should have meetings at short intervals. As our release cycle is two weeks, we have to know what to plan for the next release, and what are the most important points there." Although the project advanced significantly by the deadline, the progress was not enough. There was still a significant number of defects and missing features, even when comparing to the old ERP. New features were still about to come. Nevertheless, the vendor was happy with the progress and was not concerned about these issues. They believed the deadlines would be postponed again.

In early January, the customer CIO made a decision to terminate the 18 years long cooperation with the vendor. This was despite all others wanting to continue

the relationship and get the system ready. The CIO had lost his faith, and wanted to start from a blank slate, with a new vendor or a system provider.

4. Findings from the case

The case illustrates that at different points of time, the organization's relationship was driven by different factors: a contract, an individual person, free-form collaboration practices, or a development process. They are defined as follows:

- Contracts define roles and responsibilities, and how different incidents are handled.
- Individual person can personally react to the incidents and their consequences.
- Collaboration relies on mutual interests, voluntary cooperation, and previous experiences. Incidents and their consequences are solved by the stakeholders' common points of interests.
- Process determines the parties and procedures, and how the incidents and their consequences are resolved.

Table 1 illustrates the main project phases, incidents and different drivers for the relationship.

Each driver in Table 1 has been found appropriate in earlier literature. For example, Esteves & Pastor-Collado [10] identified the importance of an individual person for the success of ERP projects. Similarly, Saunders et al. [11] have emphasized contracts, Kotlarsky & Oshri [12] smooth and flexible collaboration and information exchange between the stakeholders, and Fitzgerald et al. [13] appropriate development process. It thus seems that the corrective actions to save the project were proven elsewhere. Why did this project then fail?

Although there were several problems, such as poor communication, undefined or ad hoc development practices, or powerless management, they were not the main sources of failures. Considering the timeline of the project, from 2007 to 2014, five years with a break, is long enough to make the corrective actions, and implement them appropriately. Those actions were made indeed. The problem was thus elsewhere, in their implementation, and in the transitions from one action and collaboration mode to the other.

The need to change the development and collaboration practices were triggered by different incidents. For example, when the old project manager resigned, his dominant role and in-depth knowledge of both the customer and the vendor could not be easily replaced. This had both positive and negative effects. The customer braced the project manager's ability "to emphasize the development priorities from our perspective. I led the opinions when there was a decision point… based on our expectations" (Customer project manager) and being informed about the vendor's problems, challenges, and directions. On the other hand, the project manager "had things so well under control because he has such a long history and he was involved with developing the old system. Now the whole initiative has not been under control in the same way even though the IT team is very professional' (Vendor Customer support). When the role disappeared, it obligated significant changes in the other actors' practices. They were not ready for the change and its entailing practices, enforced by the new project manager. The reaction to the incident failed. The

Table 1. Main phases of the project and collaboration drivers

Time	Activities and incidents	Collaboration driver
2008	Initiation	Contract
2008-10	Project halted	-
June 2010	Restart; Assigning a project manager	Collaboration
Feb 2012	Project manager appointed to the vendor's board of management	Personally by the project manager
Dec 2012	Initial roll-out	
Jan 2013	Project manager resigns	
	New project manager enters	Defined development process
April 2013	Crisis meeting because of conflicting collaboration drivers	Process and collaboration
	CIO makes an intervention	Personally by CIO
May 2013	Initial roll-out to 5 sites; numerous bugs	Defined development process, influenced by ad-hoc collaboration practices
Nov 2013	CIO makes an intervention	Personally by CIO, continue by the contract
Jan 2014	Termination of the contract	Personally by CIO

problem was thus not the incident but the actors' inability to react to it.

Large-scale software development projects are not implemented in vacuum. All kinds of incidents are possible and common; people may resign or sign in the project, organizations and environments evidently change, technologies advance, and development methods and management isms are replaced. This means that large-scale projects, potentially lasting several years, will evidently face these incidents. To cope with them is not easy and straightforward as our case illustrates (see also [14]). Yet the research has not paid attention to long-term relationships and their evolution, despite the number of failures [5, 6]. This ignorance could be one of the reasons why the organizations do not learn from past mistakes. They are not capable of reacting to incidents appropriately.

5. Conclusions

Our analysis revealed that the main reasons for the project failure were collaboration breakdowns in the transitions between the incidents. The incidents; financial crisis, the board membership and resignation of the project manager, roll-outs and materializing of the defects, and CIO interventions all affected the software development project. Meanwhile the developers at the vendor side and the IT workforce at the customer side continued their informal collaboration practices that had been successful during the past 13+ years. The implementation of corrective actions after the incidents failed.

The implementation of a change in collaboration is not easy in a hectic development project. Neves et al. [14] addressed several characteristics that affect large scale IT projects. They range from traditional project management issues such as creeping requirements, prolonged schedules and increasing costs to divergent perspectives on IT project management and its performance. Although we observed similar kind of variety in the perspectives of project management performance, we argue this disparity was not the source of failure. Development practices and ways of collaboration are often set at the beginning of the project and defined by contracts. Yet, especially in large-scale development projects, both external and internal incidents will occur and influence the project. This necessitates sensitivity to identify the incidents, responsiveness to them, adaptability to new needs, and ability to change the development practices as appropriate. For example, the financial crisis made the original contract and requirements specification obsolete. The appointment of the project manager to the board of managers overemphasized his role, which, to be successful, would have necessitated new working methods at the customer's side. We argue that the problems in reacting to different incidents and implementing corrective actions are the main reasons for collaboration breakdowns in software development. In large-scale projects, instead of focusing on rigid development practices or fine-tuning the contract, all participating organizations should also concentrate on evident incidents and how they are dealt with. This, in turn, emphasizes the social side of development; how all parties, at the customer side, at the vendor side, and at other stakeholders, react and response to incidents and corrective actions. The key to the success is on the developers' and other stakeholders' ability to adapt and change the driver of collaboration that may help in avoiding collaboration breakdowns.

Somsen et al. [15] came to similar conclusions. They emphasized the technological understanding and agility in project management practices, whereas we argue that the problem is not the technology but the ability to react to different incidents, i.e. being agile in terms of changing the collaboration driver. The importance of managing complex relationships, of which the large-scale software development projects necessitate, has also been addressed (e.g. [16]). Yet again, the focus is on a single driver, not on transition between the collaboration drivers.

We consequently suggest two contributions. First, we argue there is a need for new approaches to understand the failures in IS projects. It seems that we, as IS researchers, have studied the them from the managerial point of view without understanding the evident transitions between the managerial activities or development practices. A need for studying the dynamics and transitions in collaboration situations is thus evident. Second, our small study provides a few lessons. They can be formulated as follows:

- Prepare for evident incidents and react accordingly sooner than later.
- Ensure that the corrective action is properly implemented throughout the developer community and monitor that the new collaboration driver is respected and obeyed.
- Remember that the transitions between the drivers of collaboration will not happen instantly but will take time and face external pressures not to change.
- Do not adhere the old collaboration driver (contract, person, cooperation, process) too long as the new situation, after the incident, most likely requires new methods and modes of collaboration.

Recovering from different collaboration breakdowns are the keys to the success in large-scale software development projects.

There are some limitations in this study. First, our findings are based only on a single company and a single

case. This means all possible drivers of collaboration may have not been identified. However, we do not consider this as a problem since we were more interested in the transitions between the drivers. Second, although we interviewed the key actors in the project at its later stages, the interviews were retrospective. This means the interviewees may not be able to recall all incidents and activities that took place years before. However, we were still able to construct the big picture and the timeline of events. So, although some details may be missing, collaboration drivers and the transitions were unanimously confirmed by the interviewees.

Different incidents are indeed evident in large development projects. Those projects range from ERP renewal projects (as in this case) and large software development projects to all kinds of platforms and their development and integration. The foci of actions there should not be only on narrow project management practices but also on the agility to change the collaboration practice, mode. The larger the project, the more significant this becomes.

References

[1] P. Achimugu, A. Selamat, R. Ibrahim, and M. N. r. Mahrin, "A systematic literature review of software requirements prioritization research," *Information and software technology,* vol. 56, no. 6, pp. 568-585, 2014.

[2] J. Nandhakumar, M. Rossi, and J. Talvinen, "The Dynamics of Contextual Forces of ERP Implementation," *J Strategic Information Systems,* vol. 14, no. 2, pp. 221-242, 2005.

[3] S. Newell, C. Tansley, and J. Huang, "Social capital and knowledge integration in an ERP project team: the importance of bridging and bonding," *British Journal of Management,* vol. 15, no. S1, pp. S43-S57, 2004.

[4] A. Tarhan, O. Turetken, and H. A. Reijers, "Business process maturity models: A systematic literature review," *Information and Software Technology,* vol. 75, pp. 122-134, 2016.

[5] S. Pekkola, E. Niemi, M. Rossi, M. Ruskamo, and T. Salmimaa, "ERP Research At ECIS And ICIS: A Fashion Wave Calming Down?," *European Conference on Information Systems ECIS 2013*.

[6] L. Shaul and D. Tauber, "Critical Success Factors in Enterprise Resource Planning Systems: Review of the Last Decade," *ACM Computing Surveys,* Article vol. 45, no. 4, p. 55, 2013.

[7] E. L. Olson and G. Bakke, "Implementing the lead user method in a high technology firm: A longitudinal study of intentions versus actions," *Journal of Product Innovation Management: an international publication of the product development & management association,* vol. 18, no. 6, pp. 388-395, 2001.

[8] G. Walsham, "Doing interpretive research," *European journal of information systems,* vol. 15, no. 3, pp. 320-330, 2006.

[9] G. Walsham, "Interpretive case studies in IS research: nature and method," *European Journal of information systems,* vol. 4, no. 2, pp. 74-81, 1995.

[10] J. Esteves and J. Pastor, "Enterprise Resource Planning Systems Research: An Annotated Bibliography," *Communications of the Association for Information Systems,* vol. 7, no. 8, pp. 1-52, 2001.

[11] C. Saunders, M. Gebelt, and Q. Hu, "Achieving success in information systems outsourcing," *California Management Review,* vol. 39, no. 2, pp. 63-79, 1997.

[12] J. Kotlarsky and I. Oshri, "Social ties, knowledge sharing and successful collaboration in globally distributed system development projects," *European Journal of Information Systems,* vol. 14, no. 1, pp. 37-48, 2005.

[13] B. Fitzgerald, G. Hartnett, and K. Conboy, "Customising agile methods to software practices at Intel Shannon," *European Journal of Information Systems,* vol. 15, no. 2, pp. 200-213, 2006.

[14] F. G. Neves, H. Borgman, and H. Heier, "Success lies in the eye of the beholder: The mismatch between perceived and real IT project management performance," in *2016 49th Hawaii International Conference on System Sciences (HICSS)*, 2016: IEEE, pp. 5878-5887.

[15] Somsen, A. M., Borgman, H., Langbroek, D., & Amrit, C. (2019). Rerouting Digital Transformations-Six Cases in the Airline Industry. *Proceedings of the 52nd Hawaii International Conference on System Sciences. HICSS2019*

[16] N. A. Martin, *Project politics: A systematic approach to managing complex relationships*. Routledge, 2016.