Entrepreneurial IS Development: Why Techniques Matter and Methods Don't

Nikolaus Obwegeser(✉)

School of Business and Social Sciences, Aarhus University, Aarhus, Denmark
obweg@asb.dk

Abstract. In this article we look at the current situation of information systems development in research and mirror our findings with insights from practice. Many firms and decision makers underestimate the influence that IS development projects have on their success and competitiveness. In addition, the process of efficient development of high quality and value-creating software remains a major challenge for many organizations. After a review of general IS development approaches we draw on an analysis of the literature on method tailoring and contingency approaches in IS development in combination with qualitative empirical insights from two software development companies to posit that past research has largely focused on creating bloated, inflexible methods rather than flexible toolsets. Based on the analysis of our findings we propose an open, framework-based IS development approach that allows for the flexibility required in practice but still ensures learning processes, knowledge retention and transfer.

Keywords: IS development · Agile · Methodology · Software engineering

1 Introduction

In this article we look at the current situation of information systems development (ISD) in research and mirror our findings with insights from practice. Information systems (IS) can be the driver behind great success stories and the reason for the downfall of large companies. Nevertheless, many firms and decision makers underestimate the influence that IS development projects have on their success and competitiveness. In addition, the efficient development of high quality and value-creating software remains a major challenge for many organizations. After a review of general IS development approaches we draw on an analysis of the literature on method tailoring and contingency approaches in IS development in combination with qualitative empirical insights from two software development companies to posit that past research has largely focused on creating bloated, inflexible methods rather than flexible toolsets. Based on the analysis of our findings we propose an open, framework-based IS development approach that allows for the flexibility required in practice but still ensures learning processes, knowledge retention and transfer.

Triggered by many talks with developers and observations of ISD projects we find that, while ISD is an active field of research, academics are nevertheless largely unable to inform practitioners on practices and tools to apply in real life scenarios. Despite the

M. Kurosu (Ed.): Human-Computer Interaction, Part III, HCII 2015, LNCS 9171, pp. 218–225, 2015.
DOI: 10.1007/978-3-319-21006-3_22

vast amount of methods and different approaches to ISD discussed over time [1, 2], practitioners often openly report about their inability to apply theoretical, academic knowledge to practical settings, especially in the context of IT startups and SMEs. This situation might be traced back to the inherent complexities and challenges to ISD, as discussed in [3]. In this sense, we argue that there is a need for a radical re-thinking of the process of ISD, which has to start with an explorative investigation of the practices at the very core of the discipline instead of developing "yet another" normative method handbook.

In contrast to other ISD research efforts we start with a tabula-rasa, "an empty table" to provide an exploratory look at how highly entrepreneurial companies (young SMEs, start-ups) take up the challenge of ISD. This allows us to mirror the academic state-of-the-art with actual practices. We investigate two successful IT SMEs that have been founded after 2000 and have since grown steadily. Using a case study research design [4] including observation, interviews and document analysis we draw a picture of how these companies deal with methodology of IS development in practice. Case studies are especially useful in this situation because they allow us to investigate and understand the "how" and "why" behind observed behavior [5]. Both SMEs have decided on purpose not to have a formalized method but leave it greatly up to the specific project teams to decide which techniques, tools and practices are fit in the specific context, location and time. While ISD methods in large organizations with a long history of software development are often subject to path-dependent decision processes that are likely to hinder innovation [6], these startups can provide unique insights and learning opportunities due to their entrepreneurial spirit and affinity to innovation.

Our findings show that while research has recently moved closer to practice, from method tailoring to flexible method combinations and contingency theories [2, 7, 8], real-life practices of SMEs are yet much more radical in the way they approach ISD projects. The project teams in the two case companies decide on a mix-and-match set of techniques that they agree on in a consensual way and re-evaluate the mix regularly to adapt it to the current state of the project. Techniques found can be based on plan-based methods like the Rational Unified Process [9] (e.g. requirement planning based on use-cases), agile methods like eXtreme Programming [10] or Scrum [11] (e.g. pair-programming, iterative prototyping). Other practices we found were partially or completely self-developed (e.g. acceptance trials by non-involved colleagues). The specific mix itself is dynamic and changes whenever the need arises. This high degree of flexibility comes with an intensive need for broad knowledge about different techniques and their potential combinations – in contrast to detailed knowledge or certifications often required for specific ISD methods and processes.

Based on our findings, we propose an open framework approach to ISD, which follows the idea of the UML and other modelling frameworks (e.g. ARIS). The framework is designed to be inclusive for all ISD related topics, e.g. coding standards, user interaction, and metrics and lists potential techniques for the respective areas. In addition, the framework holds information about possible constraints and best-practice technique combinations. Thus, practitioners are given the possibility to create ad hoc

methods in a mix-and-match way, based on an informed decision process. Moreover, researchers and practitioners can add innovative techniques, combinations or practical insights to grow the knowledge base.

The remainder of this article is structured as follows. Section 2 elaborates on the methodology applied and is followed by a review of relevant literature in Sect. 3. After introducing our cases in Sect. 4 we present our findings and discussion in Sect. 5. Finally we conclude the research paper and point to future research topics.

2 Methodology

This paper builds upon an interpretive, multiple case study methodology based on qualitative data [12]. After presenting the state of the art in scientific literature on method tailoring and contingency approaches we contrast our theoretical insights with empirical data gathered from two software development companies.

Using inductive data analysis we are able to provide anecdotal evidence for our notion of a research-practice gap in IS development. A multiple case study design allows for cross-case analysis and comparison in order to generate or extend existing theory [5]. Moreover, multiple-case study research is expected to generate more robust, generalizable and testable theory than a single-case research [13].

Both case companies are SMEs according to the definition of the OECD [14], founded after 2000 and have been growing rapidly. We have chosen a qualitative research approach based on interviews and document analysis to allow for an in-depth analysis of the situation in practice. Case studies are especially valuable when studying a complex phenomenon in the practical context by means of multiple, different data collection methods in order to answer "how" and "why" questions [5].

We use interviews and document analysis in order to draw a picture of how the companies deal with methodology of IS development in practice. Mirroring real-life practices with findings from literature provides a picture of the gap that exists between theory and practice. This is in line with the claim by [7], that just as in many emerging and radically changing fields, practice is ahead of research and academia.

2.1 Selection of Case Companies

Case companies were selected among successful SME software engineering companies founded after 2000. We argue that these companies allow for better insights into the choice of ISD method since their decision processes are less framed by historically institutionalized behavior and structures. Successful companies were found by looking for continuous growth patterns in both number of employees and turnover. Both companies are mainly depending on local private and public customers and are therefore often competitors in the same market.

Table 1. Case companies

Company*	Sector	Founded	Number of employees	Interviewee
Digital Trends	ICT	2001	68	Software manager
SoftCo	ICT	2007	80	Senior developer

*Names have been changed due to confidentiality agreements.

2.2 Data Sources

Our data collection consisted primarily of interviews and analysis of ISD related documents that were provided by the companies. In addition, numerous informal talks and conversations led to the formulation of initial research hypotheses. The target informants for our interviews were software managers or senior software engineers, who were deemed to be knowledgeable about the company's choice of ISD method and the rationale behind this decision. All interviews were semi-structured and therefore partially depending on the conversational path the interviewee decided upon. All interviews and document analysis was conducted between November 2014 and February 2015. After analyzing the cases separately, we conducted a cross-case analysis to locate common themes that emerged. Our inferences are grounded on the empirical data provided by our interviews as well as the information gathered in the document analysis [15].

3 Relevant Literature

Historically, ISD has changed from small, independent experts working on specialized pieces of software to large scale software development operations. [1] provide a historical overview of ISD methods, identifying four sequential eras: pre-methodology, early methodology, methodology and post-methodology. Similarly, [2] draw a comparison between the movement from craft to industrial production of physical goods, evolving from pre-industrial to industrial and finally post-industrial making.

From the introduction of the System Development Lifecycle (SDLC), to the large scale adoption of the waterfall model [16] and heavy-weight, plan-based methods like the Rational Unified Process [9] to today's widespread use of agile methods and practices [17] as proposed in SCRUM [11] or eXtreme Programming [10]: None of the methods or method-combinations can be considered to solve all problems and complexities of ISD [3]. In addition, practitioners generally customize and adopt the formal methods according to how they see fit and need. This led to the proposal of Fitzgerald's "Method-in-Action" framework [18] of ISD, which can help to understand the multitude of influencing factors and constraint that shape a method in action, which may or may not be based on a formalized method. Reference [2] propose a contingency theory approach to method choice and tailoring. Reference [19] use a quasi-experiment to analyze the organizational challenges of enabling ambidextrous ISD processes (viz. enabling projects to use both agile and traditional methods within the same organization).

4 Case Description

In this article, we look at the current situation of ISD in research and mirror our findings with insights from practice. Being aware of fact that organizational change is often slow and follows a path-dependent logic, we have chosen not look at large scale, established companies. Our empirical data is collected from two companies in the SME sector, which have been founded after 2000 and have been growing rapidly since then. Due to the entrepreneurial atmosphere in the case companies, we were able to observe and analyze an unbiased approach to choice of ISD method. Our findings show that while the two firms pursue somewhat different approaches in the way and level they define policy and method guidelines for the development teams, they show distinct commonalities in the process of institutionalizing these processes within the organization.

5 Findings and Discussion

Our case companies provided us with detailed insights into their development environment. Both companies we investigate are software developing companies specialized on tailor-made solutions and not off-the-shelf software. In the course of our research, we were able to gather an understanding of how the organizations perceive themselves and their approach to IS development. This sometimes led to the discovery of gaps between practices that are enacted and practices as formalized. We will present both companies in detail and support our findings with anecdotal evidence from the interviews conducted. Within both companies, we could see that even in cases where only little formal requirements to the IS development process are defined, these tend to be shaped and re-defined by practitioners dynamically over and over again in a context dependent manner. Thus, any formalization of development process can be argued to be redundant, if it includes normative rules, policies, or guidelines that are interpreted context dependent.

5.1 Digital Trends

The company has been established in the early 2000 s and initially decided to start off by using SCRUM. Aside from software development, a big part of their work is related to consulting their clients in strategy, technology and project management.

As a result of constant evaluation and re-formulation of their methodological approach, Digital Trends nowadays has a very radical approach to software development methodologies. While they still consider themselves to be very close to SCRUM, the interviewee (SM) points out on many occasions that they use a different combination of tools and techniques in almost every project. Moreover, they start out with a methodological configuration (often non-formalized) at the beginning of a project which is subject to change as the project evolves. This is how they pay tribute to the truly agile nature of their work, as described by SM.

In contrast to more structured approaches to ISD, Digital Trends perceives the people as the most critical risk factors to their projects and aligns everything they do and how they do it along to this organizational context. “It’s all about creating value for the customer” is what SM points out, and not about following blindly to a specific methodology.

Every decision in Digital Trends is done in a completely democratic way. While a manager can ask specific people to join a new project, it is entirely up to them to decide if they agree and see fit to their own skillset or rather opt for a different task. Within project teams, all team member decide together on what techniques can and should be applied internally (e.g. pair programming) and externally (e.g. meeting intervals with customers). This creates a high level of both trust and responsibility which fosters the individual’s involvement above what is specified in work contracts or policies. According to SM, Digital Trends tries to apply this idea also with their customers, by not having (written) contractual agreements with them. Other research shows that contracts can in fact have a negative influence towards inter-personal and inter-organizational trust which leads to a people fulfilling contractual agreements but acting less benevolent [20].

Overall, the set of techniques used at Digital Trends is based partially on SCRUM, eXtreme Programming, Kanban and partially self-developed. The specific configuration for each project at a certain point in time is context dependent and can be altered by the team-members only.

5.2 SoftCo

After their foundation in 2007, SoftCo decided to adapt SCRUM to their own needs and requirements. A major criterion for SoftCo is to remain agile in all project phases, so they designed their own SCRUM based, internally formalized method called “BlueSky” (name changed).

The interviewee (SD) at SoftCo states that when developing a system, they always have three perspectives in mind: the end-user, the business requirements and the technological quality of the system. In order to achieve high levels of satisfaction on all three of these perspectives, they argue that that BlueSky allows them to remain as agile as necessary but provides basic structures to follow. To describe SoftCo’s understanding of ISD methods, SD uses a quote by former US President Dwight D. Eisenhower: “Plans are nothing, planning is everything”. This is to understand why they keep a model like BlueSky as a planning framework to act within, while the real planning is flexible and context-dependent.

SD points out that the simple notion of “one size fits all” methodologies will just not work in practice, so they have tailor-made approaches for their projects – based on the overall idea sketched out in BlueSky. SoftCo sees themselves as practicing a version of SCRUM that is close to the original intention of the authors - as a framework to allow for all different kinds of software developing techniques and concepts - while many companies nowadays use SCRUM in a much to formalized and by-the-book understanding.

6 Conclusion and Future Research

In this paper we point to the fact that while research on ISD methods has produced an enormous amount of literature that provides normative guidelines, what many practitioners are actually looking for is a toolbox-like collection of techniques and best practices that can be applied and combined in a mix-and-match way. Many methods - be it traditional or agile – tend to be too rigid in the way software development is approached. ISD companies nowadays often are not only engineering companies, but they have to combine skills and management support for a wide range of related tasks from strategy and technology consulting to training and education. The ISD method has to be able to fit their unique needs and dynamically adapt to changes.

Thus, we argue a framework based approach that focuses more on the level of techniques and less on the level of methods and process designs is highly valuable to practitioners. Our research is based on the findings from two case companies that allowed us insights into how entrepreneurial organizations address the problem of selecting the right ISD practices. We find that while both organizations initially aim to stick close to what literature and theory can offer them, they tend to drift away in the course of practice due to the gap between needs in practice and theoretical models. Future researchers are encouraged to pick up these ideas from the empirical base and re-consider flexible, framework-style approaches to ISD that can efficiently support practitioners in their daily work.

References

1. Avison, D., Fitzgerald, G.: Methodologies for developing information systems : a historical perspective. In: Avison, D., Elliot, S., Krogstie, J., Pries-Heje, J. (eds.) The Past Future of Information System: 1976–2006 and Beyond, pp. 27–38. Springer, Berlin (2006)
2. Austin, R.D., Devin, L.: Research commentary —weighing the benefits and costs of flexibility in making software: toward a contingency theory of the determinants of development process design. Inf. Syst. Res. **20**, 462–477 (2009)
3. Brooks Jr., F.P.: No Silver Bullet—Essence and Accidents of Software Engineering. Comput. **20**(4), 10 (1987). doi:10.1109/MC.1987.1663532
4. Yin, R.K.: Case Study Research: Design and Methods. SAGE Publications, Thousand Oaks (2003)
5. Benbasat, I., Goldstein, D.K., Mead, M.: The case research strategy in studies of information systems. MIS Q. **11**, 369–386 (1987)
6. Patel, P., Pavitt, K.: The technological competencies of the world's largest firms: complex and path-dependent, but not much variety. Res. Policy **26**, 141–156 (1997)
7. Conboy, K.: Agility from first principles: reconstructing the concept of agility in information systems development. Inf. Syst. Res. **20**, 329–354 (2009)
8. Ågerfalk, P.J., Fitzgerald, B., Slaughter, S.A.: Introduction to the special issue —flexible and distributed information systems development: state of the art and research challenges. Inf. Syst. Res. **20**, 317–328 (2009)
9. Rational: Rational Unified Process: Best Practices for Software Development. White paper (2001)

10. Beck, K., Andres, C.: Extreme Programming Explained. Addison-Wesley Publishing, Reading (2005)
11. Schwaber, K., Beedle, M.: Agile Software Development with Scrum. Prentice Hall, Englewood Cliffs (2001)
12. Eisenhardt, K.M.: Building theories from case study research. Acad. Manag. Rev. **14**(4), 532–550 (1989)
13. Eisenhardt, K.M., Graebner, M.E.: Theory building from cases : opportunities and challenges. Acad. Manag. J. **50**, 25–32 (2007)
14. OECD: OECD SME and Entrepreneurship Outlook: 2005 (2005)
15. Eriksson, P., Kovalainen, A.: Qualitative methods in business research. Narrative. pp. 227–243 (2008)
16. Royce, W.: Managing the development of large software systems Dr. Winston W. Rovce introduction. In: IEEE WESCON, pp. 328–338 (1970)
17. Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D.: Agile Manifesto. http://agilemanifesto.org/
18. Fitzgerald, B., Russo, N., Stolterman, E.: Information Systems Development: Methods in Action. McGraw Hill, New York (2002)
19. Ramasubbu, N., Bharadwaj, A., Tayi, G.: Does Software Process Ambidexterity Lead To Better Software Project Performance? (2011)
20. Woolthuis, R.K.: Trust, Contract and Relationship Development (2005)