Managing knowledge in software method adoption

Lars Mathiassen*

Center for Process Innovation
J. Mack Robinson College of Business
Georgia State University
P.O. Box 4015, Atlanta, GA 30302–4015, USA
Fax: +1(404) 463–9292
E-mail: lmathiassen@gsu.edu
*Corresponding author

Lasse Vogelsang

IT-University of Copenhagen
Rued Langgaards Vej 7
2300 København S. Denmark
Fax: +45 72 18 50 01
E-mail: vogelsang@itu.dk

Abstract: The complex and problematic relationship between software development methods and practices is well documented. Methods are difficult to adopt. There are several barriers toward successfully bringing methods to practice, and practices are, by the end of the day, quite different from method prescriptions. Methods are, however, continually developed, and organisations spend considerable resources trying to improve practices through adoption of new methods. We know relatively little about the long-term dynamics involved in such adoption efforts. This research reports from a three-year effort within an IT department of a large multinational company in which a new method was first introduced, then used to support organisation-wide projects, and subsequently extended based on emerging practices. The study applies two complementary knowledge management perspectives – networks and networking – to interpret the experiences from the case. The study reveals how method perceptions and approaches shift radically through different stages of software method adoption.

Keywords: software development; method adoption; knowledge management.

Reference to this paper should be made as follows: Mathiassen, L. and Vogelsang, L. (2005) 'Managing knowledge in software method adoption', *Int. J. Business Information Systems*, Vol. 1, Nos. 1/2, pp.102–117.

Biographical notes: Lars Mathiassen holds an MSc in Computer Science from Århus University, Denmark, a PhD in Informatics from Oslo University, Norway and a Dr. Techn. in Software Engineering from Aalborg University, Denmark. He is currently a Georgia Research Alliance Eminent Scholar and Professor in computer information systems at Georgia State University. His research interests focus on engineering and management of IT systems. More particularly, he has worked with project management, object orientation, organisational development, IT management and philosophy of computing. He is a co-author of Professional Systems Development-Experiences, Ideas and Action, Computers in Context: The Philosophy and Practice of Systems

Design, Object-Oriented Analysis and Design and Improving Software Organisations: From Principles to Practice. His research has been published in several international journals including *MIS Quarterly, Information Systems Journal, Information Systems Research, Information, Technology and People, Communications of the ACM* and *IEEE Software*.

Lasse Vogelsang holds a Master's degree in Information Technology from the IT University of Copenhagen and is PhD student at the IT University of Copenhagen. His research specialises in the use of systems development methods in practice, systems development methods and information systems in organisations.

1 Introduction

For decades, software development methods have been invented and promoted to support and improve software development. Software development is arguably a knowledge-intensive human activity, and methods for software development attempt to codify relevant knowledge to be shared among practitioners of the discipline. Several studies suggest, however, that the relationship between software practices and development methods is far from simple and straightforward. Often, methods are not used as prescribed; they are used differently than intended, or they are used in fragments in combination with other tools and adopted practices (Bansler and Bødker, 1993; Fitzgerald, 1997; Baskerville and Stage, 2001; Lyytinen and Rose, 2003).

This research studies in detail how a particular method was introduced, used and further extended within *SoftPharm*, the IT department of a large multinational pharmaceutical company. The purpose of the study is to understand the different ways in which codified knowledge in the form of methods impact and are impacted by software practices over longer periods of time. Such longitudinal studies might reveal complementary insights into the relationship between methods and practices that can help us understand and manage methods as useful knowledge resources for practice. In particular, they can suggest different types of tactics to bring methods into practice at different stages of adoption.

2 Theoretical background

Our paper draws upon three streams of research. First, it builds on existing knowledge on the use of methods in software practice (Section 2.1). Second, it relates to the existing knowledge on the adoption of software innovations in organisations (Section 2.2). These two bodies of knowledge provide insights into our area of concern – the adoption and use of software development methods. Finally, we present two complementary perspectives from knowledge management, those of networks and networking, which we use as lenses to make sense of the experiences from the studied case (Section 2.3).

2.1 Software development methods

It is generally assumed that methods have a positive influence on software development (Fitzgerald, 1996). In a survey, Chatzoglou (1997) found that the use of methods is beneficial in terms of economy and process improvement. At the same time, however, field studies show that methods are not as widely used as expected and not used in their entirety (Bansler and Bødker, 1993; Fitzgerald, 1997; 2000). Fitzgerald (1998b) found that methods are not the panacea for problems in software development, and they are not applied rigorously or in a uniform fashion. Mathiassen and Munk-Madsen (1986) argue that methods are appropriate in some situations and not in other situations. Each method has an application domain in which it works well. The literature suggests that methods are useful, but also that their use is highly situational, limited to particular application domains and often different from what one expects.

The relationship between methods and practice is indeed complex. Methods can have different roles in practice. Fitzgerald (1998a) found that methods play two diametrically opposed roles, an overt (rational) role and a covert (political) role. Baskerville and Stage (2001) argue that methods emerge in practice through interaction between the environment and the method. Bansler and Havn (2003) found that software developers work in more complex and less stable situations than assumed in most methods; therefore, we should abandon a rational and method-dominated view of practice. Method engineering is situational and offers method fragments for tailoring methods to specific situations (Brinkkemper, 1996; Welke and Kumar, 1991). Van Slooten (1996) proposes a framework for situated method engineering that includes a set of situational factors to guide the selection of method fragments. This approach also supports that methods can change as software projects improve their capabilities over time.

The inherent complexity of the relationship between methods and practice has implications for how we adopt and work with methods. Truex and Avison (2003) identify five main types of method engineering. They range from having a narrow technical focus to a broader focus that includes processes and organisational issues. Furthermore, they argue that the purpose of using methods has changed from aiming at universal support to supporting software practices in particular contexts. Nuseibeh *et al.* (1996) extend method engineering to include different project participants' perspectives on the system and the business domain it supports. This makes method engineering dependent on both the situation and the persons involved. Hidding (1997) investigates to what extend methods are read by practitioners and how to present a method to people with different roles. Hidding found that practitioners want methods to be presented in summary and additional information to be accessible as needed. Persons in different roles need different versions of a method; this is a challenge to method engineers to provide a variety of building blocs for effectively communicating methods and method fragments.

Fitzgerald (2000) argues more fundamentally that contemporary software development methods are based on assumptions about the organisational context that are derived from the period from 1967 to 1977. The organisational context has, however, changed during the last 30 years. Organisations and applications have become increasingly complex. Fitzgerald claims that new methods and work practices are needed to meet this challenge. Further, he argues that we can learn from current best practices to explicate and create new technologies because practice often is ahead of theory and research. This contention is supported by empirical studies. Bansler and Bødker's (1993) field study of how developers apply structured analysis in practice reveals that some

concepts and tools from structured analysis were used by the developers, but the procedures in structured analysis were not followed. They conclude that software development methods are based on assumptions and offer prescriptions that seldom are practiced. Fitzgerald (1997) found in his field study that methods are tailored to meet needs and that developers omit certain aspects of methods, not because they are ignorant, but because the omitted aspects are irrelevant for practice in the specific context. Further, he found that method use is correlated with developers' experience. Developers with low experience use methods as a template for practice (high usage), but eventually find out that methods are not universally applicable (low usage). They end up with a tailored version of methods (middle/average usage).

Our existing knowledge on the practical use of software development methods shows that practices are quite different from the intended use of methods, that methods are often used in a fragmented way in combination with other practices and, finally, that software practices encompass many complex issues that cannot be covered as general, codified knowledge in methods.

2.2 Adoption of software innovations

Our knowledge about the practical use of methods is supported by a portfolio of studies of organisational adoption of software innovations. Leonard-Barton (1987) has studied factors that have a positive influence on the adoption of software methods. He found that programmers are more likely to use methods if supervisors, influential peers and their clients support the methods' use. Sharmaa and Rai (2003) studied factors affecting the adoption of CASE tools. They found that positional power and job tenure of software managers are negatively related to the adoption of CASE tools. Premkumar and Potter (1995) found that the following five variables where important to differentiate adopters from non-adopters of CASE tools: existence of a product champion, strong top-management support, lower IT-related expertise, perception that CASE tools have an advantage over other technologies, and cost-effectivity of CASE tools. The non-distinguishing factors for adoption and non-adoption were complexity and organisational and technical compatibility. Kozar (1989) found adopters to be younger, to have spent less time in their organisation and in software practice, not to currently use a method and to be more optimistic about being able to fit methods into organisational practices. Zmud (1982) investigated centralisation and formalisation as factors influencing adoption of software technologies. Zmud (1984) applied the push-pull theory and found that the theory was not validated for software technology adoption. Instead, he found that the type of innovation, to some extent, influences the organisational adoption process. Sherif and Vinze (1999) investigated the barriers to adoption of software reuse at the organisation and individual levels. They found that barriers at the individual level were caused by barriers at the organisation level.

A wide range of factors that affect adoption of software innovations is summarised by Mathiassen and Sørensen (1997) into five questions that need to be addressed when adopting new software technologies. The questions are the following: why to implement, what to implement, which technology to implement, where to implement and how to implement the technology. They recommend that software technology implementation is undertaken as an iterative process, re-addressing the five questions before starting a new iteration. Mathiassen and Sørensen (1996) also discuss the adoption of CASE tools. They argue that practitioners lack knowledge about CASE tools and suggest experiments as

means for gaining this knowledge. Furthermore, they suggest that not all aspects of CASE tools have to be utilised, especially when the development organisation is on the lower levels of the capability maturity model. Finally, it should be taken into consideration that CASE tools are not introduced in an organisational 'vacuum' and that CASE tools influence the organisation while the organisation at the same time influences the CASE tool. In line with this, Fichman and Kemerer (1997) argue that the assimilation of complex technologies takes place through a process of organisational learning (Attewell, 1992). They found that organisations are more likely to initiate and sustain assimilation of software innovations when they have a greater scale of activities over which the cost of learning can be spread, when they have more extensive knowledge related to the innovation and when they have a greater diversity of technical knowledge and activities.

We have found only few studies of adoption of software methods over longer periods of time; none of these studies focus on how the role of methods changes over time. Lyytinen and Rose (2003) show that several diffusion of innovation (Rogers, 1983) factors such as past experience, own trials, ease of use, learning by doing and standards strongly affect the adoption of process innovations related to information technology development. Lyytinen and Rose focus on the factors influencing the decision to adopt a process innovation (such as a method) but not to what extent the innovation is actually used after the adoption. Further, Lyytinen and Rose focus on several innovations and the reasons for adopting them, but not on how one particular innovation is used and changed over time. Another example of a long-term study of method usage is Orlikowski's study of CASE tools implementation (Orlikowski, 1993). Orlikowski (1993) shows that the implementation of CASE tools involves organisational change over time. But Orlikowski focuses on the approach to implement CASE tools (radical and incremental) and not on how the role of CASE tools changes over time.

Existing literature identifies several factors that facilitate or inhibit adoption of software innovations. The enablers include support from stakeholders, technology advantages and prior experience. Reported barriers include technology complexity, lack of compatibility and traditions. Based on these, guidelines are reported to support adoption practices. Despite this body of knowledge, there continues to be widespread assimilation gaps between initiated and completed adoption initiatives in software organisations. Most of the available research focuses on the adoption process; the literature provides few discussions of the relationship between adoption and later use of methods. We seem to know little about the long-term dynamics that take place as methods are brought into practice.

2.3 Knowledge management

In the following, we analyse data from a longitudinal study of adoption and subsequent use of a software development method from a knowledge management point of view. To do that we adopt Swan *et al.* (1999) framework of diffusion of innovations seen as knowledge creation and sharing. This model distinguishes between two complementary perspectives on organisational implementation of technology – networks and networking (Table 1). The networks and networking perspectives represent a fundamental distinction within the knowledge management literature. It corresponds to the distinction between codification and personalisation as suggested by Hansen *et al.* (1999) and the distinction between organisational and technical as described by Tiwana (2002,p.294).

Table 1 Networks and networking

Networks	*Networking*
The critical success factor is technology.	The critical success factor is trust and collaboration.
Knowledge for innovation is equal to objectively defined concepts.	Knowledge for innovation is social constructed and based on experience.
Knowledge can be codified and transferred through networks: IT has a crucial role.	Much knowledge is tacit, shared and made sense of through active networking within and between groups and teams.
Gains from KM include exploitation through the recycling of existing knowledge.	Gains from KM include exploration through the sharing and synthesis of knowledge among different communities-of-practice.
The primary function of KM is to codify, capture and transfer knowledge through networks.	The primary function of KM is to encourage knowledge sharing through networking.
The dominant metaphors are the human memory and the jigsaw (fitting pieces of knowledge together to produce a bigger picture in a predictable way).	The dominant metaphors are the human community and the kaleidoscope (creative interactions producing new knowledge in sometimes unpredictable ways).

Source: Swan *et al.* (1999)

The networks perspective emphasises the use of technology as a means for knowledge sharing and integration. Knowledge is perceived as objective and unproblematic to codify and transfer through networks of technology. The gains from knowledge sharing and integration are achieved through the exploitation of knowledge. The main goal of knowledge sharing and integration is the support of human memory. In the context of software methods, the network perspective implies a focus on creating the right method for a development context. The gains from the method are its reuse and the sharing of best practices. Best practices are perceived as easy to codify, transfer and integrate into existing software practices.

In contrast, the networking perspective focuses on trust and collaboration among the involved practitioners. Face-to-face interaction and sharing of tacit knowledge is recognised as an important part of knowledge sharing. Knowledge is seen as being socially constructed and based on personal experience. Knowledge is shared and made sense of within communities-of-practice through exploration. The primary function of knowledge management is to encourage knowledge sharing through networking. In the context of software methods, the networking perspective implies sharing and integration of methods through networking within groups and teams. The critical success factor is the exploration through sharing and synthesis of the use of a method in social groups. The primary function of the method is to encourage knowledge sharing through networking.

3 Research method

Our study addresses the following research question: What is the role and dynamics of networks and networking during organisational introduction, use and innovation of software development methods?

We have researched this question through a longitudinal, interpretive case study (Walsham, 1993; Yin, 1994). The strengths of this approach is that it allowed us to study

the dynamics involved in bringing a method into use over a three-year period and to gain a deep understanding of how and why the relationship between method and practice evolved from a knowledge management point of view. The limitation is that our study draws upon a particular software development method and a specific organisational context. The findings are therefore exploratory in nature.

The empirical work took place between January 2000 and June 2003. The data is primarily collected in three projects: the Method Project, the Development Project and the Extension Project. The field study was undertaken as a practice study (Mathiassen, 2002).

The Method Project was undertaken to adopt a method and to overcome challenges faced in the information technology department at *SoftPharm*. The project lasted for ten months. Data collection was based on observation, semistructured interviews and document analysis. We observed and took notes during weekly meetings in the project. Eight semistructured interviews were conducted. Four of the interviews took place a couple of months into the project, and the remaining four interviews took place after the project was finished. The interviews focused on themes related to software development, to the adoption of the method and to the Development Project where elements from the method were tried out.

The purpose of the Development Project was to develop a web application that will enable doctors to collect patient data. The Development Project was initiated in the fall of 2002 and lasted for about five months. The main data collection technique was observation. Two researchers spent 1–2 days a week for four months on observing meetings and other activities in the Development Projects. Additionally, several informal talks took place, and documents from the project were collected. After the web application was developed, four semistructured interviews were conducted with key participants in the project. A working paper with preliminary findings was written and discussed with the interviewees.

The Extension Project was undertaken to codify a work practice. The work practice was to sketch user interfaces, and it emerged among usability people in the information technology department at *SoftPharm*. The work practice was codified to make it part of the method (*i.e.*, the method that was adopted in the Method Project). The project took place during the fall of 2003 and lasted for four months. The research was undertaken as participant observation. The purpose was to investigate the process of extending the method. The main contribution to the project from the researcher was the analyses of relevant work practices and participation in the meetings as a collaborator (Baskerville and Wood-Harper, 1998). Project meetings were recorded and documents were collected for analysis. After the project ended and the method was extended with new standardised practices, a meeting with the project manager took place to find out to what extent and how successfully the extension was in use.

4 The case

The introduction, use and innovation of the software development method at *SoftPharm* are presented in the following. First, we examine the three projects from a network perspective (see Table 1). We then examine the same three projects from a networking perspective (see Table 1). Finally, we synthesise the findings by discussing how the method was brought into practice from a knowledge management point of view.

4.1 A networks perspective

The interpretation of the case from a networks perspective is summarised in Table 2. The three columns named Introduction, Use and Extension relate to the three projects studied. The Introduction column covers the Method Project, the Use column the Development Project and the Extension column the Extension Project.

Table 2 A changing networks view on the method

Introduction	*Use*	*Extension*
Method introduced to solve problems in *SoftPharm's* projects	Method is used as an organisation-wide framework	Method offers framework for codifying and sharing new method fragments
Method is a codified and customised process	Templates, standards, and process models are reused from project to project	New diagramming technique was codified to fit within method
Same method reused across all projects	Exemplar documents used to guide method use	The codified diagramming technique was made available on the intranet
Method promoted as *SoftPharm's* new development practice	Knowledge about method available on intranet	The purpose was to transfer and reuse new knowledge between projects
Method transferred based on information, seminars, reading, and intranet	Method repository serves as database and frame of reference	

The aspects of the network perspective present in the Method Project were highly related to the reason for adopting the method – to solve major problems experienced within *SoftPharm's* development projects. The main problems were the *ad hoc* development and the lack of conformity among the development projects. The idea was to use a method to describe a certain development process to be used in the information technology department. The intention was to adopt and reuse the same version of the method across all projects. The method was in this way conceived by the Method Project participants, as a codified representation of processes, concepts and notations with related tools that could be transferred among projects. The main aim in the Method Project was to customise the method to *SoftPharm* by selecting a subset of features and by promoting the customised method as a comprehensive representation of *SoftPharm's* new development practice. The transfer of the method to the developers was initially based on announcements, seminars, reading and publication in *SoftPharm's* intranet. The Method Project's participants were mostly concerned about getting the method right.

The use of the method in the Development Project revealed several elements of the networks perspective. First, some of the participants had worked with the method before and also shared similar approaches to development. This helped them avoid inventing and debating everything from scratch. The project participants perceived the method to be a feasible approach to develop software in general. They also found that it provided useful support through templates, standards and process models. The templates, standards and process models were reused from project to project and used as exemplar ways of using the method. The method was seen as a repository that provided a useful database and a frame of reference for projects within *SoftPharm.* The method and knowledge about its use within *SoftPharm* was available on the intranet and used as an organisation-wide framework for software development within *SoftPharm*.

The extension of the method in the Extension Project related to the network perspective primarily through the attempt to explicate the sketching practice into a diagramming technique, which could then be codified and added to the method. The purpose of codifying the diagramming technique was to transfer and reuse it among projects and people within *SoftPharm*. The method offered the project participants a framework for codifying and sharing new method fragments. However, it also constrained the project participants through requirements given in the method to be met in order to fit the codified diagramming technique into the method. Finally, the codified diagramming technique was made available on the intranet so that the new, codified practice could be accessed and shared.

4.2 A networking perspective

The examination of the same three projects from a networking perspective is summarised in Table 3.

Table 3 A changing networking view on the method

Introduction	*Use*	*Extension*
The method represents an opportunity to reflect on and innovate practice	The method helps transfer and transform practices from previous projects	The method is a mirror for debating noncanonical practices
Experiments are conducted to assess the method	Project participants and process engineers tailor method to each project	Knowledge about diagramming is explored and shared across projects
Experiences from experiments are used to customise the method	The method complements rather than replaces developers' knowledge and experience	People with complementary diagramming skills collaborate to codify method fragment
The adoption group shares and synthesises experiences	Collaboration between project participants remains a critical success factor	Creation of a minimalist method fragment because much diagramming knowledge is tacit or unique
A community-of-practice develops amongst the adopters based on the method		Innovation leads to improved, shared understanding of diagramming issues

The adoption of the method in the Method Project was an opportunity to reflect on and innovate software development practices in *SoftPharm*. Part of the aim of the Method Project was to conduct an experiment to assess the strengths and weaknesses of the chosen method. The experiences from the experiment were used to customise the method. Additionally, the adoption group shared and synthesised a variety of experiences from different projects and departments in *SoftPharm* to facilitate the adoption of the new method. One of the outcomes of the Method Project was a community-of-practice among the project participants.

In the Development Project, the use of the method was a way to inherit and transform software development practices from previous projects to the current project. The method complements, but does not replace the project participants' knowledge and experience of working with each other. Some of the work processes were unplanned and emergent, and informal interactions between different groups in the project were a key to solving

coordination problems in the project. The project participants found that their individual skills and their ability to collaborate and solve problems were a critical factor for the success of the Development Project.

In the Extension Project, knowledge about diagramming problems and solutions was explored and shared among specialists from different projects in order to develop and explicate a diagramming technique. Much of the diagramming knowledge was considered tacit or unique for specific projects, so a complete technique that could be used for all situations was discussed, but not considered to be realistic. A minimalist approach was therefore adopted to codify a diagramming technique and to establish minimum standards for future projects. Besides the creation of the diagramming technique, an important outcome of the project was an improved, shared understanding of diagramming issues. The shared understanding was achieved because the development and explication of the diagramming technique became a mirror in which non-canonical practices could be identified and debated. This was an important activity because the project participants in the Extension Project had complementary diagramming skills and needed to collaborate and share experiences to codify the new method fragment.

4.3 Knowledge management synthesis

We have synthesised the two separate analyses of how the method was brought into practice into a comprehensive knowledge management analysis (Table 4). The 'Metaphor' illustrates the primary intention with the use of the method. The 'Method' provides a more specific description of the method use. The 'Networks' and 'Networking' provide our assessments of the importance of the two perspectives on knowledge management in each project.

The aim in the Method Project was to adopt a method to remedy problems that occurred in the development process due to new tasks and use of new technology in the information technology department. The Method Project was focused on codifying and customising the method in order to create a version which would both fit the context in the information technology department and at the same time provide a new development process. The new version of the method was created by selecting and modifying elements of a standard method. The selection and customisation of method fragments were the primary focus in the Method Project. The focus on the method shows that the introduction of the method into *SoftPharm* was dominated by the networks model. Through experiments, the adopters developed a community-of-practice around the new method. These activities were based on the networking model by emphasising the exploration of the knowledge provided by the standard method. The two complementary knowledge management approaches, networks and networking, were both present in the Method Project; the networks approach, however, played a more dominant role than the networking approach. The method was perceived as a solution for problems in the software process. Therefore, we have chosen 'method as solution' as a metaphor for method use in the Method Project.

Table 4 Changing perceptions of the method

	Introduction	*Use*	*Extension*
Metaphor	Solution	Support	Repository
Method	Codified process	Resource for projects	Framework for codification
	Customised version created through selection and modification	Language for communication about practice	Repository for storing method fragments
	Customised version represents new development process	Support is provided for tailoring method to each project	Method fragments emerge from practice
Networks	Very high weight	Medium weight	High weight
Networking	Medium weight	High weight	Medium weight

In the Development Project, the method was used directly for software development. The primary purpose was to support the project and its participants with resources such as a common language, a standard process and descriptions of roles, templates and techniques. The use of the method was optional in the project. Method fragments were explored and used by the project participants to share and synthesise a useful practice. Through the process of exploring the method, the users became a community-of-practice with a shared approach to and understanding of key development practices. The method was tailored to suit the Development Project. The tailoring shows an emphasis on the networks approach, although getting the method right was not emphasised as much as personal relations and experiences. The emphasis on personal relations indicates that the use of the method in the Development Project was dominated by the networking perspective. The method was perceived as support for the project participants in the Development Project. Therefore, we have chosen 'method as support' as a metaphor for the Development Project.

The aim in the Extension Project was to create a method fragment consisting of a technique for sketching user interfaces. The method was used as a framework for codifying an emergent development practice by providing guidelines for the creation of the new fragment. The method fragment was codified and added to the method. By codifying and including the method fragment, the method became a repository for standardising development practices and storing method fragments. The purpose of standardising and storing the method fragment was to develop a work practice that conformed better with other practices within the information technology department. The codification of development practices within the method framework was dominated by the network perspective. The main focus in the Extension Project was on codification and integration of development practices. The method innovators formed a temporary community-of-practice in which they shared and synthesised experiences. Forming this temporary community-of-practice was perceived as important, this, however, was not the main aim of the project. Therefore, networking perspective is assessed to be of medium importance. The method provided guidelines to and a repository for storing the new method fragment. The usage of the method can be perceived as a repository for work practices; therefore, we have chosen 'method as repository' as a metaphor for the Extension Project.

5 Discussion

The analysis of the case illuminates the role and dynamics of networks and networking perspectives and approaches during organisational introduction, use and extension of a software development method. The analysis shows that changes occur over time with respect to both the strategy for sharing and integrating the method and the relationship between the method and work practices.

The project participants in the Method Project emphasised getting the method 'right' and saw social aspects of the future method usage as less problematic (*i.e.*, the networks perspective dominated). The project participants in the Development Project perceived the method as helpful in developing software and found the collaboration between the project participants to be more important than the method itself (*i.e.*, the networking perspective dominated). Finally, the purpose of the Extension Project was to create a method fragment by codifying a work practice to make it conform better to other method fragments and make it available through the method (*i.e.*, the networks perspective dominated).

The study illustrates how the emphasis on networks and networking for knowledge sharing and integration changes over time as the method is brought into practice. This change in emphasis suggests that adoption of software innovations, like a software development method, is not so much about choosing one particular knowledge management strategy to facilitate the adoption. It is more about combining different strategies to suit the situations and the purposes of adopting the innovation as it evolves over time.

Orlikowski (1993) suggests two strategies for implementing a process innovation, which is a radical and incremental strategy. While this distinction is different from the networks and networking approaches, our research suggests that it is not an either-or situation, but that strategies such as the radical and incremental can successfully be combined in practice. The adoption at *SoftPharm* had definite radical aspects during the Method Project, while the Development Project and the Extension Project were approached incrementally.

McKenney and McFarlan (1990) suggest that adoption processes passes through different phases: initiating the adoption, experimenting with the innovation, controlling the way in which the innovation is used and transferring the innovation to other domains. Again, we can see the adoption process at *SoftPharm* following this rough pattern. The Method Project is part of the initiation and experimentation phase; the Development Project is part of the control phase; and the Extension Project is part of the transfer phase. The overall dynamics of networks and networking during the adoption of the software development method at *SoftPharm* is consistent with existing models of adoption of innovations. It does, however, focus on how knowledge is created, shared and integrated, and it provides valuable additional advice on how to bring software development methods into practice.

The other key aspect of our analysis is the different roles that software development methods can play as they are brought into practice. Our study shows three different roles captured through the metaphors 'method as solution', 'method as support' and 'method as repository'. There are other roles that methods can play and that are relevant for research and practice to understand (*e.g.*, a rational role and a political role) (Fitzgerald, 1998a). In these terms, our study has focused on and elaborated the rational role by

emphasising the different ways in which knowledge is created, shared and integrated to support software method adoption. Our study has not emphasised the political role of methods to the same extent. In addition, our study strongly confirms that methods are used in fragments, and it further shows how new fragments can be added as the method is brought into practice. The project participants in the Method Project selected specific method fragments to use at *SoftPharm*, and the project participants in the Extension Project added a new method fragment. The method fragments were chosen based on situational characteristics. This confirms that there are pragmatic reasons for not using methods in their entirety (Fitzgerald, 1997). The different purposes of using and adding fragments were driven by the desire to fit the emerging version of the method to the particular context at *SoftPharm*.

The three roles that software development methods can play when brought into practice are well aligned with Lundell and Ling's (Lings and Lundell, 2004; Lundell and Lings 2004) three stakeholder perspectives on software development methods: the systems developer perspective, the concept developer perspective and the product developer perspective. The systems developer perspective is the perspective of the users of a method. The concept developer perspective is the perspective of the inventor and designer of a method. The product developer perspective is the perspective of the people who provide the artifacts and support needed to use the method. These three perspectives correspond quite closely to the three metaphors we have identified in *SoftPharm's* adoption of the software method. The Development Project and the support metaphor correspond to the systems developer perspective because this is the perspective of the user of the method. The Extension Project and the method as a repository metaphor correspond to the concept developer perspective. Finally, the Method Project and the method as a solution correspond to the product developer perspective because the main aim in the Method Project was to adopt a proper method.

Our findings have implications for both research and practice. The study suggests that knowledge management theory is highly relevant in helping us to better understand the challenges involved in bringing software development methods into practice. There are no simple and general knowledge management strategies available for that purpose. Each information technology organisation has to adopt a repertoire of strategies and tactics and to select and tailor these to the different needs and situations that emerge during the implementation and use of methods. The framework suggested by Swan *et al.* (1999) and similar approaches suggested by Hansen *et al.* (1999) and Tiwana (2002,p.294) are valuable frameworks for guiding further research into this area. They also serve as useful guidelines for tailoring knowledge management practices to the needs of contemporary information technology organisations (Mathiassen and Pourkomeylian, 2003).

6 Conclusion

The analysis compares and contrasts three types of relationships between methods and practices. It relates them to existing knowledge about stages of technology adoption. It also discusses the implications for software practice and research. In particular, the analysis leads to complementary understanding of how organisations can effectively manage software method adoption to take advantage of the opportunities presented without falling into the well known traps of non-adoption, pseudo-adoption or literal-adoption.

Our analysis reveals three quite different types of relationships between method and practice. First, the method was seen as a solution to perceived problems in the existing software practices. Second, the method was seen as a support for planning and executing ongoing software projects. Third, the method was seen as a repository for inventing and codifying new knowledge based on software development experiences. The knowledge management emphasis on networks and networking, as suggested by Swan *et al.* (1999), consequently changed considerably over the different stages of bringing the method into practice.

Acknowledgements

We would like to thank the numerous people at *SoftPharm* for their invaluable help. We would also like to thank Peter Carstensen for his involvement in the data collection. Some of the work has been conducted under the DIWA project funded by the Danish Research Councils.

References

Attewell, P. (1992) 'Technology, diffusion and organizational learning: the case of business computing', *Organization Science*, Vol. 3, pp.1–19.

Bansler, J.P. and Bødker, K. (1993) 'A reappraisal of structured analysis: design in an organization context', *ACM Transactions on Information Systems*, Vol. 11, pp.165–193.

Bansler, J.P. and Havn, E.C. (2003) 'Improvisation in action: making sense of IS development in organizations', *Proceedings of Action in Language, Organizations and Information Systems (ALOIS)*, Linköping, Sweden.

Baskerville, R. and Stage, J. (2001) 'Accommodating emergent work practices: ethnographic choice of methods fragments', *Presented at IFIP WG 8.2*, Boise.

Baskerville, R.L. and Wood-Harper, T. (1998) 'Diversity in information systems action research methods', *European Journal of Information Systems*, Vol. 7, pp.90–107.

Brinkkemper, S. (1996) 'Method engineering: engineering of information systems development methods and tools', *Information and Software Technology*, Vol. 38, pp.275–280.

Chatzoglou, P.D. (1997) 'Use of methodologies: an empirical analysis of their impact on the economics of the development process', *European Journal of Information Systems*, Vol. 6, pp.256–270.

Fichman, R. and Kemerer, C.F. (1997) 'The assimilation of software process innovations: an organizational learning perspective', *Management Science*, Vol. 43, pp.1345–1363.

Fitzgerald, B. (1996) 'Formalized systems development methodologies: a critical perspective', *Information Systems Journal*, Vol. 6, pp.3–23.

Fitzgerald, B. (1997) 'The use of systems development methodologies in practice: a field study', *Information Systems Journal*, Vol. 7, pp.201–212.

Fitzgerald, B. (1998a) 'A tale of two roles: the use of systems development methodologies in practice', in N. Jayaratna, B. Fitzgerald, A.T. Wood-Harper and J. Larrasquet (Eds.) *Educating Methodology Practitioners and Researchers*, UK: Springer-Verlag, pp.89–96.

Fitzgerald, B. (1998b) 'An empirical investigation into the adoption of systems development methodologies', *Information and Management*, Vol. 34, pp.317–328.

Fitzgerald, B. (2000) 'Systems development methodologies: the problems of tenses', *Information Technology and People*, Vol. 13, pp.174–185.

Hansen, M.T., Nohria, N. and Tierney, T. (1999) 'What's your strategy for managing knowledge?', *Harvard Business Review*, March–April, pp.106–116.

Hidding, G. (1997) 'Reinventing methodology: who reads it and why?', *Communications of the ACM*, Vol. 40, pp.102–109.

Kozar, K.A. (1989) 'Adopting systems development methods. An exploratory study', *Journal of Management Information Systems*, Vol. 5, pp.73–86.

Leonard-Barton, D. (1987) 'Implementing structured software methodologies: a case of innovation in process technology', *Interfaces*, Vol. 17, pp.6–17.

Lings, B. and Lundell, B. (2004) 'Method-in-action and method-in-tool: some implications for CASE', *Proceedings of ICEIS 2004 – Sixth International Conference on Enterprise Information Systems*.

Lundell, B. and Lings, B. (2004) 'Method in action and method in tool: a stakeholders perspective', *Journal of Information Technology*, accepted for publication.

Lyytinen, K. and Rose, G. (2003) 'Disruptive information system innovation: the case of internet computing', *Information Systems Journal*, Vol. 13, pp.301–330.

Mathiassen, L. (2002) 'Collaborative practice research', *Information Technology and People*, Vol. 15, p.321.

Mathiassen, L. and Munk-Madsen, A. (1986) 'Formalizations in systems development', *Behaviour and Information Technology*, Vol. 5.

Mathiassen, L. and Pourkomeylian, P. (2003) 'Managing knowledge in a software organization', *Journal of Knowledge Management*, Vol. 7.

Mathiassen, L. and Sørensen, C. (1996) 'The capability maturity model and CASE', *Information Systems Journal*, Vol. 6.

Mathiassen, L. and Sørensen, C. (1997) 'A guide to manage new software engineering tools', in T. McMaster, E. Mumford, E.B. Swanson, B. Warboys and D. Wastell (Eds.) *Facilitating Technology Transfer through Partnership: Learning from Practice and Research*, London: Chapman and Hall, pp.257–272.

McKenney, J.L. and McFarlan, F.W. (1990) 'The information archipelago-maps and bridges', in T.D. *et al.* (Eds.) *Software State-of-the-Art: Selected Papers*, Dorset House Publishing, pp.99–116.

Nuseibeh, B., Finkelstein, A. and Kramer, J. (1996) 'Method engineering for multi-perspective software engineering', *Information and Software Technology*, Vol. 38, pp.267–274.

Orlikowski, W.J. (1993) 'CASE tools as organizational change: investigating incremental and radical changes in systems development', *MIS Quarterly*, Vol. 17, pp.309–340.

Premkumar, G. and Potter, M. (1995) 'Adoption of Computer Aided Software Engineering (CASE) technology: an innovation adoption perspective', *ACM SIGMIS Database*, Vol. 26, pp.105–124.

Rogers, E.M. (1983) *Diffusion of Innovations*, 3rd edition, New York: The Free Press.

Sharmaa, S. and Rai, A. (2003) 'An assessment of the relationship between ISD leadership characteristics and IS innovation adoption in organizations', *Information and Management*, Vol. 40, pp.391–401.

Sherif, K. and Vinze, A. (1999) 'A qualitative model for barriers to software reuse adoption', *Presented at the 20th International Conference on Information Systems*, Charlotte, North Carolina, USA.

van Slooten, K. (1996) 'Situated method engineering', *Information Resources Management Journal*, Vol. 9, pp.24–31.

Swan, J., Newell, S., Scarbrough, H. and Hislop, D. (1999) 'Knowledge management and innovation: networks and networking', *Journal of Knowledge Management*, Vol. 3, pp.262–275.

Tiwana, A. (2002) *The Knowledge Management Toolkit: Orchestrating IT, Strategy, and Knowledge Platforms*, 2nd edition, Upper Saddle River, NJ: Prentice Hall PTR.

Truex, D.P. and Avison, D. (2003) 'Method engineering: reflections on the past and ways forward', *Presented at the Ninth Association for Information Systems Americas Conference on Information Systems (AMCIS 2003)*, Navigating the Torrents of Technology, Tampa, FL.

Walsham, G. (1993) 'Interpretive case studies in IS research: nature and method', *European Journal of Information Systems*, Vol. 4, pp.74–81.

Welke, R. and Kumar, K. (1991) 'Method engineering: a proposal for situation-specific methodology construction', in W. Cotterman and J. Senn (Eds.) *Systems Analysis and Design: A Research Agenda*, New York: Wiley.

Yin, R.K. (1994) *Case Study Research: Design and Methods*, 2nd edition, Newbury Park: Sage Publications.

Zmud, R.W. (1982) 'Diffusion of modern software practices: influence of centralization and formalization', *Management Science*, Vol. 28, pp.1421–1431.

Zmud, R.W. (1984) 'An examination of "Push-Pull" theory applied to process innovation in knowledge work', *Management Science*, Vol. 30, pp.727–738.