ADOPTING THE KNOWLEDGE EMBEDDED IN NEW METHODS – THE CHALLENGE OF ALIGNING OLD AND NEW PRACTICES

Backlund, Per, University of Skövde, Högskolevägen, P.O. Box 408, SE- 541 28 Skövde, SWEDEN, per.backlund@ida.his.se

Abstract

There are many reports on how Information Systems Development (ISD) methods are adapted before they are used. The need to customise methods is explained by the variety of systems that are developed and the various situations in which information systems can be developed. This means that the development process knowledge embedded in an ISD method can be used in different ways in different situations. The study presented in this paper is a follow up of a study of how a widespread ISD method was adapted and introduced in an organisation. In this paper we focus on how the development method was actually used in a project. The paper extends an earlier model of analysis for characterising information systems development in terms of knowledge work. The model recognises four classes of knowledge work: routine, craft-like, professional, and creative knowledge work. We use that particular model to elaborate on how the various kinds of knowledge work presented interact in an actual development situation. The main contributions of the paper are the application of the analysis model on an empirical case and the extension of the analysis model to comprise the interactions between the different types of knowledge work.

Keywords: information systems development methods, method use, knowledge work , case study

1 INTRODUCTION

There are many reports on how Information Systems Development (ISD) methods are adapted before they are used, see e.g. Avison and Fitzgerald (2003), Fitzgerald (1997), Fitzgerald et al. (2002), Madsen and Kautz (2002), and Westrup (1993). This paper takes the perspective of information systems development as knowledge work (Iivari, 2000) supported by a development method. According to Iivari (2000) the need to customise methods is explained by the variety of systems that are developed and the various situations in which information systems can be developed. This means that the development process knowledge (Iivari, 2000; Backlund, Hallenborg, and Hallgrimsson, 2003) embedded in an ISD method can be used in different ways in different situations. In short, we view the ISD method as a means for managing knowledge about the ISD process.

Furthermore, Iivari (2000) suggests that there is a need for empirical studies of the interplay between tacit and explicit knowledge in ISD work. In this paper we view commercial development methods as knowledge artefacts, i.e. explicit development process knowledge, which can be introduced and used in development organisations. There are reports on method adaptation. However, these reports tend to focus on method adaptation and method use from an organisational point of view. There is also the field of method engineering (Brinkkemper, 2000; Ralyté and Rolland, 2001) with a focus on the method as such. Typically, method engineering is concerned with how new methods are constructed by combining method fragments, thus creating a new method or method configuration.

Carroll (2003) describes a situation in which the project characteristics lay a foundation for the method selection. The selected method is then adapted and used in the project. In this paper the situation is different in that the method is prescribed by company standards and the major adaptations at the organisational level have already been done. Carroll (2003) describes the activities in the above mentioned stages whereas this paper focuses on the method use stage. Hence, the aim of this study is to deepen the investigation reported in Carroll (2003) and extend the results of Backlund et al. (2003). Furthermore, we complement the findings in Carroll (2003) since we deal with a situation in which the chosen method is to be considered as mandatory. The aim of this paper is to provide a description and an analysis of how the method is actually used at the project level. In order to be able to conduct the analysis we will draw from theories in the area of knowledge management, see e.g. Nonaka and Takeuchi (1995), Davenport and Prusak (1998). We find this approach suitable since we view information systems development as knowledge work (Iivari, 2000). Furthermore, we find it fruitful to contribute to the cross-sectional debate called for by Hirschheim and Klein (2003).

As can be seen, the phenomenon can be studied at different levels, i.e. the artefact level (focusing on the method as such, see e.g. Brinkkemper (2000) and Backlund (2002)); the organisational level (focusing on how organisations adapt and introduce methods, see e.g. Fitzgerald et al. (2002) and Backlund et al. (2003)); or at an individual and project level, which is the aim of this paper. Moreover, in this paper we highlight and discuss method use from a knowledge point of view, i.e. how is the knowledge embedded in a method used. Thus this study extends and complements the above mentioned types of studies. The paper reports on a case study made at the IT department of a major car manufacturer. In summary, the contributions of this paper are:

- An extension of existing theory of ISD method use in organisations.
- An identification of knowledge work factors based on the framework of Iivari (2000) in an empirical case.
- An extension of the framework of Iivari (2000) to comprise the interaction between the various classes of knowledge work in ISD.

The rest of the paper is organised as follows. In section 2 we elaborate on our view of ISD methods and method use, and characterise ISD as knowledge work. In section 3 we give a brief introduction to the case and present our research set up. In section 4 the case findings are presented and analysed and in section 5 we provide a closing discussion.

2 IS DEVELOPMENT METHODS AND METHOD USE

In this section we will define and discuss our view of what it means to use a method. The underlying assumption is that information systems development is knowledge work that is becoming increasingly knowledge intensive. Hence the need for improved method use.

Avison and Fitzgerald (2003) define an ISD method as a collection of procedures, techniques, tools, and documentation aids which help system developers in their work. A method also consists of some sort of life cycle model that breaks down the work in phases and iterations (if applicable). Commercial methods are typically products including manuals, education and training, consultancy support, CASE tools, and different types of templates (Avison and Fitzgerald, 2003). We can view methods as a way to regularise and formalise the good practise of experienced developers. According to Wastell (1999) methods may be perceived as reified (materialised) bodies of knowledge. This view is essentially the same as perceiving the method as a cognitive artefact. A cognitive artefact is defined as an external object that helps the user to decrease the cognitive load in performing a task, i.e. representing knowledge in structures that are external to the human mind (MITECS, 2003). The knowledge embedded in the method render the expertise necessary to carry out complex tasks.

Schönström and Carlsson (2003) describe how methods can be used as knowledge enablers, i.e. to develop knowledge in organisations, by stimulating individual knowledge development; by supporting communication in terms of serving as a communicative framework, and; by facilitating the sharing of individual knowledge and its transformation to organisational knowledge. The underlying assumption is essentially the same as Iivari (2000) presents, namely that information systems development is knowledge work. Moreover, information systems development is an increasingly knowledge intensive task. Iivari (2000) and Backlund et al. (2003) define the knowledge embedded in an information systems development method as development process knowledge. *Development process knowledge* includes knowledge about how to run a development project. Development process knowledge also includes the knowledge of how to apply methodologies and technologies for IS development, as well as skills in programming languages. However, when we deal with knowledge about developing information systems there are numerous circumstances, described by e.g. Iivari and Huisman (2001), Avison and Fitzgerald (2003), and Middleton (1999), to be taken into consideration. Some examples include the organisational habits present and their effect on the utilisation of a method and organisational culture as well as the individual skills of the developers.

We view knowledge work in a similar fashion as do for example Nonaka and Takeuchi (1995), Davenport and Prusak (1998), and Wiig (1993). This view can be described in terms of the two sub-areas of knowledge-building and a knowledge-use. Modern organisations spend large efforts in organising their knowledge and its use so that knowledge work can be facilitated. Nonaka and Takeuchi (1995) and Wiig (1993) define two types of knowledge: explicit knowledge and tacit knowledge. Explicit knowledge can be articulated in natural and formal language, which makes it ‘easy’ to transfer between people via e.g. documents and other types of records. Tacit knowledge has to do with personal knowledge that is embedded in personal experience and is therefore not so easy to formalise and record. According to Nonaka and Takeuchi (1995), tacit knowledge is an important, but overlooked, factor. We are aware of the limitations of these definitions in relation to epistemological research, but they serve their purpose well in that they are frequently used.

In this paper we view ISD methods as development process knowledge that has been made explicit by the method developer. Hence the two dimensions tacit and explicit are relevant and there is also a tacit dimension of development process knowledge both in terms of the work process as such as well as in terms of how the knowledge embedded in ISD methods are actually used by the individual developers in a project setting.

According to Iivari and Maansaari (1998) the broad scope of development methods implies that there are different roles involved in using them, e.g. analysts, developers, future system users, and project managers. There may be an explicit use of a specific method in an organisation or there may be

implicit use of a method in terms of the internalised knowledge and ways of working among the developers. Finally, the method may serve different roles in systems development. A method may serve as: a rule to determine or regulate action; a resource to support action; a reminder of actions to be taken; a model of the ideal process that may not be possible to follow in practice; a vehicle of learning.

Suchman (1987) p. viii defines the concept of situated actions as "[...] however planned, purposeful actions are inevitably *situated actions*. By situated actions I mean simply actions taken in the context of particular, concrete circumstances. [...] plans are best viewed as a weak resource for what is primarily *ad hoc* activity." Hence, we do not follow plans in the strong sense that is sometimes suggested by method vendors. A main claim by Tolvanen (1998) is that methods should not be viewed as universally applicable. Instead, method knowledge is viewed as situational. According to Tolvanen (1998) it is not possible to have full knowledge of the development situation beforehand. It is rather a question of developers having a tacit method knowledge which is based on their reflection on the work situation than solely using predefined methods. This is in accordance with the main claims of, for example Nonaka and Takeuchi (1995) and Davenport and Prusak (1998), that all knowledge is context dependent. According to Tolvanen (1998) the situations that affect the applicability of a method can be found at different levels of an ISD organisation: organisation, project, or individual. We also recognize these levels but in addition we claim that they need to be complemented by taking different views of knowledge into account. Alavi and Leidner (2001) distinguish between different views of knowledge. One of these views, the process view of knowledge means that knowledge is the *process* of applying expertise. In short, this means that the knowledge (embedded in the method) has been internalised (Nonaka and Takeuchi, 1995).

When we describe the application of a method we may hence refer to both explicit and tacit development process knowledge:

- Explicit method use refers to situations when parts of the development method were explicitly used. This includes using templates, applying a certain technique such as use case modelling, using specific workflows.
- Implicit method use refers to the tacit knowledge of the developers. This comprises situations in which the developers carry on their work in the smooth fashion characterising a craftsman. This also includes situations in which developers tend to carry out their work according to their old work habits. Hirschheim and Klein (2003) refer to this phenomenon as applicative knowledge.

Iivari (2000) proposes a categorisation of ISD work into: routine, craft-like, professional, and creative knowledge work. A variation of a classification scheme, comprising the level of applicability and the level of variety, is also introduced by Iivari (2000). Low applicability means that the knowledge has to be generalised or made more concrete to suit the situation at hand. Routine work, on the other hand, has a high level of applicability. The level of variety has to do with whether or not the knowledge needs to be generalised to cover novel cases. Information systems development is not homogenous in the sense that different tasks may be of different nature. For example, documentation may be characterised as routine work, whereas visioning a new system is more creative. Furthermore, since the factors affecting the development process change over time the adaptation of the ISD process knowledge must be continuous (Iivari, 2000). According to Iivari (2000) an increased applicability of the body of knowledge will change the knowledge work towards more routine and professional work. ISD methods, techniques, and tools exemplify this trend. However, there is still an aspect of craftsmanship and creativity which is underscored by the tacit knowledge of the developers and the organisationally embedded knowledge, i.e. in organisational routines and work habits.

Craft-like knowledge work is essentially skill-based and can only be learned through apprenticeship and practical experience. According to Keller and Dixon Keller (1996) skilful operation is characterised by an ability to recognise problems, diagnosing their cause and applying appropriate corrective procedures. Professional knowledge work is characterised by judgement and adaptation and creative knowledge work is characterised by intuition and imagination; and is therefore harder to understand and analyse (Iivari, 2000). The difference between craft-like and professional knowledge

work is subtle and some of the things characterising a skilled craftsman is also applicable for professional knowledge work. Hence we may view this in terms of a sliding scale. These characteristics are summarised in Table 1.

Type of knowledge work	**Characteristics**
Routine	Specific techniques, standard documentation
Craft-like	Skill-based, learned through apprenticeship
Professional	Judgement and adaptation
Creative	Intuition and imagination

Table 1 Some characteristics of knowledge work.

Craft-like and professional knowledge work is further characterised by three features (Hirschheim and Klein, 2003): its close relationship to a person's identity which require hard work and mistakes to acquire; its connection to personal emotions and interests which makes it dependent on social interaction and socialisation, and; its holistic nature which makes hard to split into goals and means. All these characteristics describe what we refer to as implicit method use (Figure 1), as opposed to the explicit method use which characterises routine knowledge work.

To sum up, a method comprises a body of knowledge that supports the information systems development process rather than being followed in the strong sense. Since knowledge is situated in a context the method has to be adapted to each specific situation. Furthermore, the explicit development process knowledge embedded in a method is complemented by the tacit knowledge of the individual developer in the situation of use. Hence, we must consider both the explicit knowledge of the method as well as the tacit knowledge of the individual developer.

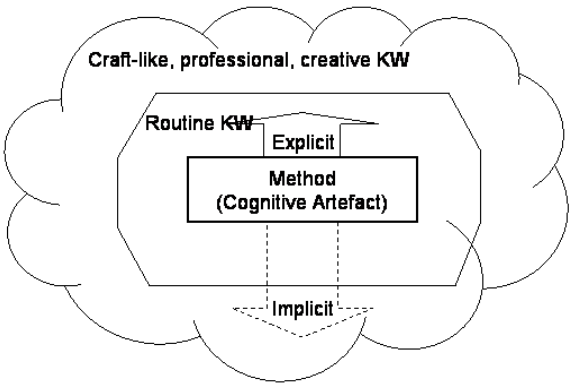

Figure 1 The dashed arrow indicates that the method is used and has an impact, but not necessarily the intended one.

In routine knowledge work we can see the indications of method work quite clearly (Figure 1). Using a template to create a new document or model according to a certain method is an example of this. Hence it is useful to refer to this level when we say that a method has been implemented in an organisation. In this sense routine knowledge work is reflected in the explicit method use. However, on the implicit level of method use it is not that easy to say that a method is implemented and has actually changed the way of working.

3 RESEARCH SET UP AND CASE PRESENTATION

This section will give an overview of the empirical setting in which the study was undertaken. It will also describe how the study was set up.

The study was made at the IT department of a major car manufacturer. In order to study how development process knowledge was utilised in an actual ISD project we followed an internal development project during the autumn of 2003. The project was initialised in June 2003 but most of the actual work was carried out during the autumn and winter. The general aim of the project is to replace the numerous system registers in use with one general register. The new system is intended to provide a more consistent record of the various systems in use, which will aid in making system maintenance more efficient. Apart from the *product* goal the project also has two *process* goals. The first one, to give the team members an opportunity to *use* the Rational Unified Process (RUP), see e.g. (Jacobson et al., 1999, Kruchten, 2000) in a real project setting; and the second one, to *introduce* a new technology and a new development tool. The major risks of the project have been identified as the lack of resources in relation to the scope of the project and the introduction of a new tool.

The project is staffed by the following roles: one project manager, one architect, four analysts, four developers, five implementers, and one test designer; with a total number of eight people involved. The project is planned to comprise of one iteration in the inception phase and two iterations in the elaboration phase. The construction phase and the deployment phase are not yet planned in detail.

Data was collected by observing project meetings and work sessions taking detailed field notes, see Table 2. Furthermore, the observation data was complemented by informal discussions with project members. We also had access to internal project documentation in different versions which has given us an opportunity to review how the different RUP artefacts have evolved over time. The different objects of observation provide different views of how the development method was used in the project. Moreover, the different sources cater for source triangulation (Williamson, 2002).

Object of Observation	**Number of Instances**
Development team meeting	8
Meeting with stakeholders	3
Tool workshop	2
Project documentation	2 (versions)

Table 2 The situations in which data was collected

The field notes and the project documentation were analysed using a combination of content analysis (Patton, 1990) and inductive analysis (Hartman, 1998). The first part of the analysis work is the *content analysis*, which aims at identifying, coding and categorising the primary patterns in the data. This has been done by reading through field notes and project documentation in order to organise the data for further analysis. Hartman (1998) describes this as *inductive analysis*; meaning that the patterns themes and categories of an analysis emerge from the data. The findings have then been mapped to the model of different types of knowledge work. Finally, we have extended the existing theory by showing how these diverse types of knowledge work interact.

4 CASE FINDINGS

In this section we will analyse the data in order to describe how the body of knowledge was used in the specific project outlined in section 3. The analysis was conducted by finding indicators of routine knowledge work, professional knowledge work, craft-like knowledge work, and creative knowledge work respectively in the observations made. By doing this we aim at describing how work was affected by the factors identified; and identifying situations of explicit/implicit use of the body of development process knowledge and how these interact.

We identified a set of indicator classes (see Table 3) which comprise a number of different instances of knowledge work indicators. We use the term artefacts to comprise the notion of artefacts in the Rational unified process, see e.g. Kruchten (2000), documents, graphical models, code, and other deliverables. We have used the term valuation to denote situations where we observed team members expressing opinions on routines as well as on their use of routine knowledge work.

In order to analyse the data collected we focused on identifying the resources that people draw on in their work. Routine knowledge work means explicitly drawing from the routines and rules of the organisation. This means that the routine aspect of knowledge work is closely associated to the organisational introduction of the new development method (Backlund et al., 2003). We also find that getting the staff to work according to the new routines (i.e. draw from the new resource) is the first step towards organisational implementation. However, we claim that this is not the same thing as having changed the craft-like and professional dimensions of knowledge work.

Routine knowledge work is easy to identify since it is found in the situations of explicit method use. Filling in document templates with specific information is an example of this. Updating an artefact refers to a situation where a model or a document in use is changed. There is a difference between these indicators since updating is more common in the daily work. The group also draws from previous experience in these situations when they use artefacts from other projects to solve project specific problems. This is typically done by using knowledge in the heads of people as opposed to using specific knowledge repositories.

Craft-like and professional knowledge work is more difficult to analyse since we must understand how experience tells (craft-like) and how theory tells (professional). These resources are harder to identify since they are not explicit as are, for example, organisational routines. Craft-like indicators are rather found in previous experiences and good practices. We also note that the concepts of craft-like and professional knowledge work are similar. The main distinction is in the more reflective character of professional knowledge work. This is an important distinction since professional knowledge work involves the use of theories, methods, and approaches, i.e. how theory tells. An example of an indicator of “Balance total situation” is that a team member tries to reach synergisms by coordinating efforts from other projects. Pattern similarities were indicated by the fact that the developers used old applications to exemplify how they had solved similar problems in the past. These similarities could be on a high level, e.g. “this is just a spread sheet application. We did that in the xx project”. The efforts to build autonomous systems and to write reusable code are illustrations of how professional strategies form the work. The professional aspects were also found in the discussions about how to interpret new instructions and the potential benefits of solving a problem in a particular way.

The purpose of creative knowledge work is to produce new ideas. In some sense creativity may also be termed a skill that can be supported by organisational routines which allow intuition and imaginative work. As expected, we found that creativity took place in all part of the process, although in different forms. For example brainstorming sessions and use of similes were more common in the early phases whereas informal use of artefacts was more common later on in the process. In this sense parts of routine knowledge work may be perceived as creative work. A typical example is the use of class diagramming techniques for sketching.

Routine KW indicators	Craft-like KW indicators	Professional KW indicators	Creative KW indicators
Use templates	Valuation of routine	Professional integrity	Informal artefact use
Use artefacts	Work habit	Strategy	Use of similes
Update artefacts	Estimation	Valuation of benefit	Brain storming
Create artefacts	Fluent work	Valuation of work habits	Excited discussion
Organise artefacts	Explicit use of routine	Valuation of approaches	
	Balance total situation		
	Pattern similarities		

Table 3 Classes of knowledge work indicators identified in the observation data

The indications of routine knowledge work are closely connected to an explicit use of the ISD method whereas craft-like, professional, and creative knowledge work rather characterises implicit method use since they presuppose that the method has been learned and internalised. Hence we can describe the interplay between implicit and explicit method use. This adds to the understanding of how methods are used since the traditional focus of research has been set on either the method as such or on the organisational view of method use, i.e. explicit use. When analysing the observation data we have studied the interplay between the different types of knowledge work presented in section two. These findings complement the analytical framework of Iivari (2000) in that they show the interplay between tacit and explicit development process knowledge. In order to describe these interactions we refer to them in terms of x supporting y, which for example could characterise a situation in which an indication of routine knowledge work has a positive relation to an indication of craft-like knowledge work. We have also identified situations where x counteracts y. These interactions can occur in various constellations and we will describe and analyse some of them.

Taking a closer look at the interactions/counteractions in Table 4 the first and obvious conclusion is that routine knowledge work should support craft-like knowledge work. This means that the knowledge embedded in the development method should support the work habits of the development team. However, we also identified a counteraction between routine and craft-like knowledge work. One of the most frequently used routine assignments were instances of use case modelling. This corresponds to the idea of RUP as being a use case driven process (Kruchten, 2000). We found some interactions between use case modelling (routine) and craft like knowledge work when existing use case descriptions were reused to develop similar system interactions. This reflects an ideal situation in which routine work supports craft-like work. Another situation in which we identified a positive interaction between use cases and craft-like knowledge work is when individual project members construct use cases which are then scrutinised by the team. This is an efficient way of using use cases to drive the analysis work.

The counteractions identified between routine and craft like knowledge work can be illustrated by a discussion between two developers about whether two use case descriptions were essentially describing the same thing; where one of the developers claimed that this is easier to implement than to write in a use case description. One reason for this may well be the good domain knowledge possessed by the developers in the project. In one situation a team member had problems to understand the concept of extension points in use cases. The document templates have a pre-specified heading for the purpose but the team members could not agree on whether it was necessary to do that work in the

particular situation. We interpret this as a situation in which routine and craft like work counteract, thus leading to a shift of focus from the problem to be solved.

The situation in which routines support routines should also be obvious. We found some indications of this situation, primarily in situations where documents were copied and adapted to suit similar situations. There are advantages of doing so but there is also the risk of cascading errors which are then tedious to correct. The opinion among the project members was that this is a limited risk when dealing with uncomplicated use cases. On the other hand, there is then the question whether this is useful duplication at all. Our impression is that routine supports routine smoothly in this fashion in simple cases. Concerning situations where routines counteract routines we have identified situations in which artefacts created in different routines do not match each other. One such example is the disagreement of concepts between the glossary and a GUI prototype. Both artefacts were used to capture and understand terms and concepts in the system but the results were contradictive.

We also identified situations in which craft-like knowledge work counteracts craft-like knowledge work. This may seem incongruous but we recognised a situation in which there was a disagreement about whether or not sequence diagrams should be drawn before the key abstractions were identified. One of the developers argued that it could be done since it is possible to identify the necessary objects while doing the sequence diagram. On the other hand, it was also argued that the objects should be created from the key abstractions.

The use case driven approach versus the data driven approach in database modelling has rendered the developers problems in the professional counteracts professional dimension. Most of the developers have a genuine data base background and according to some of them it is hard to work in a fashion where they should develop the system by modelling the way that users are supposed to interact with the system and then identify relevant data; as opposed to identifying relevant data in the domain. The developers state that this is problematic since they have good domain knowledge, as they are developing a system internal to their own organisation. Due to this situation they have preconceived notions of the domain which are problematic when trying to adopt a use case driven approach.

The interactions/counteractions are summarised in Table 4. An instance of x supporting y is denoted (x > y) and x counteracting y is denoted (x <> y). These interactions/counteractions extend the framework presented in Iivari (2000) in that they show some of the relations between the different classes of knowledge work.

	Routine	Craft like	Professional	Creative
Routine	> <>	> <>		> <>
Craft like	>	<>		
Professional			<>	
Creative				

Table 4 The interactions and counteractions identified in the material.

To summarise the case findings we claim that we have extended the existing theory in the area by identifying indicators of information systems development as knowledge work. We have showed that it is possible to classify the different indicators according to the framework of Iivari (2000). Furthermore, we have extended the framework of Iivari (2000) by showing how the different classes of knowledge work interact. By doing this we have deepened the descriptions of how a development process is introduced and used in an organisation presented by Backlund et al. (2003) and Carroll

(2003). We have also extended the results of Schönström and Carlsson (2003) by describing method use in terms of knowledge work and interactions between different types of knowledge work, hence giving an example of how information systems development methods may be used for communicative support as claimed by Schönström and Carlsson (2003). In relation to that work we have also described situations in which the development method counteracts the craft-like knowledge work represented by the tacit method knowledge of developers.

5 DISCUSSION AND CONCLUDING REMARKS

By combining the framework of Iivari (2000) with knowledge management concepts we have been able to describe how implicit and explicit knowledge interact in a set of knowledge work situations. We have extended the framework of Iivari (2000) by showing a number of interactions/counteractions between the different categories of knowledge work. The major benefit of describing these interactions is that we can gain a better understanding of how methods are used in organisations since we relate method use to the actual work habits in a project. These interactions are reflected in the interplay between implicit and explicit knowledge.

We found indications of both interactions and counteractions between the explicit and implicit development process knowledge. This is something that we expected to find and our results provide illustrations of some typical situations in which these phenomena occur. This paper suggests that there are many situations in which the explicit method use implemented through routine knowledge work counteracts the craft-like knowledge work of the professionals in the organisation. This is an essential ingredient in changing the routines of an organisation. However, if we are better aware of what routines conflict with what aspects of the craft-like work and to what extent; we stand a better chance of successfully incorporating the new work habits. We have also identified situations in which routine knowledge work counteracts routine knowledge work. These situations should be easier to identify in, for example, situations where the new development method is to be complemented by already existing processes, see e.g. Backlund et al. (2003). Hence we may view the interactions and counteractions identified here as a diagnosis instrument for organisations evaluating their incorporation of new development methods, compare Figure 1.

Even though there is routine support for class models and a glossary artefact in RUP these do not seem to cover the routine support needed for the work conducted in order to facilitate understanding of the domain concepts. There is a major element of craft-like work involved in understanding the domain which is not supported by the development method. This observation is also supported by the fact that business modelling has been substituted by an internal process (Backlund et al., 2003). This fact could be described in terms of a situation in which there is a lack of usable support for a routine task that is instead carried out in a craft like manner, i.e. in a similar way that it was done before.

Obviously there are limitations to a single case study made in one organisation. Hence we do not aim at making statistical generalisations from our material. We rather aim at a contribution of rich insight, as described by Darke et al. (1998). Furthermore, we argue that the empirical study of information systems development as knowledge work (Iivari, 2000) adds to the understanding of the area. Another potential limitation of the study is the fact that one aim of the project was for the project workers to learn a new tool and a new method. However, this could also be viewed as positive aspect of the study since it accentuates the learning perspective of information systems development (Carlsson, 2000). The largest limitation is perhaps the low priority of the project since it may have an effect on the involvement of the developers. However, we do not judge this effect as severe since the motivation for using new tools and learning the new process have been high. Furthermore, the application under development is the object of great interest in the organisation, which should serve as a motivator.

Future work associated to this study will include a more thorough investigation of the individual aspects of development process knowledge. Interesting issues concern the difference between developers with different background and level of experience. We would also suggest a more thorough

analysis of the interactions between the other types of knowledge work which have not been dealt with here.

6 ACKNOWLEDGEMENTS

This research is sponsored by the Swedish Knowledge Foundation (KK-stiftelsen). The author wishes to thank Juhani Iivari, Benkt Wangler and Anne Persson for their valuable advice.

References

Alavi, M. and Leidner, D. E. (2001) Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues. *MIS Quarterly,* **25,** 107-133.

Avison, D. and Fitzgerald, G. (2003) *Information Systems Development Methodologies, Techniques and Tools,* McGraw Hill, New York.

Backlund, P. (2002) Identifying Situational Factors for IS development Processes: Applying the Method-in-Action Framework, In *The 2002 Americas Conference on information Systems (AMCIS 2002)* Dallas, Texas.

Backlund, P., Hallenborg, C. and Hallgrimsson, G. (2003) Implementing the Rational Unified Process - A Case of Knowledge Transfer, In *The 11th European Conference on Information Systems (ECIS 2003)* (Eds, Ciborra, C., Mercurio, R., De Marco, M., Martinez, M. and Carignani, A.) Naples, Italy.

Brinkkemper, S. (2000) In *Information Systems Engineering State of the Art and Research Themes* (Eds, Brinkkemper, S., Lindencrona, E. and Sölvberg, A.) Springer, London.

Carlsson, S. (2000) Lärande systemutveckling och samarbetsformer Doctoral Thesis, Institutionen för informationsteknologi Informatik, Karlstad University

Carroll, J. (2003) The Process of ISD Methodology Selection and Use: A Case Study, In *The 11th European Conference on Information Systems (ECIS 2003)* (Eds, Ciborra, C., Mercurio, R., De Marco, M., Martinez, M. and Carignani, A.) Naples, Italy.

Darke, P., Shanks, G. and Broadbent, M. (1998) Sucessfully completing case study research: combining rigour, relevance, and pragmatism *Information Systems Journal,* **8,** 273-289.

Davenport, T. H. and Prusak, L. (1998) *Working Knowledge: How Organisations Manage What they Know,* Harvard Business school, Boston, Mass.

Fitzgerald, B. (1997) The Use of Systems Development Methodologies in Practice: A Field Study *The Information Systems Journal,* **7,** 201-212.

Fitzgerald, B., Russo, N. L. and O'Kane, T. (2002) Software Development Method Tailoring in Motorola *Communications of the ACM,* **46,** 64-70.

Hartman, J. (1998) *Vetenskapligt tänkande Från kunskapsteori till metodteori,* Studentlitteratur, Lund.

Hirschheim, R. and Klein, H. (2003) Crisis in the IS Field? A Critical Reflection on the State of the Discipline *Journal of the Association for Information Systems,* **4,** 237-293.

Iivari, J. (2000) Information Systems Development as Knowledge Work: The body of systems development process knowledge, In *Information Modelling and Knowledge Bases XI* (Eds, Kawaguchi, E., Hamid, I. A., Jaakkola, H. and Kangassalo, H.) IOS Press.

Iivari, J. and Huisman, M. (2001) The Relationship Between Organisational Culture and the Deployment of Systems Development Methodologies, In *CAiSE* (Eds, Dittrich, K., Geppert, A. and Norrie, M.) Springer, Interlaken, Switzerland.

Iivari, J. and Maansaari, J. (1998) The Usage of development methods: are we stuck to old practices? *Information and Software Technology,* **40,** 501-510.

Jacobson, I., Booch, G. and Rumbaugh, J. (1999) The Unified Process *IEEE Software,* May/June, 96-102.

Keller, C. M. and Dixon Keller, J. (1996) *Cognition and tool use,* Camebridge University Press, Cambridge.

Kruchten, P. (2000) *The Rational Unified Process An Introduction Second Edition*, Adison-Wesley, Reading, Massachusetts.

Madsen, S. and Kautz, K. (2002) Applying System Development Methods in Practice - The RUP Example, In *Information Systems Development (ISD)* (Eds, Grundspenkis, J., Kirikova, M., Wojtkowski, W., Wojtkowski, G., Wrycza, S. and Zupancic, J.) Kluwer Press, Riga.

Middleton, P. (1999) Managing information system development in bureaucracies *Information and Software Technology*, **41,** 473-482.

MITECS (2003) Electronic Encyclopedia, URL http://cognet.mit.edu/MITECS/Front/introduction.html. Accessed 2003-03-31, MIT Press.

Nonaka, I. and Takeuchi, H. (1995) *The knowledge-creating company: How Japanese companies create the dynamics of innovation*, Oxford University Press, New York.

Patton, Q. M. (1990) *Qualitative Evaluation and Research Methods*, SAGE Publications, London.

Ralyté, J. and Rolland, C. (2001) An Assembly Process for Method Engineering, In *CAiSE* (Eds, Dittrich, K., Geppert, A. and Norrie, M.) Springer, Interlaken, Switzerland.

Schönström, M. and Carlsson, S. A. (2003) Methods as Knowledge Enablers in Software Development Organizations, In *The 11th European Conference on Information Systems (ECIS 2003)* (Eds, Ciborra, C., Mercurio, R., De Marco, M., Martinez, M. and Carignani, A.) Naples, Italy.

Suchman, L. A. (1987) *Plans and Situated Action The Problem of Human Machine Communication*, Camebridge University Press, New York.

Tolvanen, J.-P. (1998) Incremental Method Engineering with Modeling Tools Theoretical Principles and Empirical Evidence Doctoral Thesis, Department of Computer Science and Information Systems, University of Jyväskylä

Wastell, D. G. (1999) Learning Dysfunctions in Information Systems Development: Overcoming the Social Defenses with Transitional Objects *MIS Quarterly*, **23,** 581-600.

Westrup, C. (1993) Information systems methodologies in use *Journal of Information Technology*, **8,** 267-275.

Wiig, K. M. (1993) *Knowledge Management Foundations*, Schema Press, Arlington.

Williamson, K. (2002) *Research methods for students, academics and professionals Information management and systems*, Centre for Information Studies, Wagga Wagga.