Summer 10-6-2011

REFLECTIONS ON THE REFLEXIVE ACTIVITIES OF EMBEDDED SYSTEM DEVELOPMENT: A CASE STUDY

Wolfgang Molnar

Follow this and additional works at: http://aisel.aisnet.org/ecis2011

Recommended Citation

http://aisel.aisnet.org/ecis2011/166

This material is brought to you by the European Conference on Information Systems (ECIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in ECIS 2011 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.
REFLECTIONS ON THE REFLEXIVE ACTIVITIES OF EMBEDDED SYSTEM DEVELOPMENT: A CASE STUDY

Molnar, Wolfgang, Centre de recherché Henri Tudor, L-1855 Luxembourg – Kirchberg, Luxembourg, wolfgang.molnar@tudor.lu

Abstract

This paper presents the findings of an in-depth investigation of situated activities of developers and managers during an embedded system development. Previous empirical investigations claim that software design may involve activities outside the continuum of methodologies, such as improvisation or situated activities. Those activities may involve reflexive practices that are purposefully applied by developers and managers. This research provides a better understanding how reflexive behaviors shape the development process by adopting an interpretive approach. The study involved the collection and analysis of qualitative data that were gathered during a 16-month field study carried out in an automotive company. This paper offers insights into everyday actions and practices, the role of reflexivity, and different consciousness levels of reflexive practices of developers and managers. The paper argues that nurturing reflexivity can positively influence information system developments. Furthermore, this paper points out how reflexivity could be encouraged to advance project work and organizational processes.

Keywords: Information System Development, Structuration Theory, Reflexivity.
1 Introduction

This paper analyses the role of reflexivity in information systems development (ISD) in the field of embedded software design. ISD processes may apply structural frameworks and methodologies in an organizational context (Samhurthy & Kirsch, 2000). Despite the availability of various structural frameworks and methods, many researchers claim that systems development may involve activities outside the continuum of methodologies, such as improvisation or situated activities (Bansler & Havn, 2005; Ciborra, 2002; Cunha, 2005; Nandhakumar & Avison, 1999). The reasons of developers and managers for moving away from structured processes are contrasting but involve purposive thoughts, hence ideas that originated through reflection. For instance, stiff methodologies may be seen as discouraging creative and critical thinking in an organization because employees are forced to work according to well-described procedures and rules (e.g. (Singels, Ruèl, & van de Water, 2001)). Furthermore, prescribed approaches seem to provide not enough margins to cope with some of the challenges of ISD in unstable circumstances. Although there are studies available concerning improvisation and situated activities in the context of ISD (e.g. (Ciborra, 2002)), scholarly understanding of improvisation and situated activities as reflexive practices in ISD is relatively thin. Therefore, it is of interest how developers and managers act purposively on the basis of their reflections when they develop information systems, such as embedded software. An empirical investigation of those emerging practices provides a better understanding of reflexive practices in ISD.

Current scholarly understanding of reflexive practices in ISD refers for example to software process improvement (Allison & Merali, 2007). Allison and Merali (2007) describe the role of the individual to shape the improvement process that may be applied by organizations to enhance the quality of their software. Turk et.al. (2005) discuss the assumptions underlying agile software development processes and state that “the agility in agile processes is achieved through continuous self-examination of the processes used and corresponding adaption of the processes” (p. 73). In other words, agility is accomplished through the ongoing monitoring of actions that involves not only the monitoring of oneself but also of others (Giddens, 1984). In addition, the ongoing flow of actions implies the capability of actors to act accordingly in the context of ISD, hence Turk et.al. (2005) state reflexive behaviors are included in agile software development. To get a better understanding about reflexive practices in ISD, this paper examines the activities of developers and managers during embedded software design in an ISO 9001-certified organization. The study mainly focuses on: a) How do the reflexive practices shape the ISD process?; b) What consciousness levels are involved during reflexive activities in the context of ISD?; c) What are the implications for developers and managers in this context?

This paper addresses these questions by examining the day-to-day work, the role of structural frameworks, and reflexive practices of developers and managers. The next section outlines a brief literature review, followed by our research approach. The fourth and fifth sections present our case description and analysis. We end this paper with a discussion of the implications for theory and practice.

2 Literature review

The metaphor of methodologies is the conventional view of researchers concerning ISD (e.g. (Avison & Fitzgerald, 2006; Madsen, Kautz, & Vidgen, 2006)). ISD methodologies and standards help to structuralise the development process of information systems from being a poorly organised pursuit towards a more controlled and stable creation process (Nandhakumar & Avison, 1999). Du Plooy (2002) argues ISD methodologies comprehend the development process as a largely rational process, managed with tools and techniques. The description of those tools and techniques comprise a great part of ISD literature by specifying prescriptive instructions on how to develop information systems (Kautz, Hansen, & Jacobsen, 2004). Avison and Fitzgerald (2006) summarise a number of arguments
for adopting and using ISD methodologies, namely: a) a better end product, b) a better development process, and c) a standardised process. Although methodologies and standards may help to design information systems better, the literature review seems to indicate that in practice the objectives are not always accomplished. For example, Nandhakumar and Avison (1999) explain how an ISD methodology was too mechanistic to be of much use in the daily organization of managers and developers activities. The authors continue, improvisation and situated development activities are involved in the development process, so that ISD methodologies provide only an image of control. Wastell (1996) describe the use of a methodology served as a social justification against involvement in the messy world of practice.

Madsen et.al. (2006) state that available literature emphasize how information systems should be developed, while empirically grounded research concentrate on how information systems are developed. Furthermore, empirical work shows that in practice managers and developers hardly adopt methodologies to its full extent, they rather employ some parts pragmatically (Fitzgerald, Russo, & Stolterman, 2002). Others argue that the development activities are evolving regards the engagement between developers and their methods – what has been referred to as the amethodical system development (Truex, Baskerville, & Travis, 2000), or the method-in-action (Fitzgerald et al., 2002). Those development methods are further investigated and they found that control is an emergent attribute of ISD when development activities are evolving (Madsen et al., 2006). However, this emergence of control is not necessarily a bad thing, since experienced developers acknowledge that customers may not comprehend the full complexity at the beginning of the development process (Walz, Elam, & Curtis, 1993). Sambamurthy and Kirsch (2000) categorize ISD processes in different types in order to author a cumulative framework for a better understanding of ISD. A better understanding is important, because ISD requires attention of different stakeholders and factors throughout the development process. Similarly, Gasson (1999) indicates that interrelated puzzles of ISD are viewed differently by various development team members, so that development capabilities are situated in the context of problem investigation.

Gasson (1999) cites Giddens merely to generalize the findings about a social action model of situated ISD. Giddens’ (1984) Structuration Theory has been widely accepted and used in the realm of information systems research. For example, Wastell (1996) relates the term methodology to ‘ontological security’ and describes the tendency of methodologists to be “cloaked by an illusion of control and omnipotence” (p. 38). Allison and Merali (2007) draws on Structuration Theory to acquire a better understanding on the process of emergent change in software process improvement. Nandhakumar and Avison (1999) use Giddens’ work to see the development process as the unfolding of interactions between activities and context of developers and methodologies. The Structuration Theory has been referenced in 225 information system articles between 1986 and 2002 (Jones & Karsten, 2003). Indeed, quite a few researchers emphasised the importance of Structuration as a theoretical lens through which a variety of information systems questions can be researched (Pozzebon & Pinsonneault, 2005).

This paper refers to the concept of reflexivity, which is an important aspect of the Structuration Theory. The perspective of reflexivity (Giddens, 1984) is used to derive a better understanding of the everyday actions and practices of embedded system developers and managers. This is different to previous studies in the available literature; hence, we seek a better understanding about reflexive practices in ISD. Our emphasis was on the improvisational and situated activities and the role of structured processes while developing an embedded system. Embedded system managers and developers, like other organizational actors, reflexively monitor their everyday actions, those of others and the contexts of social activity. Giddens (1984) argues that reflexivity operates only partly on a discursive level. He describes both practical and discursive consciousness. What individuals know about what they do and why they do it, in other words, their knowledgeability as human agents, is largely embedded in practical consciousness, which they cannot express discursively.
3 Research Approach

The research approach chosen here is an interpretive case study (Orlikowski & Baroudi, 1991; Walsham, 1993), conducted in Engineering Company (EC – a pseudonym) in Germany. Part of this study was the collection of detailed qualitative data related to the context and processes of managers’ and developers’ activities (Avison & Nandhakumar, 1995; Walsham, 1993). In particular, the study was carried out during the process of an embedded system development in the ISO 9001-certified organization. We carried out 31 semi-structured interviews, which lasted approximately 90 minutes each and were held in a relaxed atmosphere (Table 1). In addition, we carried out periodic observations of their software development team for 16 months, followed by informal visits and discussions from February 2006 until June 2007. The aim of this research was to gain a deeper understanding of EC’s embedded system development process with an emphasis on improvisational and situated activities. Participants in this research were assured that their statements would be anonymised and that they would not be explicitly linked to their company or product. Interviews were taped and transcribed, and detailed notes were taken on the observed work practices during each visit.

<table>
<thead>
<tr>
<th>Number of interviews</th>
<th>Position of interviewee</th>
<th>Responsibility of interviewee</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Senior manager, Middle managers</td>
<td>Development, Software, Quality, Service Suppliers</td>
</tr>
<tr>
<td>3</td>
<td>Project manager</td>
<td>Studied alpha</td>
</tr>
<tr>
<td>21</td>
<td>Associates (software and hardware developers)</td>
<td>Studied alpha</td>
</tr>
<tr>
<td>Total 31</td>
<td>Semi-structured interviews at EC</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: EC interviews

Company documents were also examined as a reference to the observed actions and practices of developers. These data sources included field notes from observations, interview transcripts, and company documentation that were filed electronically. They were read and re-read to understand data and determine trends. Miles and Huberman (1994) named the analysis approach applied here: three-tier coding. The first step (descriptive coding) is used to understand the various data. Therefore, meanings were attached to data to bring various events to light. The second step (interpretive coding) is needed to put these meanings into context and clarify their significance. The final step (pattern coding) helps to sort the data and identify patterns by the content of the concept rather than by the source of information. By applying this three-tier coding procedure to the collected data load, we were able to understand the phenomenon (reflexivity of developers and managers doing improvisational and situated activities during an embedded software design) through the value that participants in this research gave to the context and processes. Therefore, we were able to develop a bottom-up conceptualization from the data while using the concept of reflexivity (Giddens, 1984).

4 Case Study

4.1 Research Site Context

The site where this research was conducted is an Engineering Company (EC – a pseudonym). EC operates in the highly competitive automotive sector where automakers and suppliers’ high-tech products are sold worldwide. Its products have a reputation for creativity and customization. The company has seen significant growth since it was founded ten years ago and currently employs more than 170 people. Through this enlargement, the whole business operation underwent significant changes, including the establishment of the process management framework of ISO 9001 seven years ago. This certificate shows that EC implemented a working process management framework within the scope of the entire company. The certificate itself is provided by independent organizations, which evaluate the process management efforts of organizations. In addition to the implementation of
standardized procedures, the hierarchy of EC needed modification. Currently, it is comprised of four layers in the development department: senior management, middle management, project leaders and associates. This research is largely based on the actions of the middle management, project leaders, and associates of the development department.

The director of the hardware and software development was also the senior manager and co-founder of the EC -- Michael. The software development manager was Gabriel (middle management – software), who travelled two days a week from the main location to EC’s second office. Martin, the project leader, was vital for the success of the studied object: the development of a new embedded system (referred to here as alpha). During the initial phase of this development, he spent half of his time as a service supplier with the initiator of the new embedded system and the other half with the developers of this system. As a consequence, he had first-hand contact with customers and developers. In addition, there were several developers (associates) involved in the embedded system development, but Raphael’s role was outstanding. The EC developers described him as a ‘fire-fighter’, and meant by this that he was a person who came to solve different burning technical issues as they arose.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Responsibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Michael</td>
<td>Senior management</td>
</tr>
<tr>
<td></td>
<td>Development, Finance</td>
</tr>
<tr>
<td>Gabriel</td>
<td>Middle-management</td>
</tr>
<tr>
<td></td>
<td>Software</td>
</tr>
<tr>
<td>Martin</td>
<td>Project manager</td>
</tr>
<tr>
<td></td>
<td>Studied product development</td>
</tr>
<tr>
<td>Raphael</td>
<td>Associate</td>
</tr>
<tr>
<td></td>
<td>“Fire-fighter”</td>
</tr>
<tr>
<td>Ten other individuals</td>
<td>Associates</td>
</tr>
<tr>
<td></td>
<td>Software and hardware development, Quality</td>
</tr>
</tbody>
</table>

Table 2: Organizational actors

The development of alpha at EC was the focus of this investigation, and in particular, we studied the processes and context. This system was an electronic device (based on Linux) that used software to connect with the various bus systems of a vehicle or test rack. Its purpose was to record all transmissions on these bus systems and store them locally. Furthermore, it was possible to connect to this device using its client software on a PC for a later readout and analysis of the data. Through its technical innovation and customization, it led to some changes in that market niche.

4.2 The role of structured processes

The EC received an ISO 9001 certificate to demonstrate an established structured process management. Therefore, some organizational processes were based on this framework. According to company documents, the structured processes cover all areas of the company to a certain degree and guarantee standard procedures for EC. However, we observed a different picture of organizational processes, particularly in the focused area of software development. For example, in interviews, developers claimed: “Within a good company, ISO is not necessary!” “Individual motivation and common sense for quality is of greater value than ISO!” “ISO raises bureaucracy but is necessary for the EC products.” All software developers claimed that a strong conformance to their proclaimed standard procedures would have had a negative influence on project alpha. For example, in interviews, developers stated: “We would not have gotten the project done if we had worked according to our ISO standards.” “Frequent changes to the requirements are not part of our standard procedures, so our ISO would not have helped us with project alpha.”

In addition, Martin, the project manager, claimed that, during the development phase, it was necessary to be flexible because of the constant refinement of customer requirements. However, Martin also stated that the maintenance phase of project alpha could have benefited from more standardized procedures throughout the embedded systems development. The quality manager expressed doubt about the effectiveness of ISO procedures with the EC developers. He claimed that he dislikes the working style of some software developers, who produce only poor documentation. However, the quality manager mentioned that the ISO certificate was not ever in danger of being lost, although some
‘unorthodox’ actions and practices by software developers occurred. It is interesting to note the various opinions concerning the reasons that EC established a structured process management framework, and we identified two trends. Higher in the company hierarchy, the opinion was stronger that internal motivation was the major reason for introducing ISO 9001. In contrast, the individuals lower in the company hierarchy claimed that constraints around the EC drove the establishment of the structured process management framework. Our data shows a discrepant image concerning the role of a structured process management framework (ISO 9001) at EC.

4.3 Practiced procedures

We observed various procedures beyond the documented standards of the structured process management framework. As mentioned earlier, the documented processes of the ISO 9001 framework had only a little influence on the actions of the organization’s members, particularly the developers. However, the continuous use of software tools such as Subversion (an open source software version control system) and Bugzilla (an open source bug tracking tool), regular meetings, and improvisational and situated activities by the development team became the natural course of work. Subversion is a database system to file and structure the progress of various software modules, and it helped the software developers throughout the project alpha. One software developer mentioned that it was a valuable tool for addressing revision control because they needed to deliver new versions of the embedded system software as well as the PC client software early and frequently. All stakeholders of the embedded system development used the software tool Bugzilla. As a result, it became de facto standard to report unexpected incidents and errors with the help of that tool. Many software developers claimed that the direction and list of tasks concerning the development of alpha was to a large extent influenced by stakeholders’ reported incidents and errors. Bugzilla includes procedures to process the reports in a standardized way, and the software developers followed these procedures. In addition, Martin introduced a weekly meeting called ‘change-control-board’ for the purpose of discussing changes during the maintenance phase of alpha. Meeting participants were representative stakeholders of alpha from the departments of marketing, project management, quality, hardware development, and software development. Martin mentioned that the growing complexity of project alpha called for more structural efforts in order to manage this project well. After several ‘change-control-board’ meetings it became standard procedure at EC. Lastly, the software developers hacked things together. Software developers stated in interviews that several hacks were part of the software. Gabriel, manager of the software department, stated that the status of tinkered software was becoming more and more “ver-bastelt” (further tinkered). As a consequence, the tinkered software, which was as yet administrable, became more complex.

4.4 Improvisational and situated activities

We observed two different occurrences of improvisational and situated activities during the development of project alpha. First, the developers and managers described a demanding development phase during which the company made a leap in its technology use, from a 16-bit single-processor platform to a 32-bit multi-processor platform. Another important feature realized by Raphael, the ‘fire-fighter’ mentioned previously, and Michael, EC’s co-founder, was a remanufactured protocol. Michael claimed, “The new feature was a vital issue for the breakthrough of our 32-bit platform.”

These technological challenges created some difficulties during development, because of the inability to determine whether an error occurred in software or hardware and the complexity of alpha-reasoned complications. In addition, all developers claimed that the frequent refinement of requirements was anything but helpful. Almost all developers were focused on their individual range of duty with no larger holistic understanding. Only Raphael, Gabriel and Martin claimed to be aware of the entire architecture of the embedded device and its software. This lack of understanding caused difficulties during bug fixing, as developers mentioned.
Second, after the difficulties during the initial phase of project alpha, the focus was on the system development and maintenance. Naturally, the software developers’ day-to-day activities were significantly different from previous actions and practices. The developers learned more and more about the platform, the remanufacture technology, and the needs and desires of customers were channeled in the weekly ‘change-control-board’. Hence, the developers claimed that they had more time for observing potential side effects of changes. Previous software hacks were fixed with sophisticated code and some rudimentary documentation. Finally, the system development and maintenance phase proceeded into bug fixing and changes of alpha because the key hardware and software features were settled, and the development path was dependent upon its previous architectural decisions.

5 Analysis

5.1 Everyday actions and practices

We identified two different themes of everyday actions during the development of an embedded system by the software developers. These themes emerged from the analysis of the case study: a) “muddling through with innovative high tech solutions” and b) “proper code by finding solutions on available circumstances.” We describe the initial phase of the development as a muddling through with innovative high tech solutions because a significant number of difficulties were overcome with improvisational and situated activities. These difficulties (new platform, re-manufactured technology, refinement of requirements) were mastered through appreciation of technical solutions (skills of ‘fire-fighter’, tinkering, etc.) and non-binding negotiations with customers (service supplier task of Martin).

We recognized situated activities in various incidents, such as the introduction of the ‘change-control-board’. It was founded on a whim, when Martin invited every important stakeholder for an impromptu meeting to discuss and decide requests from customers. Previous changes were decided in passing between single stakeholders and Michael, and later arguments about the reasons for that change sometimes created a bad mood. This ad hoc meeting became routine because every stakeholder accepted its necessity for the progress of that embedded system development.

Second, we describe the later phase of the development as writing proper code by finding solutions on available circumstances because many obstacles have been overcome and the foundations of the architecture were specified. However, frequent changes led to the need for a structured way of handling the multiple needs of various customers. The ‘change-control-board’ was an important piece in the structural scaffolding that helped steer everyday actions. In addition, software tools like Bugzilla and Subversion helped to coordinate tasks and manage work output. However, some software developers claimed in interviews that even in the advanced stage of this embedded system development, some hacking and patching were still used. For example, Gabriel stated the tinkered software became more and more tinkered upon, and consequently, the software increased in its complexity. On the other side, developers spent more time on evaluating potential side effects and documented some of their work output to monitor reflexively their doings.

5.2 The role of structural frameworks

As reported in the case study, there was little evidence of an influential structured process management framework (ISO 9001) during the initial phase we observed the embedded system development. However, we were able to determine other structural procedures in use at progressed development phases. Various codes of practices were provided by the ‘change-control-board’ and software tools like Subversion and Bugzilla. The ‘change-control-board’ meetings were held weekly and brought together every important stakeholder to discuss and evaluate possible changes to alpha. This regular meeting was accepted by the organizational members of EC. In addition, software tools like Subversion and Bugzilla called for several processes in order to use these tools correctly. Relevant to those empirical findings about emerging development processes and practices is that Fitzgerald
et.al. (2002) explains that computer-aided software engineering (CASE) tools function as a type of formalised method. The software tools Subversion and Bugzilla were introduced as a result of previous experiences from some developers. The developers realized the necessity for structural tools to manage the software development, and implemented these tools in the organizational IT structure. Through their use by all stakeholders, they became part of the structural framework. For example, Bugzilla was used to file incidents as well as errors, and all stakeholders (e.g. developers, project managers, customers) had access to this database. In particular, software developers and the project manager used this database to have an overview of various tasks. These structural creations were elaborated to support and guide the tasks of the stakeholders and became part of the daily routine as well as organizational culture. However, these informal structural settings were never described in the way that the organizational ISO 9001 processes were. Interestingly, the documented processes enjoyed less attention than the less formal processes (CASE tools, ‘change-control-board’ meetings), and therefore our focus is on those structural creations that are actually practiced. Only these recurring patterns of behavior provided a structural basis for the successful management of EC.

5.3 Reflexive practices of developers and managers

Our analysis revealed a recurring behavior in the particular reflexive practices of Raphael and Martin along with other developers and managers at EC. Raphael claimed in interviews that he was an autodidactic (self-taught) person, willing to contribute to the well-being of project alpha, whereas Martin stated that he assimilated various views from stakeholders in his work. Raphael and Martin paraphrased their broader understanding of the situation around them (project alpha and organizational circumstances), but did not refer to it as reflexive practice because they stated that this broader understanding comes along with their ability to optimize available resources to fix problems. As human actors, developers and managers reflexively monitor their actions, those of others and the context of activity (Giddens, 1984). This reflexive behavior might take place in the discursive or practical consciousness. The potency of conscious reflexivity was significantly higher when questions about their actions were asked. For example, developers and managers came up with a detailed assessment of what worked and what did not work when the founders of EC asked them to answer a questionnaire about the observed actions during the initial development phase. This shows that some reflexive practices may be enhanced by particular managed actions.

The structured process management framework of ISO 9001 used at EC involved the reflexive procedures of organizational members in order to improve EC’s processes. Martin and some of the particular project managers were pinpointed as the main contributors for improvement. Nevertheless, these contributors routinely acted upon the processes in which they participated without further thinking. Consequently, activities were often unconsciously reflexive or taken-for-granted and did not pass through ‘consciousness.’ Although the developers were not normally consciously aware of reflexivity, they became aware of them through conscious reflection (strong potency of reflexivity) or when some form of disruption to routine occurred.

6 Discussion and implications

The analysis disclosed a variety of everyday activities, noteworthiness of EC’s informal structural procedures, and reflexive practices of individuals during project alpha as well as reflexive procedures in the process management framework of ISO 9001. This discussion shed some light of the role of reflexivity in embedded software design. The case study revealed that EC has not used methods and standards before EC became ISO 9001-certified five years ago. However, developers and managers maintained some control through various software tools like Bugzilla and Subversion. Hence, their purposefulness work style enabled them to cope with various constraints throughout previous development projects without having any process management framework. The introduction and establishment of the structured process management framework indicated that organisational managers were not confident enough concerning the prevalent work style. Developers and managers justified the
introduction of ISO 9001 with the need to demonstrate an established structured process management to potential customers. In addition, some managers indicated the likelihood to improve structures within EC for further prosperity. Nevertheless, the availability of new structures with the establishment of ISO 9001 did not stop developers and managers to maintain a continuing theoretical understanding of their activities. In many cases that involved the continuous use of their previous development tools and practices in order to design determined as they have gathered positive experience in previous projects. Although many developers seemed to cope with the various constraints during the initial phase of project alpha by muddling through, the project manager realized the growing complexity that required an alternative structural approach. Therefore, Martin established ‘change-control-board’ meetings that became standard procedure at EC. As the analysis highlighted, during the later phase of the development, everyday actions and practices involved proper solutions and less muddling through. Notwithstanding the importance of these solutions for EC’s procedural foundation, it may lead to unintended consequences altogether with other activities. In addition, new situations may challenge the current status quo, so that purposeful developers and managers need to cope with a different approach.

A discussion about the role of reflexivity in ISD involves an examination of the monitoring character of the ongoing flow of social life (Giddens, 1984). The case study and analysis describes repeated activities by the managers and developers that are similar to the stratification model of the agent by Giddens (1984). Giddens states the reflexive monitoring of activities is part of daily action and includes the behaviour not just of the individual but also of the ongoing flow of social life. We would paraphrase this as the monitoring of the agents’ “Umwelt” (German - environment). Individuals maintain a continuous ‘theoretical understanding’ of the reasons of their actions and practices, so that they act purposefully. The analysis and the previous paragraph indicated when the developers and managers complied with the given circumstances, when e.g. they participated in a work environment without explicit process management framework. Despite the establishment of new structures (becoming an ISO 9001-certified organisation), managers and developers continued what they were doing through purposive actions and practices. They kept improvising in situated circumstances, because they were able to overcome any difficulties through those activities previously. That confidence in their abilities to cope with troubles was founded in the maintenance of a continuing ‘theoretical understanding’ why they were doing this. Giddens (1984) differentiates the reflexive monitoring of action and rationalisation of action from its motivation. Furthermore, Giddens states that motivation relates to the potential for action. The case study and analysis reveals two different occasions when actions were linked to the wants of certain organisational members. First, EC became an ISO 9001-certified organisation and this was triggered by the potential ability to improve structures within EC. Second, the project leader established ‘change-control-board’ meetings, so that he could cope more adequately with the complexity. Therefore, purposive organisational members seek to derive their motivations by good intend, so that the organisation or their project may prosper. Figure 1 depicts the repeated activities (reflexive monitoring, rationalisation of action, motivation of action) and states the potential of unacknowledged conditions of actions and the unintended side effects that make activities unique in situated circumstances or what we call “Umwelt-influences.” In difference to the stratification model (Giddens, 1984), continuous loops symbolizes reflexive monitoring and rationalisation of activities. Those activities and the motivation of action are influenced by Umwelt-influences, such as unstated conditions and unplanned consequences. The motivation of action is not a continuous feature of daily activities and Giddens (1984) states that ‘motives tend to have a direct purchase on action only in relatively unusual circumstances, situations which in some way break with the routine’ (page 6).
The evidence suggests that the role of reflexive practices was important for developers and managers in their daily activities. We claim that formal structured processes are practiced mainly on the practical consciousness level, because processes are characteristically simply done and can be followed blindly. Formalized methods, such as CASE tools (Fitzgerald et al., 2002) and standardized meetings (“change-control-board”) were routinely followed, without fuss. However, this does not exclude the ability to reflect upon one’s own behavior as well as that of group members. Those formal structured processes are part of day-to-day life and relates to routines that are grounded in practical consciousness (Giddens, 1984). In addition, those routines supported the confidence within the developers and managers that what they are doing is proper for the development. Giddens (1984) describes this confidence as ontological security. Furthermore, we claim that improvisation and motives of actions entail reflexivity that is practiced mainly on the discursive consciousness level. The ability to improvise with strong Umwelt-influences, or the motives for action require a sophisticated understanding of the conditions. The case study reveals, that improvisation (tinkered solutions) was prevalent during the initial stage of the project and that developers and managers were able to describe what were they doing. This ability to continually reflect upon work results was a central behavior of developers and managers. In addition, the project manager explained the situation when he introduced ‘change-control-board’ meetings. Moving away from routinely behavior means that the results of the monitoring as part of the ongoing everyday actions reveal an odd situation. Giddens (1984) claims that to be a human being is to be a purposive agent who both has reasons for his or her activities and is able, if asked, to elaborate discursively upon those reasons, including lying about them. He further states “human agents always know what they are doing on the level of discursive consciousness under some description” (p. 26). Therefore, improvisation and motives of action are practiced mainly on the discursive consciousness level. Figure 2 depicts the different consciousness levels of reflexivity that we were able to identify during this case study.

These findings indicate that organizational structured processes are not fixed but may be influenced by the additional dynamics of the ongoing flow of organizational life. This is not necessarily a disadvantage because purposive agents may act to obtain better outcomes. We imply that it is worthwhile to improve reflexivity in system developments processes because it may provide better control and stability for these processes. Giddens (1984) states that reflexive monitoring of action depends upon rationalization, understood here as a process rather than a state, and as inherently
involved in the competence of agents. Furthermore, competence means maintaining a continuing ‘theoretical understanding’ of the grounds of their activity (Giddens, 1984). So, nurturing reflexivity requires a higher proficiency in the theoretical understanding of the grounds of their activities in the context of embedded system development. Therefore, these findings imply for practice that awareness about insights into the complexity of development processes is important and should be further nourished. For example, participation of software designers in other issues of development such as project management or hardware architecture, questionnaires and assessments in open-minded environments, and the feasibility to put into practice purposive thoughts of employees may encourage them to think about their actions and practices, improving confidence in their abilities. The case study evinced that questionnaires at EC encouraged developers and managers to reflexively monitor their embedded system development and to state what worked and what did not work (cf. section 4.3). Routines of frequent reflexivity would provide the feeling of security that is convenient and comforting for organizational members (Giddens, 1984). Furthermore, less trust in the systems seems to provoke a higher degree of structured processes and less situated actions. Michael stated: “We needed to make our processes more structured, because I had lost confidence in the way that some issues were handled by the developers and project managers.”

We claim that purposive developers and managers allow the maturation of situated activities in the setting they can influence to a certain extent. These situated activities increase its appropriateness so that the purposive employee may act more advantageous. If activities have reached a trustworthy degree, the same may be repeated. For example, the case study and analysis indicated that the ‘change-control-board’ meeting and tinkering software as a coping strategy were applied regularly. After continuous repetition of similar actions, they became part of the cultural sediments of that organization. Through ongoing repetition, they lost their improvisational and situated character and became part of the routine of developers and managers. Giddens (1984) states that routines are basic elements of day-to-day activities. Routines sustain trust or ontological security in the daily activities of social life (Giddens, 1984). Therefore, established procedures may be distributed within that context and shared processes may become sedimented as organizational structured processes, although they started as situated activities. For practitioners, these findings indicate that situated activities may be helpful to overcome difficulties during embedded system development.

7 Conclusions

For this study, we investigated the activities during an embedded system development. We tried to obtain a better understanding how reflective practices shape the ISD process. Managers and developers comprehend their environment through continuous reflexive monitoring and rationalisation of activities, which are part of daily actions and practices. Those activities are shaped by Umwelt-influences (cf. section 6) throughout the development and include unacknowledged conditions and unintended consequences. Those Umwelt-influences may have an impact on the motives of actions, which are not a continuing ingredient of daily activities. A further differentiation between the various consciousness levels provides a better understanding of the reflexive activities by developers and managers. This paper claims, that formal structured processes (e.g. the use of CASE tools and regular meetings) are practiced mainly on the practical consciousness level, this means that actions are simply done. Improvisational activities and motives of actions encompass reflexivity that is practiced mainly on the discursive consciousness level. Hence, developers and managers are able to comment on what they are doing. These new insights on the reflexive activities of ISD are valuable, because it raises some implications for developers and managers in this context. Giddens (1984) states that reflexive behaviours depend on the competence of agents. For that reason, better reflexive capabilities involve higher proficiency in the ability to understand the reasons of their activities. Moreover, strengthening the reflexive capabilities through routines can lead to a feeling of security concerning the appropriateness of their deliverables. This paper provides fresh insights on the reflexive activities of ISD, which could be further expanded on additional research by investigating case studies regarding companies of different sizes and complexity.
References


