INFORMATION SYSTEMS DEVELOPMENT AS A SOCIAL PROCESS: A STRUCTURATIONAL MODEL

Completed Research Paper

Christoph Rosenkranz
Goethe University
Frankfurt am Main, Germany
rosenkranz@wiwi.uni-frankfurt.de

Abstract

Prior research has shown that social interactions are important in order to understand the phenomena involved in information systems development. However, most traditional research largely ignores these issues. DeSanctis and Poole (1994) made an important contribution to the study of social dynamics in information systems research with their Adaptive Structuration Theory (AST). Although the concepts have found broad acceptance for the study of information technology (IT) uses and effects, AST has not been widely used for studying the process of designing IT artifacts and developing information systems. In this paper we transfer AST to studying information systems development as a social process. We build on Markus and Silver's (2008) redefinition of AST's core concepts 'structural features' and 'spirit' as technical objects, functional affordances, and symbolic expressions, and we extend them with relational concepts for agents and activities that we derive from social construction of technology (SCOT) studies. The result is an AST-based model that describes the information systems development process. We illustrate and discuss how researchers might use these concepts to generate hypotheses in studies of information systems development processes.

Keywords: Information system development, Information system development process, Structuration Theory, Adaptive Structuration Theory, Social construction of technology

Introduction

Information systems (IS) research and related fields provide a large body of knowledge on different aspects of IS development (Hirschheim et al. 1995; Iivari et al. 2004), and researchers have investigated the involved phenomena over many decades by adopting different perspectives (e. g., Chae and Poole 2005; Gallivan and Keil 2003; Hirschheim et al. 1995; Joshi et al. 2007; Kautz and Nielsen 2004; Ko et al. 2005; Levina and Vaast 2005; Sambamurthy and Kirsch 2000; Vidgen and Wang 2009; Xia and Lee 2005). The research necessity remains obvious, however, as various studies still point to the problems arising in IS development (e. g., Agrawal and Chari 2007; Avison and Fitzgerald 2003; Hansen 2008; Nelson 2007). Despite different circumstances, the basic message has persisted over the last four decades: the challenge of developing IS still exists today (Berry 2004; Boehm and Basili 2000; Fraser and Mancl 2008; Jarke et al. 2009; Kautz et al. 2007).

IS are often developed in the form of structured approaches or projects (Hirschheim et al. 1995, p. 33). These projects are inherently complex because they must deal with not only technological issues but also organizational factors that by and large are outside of the project team's control (Xia and Lee 2005). IS are *socio-technical systems* in a specific organizational context, which includes both technical and organizational sub-systems (Bostrom and Heinen 1977). An IS "is not the information technology alone, but the system that emerges from the mutually transformational interactions between the information technology and the organization" (Lee 2004, p. 11). The IS is a result of an information technology (IT) enabling an organization, as much as the IS is the result of an organization enabling an IT (Lee 2004, p. 12). Therefore IS development is often characterized by multiple stakeholders and multiple influences, many of which relate to pre-existing IT systems, organizational or cultural elements, and social structures that have evolved in organizations over decades (Chae and Poole 2005).

Consistently, a central position is the understanding of IS development as a *social process* that is carried out in an organizational setting and involves users, systems analysts, developers, and other stakeholders (Hirschheim et al. 1995; Newman and Robey 1992; Newman and Robey 1996; Truex et al. 1999). For example, a coherent and meaningful model is created during IS development by the consolidation of different stakeholders' perspectives and multiple requirements in an organization (Alvarez and Urla 2002; Curtis et al. 1988; Holmqvist 1989). Stakeholders are confronted with many uncertainties and ambiguities, which can lead to failure of the whole process (Cule et al. 2000). This is contrary to viewing IS development as akin to a consciously planned, rational design by developers and managers (e. g., Chan et al. 1997; Teo and King 1997). Consequently, the major problems of IS development are not so much technological as sociological in nature (DeMarco and Lister 1987, p. 4; Hirschheim et al. 1995, pp. 1-3).

However, the IS development process "has received relative little research attention in prior literature" (Siau et al. 2010, p. 88). Not much research goes beyond IS development methods, and there is a general paucity regarding theory and studies of organizational and social processes in IS development (Kautz et al. 2007). Few theories exist that help to understand and explain the "structuring of the IS during development" (Chae and Poole 2005, p. 21). To remedy this deficit, we propose that the use of Adaptive Structuration Theory as a comprehensive framework for theorizing (Bostrom et al. 2009) could help us to get a deeper understanding of the social phenomena involved in IS development. We suggest that the application of Adaptive Structuration Theory to study the development of IS as a social process yields important insights for research and practice. Our goal in this paper is to examine concepts of Adaptive Structuration Theory for use in research on the social processes involved in IS development. The research question we ask is this: *How can we conceptualize IS development in a way that helps us to hypothesize about, and investigate, the social process of developing IT artifacts?*

The remainder of the paper is structured as follows. We first describe related work in the domain of IS development. Next, we describe Adaptive Structuration Theory, especially recent developments in theorizing concepts about the relationship between users and IT artifacts. We examine these concepts in light of another research stream in which social processes and technology play important roles: social construction of technology (SCOT) studies. From this analysis, we derive a structurational model of the IS development process. For illustrational purposes, we present an application of the model. In the concluding section of the paper, we discuss the findings and implications, and we outline how our model may be used in future research studies.

Related Work and Theoretical Background

Information Systems Development

In practice, proposals for developing IS range from sequential methodologies (Royce 1970) to cyclic, iterative approaches (Boehm 1988). However, the nature of IS development is in many aspects intangible (Cule et al. 2000, p. 65), making the coordination of activities during the IS development process difficult (Kraut and Streeter 1995). Furthermore, early studies already point to the importance of human and social factors in IS development (e. g., Bartol and Martin 1982; Laughery and Laughery 1985; White 1984).

During the last decade, agile approaches and new management concepts have complemented the iterative approach (Beck and Andres 2004; Beck et al. 2001; Martin 1991; Poppendieck and Poppendieck 2003; Schwaber 1995). The resulting agile IS development methodologies (Cao et al. 2009; Vidgen and Wang 2009) appear to incorporate most of the lessons learned about IS development during the past (Cockburn and Highsmith 2001; Highsmith and Cockburn 2001): they trade strict control for more flexibility and autonomy, the overall development process is not planned and scheduled upfront, and progress is made in small iterative phases, while encouraging constant change, frequent interaction, and informal face-to-face communication. Planning becomes a permanent task, and leadership is established via collaboration, while team leadership is separated from project lead. Many new challenges have also arisen in the meantime such as 'Internet-speed' application development (Baskerville and Pries-Heje 2004; Baskerville et al. 2003; Slaughter et al. 2006), distributed development teams (Ramesh et al. 2006), or free/libre open source software (Crowston and Wade 2010).

However, research lags behind practice in understanding these phenomena, although various, often disparate concepts and research frameworks have been proposed (e. g., Chakraborty et al. 2010; Conboy 2009; Kappos and Rivard 2008; Kautz 2004; Levina 2005; Levina and Vaast 2005). For example, communication between stakeholders (Gallivan and Keil 2003; He et al. 2007), the creation of shared understanding (Tan 1994), or successful knowledge transfer (Joshi et al. 2007; Ko et al. 2005; Williams 2010) are deemed to be major drivers for IS development success. Nevertheless, the IS development process is often treated as a "black box" (Siau et al. 2010, p. 92), and the details and the importance of the involved *social interactions* for IS development are still not well understood (Sawyer et al. 2010). More than 25 years after IS research has first focused on IS development as a social process (e. g., Barley 1986; Guinan and Bostrom 1986; Hirschheim et al. 1991; Ives and Olson 1984; Newman and Robey 1992; Orlikowski 1992; Robey and Markus 1984), researchers still voice "a need for theory and studies about social behavior and processes of communication, negotiation, and learning and their relation to the broader (historical, political and social) context" (Kautz et al. 2007, p. 235).

Structuration Theory and Information Systems Research

Giddens' (1984) Structuration Theory is one of the most used social theories in IS research (Jones et al. 2004). Recent reviews have remarked Structuration Theory's importance for IS research and have called for further development and use (Jones and Karsten 2008; Poole and DeSanctis 2004). Therefore, it also provides a good starting point for investigating IS development as a social process.

Structuration Theory is a theory of social action, which claims that society should be understood in terms of *action* and *structure*; a duality rather than two separate entities (Jones et al. 2004; Jones and Karsten 2008; Poole and DeSanctis 2004). This insight has influenced important theoretical proposals in IS research (e. g., DeSanctis and Poole 1994; Orlikowski 1992; Walsham and Han 1991).[1]

For example, Orlikowski's (1992) notion of the 'duality of technology' put emphasis on the human agency in developing and using IT as structures in organizations. Subsequent work extended and applied this model especially to the organizational use of different kinds of IT and the effects of IT use (Jones et al. 2004, pp. 317-319). Although Orlikowski and Robey (1991) also proposed to focus attention on the social processes through which technologies are developed, this call has led to few studies (e. g., Mathiassen 1998; Newman and Robey 1992; Robey 1995).

[1] The difference between these approaches are the subject of vigorous, sometimes strident debate (e. g., Jones and Karsten 2009; Poole 2009).

Moreover, while Orlikowski (1992) acknowledged the importance of technology's material properties (Jones et al. 2004, p. 319), her recent contributions argue for the notion of "technologies-in-practice" (Orlikowski 2000, p. 407), and criticize structurational perspectives for their "ontology of separateness" (Orlikowski 2010, p. 134), that is, that they treat technology and humans as essentially different and separate realities. However, if we adopt a purely sociomaterialist view (Orlikowski and Scott 2008), technical objects have no inherent properties, boundaries, or meanings. This then raises the question of where we draw the line between things for analytical purposes or in practice (Faulkner et al. 2010, p. 9). According to Orlikowski (2010, p. 135), the answers to these questions are 'entanglement in practice' and 'agential cuts' (Barad 2003; Barad 2007) that are performed and temporarily stabilized through human practices.

However, a recent paper by Markus and Silver (2008) criticizes that such proposals undermine the basic assumption that IT itself can play a causal role. We acknowledge IS development as a social process, "not as the necessary result of a powerful technological infrastructure, or as principally reflecting the interpretations and interactions of the human developers or users" (Orlikowski 2010, p. 136). Nevertheless, for us, the only consequence is that it is never clear a priori and independent of context whether an observed phenomenon should be treated as technical or social (Bijker 2010, p. 67). Therefore we share Markus and Silver's (2008) reservations and their call for other concepts that refer to relations between IT artifacts and human agents (cf. also the arguments by Pinch 2010, p. 87; Poole and DeSanctis 2004, p. 211).

Adaptive Structuration Theory and Markus and Silver's (2008) Extension

Another structurational approach that has been very influential in IS research over the recent years is *Adaptive Structuration Theory* (AST). AST was originally developed by DeSanctis and Poole (1994) to study the behavior of groups and organizations using IT for their activities. AST explains the interplay between technology, social structures, and human action, and is a holistic attempt to examine the use and the impacts of advanced technologies in organizations. It describes an IS as a socio-technical system in which each element may impact others, and the nature of interaction may change over time (Bostrom et al. 2009, p. 30). AST focuses on social structures, which are defined as rules and resources that are provided by technologies and institutions as the basis for human activity. On the one hand, social structures serve as templates for planning and accomplishing tasks; on the other hand they are reproduced and altered through human interaction (DeSanctis and Poole 1994; Poole and DeSanctis 2004). IT artifacts are real in the sense that they have embedded material features and components, such as the hardware or the software. Then again, IT artifacts also are socially constructed in the sense that new structures emerge in human action as human agents interact with IT artifacts. Therefore human agents using IT for their work vary widely in their perceptions about the role and utility of IT artifacts, and how they can be used.

Initially, DeSanctis and Poole (1994) considered social structures (rules and resources as the basis for human behavior) embedded in technology in form of the concepts of 'structural features' and 'spirit'. Structural features are said to bring meaning and control to group interaction. For a group support system, for example, these might include voting algorithms and anonymous recording of ideas. The spirit of a structural feature set is described as its underlying general intent with regard to values and goals. Both concepts serve as a source for social structure and influence the way people use IT. However, these definitions are highly controversial as the concepts of structural feature and spirit are conceptualized as properties of an IT artifact, although such values are fundamentally attributed to human agents (Jones and Karsten 2008; Poole 2009). Markus and Silver (2008) tried to address the above concerns by redefining the core concepts of structural features and spirit for IT effects studies. Building on DeSanctis and Poole's (1994) concepts and combining them with other theories, Markus and Silver (2008) propose three concepts to describe IT artifacts and their connection with human agents: *technical objects, functional affordances*, and *symbolic expressions* (cf. detailed description of the first three entries in Table 1). The concept of technical objects pertains to the IT artifacts themselves; the functional affordances and symbolic expressions concepts refer to relations between technical objects and users. The two last concepts are bridging concepts between IT artifacts and users' use and interpretation of IT artifacts.

At the heart of Markus and Silver's (2008) conceptualization is the common assumption of structurational perspectives that IT artifacts have as their kernel a dual nature (Houkes and Meijers 2006; Kroes 2010). IT artifacts, as objects intentionally produced by human beings, are neither simply physical objects nor intentional/social objects (Kroes 2010, p. 61). They are different from physical (natural) objects in that their function is a defining feature of what kind of objects they are. They are different from social ob-

jects (e. g., money) because these perform their function on the basis of collective intentionality (Searle 1995), whereas IT artifacts perform their function on the basis of their physical structures. This means that the conception of IT artifacts involves three key notions (Kroes 2010, p. 56), namely the notion of a physical structure (the technical object), of a (technical) function, and of a context of intentional human action. The actual uses of an IT artifact are dependent on functional affordances and symbolic expressions for a specific user in a specific context of intentional human action.

Use versus Engineering and Design

Two specific contexts of intentional human action are of particular interest for IS research, namely the *user context* and the *engineering design context* (Kroes 2010, p. 56). In our case, this relates to the use of an IT artifact and the development and design of an IT artifact respectively. As with most structurational perspectives in the IS field, researchers using AST have studied the effects of using existing IT artifacts in the organization. This has helped to increase our understanding of phenomena in relation to the effects of existing IT in organizations, and how IT and organization shape each other. We agree with Markus and Silver (2008) that there is a conceptual gap in AST between IT artifacts and human agents' interpretations of them, which can be closed using relational concepts linking IT artifacts with agents or groups. However, there is also another gap both in AST and in Markus and Silver's (2008) extension: although AST has been used to describe the activities and processes of developers as users of IT (Sawyer et al. 1997), the processes of *designing* new IT artifacts and *developing* IS have scarcely been in the focus of attention (e. g., Chae and Poole 2005; Rose and Scheepers 2001). Likewise, Markus's and Silver's (2008) statements relate only to the user context – the effects of IT artifacts on their users. In the engineering design context, the focus is on inventing/constructing a physical structure that may realize a function by providing functional affordances to users as intended or unintended by designers (Kroes 2010, p. 56).

AST represents an interesting theoretical lens on socio-technical systems and provides a strong meta-framework for reasoning and theorizing (Bostrom et al. 2009, p. 36); it is a *meta-theory* that provides a lens to understand, investigate, and predict outcomes of IT-induced change in a socio-technical work system (Bostrom et al. 2009, p. 37). We suggest that applying AST for studying IS development as a social process can add value because it allows us to draw on the same theory that is used for studying effects of IT use for also studying IT design. Design and development lead to the creation of IT artifacts, and the use of IT artifacts leads to effects. For example, how are the social aspects of the IS development process related to the use process and its effects? Using the same theoretical foundation for studying both IT use processes and IS development processes tightens the connection between the two. As Giddens (1991) himself has noted, structurational concepts should be regarded as "sensitizing devices", "being relevant to anyone writing about very broad questions of social organization and transformation" (p. 213). To do so, we have to extend AST and the explanations of Markus and Silver (2008) because they do not focus on IS development or the process of designing IT artifacts so far.

The Social Construction of Technology

How to characterize material objects and artifacts is also a central challenge for researchers in constructivist studies of technology as exemplified in the seminal works of Bijker (Bijker 2001; Bijker 2010; Bijker et al. 1987; Pinch and Bijker 1984). Whereas Markus and Silver (2008) ask "what are *effects* of IT artifacts?", researchers applying *social construction of technology* (SCOT) perspectives trace the process "how to *make* IT artifacts" (Bijker 2010, p. 63). SCOT studies range from having realist ontologies of technology to radical constructivist ontologies. The former are relativistic only in the sense of methodology: "This is the central idea of the methodological relativism that SCOT advocates: do not assume any a priori preference for one relevant social group over another" (Bijker 2010, p. 68). That is, the methodology is to follow the social processes and empirically find out what makes up well-working, successful, and winning technologies. The focus is more on understanding the process than on describing the product (Bijker 2010, p. 68).[2]

Pinch and Bijker's (1984) key argument is that artifacts are socially constructed by social groups that give different meanings to technical objects. All members of a certain social group share similar meanings at-

[2] Employing SCOT with a realist ontology and a relativistic methodology is certainly compatible with the critical realist view of causality that is proposed by Markus and Silver (2008).

tached to a specific artifact, and social groups may disagree over definitions of what constitute 'success' or 'failure' of an artifact to solve a problem (Howcroft et al. 2004, pp. 338-340). An artifact can stabilize and reach 'closure' when relevant social groups see their problems as having been solved, and as different social groups may define problems differently, they also have different opinions about achievement of closure (Howcroft et al. 2004, p. 339). Therefore, development involves constant negotiation of different social groups with alternative readings of the artifact.

Bijker (2010) proposes to follow a three-step research process to analyze the development of artifacts: (i) sociological deconstruction of an artifact to demonstrate its interpretative flexibility; (ii) description of the artifact's social construction; and (iii) explanation of this construction process in terms of the technological frames of relevant social groups (Bijker 2010, p. 69). The central concept here is *technological frame*. A technological frame structures the interactions among the members of a relevant social group, and shapes their thinking and acting in interaction 'around' an artifact. The technological frames located between human agents build up as interaction around an artifact also builds up (Howcroft et al. 2004, p. 340). Elements of a technological frame of software developers, for example, could comprise organizational goals and constraints, development methodologies, or testing procedures (Howcroft et al. 2004, p. 340). Typically, a person will be included in more than one social group and thus also in more than one technological frame. Researchers need to identify relevant social groups and their characteristics, for a specific artifact "is described through the eyes of relevant social groups. Social groups are relevant for describing an artefact when they attribute explicitly a meaning to that artefact" (Bijker 2010, p. 69). The description of an artifact through the eyes of different social groups produces different descriptions. In this methodological – and not ontological – sense, there is not one artifact, but many (Bijker 2010, p. 68).

SCOT and its methodological relativism resonate well with our intention to provide a model of the IS development process within the framework of AST. It also has a history with the other structurational approaches in IS research (e. g., Orlikowski (1992) applied Pinch and Bijker (1984) to characterize interpretive flexibility and relevant actors, and Orlikowski and Gash (1994) employed technological frames as an analytic perspective). In the following section, we build on Markus and Silver's (2008) extension and adapt it with ideas from SCOT to provide a model for the study of IS development as a social process.

A Structurational Model of Information Systems Development

In adapting SCOT's three-step research process for use with AST to understand and explain the IS development process, we have to develop concepts that help us to (i) sociologically *deconstruct* IT artifacts, (ii) *describe* the construction process of the IT artifacts, and (iii) *explain* the construction process in terms of the technological frames of relevant social groups. The first step in the model building is to map the concepts of AST as our meta-theory to the phenomenon of interest (Bostrom et al. 2009, p. 34). We use Markus and Silver's (2008) extension of AST as a starting point. We adapt their concepts to the IS development process, we detail them with ideas from SCOT, and we propose three new concepts for research on the IS development process: *designers*, *design affordances*, and *development activities*.

Figure 1 summarizes our AST-based process model. In the following sections, we explain the model's components (the concepts) by following SCOT's three-step research process. This includes a (somewhat minimalistic) running example that refers to and describes the components and the IS development process in a typical situation: the development of an 'individual application system' to support business processes within an organization (a custom-built human resource management system).

Concepts for the Sociological Deconstruction of IT Artifacts

Users and Designers

Markus and Silver (2008) noted that "no explanation of IT effects would be complete without careful conceptualizations of users and use environments" (p. 620). Naturally, any explanation of the IS development process also cannot be successful without such conceptualizations. However, we argue that it is also necessary to distinguish users of IT artifacts from designers of IT artifacts. Our concepts of *users* and *designers* are based on the distinction of the user context and the engineering design context as two specific contexts of intentional human action (Kroes 2010, p. 56). In a fundamental sense, designers and users are performing complementary intentional human actions that shape the dual nature of IT artifacts: without a

user context, the engineering design context would be meaningless; without an engineering design context, the user context would not come to exist in the first place. (Note that other relevant social groups that are neither users nor designers can be important for IS development and in shaping IT-induced organizational change. For example, powerful human agents that have to give sponsorship are central concerning organizational constraints or politics – that is, to the technological frames of users and designers.)

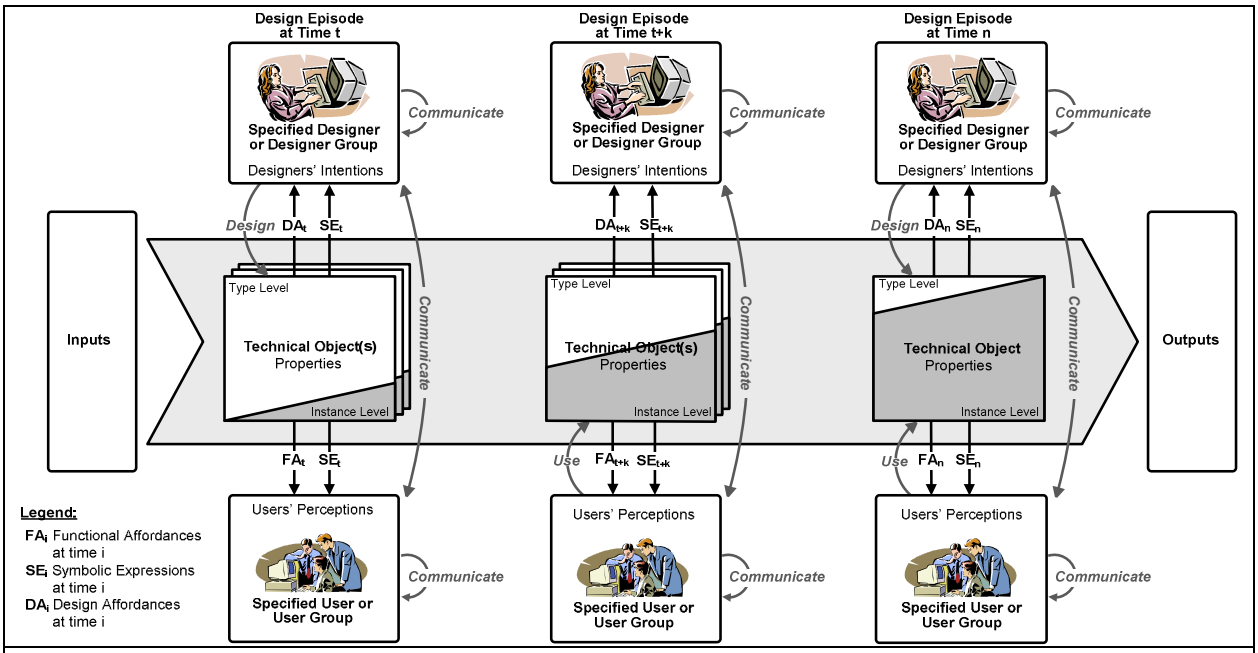

Figure 1. AST-based Structurational Model of the IS Development Process

Users use technical objects, in conjunction with their individual capabilities, to realize goals (Kroes 2010, p. 52). Therefore, in the user context, the function of a technical object is primarily related to the ends of human actions; what matters here is how the functional affordances and symbolic expressions of a technical object for a specified user may contribute to realizing those ends (Kroes 2010, p. 56). *Designers* invent/construct, using their capabilities, a physical structure or technical object that may realize (intended and unintended) functional affordances for users. Therefore, in traditional studies of the IS development process, the functional affordances of technical objects (as perceived by the designer) and the designers' intentions have often moved to the foreground, and the symbolic expressions have moved to the background. In contrast, we acknowledge that designers can often pursue their own goals and agendas when constructing technical objects.

A *user group* or *designer group* is a social group. For both users and designers, we have to identify relevant social groups and sub-groups. Relevant social groups can be identified by looking for persons who mention an IT artifact in the same way (Bijker 2010, p. 68). For example, in the case of an 'ordinary' desktop personal computer, there may be the "unreliable apparatus" (through the eyes of 'IT-resistant' users) and there may be the "dependable gadget" (through the eyes of 'IT-affine' users). As such, a social group does not necessarily have to correspond to stereotypical roles such as 'software engineers' or 'testers'. An individual person may be member of several such groups depending on her or his technological frames. This is also reflected in the "boundary spanners" and "brokers" (e. g., Levina and Vaast 2005; Pawlowski and Robey 2004) examined in the IS development literature.

Examples for user and designer groups: in case of the human resources management system, user groups may include, for example, end-users in different locations or business units, 'ordinary' users or 'power' users, client/customer representatives, or testers. Exemplary designer groups might be product owners, IT architects, project management committee members, project managers, (senior) developers and programmers, or consultants. Users and designers can be both full time employees and contractors, and they might be collocated or distributed.

Technical Objects

Technical objects are IT artifacts and their component parts (Markus and Silver 2008, p. 620). These form the technical system, for there is nothing self-evident about the delineation of even a 'singular' IT artifact: is a personal computer an IT artifact or is it a technical system comprising technical objects such as hard drive, microchips, monitor, cables, and input devices (Bijker 2010, p. 66)? Technical objects are made by humans and they are outcomes of intentional design and manufacturing processes. They are real things, material or abstract (Faulkner and Runde 2009; 2010), with properties that may have causal potential (Markus and Silver 2008, p. 621). These properties may be intended or not; technologies "do not merely assist in everyday lives, they are also powerful forces acting to reshape human activities and their meanings" (Bijker 2010, p. 67). Moreover, existing technical objects limit and enable IS development: through the material constraints and directions inherent in present systems and through the experiences from previous systems that shape designers' approaches to building new systems (Chae and Poole 2005, p. 20).

Markus and Silver (2008) also include interface components and outputs of IS such as documents and other boundary objects. We extend the concept of technical objects to also include the *design input* of IT artifacts. These objects are important in the IS development process. This includes not only hardware or software components, but also requirements specification documents, data and process models, diagrams, program code and print-outs of code examples, all of which are instances of the "design boundary objects" discussed in the IS development literature. Design boundary objects are "any representational artefact that enables knowledge about a designed system, its design process, or its environment to be transferred between social worlds and that simultaneously facilitates the alignment of stakeholder interests populating these social worlds by reducing design knowledge gaps" (Bergman et al. 2007, p. 551).

We also distinguish technical objects at *type level* and at *instance level* from each other. Abstract types are schemas in the sense of 'class' versus 'instantiation': a particular object of the described type is an instance of that type. Technical objects at type level describe and represent the design of technical objects at instance level. For example, a requirements specification document is a technical object at type level that describes aspects of the design of a technical object at instance level. Technical objects at type level are often used for communicating *about* other technical objects, whereas technical objects at instance level are used *in* the work. For example, designers construct technical objects at type level as schemas that try to communicate the designers' intentions for technical objects at instance level to other designers.

Examples for technical objects: the 'final' technical object to-be-developed in case of the human resources management system may comprise several hardware and software components. These form a technical system. Typical design inputs in 'traditional' collocated projects may include requirements specification documents on the type level, project and test plans, UML diagrams, mock-ups, or whiteboard drawings, or sticky notes on a task board, code and prototypes, or a burndown chart on the wall in 'agile' projects. Technical objects that are used during development may include networked computer systems with various software tools such as source code management tools (e. g., CVS), computer-aided software engineering (CASE) tools, or project management software. In distributed situations when teams are not in the same room they may use a number of collaboration tools (e. g, conferencing tools or instant messengers).

Functional Affordances

Functional affordances in the user context identify what the user may be able to do with the object, given the user's capabilities and goals (Markus and Silver 2008, p. 622). In that they are "potentially necessary (but not necessary and sufficient) conditions for 'appropriation moves' (IT uses)" (Markus and Silver 2008, p. 625). Functional affordances refer to the *potential uses* of a technical object (Markus and Silver 2008, p. 622), thereby recognizing how the properties and the materiality of a technical object favor, shape, invite, or constrain a set of specific uses (Zammuto et al. 2007, p. 752). Functional affordances of a technical object for users (or user group) can be perceived and interpreted differently by other users or designers in relation to their technological frames. Applying this interpretive flexibility allows to understand a seemingly unambiguous technical object by identifying users and designers different interpretations of 'working' or 'nonworking', which are not intrinsic properties of IT artifacts (Howcroft et al. 2004, p. 340). We need to pay attention to what a technical object "lets users do, to what it does not let them do, and to the workarounds that users develop to address the latter" (Leonardi and Barley 2010, p. 35).

Examples for functional affordances: in case of the human resources management system, a developer might design and implement a feature that 'supports printing of payroll checks'. This function is a functional affordance for the users *as perceived by the designer*. However, since the only secure payroll printer is located in a building that is not easily accessible and checks must be manually signed, this function is 'nonworking' from the point of view of 'ordinary' end-users in the human resources department. Hence, it may not be a functional affordance *as perceived by the 'ordinary' end-users*. Instead, the end-users might fall back to cash or electronic funds transfer for payment that employees receive.

Design Affordances

From the previous discussion also follows that the concepts of functional affordances and symbolic expressions are not sufficient to describe relationships between a technical object and human agents in the IS development process. In the same way, in the engineering design context, we define the concept of *design affordances* as the possibilities for goal-oriented action afforded by technical objects to a specified designer (or designer group). In that they are potentially necessary (but not necessary and sufficient) conditions for IT designs. Design affordances can also be interpreted (perceived) by designers and by users and in relation to designers' technological frames.

Examples for design affordances: for example, the Java programming language offers other design affordances than the C/C++ programming languages. Properties of Java are different from properties of C/C++, such as the former having multi-platform capabilities and the latter requiring specific platforms. So the design affordances of Java and C/C++ can also be different for different designer groups.

Symbolic Expressions

Originally, Markus and Silver (2008) proposed the concept of symbolic expressions as a relational concept bridging IT artifacts and how users may interpret them. For example, symbolic expressions include 'messages' that help users interact with IT artifacts, or messages pertaining to designers' or users' goals and values (Markus and Silver 2008, p. 623). Symbolic expressions can also refer to expressions about functionality. Such expressions may be erroneous, and functional and values-oriented symbolic expressions may be in conflict with each other (Markus and Silver 2008, p. 623). Moreover, an IT artifact may have many different symbolic expressions for a specified user group, just as it may have many functional affordances. With respect to our extensions so far, we also include the engineering design context so that symbolic expressions also exist as relations between a technical object and a designer (or designer group).

Examples for symbolic expressions: in case of the human resources management system, the technical system may be seen by 'power' users as "a great way to make my work easier and more efficient". In contrast, flexibility of features and many options may lead 'ordinary' users to view it as "too complicated to be of much use" or as "a repressive mechanism that forces me to do things in a specific way".

Concepts for Describing the Construction Process of IT Artifacts

The IS development process is, in a fundamental sense, exactly that – a *process* or "sequence of episodes, punctuated by encounters, that follows patterns established in previous development work"; it is "a dynamic, social process that is simultaneously constrained by past experience and capable of constructing new patterns of interaction" (Newman and Robey 1992, p. 250). Thus, IS development can be *described* as an input-process-output model (cf. Figure 1). Many of the input conditions and outcome variables can be culled from the comprehensive list of success dimensions provided by Siau et al. (2010). The process creates and re-configures socio-technical elements and their relationships within and between: (1) signs and symbols deployed, (2) tasks, structures, and processes, and (3) its technological core (Lyytinen and Newman 2008).

The process is structured by *activities* (Giddens 1984, p. 3) of users and designers, and revolves around their interaction with each other and with technical objects. The technological frames of members of relevant social groups structure these interactions during IS development. Most prior research views the IS development process as a black box (Siau et al. 2010, p. 92). We want to open this box and therefore propose three *development activities* that fundamentally shape the interaction of the IS development process: *use activities*, *design activities*, and *communication activities*.

Use Activity

The activity of *use* describes that and how a specified user (or user group) uses a technical object. It is similar to DeSanctis and Poole's (1994) concept of 'appropriation moves'. Whereas functional affordances refer to potential uses, appropriation moves refer to *actual uses* of an IT artifact (Markus and Silver 2008, p. 622). Appropriations, the use of features provided by IT, are immediate visible actions that evidence deeper structuration processes and therefore instantiate structures (DeSanctis and Poole 1994, p. 128). However, observing an actual use is just that – a user (or user group) uses a technical object in whatsoever ways. It does not give any details on the type of appropriation. Uses are not automatically determined by IT designs. Rather, people actively select how technology structures are used, and therefore the use practices vary across different users. For example, users may choose to directly use the technical object, to relate it to other technical objects, or to constrain it as it is used (DeSanctis and Poole 1994, p. 129).

Examples for use activity: in case of the human resources management system, a direct appropriation of a prototype of the to-be-developed human resources management system involves simply using some feature of the technical system. For example, 'power' users may test and use the 'supports printing of payroll checks' feature (cf. Section "Functional Affordances") and view it as 'working'. In contrast, 'ordinary' users might reject appropriation of the 'nonworking' feature by ignoring it. A relational appropriation move melds the technical system's use with another technical object, as when 'ordinary' users also use spreadsheets to run calculations and combine those with the use of the human resources management system.

Design Activity

Design as an activity describes that a specified designer (or designer group) constructs a new technical object or configures existing technical objects as components of a new technical object. It includes inventing/constructing a material or abstract structure that may realize some functional affordances (Kroes 2010, p. 56). The activity of design is fundamentally different from the activity of use. The basic argument "revolves around a crucial difference between the notions of creating (that is, designing and making) and using technical artefacts" (Kroes 2010, p. 58). The creation of a technical object, whether in the sense of a new instance of an already existing type or the first instance of a new type, has ontological significance; new objects are set into the world. In contrast, for example, when somebody uses a screwdriver to drive screws, no new technical object is created. Similarly, in situations of creative use of already existing technical objects (e. g., using a knife as a screwdriver), a user attributes new functional affordances to already existing objects (Kroes 2010, p. 58). This is a limiting case of the creation of technical objects, for example, when a software package is customized; no new physical objects are created (Kroes 2010, p. 58).

Users are usually involved in IS development processes by using technical objects, for example, using or testing prototypes. Users can also be involved in the design. However, for analytical purposes, we argue that a user that also participates in a design activity of an IT artifact is becoming a designer at that time. Similarly, we regard a designer that simply uses an existing IT artifact as a user.

Examples for design activity: in case of the human resources management system, when using a traditional, plan-driven development methodology, analysts may create functional specifications and diagrams for the to-be-developed human resources management system. Developers then take these as design inputs from which they develop hardware and software components, for example, the 'supports printing of payroll checks' feature (cf. Section "Functional Affordances"). Test users' role is to find and report defects and usability issues, which are then fixed by designers. Most of the test users are simply using the application and are plain beta testers. When using an agile development methodology, analysts may create no formal documentation up-front, designers may frequently produce 'working' technical systems in fast-paced release cycles, and users may continually test these releases.

Communication Activity

Markus and Silver (2008, p. 624) concluded that examining the diversity of symbolic expressions is useful in explaining users' interpretations of, and interactions with, IT artifacts. However, they limited their concept of symbolic expressions to technical objects and the 'messages' that they send as 'signs' to users. Sure enough, users and designers interact more directly in IS development processes, engaging into *communication*, a relational activity between both groups (Guinan and Bostrom 1986; Te'eni 2001). Communica-

tion between involved participants is identified as a relevant part of the IS development process because it is necessary for knowledge transfer (Levina and Vaast 2005; Robillard 1999). Communication is not only knowledge transfer, of course, but also includes dispute, negotiation, persuasion, and so forth. Constant negotiation, ongoing renegotiations, and conflicts between social groups with competing interests and visions about what the technology should do drive the development of technical objects (Leonardi and Barley 2010, p. 38). Therefore we regard communication activities as crucial in IS development. Even the most reclusive programmer has to spend some time communicating with colleagues.

In this context, requirements engineering often plays a central role in IS development (Ambriola and Gervasi 2006; Chakraborty et al. 2010; Galliers and Swan 2000; Hofmann and Lehner 2001), as it deals with the elicitation, analysis, documentation, and validation of stakeholder requirements that are to be met by software-intensive systems (Abran et al. 2004). For example, data models are examples of type level technical objects that are created in design activities by designers *after* requirements have been elicited from (and negotiated with) future users in *communication activities* such as workshops or face-to-face conversations. (Note that 'messages' between technical objects and human agents are symbolic expressions. However, agents can use technical objects as a medium or channel for communication. For example, a written document can be part of a communication activity between human agents, but it can also be a technical object, of which the symbolic expressions may be interpreted.)

The activity of *communication* is composed of two primary processes whereby information is transferred from a sender to a receiver, conveyance of information and convergence on meaning (Miranda and Saunders 2003). All work requiring more than one individual is composed of different combinations of these two fundamental processes (Dennis et al. 2008, p. 576). A sender/receiver can only be a human agent, that is, a user or designer. The information sent by the sender is interpreted in sense-making processes by the receiver (Weick et al. 2005). Structures such as technical objects are linked to communication by *interpretative schemes* of human agents (Giddens 1984, p. 29). Interpretative schemes are "standardized stocks of knowledge, applied by actors in the production of interaction" (Giddens 1979, p. 83). Stocks of knowledge are words, phrases, or images that are known in common by the actors and are available to them for use in language games to make meaning (Boland 1996, p. 692). Consequently, all processes of the structuration of systems of social interaction involve the communication of meaning as a key element (Giddens 1984, p. 29). Social encounters are sustained above all through talk (Giddens 1984, p. 73), through everyday conversation using language (Giddens 1984, pp. 264-265).

Examples for communication activity: in case of the human resources management system, when using a traditional, plan-driven development methodology, analysts may elicit requirements up-front in workshops with user 'representatives'. Designers later transfer these into a technical system. Some 'power' users may then test the technical system and communicate to the designers that the 'supports printing of payroll checks' feature (cf. Section "Functional Affordances") is 'working'. When using an agile development methodology, developers constantly communicate with user 'representatives'; business people and developers ideally work together daily in face-to-face conversation. 'Ordinary' users may be earlier involved in testing prototypes of the technical system, and thus may communicate to designers that the 'supports printing of payroll checks' feature is 'nonworking' from their point of view.

Summary and Recap

Table 1 summarizes the conceptual extensions that we made in order to provide a structurational account and description of the IS development process and compares them to AST's and Markus and Silver's (2008) original concepts. The concepts can be used to *deconstruct* (step i) and *describe* (step ii) the IS development process. The IS emerges from the process of social construction and the degree of stabilization increases up to the moment of closure (Bijker 2010, pp. 68-69; Howcroft et al. 2004, p. 339).

In the next step, the described IS development process can be analyzed and *explained* by interpreting it in a broader theoretical framework (step iii). AST and our model provide an overarching perspective, where IT artifacts are included in the structures of an organization. Thereby it provides a framework into which other theories can be fitted to make contextually appropriate predictions about system behavior (Bostrom et al. 2009, p. 39). To illustrate how our AST-based model enables the creation of specific hypotheses, we present an example: the development of a research model for communication in IS development processes and its relation to factors such as different IS development methodologies.

Table 1. Summary of Key Concepts

Concept	Definitions and Explanations	Mapping to AST	Extensions
Technical objects (Markus and Silver 2008, p. 625)	Technical objects are IT artifacts and their component parts. They are understood as material and abstract things of which the properties are potentially causal, that is, necessary conditions for people to perceive them and use them in particular ways with particular consequences. Technical objects do not depend on people's perception of them to exist and their properties do not necessarily correspond to users' perception. As artifacts people make them but their properties are not always intentional and their properties do not necessarily correspond to designers' intentions.	Conceptualized by Markus and Silver (2008); related to the concept of structural features (DeSanctis and Poole 1994).	Technical objects also include the design input of IT artifacts. Technical objects are distinguished from each other at type level and at instance level.
Functional affordances (Markus and Silver 2008, p. 625)	Functional affordances are the possibilities for goal-oriented action that are afforded by technical objects to a specified user group. Therefore functional affordances are relations between technical objects and users that identify what the user may be able to do with the object, given the user's capabilities and goals. Functional affordances refer to potential uses of IT artifacts and the concept focuses on issues related to technical functionality. They are potentially necessary (but not necessary and sufficient) conditions for IT uses and the consequence of IT use.	Conceptualized by Markus and Silver (2008); related to the concept of structural features (DeSanctis and Poole 1994).	Functional affordances of a technical object for users (or user groups) can be perceived and interpreted by the users or designers.
Design affordances	Design affordances are the possibilities for goal-oriented action afforded by technical objects to a specified designer (or designer group). Therefore design affordances are relations between technical objects and designers that identify what the designer may be able to do with the technical object, given the designer's capabilities and goals.	Related to the concept of structural features (DeSanctis and Poole 1994); not conceptualized by Markus and Silver (2008)	Counterpart of functional affordances based on the distinction of the user context and the engineering design context.
Symbolic expressions (Markus and Silver 2008, p. 625)	Symbolic expressions are the communicative possibilities of technical objects for a specified user group. Symbolic expressions are also relations between technical objects and users. The concept focuses on issues related to the interpretation of technical objects by users. They are potentially necessary (but not necessary and sufficient) conditions for user interpretations of IT and the consequences resulting from those interpretations.	Conceptualized by Markus and Silver (2008); related to the concept of spirit (DeSanctis and Poole 1994).	Symbolic expressions also are relations between a technical object and a specified designer (or designer group).
User (user group)	Users use technical objects in conjunction with their individual capabilities to realize goals according to their individual needs, problems, and preferences. Different user groups may be characterized by different aspects (experience with technical object, frequency of use, mode of operation, and so forth).	Related to the concept of group members/actors (DeSanctis and Poole 1994); not explicitly conceptualized by Markus and Silver (2008).	Based on the distinction of the user context and the engineering design context as two specific contexts of intentional human action in IS development.
Designer (designer group)	Designers invent/construct, using their capabilities, a technical object that may realize (intended and unintended) functional affordances. Different designer groups may be characterized by different aspects (experience with problems requiring a technical object, communication skills, mode of operation, and so forth).		
Communication activity	Communication describes the process whereby information is transferred from a sender to a receiver according to their interpretative schemes, using language and alternative forms. Communication is contextual and is subject to the situation, environment, culture, and so forth in which it is embedded. (Not considered is the other side of a more complete explanation of IS development, in which users and designers may interpret each others' activities).	Not explicitly conceptualized by DeSanctis and Poole (1994) or by Markus and Silver (2008).	Based on the importance of signification and interpretative schemes for social interaction in IS development.
Use activity	Use describes that a specified user (or user group) uses a technical object. Users may make (a) direct use of a technical object, (b) relate the technical object to other structures, or (c) constrain the technical objects as they are used.	Related to the concept of appropriation moves (DeSanctis and Poole 1994); not conceptualized by Markus and Silver (2008).	Based on the distinction of the user context and the engineering design context as two specific contexts of intentional human action in IS development.
Design activity	Design describes that a specified designer (or designer group) constructs a new technical object or configures existing technical objects as components of a new technical object.	Not explicitly conceptualized by DeSanctis and Poole (1994) or by Markus and Silver (2008).	

An Illustration and Application Example of the Model

It is a commonplace in IS development research that a "communication gap" (Alter 2001, p. 46; Ryan 1993, p. 240) exists between users and designers, and it "has repeatedly been observed that business and IT professionals 'speak different languages' and apply differential yardsticks for desired outcomes" (Hansen and Lyytinen 2010, p. 4). While the existence of the communication gap seems to be a truism, it is astounding that studies investigating this phenomenon are scarce. Despite early calls for investigating the relationship between project performance and communicator competence, identifying effective communication patterns and behaviors used by successful developers, and investigating the relationships between project performance, external behaviors, and internal processes (Guinan and Bostrom 1986), there are almost no studies that have explored this gap empirically or systematically. Only little IS development research investigates social interactions or communication of participants in detail (e. g., Guinan and Scudder 1989; Sawyer and Guinan 1998; Sawyer et al. 2010; Watts Sussman and Guinan 1999).

So even as many authors stress the importance of effective communication in the IS development process, few provide empirical evidence or concrete suggestions (Bostrom 1989, p. 280). This statement is still valid today; communication is named one of the persistent and salient problems of IS development (Kautz et al. 2007; Siau et al. 2010). For example, what *is* the amount and frequency of communication in agile teams compared to traditional teams? Does a certain kind of communicative behavior explain or predict project performance or project success? Few studies have examined communication and its relation to overall project performance and quality (Sawyer et al. 2010), or compared communication in projects using different IS development methodologies (Pikkarainen et al. 2008). Consequently, a systematic and insightful understanding of communication in IS development is still missing.

Structuring Communication in the Information Systems Development Process

Figure 2 derives from our model and illustrates a *communication activity* in IS development in more detail based on the sender-receiver model (Guinan and Bostrom 1986; Hesse et al. 2008; Te'eni 2001).

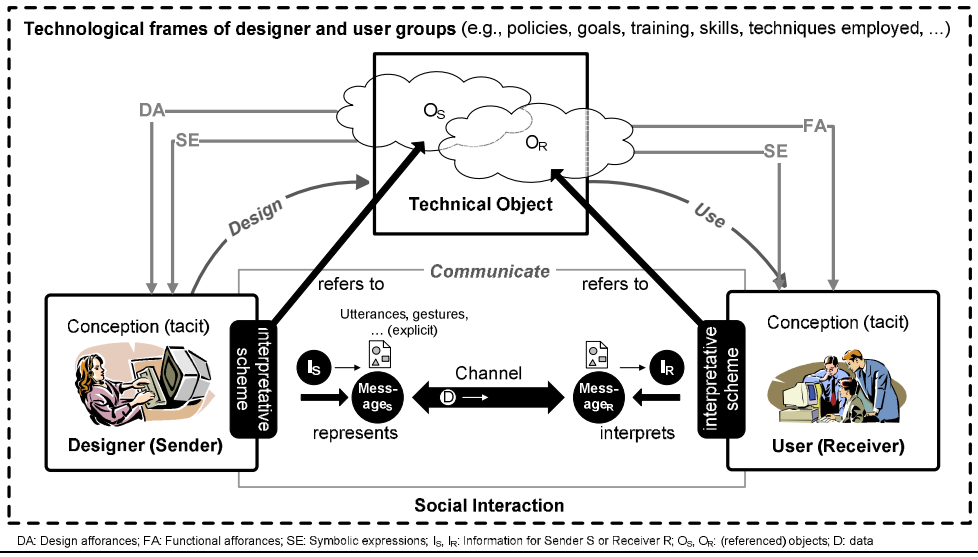

Figure 2. Communication Activity in IS Development (adapted from Hesse et al. 2008)

On the left side, whenever a sender passes a representation in form of a message to a receiver, s/he chooses a representation – for example, an (explicit) utterance in natural language, or a document, or a conceptual model – according to her or his interpretative scheme and (tacit) conception, and doing so, s/he presumes an (initially subjective) domain reference. On the other side, the receiver interprets the received message according to her or his interpretative scheme in sense-making processes and creates her or his own conception of it, which again leads to an (initially subjective) domain reference. The technological frames of the relevant social groups add the context to the communication activities (and all other activi-

ties in social interaction) of sender and receiver and their interpretative schemes. It can only be clarified in an adjoining feedback process by additional communication loops whether or not an interpretation by a receiver corresponds with the one intended by the sender.

Consequently, linguistic alignment in interactions between sender and receiver plays a critical role in achieving successful communication (Branigan et al. 2010; Clark 1996; Deacon 1997; Tomasello 2008). Without linguistic alignment, interpretative schemes are not aligned and the meaning of a message may be different for sender and receiver (Weick 1979, p. 4): "Individual groups of humans develop their own unique ways of symbolizing and doing things – and these can be very different from the ways of other groups, even those living quite nearby" (Tomasello et al. 2005, p. 721). Only if people establish and share the conventions making the syntax, semantics, and pragmatics of signs they form a *language community* (Kamlah and Lorenzen 1984, pp. 45-47). To really align the meaning of signs, the signs need to be learned *empractically*, that is, people have to experience what the meaning of a sign in specific situations really is through practical use in the given situation (Bühler 1990, pp. 176-179; Kamlah and Lorenzen 1984, p. 36).

Without communication and empractical learning, it may be hard for participants themselves to detect significant differences in functional affordances and symbolic expressions of different user groups and designer groups: "Things can only function as signs for members of shared culture or *language communities*" (Markus and Silver 2008, p. 623, emphasis added). For example, groups of people in the same organization carry out different functions, and these groups develop different sub-languages or jargons on the basis of their professional backgrounds and the nature and organization of their functions (Weber and Camerer 2003); therefore IS development is also influenced by these languages (Holmqvist 1989, p. 73).

Examples for language communities and empractical learning: in case of the human resources management system, the use of the word "Oracle" in a communication activity may mean, depending on the domain reference, 'the name of a company that sells software' or 'a specific type of relational database that is sold by the company of the same name and that is used within the current project'. Test users may have learned both meanings in their testing and previous communication activities with developers. However, both meanings may be completely unknown (and thus incomprehensible) to most end-users, whereas some end-users may know the first meaning and may not understand its usage when test users, users, and developers discuss features of the database (and, accordingly, functional affordances). Only test users and developers form a language community.

The Effect of Different Information Systems Development Methodologies

Most IS development methodologies supposedly aim to facilitate the communication among different participants and stakeholders. For example, Rational Unified Process and various other development methodologies are often stated to be created just for this purpose (Kroll and Kruchten 2003, pp. 145-149; Kruchten 2004, pp. 5, 92). The majority of traditional development methodologies, both sequential or iterative, are plan-driven and rely on *formal* communication such as specification documents to control communication among development participants (Black et al. 2009; Boehm and Turner 2004; Kraut and Streeter 1995). But in rapidly changing environments, it is hard for formal mechanisms of communication such as specification documents to react quickly enough (Herbsleb and Mockus 2003): "Rather than being bastions of order in an uncertain world, traditional teams may indeed become chaotic should their plan-driven organization be overwhelmed by events" (Vidgen and Wang 2009, p. 374).

In contrast, many agile development methodologies such as Extreme Programming (Beck and Andres 2004) or Scrum (Schwaber 1995) suggest that business people and developers must work together daily and project information should be shared through *informal*, face-to-face conversation rather than through documentation. This goes along with principles such as iterative cycles, early delivery of working software, and simplicity as defined in the Agile Manifesto (Beck et al. 2001). Agile development methodologies focus on intensive communication and quick change, involving constant feedback (Chow and Cao 2008; Conboy 2009; Vidgen and Wang 2009). However, 'over-communication' between the team and customers and between team members may also become an inhibitor (Vidgen and Wang 2009). Even though agile practices have positive effects, communication hurdles are still present in extended environments where many stakeholder groups and development teams are involved in the same IS development process (Pikkarainen et al. 2008). Agile practices may not always be 'best' (Lee and Xia 2010); purely in-

formal communication may not be effective when dealing with a large number of stakeholders and vast amounts of information (Cao et al. 2009).

Different Technological Frames in Social Interactions

An IS development methodology is an element of the technological frame that structures the interaction among and between user groups and designer groups around technical objects. It is one aspect (of many) of structure that influences and shapes the IS development process, and is perceived and established differently at different contextual levels (Kautz et al. 2007, p. 233). Different IS development methodologies structure the interaction differently, leading to different technological frames and differences in use activities, design activities, and communication activities of designer groups and user groups.

The practices or methodologies-in-practice observed in IS development projects may deviate from the methods recommended in textbooks, and developers are reported to combine elements of different methods and tools based on prior experience and intuition (Baskerville et al. 2003, p. 73; Kautz et al. 2007, p. 223). On the one end of the spectrum, a formal IS development methodology can become a "fetish" (Wastell 1996, p. 34), on the other hand, a methodology-in-practice can instead be emergent and "amethodical" (Truex et al. 2000, p. 54). For example, we classify an IS development methodology-in-practice as agile if it applies practices that predominantly follow agile principles.

Examples for elements of technological frames for social groups given by agile and plan-driven development methodologies-in-practice: when using an iterative, plan-driven development methodology, all participants typically follow an overall plan and rely on formal communication. Requirements usually are elicited by analysts in communication with user 'representatives' and specified in documents ('big design up-front'). These specifications may then be read and transferred by developers into a 'prototype' technical system, which may then be tested by test users, who document issues. Developers fix reported issues and this cycle is repeated iteratively until the 'final' technical system is released. When an agile development methodology is used, change is typically core to the approach. Instead of following a detailed plan, participants may engage into a daily 'planning game'. Requirements usually are of small granularity and are gathered continuously. Developers frequently and iteratively deliver a 'working' technical system that is then tested by users. Users and developers interact closely and may communicate together daily in face-to-face conversation. While an agile approach emphasizes face-to-face communication and running systems over written documents, participants may adapt this and employ some planning and documenting.

Consequences for Information Systems Development

We can now analyze communication activities in IS development processes with regard to different IS development methodologies and address questions such as "What is the amount and frequency of communication in teams using agile approaches compared to teams using traditional approaches?" or "What is the relation between communicative behavior and project performance?" The goal is to *explain* the process of IS development, and the interactions of humans and technology within it, in a broader theoretical framework (step iii). Moreover, we can focus on developing explanatory hypotheses with regard to these differences and their effects on communication activities in IS development. This may include adequately explaining the micro-social dynamics during IS development while simultaneously considering the macro-social processes (Leonardi and Barley 2010, p. 41). For example, using our AST-based model we can argue that, given similar complexity, then IS development processes in projects applying different IS development methodologies will show differences in communication activities between designers and users and will have different project performance.

Similarly, we can generate hypotheses on other concepts (e. g., design activities, use activities, functional affordances, or symbolic expressions). In the same way, other theories could be used to expand the model to generate hypotheses on other dimensions of social interaction (e. g., legitimation, domination, or culture). These theories can be combined and tested in the coherent AST-based model (Bostrom et al. 2009, p. 36). The use of specific theories makes the model falsifiable because expected outcomes and process results are specified and alternatives can be identified. This may include both variance theory and process theory, each being a form of representation that focuses on different questions and provides a different understanding of the IS development process (Van de Ven and Poole 2005).

Examples for hypotheses: the differences in IS development methodologies-in-practice lead to differences in the technological frames and in the development activities. These differences provide for the creation of falsifiable hypotheses that allow description, explanation, or prediction (this is by no means exhaustive):

H1a. *The overall amount and frequency of informal (formal) communication activities in an agile team is higher (lower) compared to a traditional team.*

H1b. *Use of agile practices increases (decreases) the amount and frequency of informal (formal) communication activities between users and designers.*

H2a. *Increases in the amount and frequency of informal (formal) communication activities between users and designers lead to an earlier (later) detection of differences between users and designers in functional affordances and symbolic expressions for technical objects.*

H2b. *Earlier detection of differences between users and designers in functional affordances and symbolic expressions for technical objects leads to more rapid and more accurate requirements and identification of problems.*

H3. *Use of agile practices increases the number of users and user groups (i. e., more sub-teams) involved in the IS development process.*

H4a. *Increases in the number and diversity of user groups and designer groups lead to increases in the number of languages communities and more differences between users and designers in functional affordances and symbolic expressions for technical objects.*

H4b. *Increases in the number of language communities involved in informal communication activities lead to 'over'-communication to resolve differences in interpretative schemes.*

Discussion

Our goal in using AST and detailing its concepts is to develop a foundation for the study of phenomena in IS development processes. Several studies have found the same technology implemented in highly similar organizational settings to be associated with very different consequences for structure and process (Barley 1986; DeSanctis and Poole 1994; Markus and Robey 1988). Transferred to IS development, the same technology implemented in different organizational settings may be associated with different IS development processes. We need more research on social interactions and the consequences of both material and social influences. This paper has described IS development as a social process. We provide a structurational, AST-based model as a theoretical framework for IS development research. The developed model integrates concepts of AST, extensions of AST, and SCOT to describe the human agents, the technological objects, their relations, and the activities within IS development. We propose to use the same theoretical foundation that is often applied in studies of organization-technology interaction and IT effects for research on IS development. We suggest that our concepts can be of great help in the attempt to trace and analyze the processes of both material (technical object-human) and social (human-human) interaction during the IS development process. The model was exemplary applied to illustrate how it can be used for giving form to, for analyzing, and for explaining IS development processes.

We primarily consider our concepts as a lens for researchers looking for a suitable means for analyzing the material and social interactions in IS development processes. Using the model provides us a lens to compare settings with regard to generalization, explanation, and prediction. Why does an IS development social process follow this way, rather than that? The model can be detailed and complemented with other, context-specific theories to derive falsifiable and testable hypotheses. For example, a specific IS development process in time as an instance of an "organizing process" (Van de Ven and Poole 2005, p. 1380) could be analyzed using different explanatory devices. It could be investigated as a teleological process, with the IS as a socio-technical system or "organizational entity" that moves toward "some final goal or state of 'rest' (however temporary)" (Van de Ven 1992, p. 178). Or it could be examined as a dialectic process, with the assumption that the IS exists in a pluralistic world of colliding forces, with multiple conflicting goals or teleologies, which compete with each other for domination (Van de Ven 1992, p. 178). Researchers could decompose the entity using our AST-based model to examine its components (users or user groups, designers or designer groups, and technical objects) and their interactions (Van de Ven and Poole 1995, p. 522).

The model offers several advantages to researchers and practitioners interested in the social dynamics of IS development. First, the model offers a form for carrying out description, explanation, and prediction. The different components of the model can be compared within and between settings and cases. Hypotheses derived from the application of the model can be applied to data of real IS development processes in different organizational settings. Second, our model complements existing models of IS development success (e. g., Lee and Xia 2010; Siau et al. 2010; Xia and Lee 2005) and existing approaches investigating the IS development process (e. g., Benbya and McKelvey 2006; Chae and Poole 2005; Falkenberg et al. 1998; Kautz 2004; Lyytinen and Newman 2008; Maruping et al. 2009; Newman and Robey 1992; Vidgen and Wang 2009). For example, our model is similar to existing frameworks that examine and describe user-analyst relationships (Chakraborty et al. 2010; Newman and Robey 1992). However, these models have thus far examined social interaction in isolation and have neglected the influence of technology. Likewise, constructs such as user participation (Iivari et al. 2010; Markus and Mao 2004) could be investigated more broadly in relation to specific IS development processes and different contexts, as described by our model. We provide a more holistic view of how human agents and technical objects interact in the IS development process, which extends existing approaches and places them within the AST framework.

Third, the model is capable of guiding both research and practice of IS development. If a solid and substantial understanding of IS development as a social process is among the desired goals, researchers and practitioners may benefit from our insights on how to conceptualize the relationship between users, designers, and technical objects. For example, the agile development literature so far has been largely anecdotal and prescriptive, lacking empirical evidence and theoretical foundation to support agile principles (Lee and Xia 2010). Our model could be a first building block for providing the missing "theoretical glue" (Conboy 2009, p. 330) and for a succeeding empirical investigation. Our model may also help both researchers and practitioners to better understand the diversity of IS development projects or the relationship between structure and IS development practice (Kautz et al. 2007). For example, observing and tracing use activities, design activities, and communication activities could allow us to explore how complexity and diversity are reflected in and dealt with by project members, both on an individual and on the group level. Similarly, the observation of functional affordances and symbolic expressions could allow us to explore how formal and social structures in IS development projects are perceived and established by and between individual stakeholders and groups.

We do not claim that our concepts are the only valid ones for use to examine IS development. We acknowledge other areas dealing with the dynamics of social interaction as important. Pragmatically, there may be several perspectives for examining social interaction in IS development processes. In terms of Giddens' (1984) dimensions of the duality of structure, our application example only deals with signification, interpretative schemes, and communication, and ignore other dimensions such as domination or legitimation. Other factors and conditions such as users' capabilities, characteristics and goals, their institutional contexts, power, or culture may also play key roles in causal explanations (e. g., Iivari and Iivari 2011; Robey and Markus 1984). We agree with Markus and Silver (2008, p. 627) that the continual emergence of new technologies inevitably requires ongoing conceptual development. Our extensions of AST are intended to allow researchers to develop specific hypotheses and measurement instruments for investigating the IS development process. We encourage others to comment on and challenge our conceptualization. In this way, we hope that we may advance the development of theory in IS research.

Conclusion

The lack of theories about IT artifacts, the ways in which they are developed, emerge and evolve over time, is still a key unresolved issue (Orlikowski and Iacono 2001). DeSanctis and Poole (1994) made important contributions to IS research with their insightful adaptation of Structuration Theory. Although their concepts have found broad acceptance in the IS research community for the study of IT uses and effects, the concepts have not been widely used for understanding the IS development process. In this paper, we tried to transfer AST and Markus and Silver's (2008) extensions of AST to the study of IS development as a social process, and by illustrating and discussing how IS researchers might use these concepts in IS development studies. Our research also comes together nicely with Jones and Karsten's (2008) call for more attention on the "interaction between technology and human action" (p. 150). We hope that our transfer of the concepts will be useful for other IS development researchers and practitioners.

References

Abran, A., Moore, J.W., Bourque, P., and Dupuis, R. (eds.). 2004. *Guide to the Software Engineering Body of Knowledge (SWEBOK®)*. Los Alamitos, CA, USA et al.: IEEE Computer Society.

Agrawal, M., and Chari, K. 2007. "Software Effort, Quality, and Cycle Time: A Study of CMM Level 5 Projects," *IEEE Transactions on Software Engineering* (33:3), pp. 145-156.

Alter, S. 2001. "Which Life Cycle - - - Information System, Work System, or Software?," *Communications of the AIS* (1:17).

Alvarez, R., and Urla, J. 2002. "Tell Me a Good Story: Using Narrative Analysis to Examine Information Requirements Interviews during an ERP Implementation," *The DATA BASE for Advances in Information Systems* (33:1), pp. 38-52.

Ambriola, V., and Gervasi, V. 2006. "On the Systematic Analysis of Natural Language Requirements with CIRCE," *Automated Software Engineering* (13), pp. 107-167.

Avison, D.E., and Fitzgerald, G. 2003. "Where now for development methodologies?," *Communications of the ACM* (46:1), pp. 78-82.

Barad, K. 2003. "Posthumanist performativity: toward an understanding of how matter comes to matter," *Signs* (28:3), pp. 801-831.

Barad, K. 2007. *Meeting the University Halfway: Quantum Physics and the Entanglement of Matter and Meaning*. Durham, NC, USA: Duke University Press.

Barley, S.R. 1986. "Technology as an Occasion for Structuring: Evidence from Observations of CT Scanners and the Social Order of Radiology Departments," *Administrative Science Quarterly* (31:1), pp. 78-108.

Bartol, K.M., and Martin, D.C. 1982. "Managing Information Systems Personnel: A Review of Literature and Managerial Implications," *MIS Quarterly* (6:1), pp. 49-70.

Baskerville, R., and Pries-Heje, J. 2004. "Short cycle time systems development," *Information Systems Journal* (14:3), pp. 237-264.

Baskerville, R., Ramesh, B., Levine, L., Pries-Heje, J., and Slaughter, S. 2003. "Is Internet-Speed Software Development Different?," *IEEE Software* (20:6), pp. 70-77.

Beck, K., and Andres, C. 2004. *Extreme Programming Explained: Embrace Change*, (2. ed.). Addison-Wesley Professional.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas, D. 2001. "The Agile Manifesto." Retrieved 2008-09-20, from http://www.agilemanifesto.org/

Benbya, H., and McKelvey, B. 2006. "Using coevolutionary and complexity theories to improve IS alignment: a multi-level approach," *Journal of Information Technology* (21:4), pp. 284-298.

Bergman, M., Lyytinen, K., and Mark, G. 2007. "Boundary Objects in Design: An Ecological View of Design Artifacts," *Journal of the Association for Information Systems* (8:11), pp. 546-568.

Berry, D.M. 2004. "The Inevitable Pain of Software Development: Why There Is No Silver Bullet," *Radical Innovations of Software and Systems Engineering in the Future,* M. Wirsing, A. Knapp and S. Balsamo (eds.), Venice, Italy, pp. 50-74.

Bijker, W.E. 2001. "Social construction of technology," in: *International Encyclopedia of the Social & Behavioral Sciences,* N.J. Smelser and P.B. Baltes (eds.). Oxford, Amsterdam: Elsevier, pp. 15522–15527.

Bijker, W.E. 2010. "How is technology made?--That is the question!," *Cambridge Journal of Economics* (34:1), 1/1, pp. 63-76.

Bijker, W.E., Hughes, T.P., and Pinch, T.J. (eds.). 1987. *The Social Construction of Technological Systems: New Directions in the Sociology and History of Technology*. Cambridge, MA, USA: MIT Press.

Black, S.E., Boca., P.P., Bowen, J.P., Gorman, J., and Hinchey, M.G. 2009. "Formal versus agile: Survival of the fittest," *IEEE Computer* (49:9), pp. 39-45.

Boehm, B., and Basili, V.R. 2000. "Gaining Intellectual Control of Software Development," *Computer* (33:5), pp. 27-33.

Boehm, B., and Turner, R. 2004. *Balancing Agility and Discipline: A Guide for the Perplexed*. Boston, MA, USA: Addison-Wesley.

Boehm, B.W. 1988. "A Spiral Model of Software Development and Enhancement," *IEEE Computer* (21:4), pp. 61-72.

Boland, R.J. 1996. "Why shared meanings have no place in structuration theory: A reply to scapens and macintosh," *Accounting, Organizations and Society* (21:7-8), 1996/11//, pp. 691-697.

Bostrom, R., and Heinen, J.S. 1977. "MIS Problems and Failures: A Socio-Technical Perspective," *MIS Quarterly* (1:3).

Bostrom, R.P. 1989. "Successful application of communication techniques to improve the systems development process," *Information & Management* (16:5), pp. 279-295.

Bostrom, R.P., Gupta, S., and Thomas, D. 2009. "A Meta-Theory for Understanding Information Systems Within Sociotechnical Systems," *Journal of Management Information Systems* (26:1), pp. 17-48.

Branigan, H.P., Pickering, M.J., Pearson, J., and McLean, J.F. 2010. "Linguistic alignment between people and computers," *Journal of Pragmatics* (42:9), pp. 2355-2368.

Bühler, K. 1990. *Theory of language: the representational function of language*. Amsterdam, Philadelphia: J. Benjamins Pub. Co.

Cao, L., Mohan, K., Peng, X., and Ramesh, B. 2009. "A framework for adapting agile development methodologies," *European Journal of Information Systems* (18:4), pp. 332-343.

Chae, B., and Poole, M.S. 2005. "The surface of emergence in systems development: agency, institutions, and large-scale information systems," *European Journal of Information Systems* (14:1), pp. 19-36.

Chakraborty, S., Sarker, S., and Sarker, S. 2010. "An Exploration into the Process of Requirements Elicitation: A Grounded Approach," *Journal of the Association for Information Systems* (11:4), pp. 212-249.

Chan, Y.E., Huff, S.L., Barclay, D.W., and Copeland, D.G. 1997. "Business Strategic Orientation, Information Systems Strategic Orientation, and Strategic Alignment," *Information Systems Research* (8:2), pp. 125-150.

Chow, T., and Cao, D.-B. 2008. "A survey study of critical success factors in agile software projects," *Journal of Systems and Software* (81:6), pp. 961-971.

Clark, H.H. 1996. *Using Language*. New York, NY, USA: Cambridge University Press.

Cockburn, A., and Highsmith, J. 2001. "Agile Software Development: The People Factor," *IEEE Computer* (34:11), pp. 131-133.

Conboy, K. 2009. "Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development," *Information Systems Research* (20:3), pp. 329-354.

Crowston, K., and Wade, M. 2010. "Introduction to JAIS Special Issue on Empirical Research on Free/Libre Open Source Software," *Journal of the Association for Information Systems* (11:11).

Cule, P., Schmidt, R., Lyytinen, K., and Keil, M. 2000. "Strategies for Heading off IS Project Failure," *Information Systems Management* (17:2), pp. 65-73.

Curtis, B., Krasner, H., and Iscoe, N. 1988. "A field study of the software design process for large systems," *Communications of the ACM* (31:11), pp. 1268-1287.

Deacon, T.W. 1997. *The Symbolic Species. The Co-evolution of Language and the Brain*. New York, NY, USA: W. W. Norton & Company.

DeMarco, T., and Lister, T. 1987. *Peopleware: productive projects and teams*. Dorset House Publishing Co., Inc. New York, NY, USA.

Dennis, A.R., Fuller, R.M., and Valacich, J.S. 2008. "Media, tasks, and communication processes: A theory of media synchronicity," *MIS Quarterly* (32:3), pp. 575-600.

DeSanctis, G., and Poole, M.S. 1994. "Capturing the Complexity in Advanced Technology Use: Adaptive Structuration Theory," *Organization Science* (5:2), pp. 121-147.

Falkenberg, E.D., Hesse, W., Lindgreen, P., Nilsson, B.E., Oei, J.L.H., Rolland, C., Stamper, R.K., Assche, F.J.M.V., Verrijn-Stuart, A.A., and Voss, K. 1998. "A Framework of Information Systems Concepts. The FRISCO Report, IFIP WG 8.1 Task Group FRISCO (Web edition)." Retrieved 2009-09-23, from http://www.mathematik.uni-marburg.de/~hesse/papers/fri-full.pdf

Faulkner, P., Lawson, C., and Runde, J. 2010. "Theorising technology," *Cambridge Journal of Economics* (34:1), pp. 1-16.

Faulkner, P., and Runde, J. 2009. "On the Identity of Technological Objects and User Innovations in Function," *The Academy of Management Review* (34:3), pp. 442-462.

Faulkner, P., and Runde, J. 2010. "The social, the material, and the ontology of non-material technological objects." Retrieved 2011-08-25, from http://www.lse.ac.uk/collections/informationSystems/newsAndEvents/2011events/Non-MaterialTechnologicalObjects.pdf

Fraser, S., and Mancl, D. 2008. "No Silver Bullet: Software Engineering Reloaded," *IEEE Software* (25:1), pp. 91-94.

Galliers, R.D., and Swan, J.A. 2000. "There's More to Information Systems Development than Structured Approaches: Information Requirements Analysis as a Socially Mediated Process," *Requirements Engineering* (5:2), pp. 74-82.

Gallivan, M.J., and Keil, M. 2003. "The user-developer communication process: a critical case study," *Information Systems Journal* (13:1), pp. 37-68.
Giddens, A. 1979. *Central Problems in Social Theory: Action, Structure and Contradiction in Social Analysis*. Berkeley, CA, USA: University of California Press.
Giddens, A. 1984. *The Constitution of Society: Outline of Theory of Structuration*. Berkley, CA, USA: University of California Press.
Giddens, A. 1991. "Structuration theory: past, present and future," in: *Giddens' Theory of Structuration: A Critical Appreciation,* C.G.A. Bryant and D. Jary (eds.). London, UK: Routledge, pp. 201-221.
Guinan, P., and Bostrom, R.P. 1986. "Development of computer-based information systems: A communication framework," *SIGMIS Database* (17:3), pp. 3-16.
Guinan, P.J., and Scudder, J.N. 1989. "Client-Oriented Interactional Behaviors for Professional-Client Settings," *Human Communication Research* (15:3), pp. 444-462.
Hansen, S., Berente, N., Lyytinen, K.J. 2008. "Requirements in the 21st Century: Current Practice & Emerging Trends," in: *Design Requirements Engineering: A Ten-Year Perspective,* K.J. Lyytinen, P. Loucopoulos, J. Mylopoulos and W. Robinson (eds.). Berlin, Germany et al.: Springer, pp. 44-87.
Hansen, S., and Lyytinen, K. 2010. "Challenges in Contemporary Requirements Practice," *43th Hawaii International Conference on System Sciences (HICSS 2010)*, Koloa, HI, USA.
He, J., Butler, B.S., and King, W.R. 2007. "Team Cognition: Development and Evolution in Software Project Teams," *Journal of Management Information Systems* (24:2), pp. 261-292.
Herbsleb, D., and Mockus, A. 2003. "An empirical study of speed and communication in globally-distributed software development," *IEEE Transactions on Software Engineering* (29:6), pp. 1-14.
Hesse, W., Müller, D., and Ruß, A. 2008. "Information, information systems, information society: interpretations and implications," *Poiesis & Praxis: International Journal of Technology Assessment and Ethics of Science* (5:3), pp. 159-183.
Highsmith, J., and Cockburn, A. 2001. "Agile Software Development: The Business of Innovation," *IEEE Computer* (34:9), pp. 120-127.
Hirschheim, R., Klein, H., and Lyytinen, K. 1995. *Information Systems Development and Data Modeling. Conceptual and Philosophical Foundations*. Cambridge, UK et al.: Cambridge University Press.
Hirschheim, R., Klein, H., and Newman, M. 1991. "Information Systems Development as Social Action: Theoretical Perspective and Practice," *OMEGA* (19:6), pp. 587-608.
Hofmann, H.F., and Lehner, F. 2001. "Requirements engineering as a success factor in software projects," *IEEE Software* (18:4), pp. 58-66.
Holmqvist, B. 1989. "Work, Language and Perspective: An Empirical Investigation of the Interpretation of Computer-Based Information Systems," *Scandinavian Journal of Information Systems* (1:1), pp. 72-96.
Houkes, W., and Meijers, A. 2006. "The ontology of artefacts: the hard problem," *Studies In History and Philosophy of Science Part A* (37:1), pp. 118-131.
Howcroft, D., Mitev, N., and Wilson, M. 2004. "What We May Learn from the Social Shaping of technology Approach," in: *Social theory and philosophy for information systems,* L. Willcocks and J. Mingers (eds.). Chichester, UK et al.: John Wiley & Sons, pp. 329-371.
Iivari, J., Hirschheim, R., and Klein, H.K. 2004. "Towards a distinctive body of knowledge for Information Systems experts: coding ISD process knowledge in two IS journals," *Information Systems Journal* (14:4), pp. 313-342.
Iivari, J., and Iivari, N. 2011. "The relationship between organizational culture and the deployment of agile methods," *Information and Software Technology* (53:5), pp. 509-520.
Iivari, J., Isomäki, H., and Pekkola, S. 2010. "The user – the great unknown of systems development: reasons, forms, challenges, experiences and intellectual contributions of user involvement," *Information Systems Journal* (20:2), pp. 109-117.
Ives, B., and Olson, M.H. 1984. "User Involvement and MIS Success: A Review of Research," *Management Science* (30:5), pp. 586-603.
Jarke, M., Lyytinen, K., Mylopoulos, J., Kappel, G., Leite, J., Mark, G., Ramesh, B., Schmitz, D., and Sutcliffe, A.G. 2009. "High-Impact Requirements for Software-Intensive Systems:Seminar Outlines and Working Group Summaries," *Perspectives Workshop: Science of Design: High-Impact Requirements for Software-Intensive Systems,* M. Jarke, K. Lyytinen and J. Mylopoulos (eds.), Dagstuhl, Germany: Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany.

Jones, M., Orlikowski, W., and Munir, K. 2004. "Structuration Theory and Information Systems: A Critical Reappraisal," in: *Social Theory and Philosophy for Information Systems,* J. Mingers and L. Willcocks (eds.). Chichester, UK et al.: John Wiley & Sons.

Jones, M.R., and Karsten, H. 2008. "Giddens's Structuration Theory and Information Systems Research," *MIS Quarterly* (32:1), pp. 127-157.

Jones, M.R., and Karsten, H. 2009. "Divided by a Common Language? A Response to Marshall Scott Poole," *MIS Quarterly* (33:3), pp. 589-595.

Joshi, K.D., Sarker, S., and Sarker, S. 2007. "Knowledge transfer within information systems development teams: Examining the role of knowledge source attributes," *Decision Support Systems* (43:2), pp. 322-335.

Kamlah, W., and Lorenzen, P. 1984. *Logical Propaedeutic. Pre-School of Reasonable Discourse*. Lanham, MD, USA: University Press of America.

Kappos, A., and Rivard, S. 2008. "A Three-Perspective Model of Culture, Information Systems, and Their Development and Use," *MIS Quarterly* (32:3), pp. 601-634.

Kautz, K. 2004. "The Enactment of Methodology: The Case of Developing a Multimedia Information System," *International Conference on Information Systems (ICIS 2004), Paper 55*.

Kautz, K., Madsen, S., and Nørbjerg, J. 2007. "Persistent problems and practices in information systems development," *Information Systems Journal* (17), pp. 217–239.

Kautz, K., and Nielsen, P.A. 2004. "Understanding the implementation of software process improvement innovations in software organizations," *Information Systems Journal* (14:1), pp. 3-22.

Ko, D.-G., Kirsch, L.J., and King, W.R. 2005. "Antecedents of Knowledge Transfer from Consultants to Clients in Enterprise System Implementations," *MIS Quarterly* (29:1), pp. 59-85.

Kraut, R.E., and Streeter, L.A. 1995. "Coordination in software development," *Communications of the ACM* (38:3), pp. 69-81.

Kroes, P. 2010. "Engineering and the dual nature of technical artefacts," *Cambridge Journal of Economics* (34:1), 1/1, pp. 51-62.

Kroll, P., and Kruchten, P. 2003. *The rational unified process made easy: a practitioner's guide to the RUP*, (4 ed.). Boston, MA, USA: Addison-Wesley.

Kruchten, P. 2004. *The rational unified process: an introduction*, (3 ed.). Boston, MA, USA: Addison-Wesley.

Laughery, K.R., and Laughery, K.R. 1985. "Human factors in software engineering: A review of the literature," *Journal of Systems and Software* (5:1), pp. 3-14.

Lee, A.S. 2004. "Thinking about Social Theory and Philosophy for Information Systems," in: *Social Theory and Philosophy for Information Systems,* L. Willcocks and J. Mingers (eds.). Chichester, UK et al.: John Wiley & Sons, pp. 1-26.

Lee, G., and Xia, W. 2010. "Toward Agile: An Integrated Analysis of Quantitative and Qualitative Field Data," *MIS Quarterly* (34:1), pp. 87-114.

Leonardi, P.M., and Barley, S.R. 2010. "What's Under Construction Here? Social Action, Materiality, and Power in Constructivist Studies of Technology and Organizing," *The Academy of Management Annals* (4:1), 2011/08/24, pp. 1-51.

Levina, N. 2005. "Collaborating on Multiparty Information Systems Development Projects: A Collective Reflection-in-Action View," *Information Systems Research* (16:2), pp. 109-130.

Levina, N., and Vaast, E. 2005. "The emergence of boundary spanning competence in practice: implications for implementation and use of information systems," *MIS Quarterly* (29:2), pp. 335-363.

Lyytinen, K., and Newman, M. 2008. "Explaining information systems change: a punctuated socio-technical change model," *European Journal of Information Systems* (17:6), pp. 589-613.

Markus, M.L., and Mao, J.-Y. 2004. "Participation in Development and Implementation - Updating An Old, Tired Concept for Today's IS Contexts," *Journal of the Association for Information Systems* (5:11), pp. 514-544.

Markus, M.L., and Robey, D. 1988. "Information Technology and Organizational Change: Causal Structure in Theory and Research," *Management Science* (34:5), pp. 583-598.

Markus, M.L., and Silver, M.S. 2008. "A Foundation for the Study of IT Effects: A New Look at DeSanctis and Poole's Concepts of Structural Features and Spirit," *Journal of the Association for Information Systems* (9:10/11), pp. 609-632.

Martin, J. 1991. *Rapid application development*. Macmillan Publishing Co., Inc.

Maruping, L.M., Venkatesh, V., and Agarwal, R. 2009. "A Control Theory Perspective on Agile Methodology Use and Changing User Requirements," *Information Systems Research* (20:3), 9/1, pp. 377-399.

Mathiassen, L. 1998. "Reflective Systems Development," *Scandinavian Journal of Information Systems* (10:1/2), pp. 67-117.

Miranda, S.M., and Saunders, C.S. 2003. "The Social Construction of Meaning: An Alternative Perspective on Information Sharing," *Information Systems Research* (14:1), pp. 87-106.
Nelson, R.R. 2007. "IT project management: infamous failures, classic mistakes, and best practices," *MIS Quarterly Executive* (6:2), pp. 67-78.
Newman, M., and Robey, D. 1992. "A social process model of user-analyst relationships," *MIS Quarterly* (16:2), pp. 249-266.
Newman, M., and Robey, D. 1996. "Sequential Patterns in Information Systems Development: An Application of a Social Process Model," *ACM Transactions on Information Systems* (14:1), pp. 30-63.
Orlikowski, W.J. 1992. "The Duality of Technology: Rethinking the Concept of Technology in Organizations," *Organization Science* (3:3), 1992/08, pp. 398-427.
Orlikowski, W.J. 2000. "Using Technology and Constituting Structures: A Practice Lens for Studying Technology in Organizations," *Organization Science* (11:4), pp. 404-428.
Orlikowski, W.J. 2010. "The sociomateriality of organisational life: considering technology in management research," *Cambridge Journal of Economics* (34:1), 1/1, pp. 125-141.
Orlikowski, W.J., and Gash, D.C. 1994. "Technological frames: making sense of information technology in organizations," *ACM Transactions on Information Systems* (12:2), pp. 174-207.
Orlikowski, W.J., and Iacono, C.S. 2001. "Research commentary: desperately seeking the" IT" in IT research-A call to theorizing the IT artifact," *Information Systems Research* (12:2), pp. 121-134.
Orlikowski, W.J., and Robey, D. 1991. "Information technology and the structuring of organizations," *Information Systems Research* (2:2), pp. 143-169.
Orlikowski, W.J., and Scott, S.V. 2008. "Chapter 10: Sociomateriality: Challenging the Separation of Technology, Work and Organization," *The Academy of Management Annals* (2), pp. 433-474.
Pawlowski, S., and Robey, D. 2004. "Bridging User Organizations: Knowledge Brokering and the Work of Information Technology Professionals," *MIS Quarterly* (28:4), pp. 645-672.
Pikkarainen, M., Haikara, J., Salo, O., Abrahamsson, P., and Still, J. 2008. "The impact of agile practices on communication in software development," *Empirical Software Engineering* (13:3), pp. 303-337-337.
Pinch, T. 2010. "On making infrastructure visible: putting the non-humans to rights," *Cambridge Journal of Economics* (34:1), January 1, 2010, pp. 77-89.
Pinch, T.J., and Bijker, W.E. 1984. "The social construction of facts and artefacts: or how the sociology of science and the sociology of technology might benefit each other," *Social Studies of Science* (14), pp. 399-441.
Poole, M.S. 2009. "Response to Jones and Karsten, "Giddens' Structuration Theory and Information Systems Researched"," *MIS Quarterly* (33:3), pp. 583-587.
Poole, M.S., and DeSanctis, G. 2004. "Structuration Theory in Information Systems Research: Methods and Controversies," in: *The Handbook of Information Systems Research,* M.E. Whitman and A.B. Woszczynski (eds.). Hershey, PA, USA: IGI Global, pp. 206-249.
Poppendieck, M., and Poppendieck, T. 2003. *Lean Software Development: An Agile Toolkit*, (1 ed.). Addison-Wesley Longman, Amsterdam.
Ramesh, B., Cao, L., Mohan, K., and Xu, P. 2006. "Can distributed software development be agile?," *Communications of the ACM* (49:10), pp. 41-46.
Robey, D. 1995. "Theories that Explain Contradiction: Accounting for the Contradictory Organizational Consequences of Information Technology," *16th International Conference on Information Systems (ICIS 1995),* J.I. DeGross, G. Ariav, C. Beath, R. Høyer and C. Kemerer (eds.), Amsterdam, The Netherlands 1995, pp. 55-63.
Robey, D., and Markus, M.L. 1984. "Rituals in Information System Design," *MIS Quarterly* (8:1).
Robillard, P.N. 1999. "The role of knowledge in software development," *Communications of the ACM* (42:1), pp. 87-92.
Rose, J., and Scheepers, R. 2001. "Structuration theory and information systems development - frameworks for practice," *Proceedings of the 9th European Conference on Information Systems,* S. Smithson, J. Gricar, M. Podlogar and S. Avgerinou (eds.), Bled, Slovenia, pp. 217-231.
Royce, W.W. 1970. "Managing the Development of Large Software Systems: Concepts and Techniques," *Proceedings of WesCon*.
Ryan, K. 1993. "The Role of Natural Language in Requirements Engineering," *IEEE International Symposium on Requirements Engineering 1993*, San Diego, CA, USA, pp. 240-242.
Sambamurthy, V., and Kirsch, L.J. 2000. "An Integrative Framework of the Information Systems Development Process," *Decision Sciences* (31:2), pp. 391-411.

Sawyer, S., Farber, J., and Spillers, R. 1997. "Supporting the social processes of software development," *Information Technology and People* (10:1), pp. 46-62.
Sawyer, S., and Guinan, P.J. 1998. "Software development: Processes and performance " *IBM Systems Journal* (37:4), pp. 552-569.
Sawyer, S., Guinan, P.J., and Cooprider, J. 2010. "Social interactions of information systems development teams: a performance perspective," *Information Systems Journal* (20:1), pp. 81-107.
Schwaber, K. 1995. "Scrum Development Process," *Conference on Object-Oriented Programming Systems, Languages, and Applications*, pp. 117-134.
Searle, J. 1995. *The Construction of Social Reality*. London, UK: Penguin Books.
Siau, K., Long, Y., and Ling, M. 2010. "Toward a Unified Model of Information Systems Development Success," *Journal of Database Management* (21:1), pp. 80-101.
Slaughter, S.A., Levine, L., Ramesh, B., Pries-Heje, J., and Baskerville, R. 2006. "Aligning software processes with strategy," *MIS Quarterly* (30:4), pp. 891-918.
Tan, M. 1994. "Establishing Mutual Understanding in Systems Design: An Empirical Study," *Journal of Management Information Systems* (10:4), pp. 159-182.
Te'eni, D. 2001. "Review: A Cognitive-Affective Model of Organizational Communication for Designing IT," *MIS Quarterly* (25:2), pp. 251-312.
Teo, T.S.H., and King, R.W. 1997. "Integration between business planning and information systems planning: an evolutionary-contingency perspective," *Journal of Management Information Systems* (14:1), pp. 185-214.
Tomasello, M. 2008. *Origins of Human Communication*. Cambridge, MA, USA: MIT Press.
Tomasello, M., Carpenter, M., Call, J., Behne, T., and Moll, H. 2005. "Understanding and sharing intentions: The origins of cultural cognition," *Behavioral and Brain Sciences* (28:675-691).
Truex, D., Baskerville, R., and Travis, J. 2000. "Amethodical Systems Development: The Deferred Meaning of Systems Development Methods," *Accounting, Management and Information Technology* (10), pp. 53-79.
Truex, D.P., Baskerville, R., and Klein, H. 1999. "Growing systems in emergent organizations," *Communications of the ACM* (42:8), pp. 117-123.
Van de Ven, A.H. 1992. "Suggestions for studying strategy process: A research note," *Strategic Management Journal* (13:S1), pp. 169-188.
Van de Ven, A.H., and Poole, M.S. 1995. "Explaining Development and Change in Organizations," *The Academy of Management Review* (20:3), pp. 510-540.
Van de Ven, A.H., and Poole, M.S. 2005. "Alternative Approaches for Studying Organizational Change," *Organization Studies* (26:9), pp. 1377-1404.
Vidgen, R., and Wang, X. 2009. "Coevolving Systems and the Organization of Agile Software Development," *Information Systems Research* (20:3), 9/1, pp. 355-376.
Walsham, G., and Han, C. 1991. "Structuration theory and information systems research," *Journal of Applied Systems Analysis* (17), pp. 77-85.
Wastell, D.G. 1996. "The fetish of technique: methodology as a social defence," *Information Systems Journal* (6:1), pp. 25-40.
Watts Sussman, S., and Guinan, P.J. 1999. "Antidotes for high complexity and ambiguity in software development," *Information & Management* (36:1), pp. 23-35.
Weber, R.A., and Camerer, C.F. 2003. "Cultural Conflict and Merger Failure: An Experimental Approach," *Management Science* (49:4), pp. 400-415.
Weick, K.E. 1979. *The Social Psychology of Organizing*, (2 ed.). Reading, MA, USA: Addison Wesley.
Weick, K.E., Sutcliffe, K.M., and Obstfeld, D. 2005. "Organizing and the Process of Sensemaking.," *Organization Science* (16:4), pp. 409-421.
White, K.B. 1984. "MIS Project Teams: An Investigation of Cognitive Style Implications," *MIS Quarterly* (8:2), pp. 95-101.
Williams, C. 2010. "Client–vendor knowledge transfer in IS offshore outsourcing: insights from a survey of Indian software engineers," *Information Systems Journal* (21:4), pp. 335-356.
Xia, W., and Lee, G. 2005. "Complexity of Information Systems Development Projects: Conceptualization and Measurement Development," *Journal of Management Information Systems* (22:1), pp. 45-83.
Zammuto, R.F., Griffith, T.L., Majchrzak, A., Dougherty, D.J., and Faraj, S. 2007. "Information Technology and the Changing Fabric of Organization," *Organization Science* (18:5), pp. 749-762.