Annosi M.C., Magusson M., Martini A., Appio F.P. (2015) "*Social conduct, learning and innovation: an abductive study of the dark side of agile software development*", Creativity and Innovation Management, forthcoming

ABSTRACT

Agile methodologies have been adopted by an increasing number of organizations to improve their responsiveness. However, few studies have empirically analysed the effect of Agile on long-term organizational goals such as learning and innovation. Using an abductive approach, this study examines the relationships between self-regulated teams' social conduct and their resulting learning and innovation. Results indicate that the time pressure induced by the implementation of Agile impedes team engagement in learning and innovation activities. Time pressure is affected by a set of different control strategies, more specifically concertive, belief, diagnostic, and boundary controls, and these need to be adequately addressed in order to minimize the potential dark side of Agile.

> *"Now we are working with new products and we have to learn how they work - at least so much that we can see where and how to do the implementations. But to really understand the product, to be able to do improvements, that takes time. Before you had knowledge about the product, now you even don't know if you can neither propose nor do any improvements - spending time of investigation without any outcome isn't so popular I guess. Teams don't spend time on digging the product, people are just making features"* [Team Member]

INTRODUCTION

Investigating the effects of Agile software development methods, we notice that at least for one of them everything that glitters is not gold. We are talking about Scrum (Schwaber and Beedle, 2002). It is widely recognized that changes from traditional, planning-intensive and linear development approaches to more iterative and self-organized approaches inspired by agile methodologies (Beck et al., 2001) are mind-bogglingly complex and nuanced. This does not only pertain to the methodology in itself. The changes imposed to social factors regulating interaction processes matter as well. In an increasingly inter-connected and rapidly changing environment, many firms are faced with continuous renewal of - and adaptation to - products and services to match customer needs and requirements (Beck and Andres, 2005). Scrum enters the stage to solve

the tension between its inherent complexity and the outside world's quest for novelty.

Scrum is an Agile software development method (Schwaber and Beedle, 2002) that has been adopted widely by various industries. Its peculiar characteristics allow the firms implementing it to cope with environmental unpredictability by deploying strict control mechanisms (Schwaber, 1997) and leveraging the autonomy of the team (Cohen and Bailey, 1997). Scrum involves a strong emphasis on team's self-organization and use of iterative processes. This results in high levels of autonomy, information redundancy, and requisite variety, as well as shared visions and goals with a potential positive effect on learning and creativity. Autonomy favors creativity in problem solving and increases team learning in uncertain environments (Imai et al., 1984). Additionally, self-organization and local control create the conditions for teams to be open to innovative ideas (Lyytinen and Rose, 2006). Further, in enabling a good balance between flexibility and structure, Agile creates a slightly chaotic environment, not a primarily structured context where creativity and innovation can occur (Highsmith, 2002).

However, the social factor can become a problem. Agile software development methods depend heavily on teamwork (Nerur and Balijepally, 2007) as opposed to the individual role assignment that characterizes plan-driven development (Nerur et al., 2005). Combining team members' individual knowledge and skills does not secure a creative outcome unless there is a proper team environment, where team members have the ability and the motivation to utilize their potential (Aalbers et al., 2013; Shin et al., 2012). Team-level knowledge creation activity requires members not only to produce novel ideas, but also to share them within the team, to pay attention to one another's thinking, and to create new associations that merge team members' insights into new workable solutions (Baer et al., 2010; Harrison and Rouse, 2014). Thus, the production of new knowledge in teams is the result of the joint effect of

team members' personal characteristics and the social context (Shalley et al., 2004).

Based on this reasoning, the study analyzes the role of team's social conduct in predicting team learning and innovation. Precisely, by using an abductive approach (Peirce, 1903; Dubois and Gadde, 2002, 2014) and the modeling of paradigms emerging from the context of this investigation, we disentangle how different sources of controls, formal and informal, as disciplinary mechanisms and techniques of surveillance (Clegg et al., 2002), influence the formation of team's social conduct (Latham and Locke, 1991). Also, we analyze the effect of team's social conduct on team's learning and innovation as it acts as source of influence that guides team members' behaviors. Result show that managerial practices and work routines deriving from the implementation of Scrum and realizing different form of controls, serve as efficiency carriers and induce team members to focus on what should be done. This greatly affects team orientation to learn and to innovate and consequently weakens their self-regulated learning activities.

In the next section, the key elements of Scrum are described, followed by a discussion of the received theory on Agile methodology and Scrum in the learning and ideation domains. Following a detailed description of the research method, empirical observations are presented and analyzed. Finally, results are discussed, and propositions for further research proposed.

THEORETICAL BACKGROUND

Agile Software Development

The popularity of Agile methods has grown especially in relation to technology projects as they have been designed to address high volatility in the market environment (Lindvall et al., 2002). They cope with different types of changes that occur in projects, in terms of: (1) goals; (2) materials, resources, tools, and techniques; and (3) relationships with

other related projects, services, or products (Collyer et al., 2010). Consistent with this, Williams and Cockburn (2003, p.39) define Agile software development as being ''about feedback and change", and argue that agile methodologies have been conceived to ''embrace, rather than reject, higher rates of change".

A U.S. Department of Defense study reports that 45 percent of software features fail to meet user needs and requirements (Larman, 2004). This has led organizations to embrace alternative approaches to traditional planning-driven development processes (Vidgen and Wang, 2009), and to adopt Agile methods. The concept of agility understood as the ability to efficiently and effectively react to user requirement changes is at the basis of agile development methodologies (Lee and Xia, 2010). Agile methods imply the adoption of underlying principles such as lightness and leanness (Cockburn, 2007), and related notions such as nimbleness, quickness, dexterity, suppleness and alertness (Erickson et al., 2005).

According to Agile principles, business value is created by delivering working software to customers at regular short intervals (Dingsøyr et al., 2012) and having customers (or their representatives) fully involved in the development process to encourage feedback and reflection which can lead to more satisfactory outcomes. The iterative nature of Agile allows for regular stakeholder interaction, corrections made "on the fly", and the re-scoping of project requirements supported by updated information or a new customer request. Self-managing teams are the basic units used to realize effective outcomes and eliminate waste and inefficiencies (Conboy, 2009).

Scrum Methodology

Scrum is one of the most common Agile methods (Dingsøyr et al., 2012). The Scrum development process was originally proposed by Schwaber (1997) and is arguably based on the product development methodology described in Takeuchi and Nonaka (1986). In this methodology, whose

characteristics and development flow are described in Table 1 and Figure 1, respectively, a new approach to commercial product development was introduced to increase speed and flexibility. Scrum brings decision-making authority to the operational level by leveraging self-managing teams. The Scrum development process as proposed by Schwaber (1997) comprises various ceremonies which self-managing teams have to respect (Cohn, 2009). More specifically, teams are expected to have:

- sprint planning meetings to determine a list of prioritized features for the team;
- daily Scrum meetings, focusing on what each team member accomplished the day before and should accomplish the coming day;
- demo meetings where the team shows what was completed during the sprint;
- retrospective meetings, aiming to reflect on how the team is doing and find ways to improve.

As evident from Figure 1, software development is realized by iterative increments (or "Sprints"). Sprints usually last between one and four weeks. The features to be developed in the target system are registered in a common backlog. Team members coordinate the work within their daily Scrum meeting. Each sprint starts with a planning and ends with a demo meeting where the delivered feature is shown to the customer (or its representative).

Figure 1: Agile Scrum development process

Agile literature review

Few studies empirically analyzed Agile key concepts and their basic theoretical relationships (Baskerville, 2006; Boehm and Turner, 2004; Larman, 2004). Erickson and co-authors (2005) argued that Agile

development lacks theoretical underpinnings and scientific evidence that support its claimed benefits and key principles (Erickson et al., 2005). For instance, although Scrum stresses the importance of team autonomy and team diversity as important to improve software development agility (Larman, 2004), little empirical research has examined how team autonomy and team diversity affect software development agility (Lee and Xia, 2010). In addition, the positive effect of Agile on organizational performance is supported mainly by anecdotal evidence and rhetorical arguments (Lee and Xia, 2010) with few field studies systematically verifying if, how or why Agile development is effective (Fruhling and De Vreede, 2006; Moe et al., 2008). An exception is found in Lee and Xia (2010), which proposes a tradeoff relationship between extensiveness, which refers to the amount of different types of requirements a team is able to implement and the efficiency of the team's response to user requirement change. These two agility dimensions have distinct impacts on team performance: response efficiency has a positive impact on on-time completion, on-budget completion, and software functionality, and response extensiveness has a positive impact on software functionality. Lee and Xia found also that team autonomy has a positive effect on response efficiency and a negative effect on response extensiveness, while team diversity has a positive impact on response extensiveness. Some preliminary results provides evidence of some issues related to managing effective coordination and control of information in global software projects applying agile software development (Dingsøyr and Smite, 2014).

Magazinius and Feldt (2011) compared Agile with non-Agile companies and found that the accomplishment of objectives such as meeting time and budget goals, and the causes of failures is not dramatically different between the two types of organizations. This suggests the need for more and better investigations of the effects of using Agile. Another shortcoming of much of the research on Agile methodologies is that it

has primarily focused on the phases of introduction and adoption of Agile (Dyba˚ and Dingsøyr, 2008), and mostly ignores its effects on long-term organizational goal, such as learning and innovation (Abrahamsson et al., 2009). Also, Dybå and Dingsøyr (2008) find that the quality of empirical studies on Agile software development is poor and that research methods are not well described, and the validity and reliability of results is not always addressed.

They call for more rigorous studies on benefits and limitations of Agile methods, which would inform organizational decision makers pondering the adoption of Agile methodologies.

Table 1: Main characteristics of Scrum methodology according to Dybå and Dingsøyr (2008) and Schwaber (1997)

Characteristics of Scrum methodology	Detailed explanation
Basic assumption	The main difference between the defined waterfall, spiral and iterative development processes and Scrum approach is that Scrum assumes that the analysis, design, and development processes are unpredictable. Thus control mechanisms are embraced to deal with the unpredictability and control the risk
Expected outcomes	Flexibility, responsiveness, and reliability are the main expected results from the implementation of Scrum
Applicability	It is suitable for situations where it is difficult to plan ahead
Core Process Levers	Scrum embeds mechanisms enabling a strict empirical control, which relies on short feedback loops. They allow tolerance of schedule deviations and are a core element of the Scrum framework
Scrum Process roles	The product owner, assigned to a specific team and representing the customer, takes the decision about which backlog item should be developed in a specific sprint. One team member, the Scrum master, is in charge of solving

	problems that stop the team from working effectively
Basic Organization form/structure	Software is developed by a self-managing team which operates in autonomously and has full responsibility for the software deliveries. The self-managing team is cross-functional. Thus, each person in the team can have different experience and background to turn software requirements into the best possible solutions. Teams are usually small including at most six/seven people
Knowledge management approach	Instructions on how to do the first and the last phases of each sprint are explicit. The flow is linear, with some iterations in the planning phase. For the rest when available, explicit process knowledge is used; otherwise tacit knowledge and trial and error is used to build process knowledge
Communication	The team is open to the environment until the closure phase of each sprint. In fact the software deliverable can be modified at any time during the planning and sprint phases of the project. The team is subject to competitive, time, quality, and financial pressures
Management	Management defines the content and timing of the each software release, and controls its evolution. When it believes the constraints on the cost, and quality requirements have been overcome for a new release, it declares it as "closed". It supervises work by controlling the backlog, the change in the project scope, the software releases basing on the variables of requirement, cost, time and quality; and monitoring the technical problems that can arise, the efficacy of related solutions and the risk potentially affecting the success of a given project

Scrum software development process and learning: what we know and what we do not know

It has been argued that Agile software development methods, such as Scrum, are best suited to investigate new fields and to sustain individual

who prioritize innovation and creativity (Highsmith, 2002). Some authors believe that Scrum is based largely on Nonaka's (1994) theory of knowledge creation (e.g., Sutherland, 2010), intended to organize teams for effective and efficient knowledge creation (Beedle et al., 1999). Arguably, some of the basic characteristics of Scrum are derived from this theory: for example, the creation of new knowledge through direct interaction among people in teams, emphasis on tacit knowledge, feedback from working software, and a significant amount of information sharing activities (Beedle et al., 1999; Sutherland, 2010). Scrum implies also the transition from a traditional bureaucratic management to one of empowerment and ownership, intrinsically increasing the opportunities for double-loop learning. By encouraging and accepting changes, the Scrum approach reduces the stability that is detrimental to learning (Argyris, 1976). Furthermore, the team goals and feedback loops characterizing the Scrum framework are relevant to double-loop learning, since inquiry and feedback are more dynamic in such settings (Yeo, 2002). Thus, we could argue that Scrum seems to introduce several processes to enable knowledge sharing, knowledge creation, and learning throughout its software development activities, via support for team members' flexibility, and a collaborative organizational approach (Nerur et al., 2005). However, as Argyris (1995) points out, learning happens also when there is an initial match between the intentions and the consequences of action. This type of learning is related primarily to a pattern of beliefs and the qualities of the interaction among organizational members that favor (or humper) organizational capabilities (Lumpkin et al., 2005). In fact, it requires a commitment to enhance the integrity of individual action, as well as the alignment of activities within the social environment (e.g. Schön, 1983). Thus, by focusing on team's conduct and inquiring whether the rules of team's engagement are appropriate to create a culture of openness and creativity, awareness of team members' espoused theory (what individuals say they do) and their theory-in-use (what individuals

actually do) (Argyris, 1995) is broadened. This establishes the conditions for a more refined evaluation of the team's abilities to learn and innovate. Describing the team's rules of engagement, which allow individual team member to express honestly and behave with fewer defenses, contributes to a more reliable picture of the team's abilities to learn and to innovate. This implies a deeper understanding of disciplinary mechanisms and techniques of surveillance as a basis to obtain the '*active consent and subjugation of subjects [workers], rather than their oppression, domination or external control'* (Clegg et al., 2002, p.317).

This contribution seeks to add to existing team learning and innovation theory by providing a means to analyze situated organizational interaction: it allows people to operate not just for or against socially and organizationally defined conducts, but also in a way that goes beyond these. People's conduct can be interpreted not just through the way they react to managerial control (Deetz, 1998), but also in terms of how they ingeniously shape their conduct, negotiate their norms, goals and develop emotions in the face of given managerial practices, as part of their "immaterial labor" (Hardt and Negri, 2004) accompanying and exceeding the imposed managerial control.

We investigate the following main research questions: (i) To what extent do Agile managerial practices and routines, as disciplinary mechanisms and techniques of surveillance, influence team's social conduct in a way that impacts team's learning and innovation? (ii) To what extent do the Agile managerial practices and routines act in combination and affect team's learning and innovation?

The points above are investigated using relevant identified theoretical approaches constituting the base for an abductive research framework. The specific theoretical perspectives that contribute to the development of the research questions are in particular: social control (Barker, 1993) and managerial organizational control systems (Simons, 1991, 1994). These theoretical perspectives are highly correlated because of their

complementarity nature and arguably, both are needed to identify potential learning and innovation issues in an Agile context. In the following, we brief explain the choice for these theoretical perspectives.

Strategies of control for self-regulating team's conduct: the dark side of Agile

Self-regulation is related to how individuals absorb social values and extrinsic contingencies and gradually turn them into personal values and self-motivations (Ryan and Deci, 2000). Self-regulating processes determining team's social conduct, as volitional, strongly depend on their goal setting and its consequent translation into action (Latham and Locke, 1991). However, having a specific goal (or performance level to aim for) does not trigger self-regulated modulation of thought or behavior. In most experimental research, standards are activated by clear and crucial instructions or routines (Greenwald 1982). As Simon (1967) points out most goals are "in a queue" waiting to be called by appropriate contextual circumstances. Also, Brief and Hollenbeck (1985) describe self-regulation through self-monitoring, and self-rewarding or self-punishing related to the degree of the discrepancy between an individual's behavior within a group and the shared goals of group members. Upon the perceived discrepancy, peer control exercises their influence over people's behavior. Specifically peer control is related to the team members' influence over the behavior of peers through verbal expression of their disapproval (Druskat and Kayes, 2000). This can generate pressure which is revealed in various ways, such as direct requests for improved effort, public discussion of deviations in the presence of others, or trials to create a sense of guilt for not being a good team members (Barker, 1999; Hackman, 1992; LePine and Van Dyne, 2001). Barker's (1999) ethnographic study gives many real examples of how peer control is a concrete phenomenon in teams. The present study investigates how the Scrum framework, through a concert of interrelated mechanisms, induces

pressure within the teams, which controls team members' actions more powerfully than any bureaucratic form of control. This work provides a detailed account of the dynamics manifested by team members' interactions depicting the effect of peer control and of a more complex system of organizational controls established over teams as a consequence of Agile.

RESEARCH DESIGN

In this section, we describe the research settings and the methods, including data collection and data analysis.

Research setting

Using purposeful sampling (Patton, 1990), we selected four research and development (R&D) organizations belonging to the same multinational telecommunication company (Table 2).

Table 2: Organization and roles involved

Organization Name	Interviewees	Product Characteristics	Main Activities in the organization	Countries
Organization A (20 teams)	2 Product Guardians 1 Scrum Master 2 Line Managers 3 Team Members 2 Product Owners 1 Head of Organization	Mobile Broadband	New Product Development and test consultancy	Poland, China and Sweden
Organization B (15 teams)	1 Head of Org. 2 Line Managers 1 Scrum Master 1 Product Guardian	WCDMA Radio Access Network	System Management supporting the product development	Sweden

	3 Team Members		activities for the whole company	
Organization C (24 teams)	1 Scrum Master 2 Product Guardian 2 Line Managers 3 Team Members 2 Product Owners 1 Head of Org.	The product developed is related to the communication between the Radio Network Controller and the Radio Base Station	New product development and test activities	Sweden, Poland, China and Ireland
Organization D (20 teams)	1 Scrum Master 2 Product Guardians 2 Line Managers 2 Product Owners 3 Team Members 1 Head of Org.	Radio Network Controller	New product development and test activities	Poland and Sweden

These organizations apply Agile software methodologies, especially the scrum method (Schwaber, 2004), for their product development activities. Teams have an average of seven members with different backgrounds and experiences, in order to maximize cross-functional and cross-product capabilities. In particular, in the later stages of product development activity, teams are composed with people coming from different product subsystems and having covered different types of roles (e.g. designer, tester). The criteria for selecting individuals for team working on the early product development stages are different and the emphasis is on people with experience that is more similar, and backgrounds in terms of roles covered and activities performed. Teamwork is organized in short iterations with Scrum sprints lasting three weeks. The transition to Agile was adopted in the organizations studied in 2011 and was considered

complete when all employees working in Scrum teams, which was achieved by end 2012 with the formation of 90 teams cross the four selected R&D organizations. Our analysis started one year after this completed transition to Agile.

Overview of the research method

Given the topic's complexity and limited extant knowledge, an abductive research (Dubois and Gadde, 2002, 2014) was identified as an appropriate research method. In research processes aimed at revealing complex links among interrelated set of variables, 'abduction' is a useful research approach (Suddaby, 2006). As long as 1903, Peirce suggested that discovery research rests primarily on abductive reasoning, considered to be a means of assigning 'primacy to the empirical world but in the service of theorizing' (van Maanen et al., 2007, p.1149). Abduction is a continuous undertaking, and affects every phase in the research process when 'analysis proceeds by the continuous interplay between concepts and data' (van Maanen et al., 2007, p.1149). Also Woodside (2010) argues that an abductive lens improves already very high levels of accuracy, generality and coverage of 'add-on objectives.'

At the basis of this method is a series of characteristics identified in Dubois and Gadde (2002). First, in order to exploit the full potential of the information derived from a case study, the researcher must move continuously back and forth from one research activity to another and between the empirical evidence and the theory. The recursive nature of the abductive approach forces the researcher to proceed stepwise splitting the process into specific phases. Our research involved five phases (see Figure 2).

The first phase, *Detecting*, helped to reveal the problem by comparing the empirical evidence with the existing concepts proposed in the literature. The resulting mismatch between the empirics and the theory set the direction of our investigation.

The second phase was *Sensing*. Abduction was considered a suitable approach to resolve the mismatching resulting from the first detecting phase. In parallel, we began a literature review to identify the determinants of innovation performance. Together with the people involved we tried to make sense of and focus on the main constructs of the innovation process. In this phase, we conducted a matching between the theory and the cases. This suggested a return to the theory for an in-depth analysis of the identified constructs.

The third phase was *Analyzing*. Our selection of sampling method and qualitative analysis related to interviewing was based on the theory. We performed a coding analyses on the interview material and other relevant secondary data.

The matching between theory, case study, and empirical evidence allowed us to identify the first paradigm (Figure 5) resulting in a fourth phase called *Framing*. At this stage, we needed a deeper theoretical understanding in order to investigate the corollaries of the constructs in the identified paradigm. We did this by exploring the literature on Agile and self-regulation processes.

Figure 2: Abductive framework generation (adapted from Dubois and Gadde, 2002) (EW=empirical world, FW=framework, TH=theory, CS=case)

The matching between theory and our identified framework suggested the need for more insights from the field based on focus groups and follow-up interviews, which led to the fifth phase called *Reframing*. We identified other constructs, which helped us to begin disentangling the characteristics of the first paradigm. This resulted in further theoretical ramifications and corrections in order to make sense of the constructs and formulate propositions.

Based on the above, abduction can be defined as a nonlinear, path-dependent process aimed at the closest possible match between the

theory and the reality. It highlights the relevance of parallel theoretical framework development, fundamental for a categorization exercise.
In the following, we describe data collection and analysis during the different phases of the process in order to generate the paradigms.

Data collection

Semi-structured interviews (face to face or by telephone, N=44) were carried out by two of the authors during August-October 2013. The interviews focused on capturing the effects of Agile on learning and innovation. All Agile roles were represented for the four R&D organizations. Table 2 shows the distribution of interviewees' roles and their distribution across the four organizations.
The first round of interviews was performed using purposeful sampling (Schatzman and Strauss, 1973), with the idea of selecting information-rich cases that purposefully fit our study. We considered Patton's (1990) strategies, as outlined below:

- stratified purposeful sampling was used to gather information from different subgroups: main Agile roles (product owner, Scrum masters, Agile team members), organizational entities (teams, people, supporting teams), and organizations (all 4 were involved);
- maximum variation sampling was used to capture a wide range of perspectives related to our study. In particular, we interviewed all members of senior management in the four firms.

The interviewees for the second round of the interviews were selected using theoretical sampling (Draucker et al., 2007) in order to develop the emerging theory. Our interviewees included middle level managers in order to obtain information on: (i) the sources of control mechanisms acting against Agile teams; (ii) the perceived and objective performance

at the individual team and organization levels; (iii) how the information flow was achieved. In the third stage, we re-contacted some of the interviewees from first phase in order to confirm specific emerging concepts, and also included additional Agile team members from the different organizations. We focused on particular aspects, in order to validate our initial theoretical concepts. The interview data collection processes ended when interviews were not producing new relevant concepts and resulting code categories appeared to be saturated. Interviews lasted between 60 and 80 minutes and were recorded and transcribed. The transcriptions were sent to the interviewees for validation.

A second round assessment consisted of 121 free text comments from a survey of the four organizations, administered in August 2013 (secondary data source) in parallel with the interviews. Most of the questions in the survey focused on the effects of Agile on quality, productivity and adaptability. Line managers, product owners, project/program managers, Test consultants, and cross-functional team members were involved.

Several internal documents were made available by informants including general documentation (e.g., continuous improvement framework, product improvement report, Lean and Agile strategy reports, operational descriptions, metrics, etc.), web pages (e.g., competence communities of practices), and R&D documents (e.g., description of requirement areas, competence goals, Agile amplifier for people's evaluation, team development tools, etc.).

Data analysis and reliability checks

Data were analyzed alongside sampling activities based on a coding process conducted in different phases of the analysis (see Figure 3).

Figure 3: Overall description of the analytical process.

The first step involved *open coding*. The interview transcripts and other information were closely scrutinized, followed by a close line-by-line examination of the data to develop provisional concepts. Coding was done in pairs, repeated twice and double checked starting the first time with team members' texts and the second time from the middle managers' texts. Through the comparison involving going to and from between the themes that emerged in each interview, we were able to group the concepts identified into categories.

The second step, *axial coding* (hierarchical and non-hierarchical) was designed to sort the large number of concepts into macro categories, to identify related sub-categories and to define various hierarchical relationships among them. The existing literature was part of the 'data' and was used for comparing emerging categories to be integrated into the nascent theory (Glaser, 1992). We identified macro categories, which were connected to existing theoretical concepts, which informed the construction of the first theoretical framework comprising an initial codebook. We used the computerized data management program (MAXQDA 10plus®) to organize the huge amount of data and identify codes, categories and sub categories. During the axial coding phase, in order to better analyze the collected observations, different theoretical perspective were considered. These included managerial and social control theories, augmented by team's self-efficacy and team's prior knowledge, and innovation and team's self-regulated learning as the main team outcomes. Although the authors are familiar with these theories and the related concepts, the axial coding enabled analysis of their relevance to the empirical evidence collected.

Having completed this preliminary codebook, we applied it to the set of secondary data and conducted parallel analysis of both the 44 individual

semi-structured interviews (comprising 979 codes) and the 121 free text comments (comprising 334 codes). Analysis of text from the secondary data sources was guided, but not confined, by the first codebook, which allowed us to assign new inductive codes to the data segments describing new emerging themes. This allowed us to confirm the findings from the analysis of the first data set and enrich our initial understanding of the phenomenon. The results of this step are presented inTable 3. They served as the basis for refining our initial theoretical framework.

We next performed code frequency analysis on the whole data set, using MAXQDA® tool. This allowed identification of ideas that were repeated in our data set (Ryan and Bernard, 2000, p.776), and gave an indication of the prevalence of codes across all participants. Code frequency analysis resulted in a histogram for each specific code and stratified onto the identified groups of respondents. This allowed us to infer the main characteristics of certain groups, isolate extreme or deviant cases, understand information-rich cases, which manifested the effects of the phenomenon more intensively, and gain information about the criticality of a specific theme as a function of its recurrence within the complete data set and within a specific group.

Table 3: Result of the axial coding step

Category	Sub-Category	Sample of the codes within a category
Team's prior related knowledge	Knowledge Breadth	- Team's competence naturally broadens (different roles) - People are pushed to broaden their competence - Knowledge broadening damages expertise
	Knowledge Depth	- Experts are unable to evolve their knowledge: no interaction with peers - Difficulty to recover the big picture of the product
Team's Self-Regulated Learning (SRL)		- Competence building based on the need to implement new features - Team effort devoted to increase competence

Strategies		and performance - Teams do not implement specific activities to foster learning and innovation
Team's self-efficacy		- Teams are unable to handle the wider scope of their activities - -Forced knowledge broadening creates frustration
Control systems	Belief control systems	- Importance of broadening team competence - Learning and innovation are not prioritized as developing features
	Management shared beliefs	- Importance of broadening team's competence - Achievement of product quality is important - Need to foster knowledge sharing
	High level management shared beliefs	- Importance of allowing people to develop their competence - Importance of broadening competence to generate new insights - Teams have to re-engineer the process
	Diagnostic control systems	- Managerial attendance to Scrum ceremonies as team's observer - Short feedback loops to ensure work flow - Pressure inhibits team's ability from allocating time for leaning and innovation (i.e. Time Pressure)
	Interactive control systems	- Managerial beliefs reinforced through systematic interaction with scrum master - Increased distance between line manager and teams - Product Owner ensures right focus and timing of work
	Boundary control systems	- Product owner sets limits and rules on product quality - Product backing regulates what to do and usage of extra time
	Transformational leadership from Scrum roles	- Strong support for team's work flow and performances - Strong support for the process and

		improvements - Less support for technical implementation and product innovation
	Concertive control systems	- Social pressure within teams
Outcomes	Innovation	- Lack of strategic knowledge about how to incorporate innovation - Absence of product ideas within the team - No mechanisms to drive innovation

As example, Figure 4 shows the distribution of codes, stratified on the roles of interviewees, under Diagnostic Control Systems' sub-category. It demonstrates the importance of time pressure, which resulted to be the most frequent code across all the collected data.

Code co-occurrence models (Namey et al., 2007, p.145) were generated by MAXQDA®. When these models were run on the most critical codes, they gave an initial idea of the relationships with other codes and of potential relationships among the different subordinated code categories.

Figure 4: Code frequency-Diagnostic control systems

Next, we conducted *selective coding*. The most frequent codes related to team outcomes (perceived product quality, job satisfaction, perceived productivity and innovation) were chosen as initial core categories (Strauss and Corbin, 1990) combined with codes referring to team self - regulated learning processes. The aim was to identify the determinants of outcomes and self-regulated learning processes which were assumed to be fundamental to team behavior. We constructed models showing the relationships between the most relevant concepts linked to the selected core categories and the most significant codes belonging to subsidiary dimensions. The paradigm models were constructed based on:

- an overview of the coded segments linked to each selected code category, all the resulting clusters of codes that resulted directly involved were identified;
- for each selected cluster, reading the texts related to each individual code and to all the adjacent parts; this was done to detect the nature of relationships between the codes: causality, mediation, moderation;
- identified relationships collected for all clusters obtained from the «overview of coded segments» relationships among dimensions in the models were compared with the literature in order to validate and refine them.

In order to refine and validate the empirical models generated, the most influential codes appearing in the paradigm models were chosen as second step core categories and the process described above was repeated again.

The researchers followed strict rules in their methodological procedures, which means that the results can be considered reliable:

- data analysis started with repeated reading of all the data to get sense of the whole;
- word by word close reading of the data to identify codes or derive new ones;
- identification of codes was repeated twice from scratch, each time starting from a different perspective;
- in both the cases peer debriefing was conducted to reduce bias;
- triangulation of data from different organizational roles was attempted.

RESULTS AND DISCUSSION

We present the results of the cross-case analysis and discuss the data-driven model, resulting from the thematic analysis. We consider only

evidence related to the research questions in this paper. Core categories are aimed at determining and delimiting the attention mainly on the identified research questions. We include time pressure as construct to describe the social conduct within the team and its determinants.
Results were triangulated with the secondary data (i.e. the 121 free comments). They are displayed in the paradigm model describing the most common relationships among concepts emerging from the data.

Results

In this section, we first briefly define the constructs related to Time Pressure (Table 4) - which is the most critical code (as shown in Fig. 4) - illustrated by comments from Agile team members and other relevant organizational roles. We describe and discuss the mechanisms, managerial practices and work routines, related to the enactment of time pressure within the Agile team.

Table 4: Illustration of time pressure and related constructs emerging from the paradigm model

Construct definition	Illustration
Deadline pressure inhibits teams' from allocating time to learning and innovation	"*Some people are active to go to learning days and courses, others are less, but when the pressure gets up, and we feel that we are getting delayed the attendance to learning days is reduced. We need to plan properly time for learning*" [Operative Product Owner] *"They are struggled with that all the time because they're focused on delivering features all the time but they also see they need to help each other, the testers need to help with design as well and vice-versa but it's a challenge for them. I do have to keep reminding them during the sprint planning. They tend as default to get finished the feature"[high level*

	manager].
Absence of product improvement ideas since people are losing competences	"*I lost the overview of the product. In other case, I am not always aware about some updates, I can feel that my own inventiveness is going down due to this because I cannot see the problem before of coming up with solutions*" [Product Guardian]
Teams' competence build-up is based on the need to implement new features Despite management encouragement, low commitment towards learning and innovation	"*We learn only what it is needed for completing the tasks*" [Product Guardian] "*At the start of every sprint they [team members] decide which part of the user stories to implement, on many days it would take, which tasks implement, they write it up, sometimes meetings can take two hours, they plan who is going to pair up, they take away the time as well for learning in general but not so much, I still challenge them. They usually list everything, they list the capacity, they take away the learning days from the capacity and everything else from the capacity, then they match the two and what is left goes to the next sprint. The challenge is to encourage them to take more of their capacity to learning, they started to take some but the challenge is taking more. But it's very hard to do that after they have committed for the whole feature. That's the problem*" [Line Manager]
Lack of support for technical implementation and product innovation	*Before when we worked in the waterfall, we had a competence area were we were quite taught now we really don't know how to do, we need to ask a run or we need to find out by yourself, we don't have a real support. We have a lot of areas with no one supporting so it could take some time. You need to get into, it could take weeks.*[team member]
Knowledge broadening damages expertise	*"I'm learning a lot because of these new areas all the time to work with, but my feeling is that we are learning small pieces from all the places, you are not good in anything, you know a little bit of that and a little bit of that, you are not succeeding in being good*

	in anything, you are not expert in anything. This is the downside of Agile" .[team member]
Difficulty to recover the big picture of the product	"*Before we have more broad of the system, we were responsible of functions, if there was problem you have to solve all these problems. Now in Agile team as a systemist I'm limited because I have to focus on the features within the team*" [Team member]
Teams are unable to handle the wider scope of activity	*"Earlier we worked in a certain area, but now our features strike on every subsystem, and we don't have enough knowledge about it so it's difficult" [*Team member*]*
Boundary: product backlog regulated team's activities and extra time usage	"*In Agile we are in quite regular mode, working in a regular and constant time box, which is called sprint, three weeks long. At the beginning of each sprint we have half day meeting called sprint planning, where the Agile team members are looking at the sprint backlog. As team, we know our capacity and according to our estimation of it, we take items in the product backlog, pulling out user stories. Among the user stories we have also some bonus. It deals with normal work as a normal user story, but differently from it, it represents something for which team does not take a specific commitment to implement by the end of the sprint [...] It aims to full utilize the team's capacity if some spare time occurs*" [Scrum master]
Belief: learning and innovation are not priorities in feature development	"*My view about the main problem is that we need to be more efficient to produce more and then to be able to innovate. Becoming more efficient is a condition to have innovation in place, [...] it is hard to get things into the product because the demand is there but the capability was low*" [High level manager] *"Our team learning opportunity is not much, we have been working with 2 features at the same time, we had pressure for deliver those features and we don't have much time to dedicate to learning. It is not the*

	priority as developing and delivering features so we don't spend time on learning" [Team member]
Diagnostic: short feedback loops to maintain flow	"*I get updated info through several ways: weekly reports about the progress and then I have to report their progress to other forum like the release project*" [Product owner]
	"*Normally the teams have, according to the Scrum, meetings every day in the morning in which they discuss about the progresses, we try to be present to this meeting to listen what it is going on*" [Line manager]
Concertive: social pressure	*"Now we are working with new products and we have to learn how they work - at least so much that we can see where and how to do the implementations. But to really understand the product (to be able to make improvements) that takes time. Before you have knowledge about the product, now you even don't know if you can propose/do any improvements - spending time of investigation [sic] without any outcome isn't so popular I guess... Teams don't spend time on digging the product, people are just making features"* [Team Member]

The paradigm model in Figure 5 illustrates the pressure to 'get the job done' as an underexplored source of change in Agile context and show that pressure, emerging from day-to-day work, can spread to the whole organization and undermine organizational learning and innovation performance. It shows clear inter-relationship between the concepts tied to the Agile control systems and those tied to time pressure.

Figure 5*:* Paradigm model for Time Pressure.

In particular, we see that implementation of Agile introduces a complex system of controls, which, although different in nature, altogether

contribute to fostering product development activities, multiplying their effects and creating a feeling of time pressure within the Agile teams. The controls identified below are based on Simons' (1994) levers of control and Barker's (1993) concertive control systems.

Boundary control systems, defined as the formal system used by top management to establish obligatory limits and rules (Simons, 1994), in the Agile context, consist of complex Agile routines/ceremonies that apply to team members. Among these, product backlog seems to limit the team's freedom to allocate time to anything not clearly included in the specific time-period (or sprint). The following quotes are illustrative:

> *"[It is] very difficult to get time for competence development, very tight time schedule."*
>
> *"Transparency of the organization means that even small prioritization issues are quickly escalated. There are no buffers since product management keeps the teams 100% busy. This means that small additional tasks require involvement of product management for decisions."*

Belief control systems, or the formal systems used by top management to define, communicate and reinforce the organization's basic values (Simons, 1994). In Agile, this system is represented by team beliefs. These mirror the values transmitted through line management together with their social environment and represent the basic values, which should drive team behaviors and choices. In Agile, team members consider development of features not innovation and learning to be priorities. As a result, they prioritize project deadlines, which they feel adds pressure, but do not implement strategies to foster learning. One interviewee told us that:

> *"They [teams] have all the possible means to do that [reserve time for learning], so it's up to teams to really use them. They*

> *don't prioritize learning. We are still working for project releases that give us task deadlines, so there is no change on that aspect. We still have a deadline to meet and we have to deliver by that date, so that keeps the pressure up. That's why it's difficult for them to really plan time for learning."*

Diagnostic control systems, defined as formal feedback systems used to monitor team outcome and correct deviations from preset performance standards (Simons, 1994). They are represented by the short feedback loops in the Agile framework. Examples of feedbacks include daily standup meetings, continuous integration activities, demo and retrospective every three weeks, frequent meetings with product owners to track team progress, and the information radiators to constantly monitoring team competence. The presence of these short feedback loops ensures the correct focus and allocation of time to team activities but generates stress and pressure among team members. One informant said that:

> *"Concerning the stress you feel, in Agile the way of working is stressful. Management wants us to deliver code every day for testing, to find out if new code breaks legacy functionality. But the delivery process is not good enough. When people make mistakes, you have to roll back and many people are waiting for you. This way of working is not so effective. It should be modified somehow."*

Barker (1993) defines *concertive control systems* for self-regulating teams as normative controls that become restrictive for the individual team members, creating high levels of stress. The effect of concertive control means that people feel they are being watched and their contribution to team goals checked on. They feel unable to divert to activities not strictly related to those of other team members and the project. There is implicit

pressure to finish a task as soon as possible in order to start on the next one. The following statement highlights this situation:

> *"[y]ou have this tight control on what you are doing. As soon as you are ready, you go to the board and take a new task. So there is pressure to go through this kind of work packages as quickly as possible. And there is also "peer" pressure - if you are in a team, everyone knows what everyone else is doing."*

According to Li and co-authors (2010), Scrum increases stress and time pressure on team members by emphasizing a continuous functional delivery and inhibiting teams from performing other activities. This is clearly reflected in our findings.

Discussion

Our model reveals the mechanisms, managerial practices and work routines, that give rise to pressure and its effects on teams, and eventually the whole organization. Importantly, we do not deal with these mechanisms in isolation, but show their 'concatenation', underlining how the nature of the origin of pressure encourages distinct forms of theorization. We showed that these mechanisms are mutually reinforcing. By revealing these mechanisms, their combined effect, and recursive interactions, we have provided a more comprehensive model of an institutional change and forces that keep employees focused on the project performance at the expense of learning and innovation activities.

We found that the Scrum framework is a comprehensive system of controls, deriving from managerial practices and work routines, which includes micro-techniques to maintain discipline through normalized individual and collective actions. The first fundamental control, which emerged from our coding as contributing to time pressure is the team's identification with a set of values that guide their decision and work

activities. This is in line with Barker and Tompkins (1994). Through this process of identification, team members consider the organization's values and interest to be crucial factors in their choices (Tompkins and Cheney, 1983) and they orientate their behavior accordingly. Their conduct is governed only partly by their immediate work task and the work structure imposed on them. The team's value systems are the results of forms of discipline which are much more advanced than formal managerial hierarchies, and which are "*directed towards the soul, the mind and the will* (Burrell, 1988, p.225) and which conceive workers not as laboring bodies but as subjects whose values lies in their identities, knowledge and productivity (Rhodes and Garrick, 2002).

There is a clear link between the micro-techniques of discipline enacted by concertive control and team members' identification. Barker and Cheney (1994, p.30) pointed out that: "*Disciplinary mechanisms are perhaps most potent when they are grounded in highly motivating values that appeal to the organization's actors*" (p.30). This results in self-disciplining behaviors because workers believe that they are being watched and operate as if they are under surveillance.

Concertive control systems can be considered worker-maintained system of control. Regular events such as daily scrum meetings, which are central to this disciplinary apparatus, allow team members' behaviors to be observed by peers with the purpose to qualify, classify and punish behaviors and their normalization is induced. These regular meetings allow individuals to be continuously evaluated against established standards and rewarded for if they result to be significantly better than the developed norms.

Thus, constant surveillance and normalizing actions induce self-control from the organizational members affecting the distinct characteristics of their organizational conscience (Merton, 1968; Scott and Hart, 1979).

This finding extends research on the conditions affecting team innovative behavior in teams, suggesting that the effect of concertive control on

team members can prevent their investing effort in learning and innovation tasks if these are not recognized as core organizational values. Acceptance of organizational values by employees will limit their selection of choices, which will confirm their identification (Tompkins and Cheney, 1985) with organizational goals.

Concerning the role of diagnostic and boundary control systems within Scrum framework, they also impose a disciplinary system, which acts differently than social control. Based on the malleability of human behavior, these disciplinary micro-techniques are more in line with Foucault's (1977) concept of training people as involving people in disciplinary processes which create *"a set of habits and orientation that forms the initial object of psychological science*" (May, 1993, p.43).

The repetitiveness of Agile routines creates the conditions for what Foucault would describe as docile body. Individuals' actions become so standardized that they are performed automatically without any changes. The employee becomes mentally docile, does not design alternative ways to perform the task and performs actions without thinking through each step. Over time, these routines become internalized and the speed of performance expected becomes so natural that the team members risks to not become aware of external constraints. When the discipline linked to the team's task performance becomes internalized, then the employee self-regulates his or her behavior.

On the other hand, information and feedback systems, realized through the implementation of Scrum routines, favor the establishment of standards and discrepancy tolerance, which trigger the discrepancy-reduction process, involving possible modifications in team members' behaviors and in their interpretation of organization values and goals. In this sense, diagnostic and boundary control systems act to reinforce the team's perception of organizational values and team members may become more resilient to change (Bem, 1972).

New product development under time pressure implies a crisis prone situation. Organization and consequently self-managing teams under crisis need a huge amount of correct information and innovative ideas-real time in order to cope with the complexity of the situation. They need also to pool information, since no one individual can manage the amount of information needed to fully handle the situation (Pauchant and Mitroff, 1992). Teams with a centralized decision making mechanisms do not have the capacity to deal with the large amount of information they need to perform complex task, and transmission may misleads the information they do receive.

Previous research shows that organizations in crisis should decentralize to reduce these dysfunctions (Smart and Vertinsky, 1977). However, conversely, Perrow (1984) argues that in this situation they should centralize to enable a rapid and collective response. It would seem that firms in crisis require structure and processes that both secure access to large amount of high quality ideas and information to integrate them so as to inform collective action.

Based on the empirically derived model and on the theoretical implication highlighted above we predict that:

Proposition 1: Diagnostic and boundary control systems, in combination with team identification and concertive controls, have a reinforcing effect on teams' perceived time pressure

Proposition 2: A combination of team identification and concertive control negatively influences a team's learning and innovation, unless legitimized by the team's value systems.

Proposition 3: A combination of diagnostic and boundary control systems negatively influences a team's learning and innovation, unless they are implemented through routinized behaviors.

Managerial implications

Our results focus on two important implications for managerial practice: (1) the variety of interrelated controls upon which team's behavioral conduct rely; (2) the roles of managers and the management function in driving teams since all the highlighted controls are organizationally induced. These two aspects are intimately intertwined.

Managers implementing this new approach to work may believe that they are empowering individuals giving teams autonomy of behavior but, instead, they are realizing surveillance over individuals, even if they are not actively seeking it. In fact, we showed the existence of a direct intentional link between controls and surveillance.

As for (1): managerial controls alone would never be able to realize the surveillance, but the gaze of peers may be difficult to sustain especially when interdependence among team members is high, given the way the work is organized in Agile with a common backlog of activities to collectively share. So it is the concatenation between horizontal controls (team identification with organization values and concertive control) and the vertical managerial controls (diagnostic and boundary control systems) that creates conditions for monopolizing team attention and their knowledge behind a feeling of time pressure in a way to reduce the incidence of unproductive activities.

As for (2): managers should identify ways to reduce even partially the negative effects of this concert of controls. The propositions formulated above offer a means to realize this attempt suggesting to develop a combination of commitments within teams with a balanced set of organizational values and of organizational routines fostering also learning and innovation.

CONCLUSIONS AND LIMITATIONS

This study contributes to the understanding of the effects of an Agile social context on team learning and innovation

Results highlight the importance of managerial practices and work routines, derived from the implementation of Scrum, which serve as efficiency carriers, and thereby contribute to the achievement of several aspects of team performance. Specifically results underline the role of formal and informal controls as a prerequisite for the team's self-regulating learning strategy and innovation. They confirm also that team innovation and self-regulated learning depend greatly on the team's internal pressure. A mix of organizational control systems contributes to increased monitoring of team members' behaviors, generation of behavioral alternatives within the team, and multiple interpretations of external input, all of which are crucial for innovation and learning.

This study complements previous work by highlighting the determinants of team innovation and learning, and the importance of the interaction processes.

Our results indicate that, although formal and informal controls are important, the interaction processes leading to time pressure on team members outweigh the managerial controls in predicting innovation.

We built an empirically based model framework using an abductive approach. This methodology allowed combining of empirical and case study evidence with the theory. Our matching and analytical phases allowed us to formulate some propositions. Although we do not test these propositions, our investigation of four R&D organizations shows that this framework could explain how different types of controls resulting from the implementation of Scrum might justify the widespread perception among Agile teams of time pressure. A large-scale survey is needed to test and validate the proposed model. The applicability of our results to other contexts needs to be verified. Although our sample was homogeneous across the four R&D organizations, the single organizational context may affect the generalizability of our findings. Future research should consider teams from multiple firms and examine additional types of team activities, such as services or less knowledge-intensive activities. This would allow a

better understanding of how the implementation of Scrum methods can affect team innovation performance and absorptive capability. The present study represents a step in this direction, but much work is needed to increase our understanding of the management of organizational learning and innovation in an Agile context.

REFERENCES

Abrahamsson, P., Conboy, K., and Wang, X. (2009). 'Lots done, more to do': the current state of Agile systems development research. *European Journal of Information Systems*, 18, 281-284.

Aalbers H.L., Dolfsma, W. and Koppius, O. (2013). Individual connectedness in innovation networks: On the role of individual motivation. *Research Policy*, 42(3), 624-634.

Argyris, C. (1976). Single-loop and double-loop models in research on decision making. *Administrative Science Quarterly*, 21(3), 363-375.

Argyris, C. (1995). Action science and organizational learning. *Journal of Managerial Psychology*, 10(6), 20-26.

Baer M., Leenders, R.T.A., Oldham, G.R., and Vadera, A.K. (2010). Win or lose the battle for creativity: The power and perils of intergroup competition. *Academy of Management Journal*, 53(4), 827-845.

Barker, J.R. (1993). Tightening the iron cage: Concertive control in self-managing teams. *Administrative Science Quarterly*, 38(3), 408-437.

Barker, J.R. (1999). *The discipline of teamwork: Participation and concertive control*. Newbury Park: Sage.

Barker, J.R., and Cheney, G. (1994). The concept and the practices of discipline in contemporary organizational life. *Communications Monographs*, 61(1), 19-43.

Barker, J.R., and Tompkins, K.P. (1994). Identification in the Self-Managing Organization Characteristics of Target and Tenure. *Human Communication Research*, 21(2), 223-240.

Baskerville, R.L. (2006). Artful Planning. *European Journal of Information Systems*, 15(2), 113-115.

Beck, K. et al. (2001). Manifesto for Agile Software Development. Online: http://www.agilemanifesto.org/ (Accessed on 15 October 2014).

Beck, K., and Andres, C. (2005). *Extreme Programming Explained: Embrace Change*, Boston: Addison-Wesley.

Beedle, M., Devos, M., Sharon, Y., Schwaber, K., and Sutherland, J. (1999). Scrum: a pattern language for hyperproductive software development. In *Pattern Languages of Program Design*, N. Harrison (Ed.), pp.637-651. Boston: Addison-Wesley.

Bem, D.J. (1972). Self-perception theory. In L. Berkowitz (Ed.), *Advances in Experimental Social Psychology*, pp.186-199. New York: Academic Press.

Boehm, B.W., and Turner, R. (2004). B*alancing Agility and Discipline: A Guide for the Perplexed*. Boston: Addison-Wesley.

Brief, A.P., and Hollenbeck, J.R. (1985). An exploratory study of self-regulating activities and their effects on job performance. *Journal of Occupational Behavior*, 6(3), 197-208.

Annosi M.C., Magusson M., Martini A., Appio F.P. (2015) "*Social conduct, learning and innovation: an abductive study of the dark side of agile software development*", Creativity and Innovation Management, forthcoming

Burrell, G. (1988). Modernism, post modernism and organizational analysis 2: The contribution of Michel Foucault. *Organization Studies*, 9(2), 221-235.

Clegg, S.R., Pitsis, T.S., Rura-Polley, T., and Marosszeky, M. (2002). Governmentality matters: designing an alliance culture of inter-organizational collaboration for managing projects. *Organization Studies*, 23(3), 317-337.

Cockburn, A. (2007). *Agile Software Development: the cooperative game*. Boston: Addison-Wesley.

Cohen, S.G., and Bailey, D.E. (1997). What makes teams work: Group effectiveness research from the shop floor to the executive suite. *Journal of Management*, 23(3), 239-290.

Cohn, M. (2009). *Succeeding with Agile: software development using Scrum*. Pearson Education.

Collyer, S., Warren, C., Hemsley, B., and Stevens, C. (2010). Aim, fire, aim - Project planning styles in dynamic environments. *Project Management Journal*, 41(4), 108-121.

Conboy, K. (2009). Agility from first principles: reconstructing the concept of agility in information systems development. *Information Systems Research*, 20(3), 329-354.

Deetz, S. (1998). Discursive formations, strategized subordination and self-surveillance. In A., McKinlay and K. Starkey (Eds.), *Foucault, Management and Organization Theory*, pp.151-172. London: Sage.

Dingsøyr, T., Nerur, S., Balikepally, V., and Moe, N.E. (2012). A decade of agile methodologies: towards explaining agile software development. *Journal of Systems and Software*, 85(6), 1213-1221.

Dingsøyr, T., and Smite, D. (2014). Managing Knowledge in Global Software Development Projects. *IT Professional Magazine*, 16(1), 22-29.

Draucker, C.B., Martsolf, D.S., Ross, R., and Rusk, T.B. (2007). Theoretical sampling and category development in grounded theory. *Qualitative Health Research*, 17(8), 1137-1148.

Druskat, V.U., and Kayes, D.C. (2000). Learning versus performance in short term project teams. *Small Groups Research*, 31(3), 328-353.

Dubois, A., and Gadde, L. E. (2002). Systematic combining: an abductive approach to case research. *Journal of Business Research*, 55(7), 553-560.

Dubois, A., and Gadde, L. E. (2014). "Systematic combining" - a decade later. *Journal of Business Research*, 67(6), 1277-1284.

Dybå, T., and Dingsøyr, T. (2008). Empirical studies of Agile software development: A systematic review. *Information and Software Technology*, 50(9), 833-859.

Erickson, J., Lyytinen, K., and Siau, K. (2005). Agile modeling, agile software development, and extreme programming. *Journal of Database Management*, 16(4), 88-100.

Foucault, M. (1977). *Discipline and punish: The birth of the prison*. Vintage Books.

Fruhling, A., and De Vreede, G.-J. (2006). Field experiences with eXtreme programming: developing an emergency response system. *Journal of Management Information Systems*, 22(4), 39-68.

Glaser, B.G. (1992). *Emergence vs forcing: Basics of grounded theory analysis*. Sociology Press.

Greenwald, A.G. (1982). Ego task analysis: An integration of research on ego-involvement and self-awareness. In A. Hastorf, and A. Isen (Eds.), *Cognitive Social Psychology*, pp.109-147. Amesterdam: Elsevier.

Annosi M.C., Magusson M., Martini A., Appio F.P. (2015) "*Social conduct, learning and innovation: an abductive study of the dark side of agile software development*", Creativity and Innovation Management, forthcoming

Hackman, J. R. (1992). Group influences on individuals in organizations. In M.D. Dunnette, and L.M. Hough (Eds.), *Handbook of industrial and organizational psychology* Vol. 3, pp.199-267. Palo Alto: Consulting Psychologist Press.

Harrison S.H., and Rouse, E.D. (2014). Let's dance! elastic coordination in creative group work: A qualitative study of modern dancers. *Academy of Management Journal*, 57(5), 1256-1283.

Hardt, M., and Negri, A. (2004). Moltitudine. *Guerra e democrazia nel nuovo ordine imperiale.* Milano: Rizzoli.

Highsmith, J.A. (2002). *Agile Software Development Ecosystems*. Boston: Addison-Wesley.

Highsmith, J.A., and Cockburn, A. (2001). Agile software development: The business of innovation. *Computer*, 34(9), 120-127.

Imai, K., Nonaka, I., and Takeuchi, H. (1984). *Managing the new product development process: how Japanese companies learn and unlearn.* In R. Hayes, K. Clark, and C. Lorenz ∗Eds.), *The Uneasy Alliance: Managing the Productivity-Technology Dilemma*. Boston: Harvard Business School Press.

Larman, C. (2004). *Agile and Iterative Development: A Manager's Guide*. Boston: Addison-Wesley.

Latham, G.P., and Locke, E.A. (1991). Self-regulation through goal-setting. *Organizational Behaviour and Human Decision Processes*, 50(2), 212-247.

Lee, G., and Xia, W. (2010). Toward Agile: an integrated analysis of quantitative and qualitative field data. *MIS Quarterly*, 34(1), 87-114.

LePine, J.A., and Van Dyne, L. (2001). Voice and cooperative behavior as contrasting forms of contextual performance: evidence of differential relationships with big five personality characteristics and cognitive ability. *Journal of Applied Psychology*, 86(2), 326-336.

Li, J., Moe, N. B., and Dybå, T. (2010). Transition from a plan-driven process to Scrum: a longitudinal case study on software quality. *Proceedings of the 2010 ACM-IEEE international symposium on empirical software engineering and measurement*.

Lindvall, M., Basili, V., Boehm, B., Costa, P., Dangle, K., Shull, F., Tesoriero, R., Williams, L., and Zelkowitz, M. (2002). Empirical findings in Agile methods. *Proceedings of the XP/Agile University 2002; Second XP Universe and First Agile Universe Conference*, Chicago: Springer-Verlag.

Lumpkin, G.T., and Lichtenstein, B.B. (2005). The role of organizational learning in the opportunity-recognition process. *Entrepreneurship Theory and Practice*, 29(4), 451-472

Lyytinen, K., and Rose, G.M. (2006). Information System Development Agility as Organizational Learning. *European Journal of Information Systems*, 15(2), 183-199.

Magazinius, A., and Feldt, R. (2011). Confirming distortional behaviors in software cost estimation practice. *Proceedings of the 37th EUROCMICRO Conference on Software Engineering and Advanced Applications (SEAA*), pp.411-418.

Merton, R.K. (1968). *Social theory and social structure*. Simon and Schuster.

Moe, N.B., Dingsøyr, T., and Dybå, T. (2008). Understanding self-organizing teams in agile software development. *Proceedings of the Australian Software Engineering Conference* - ASWEC.

Namey, E., Guest, G., Thairu, L., and Johnson, L. (2007). Data reduction techniques for large qualitative data sets. In G., Guest and M. MacQueen (Eds.), *Handbook for team-based qualitative research*, pp.137-162. Lanham, MD: Rowman and Littlefield.

Annosi M.C., Magusson M., Martini A., Appio F.P. (2015) "*Social conduct, learning and innovation: an abductive study of the dark side of agile software development*", Creativity and Innovation Management, forthcoming

Nerur, S., and Balijepally, V. (2007). Theoretical reflections on Agile development methodologies. *Communications of the ACM*, 50(3), 79-83.

Nerur, S., Mahapatra, R., and Mangalaraj, G. (2005). Challenges of migrating to Agile methodologies. *Communications of the ACM*, 48(5), 72-78.

Nonaka, I. (1994). A dynamic theory of organizational knowledge creation. *Organization science,* 5(1), 14-37.

Patton, M. (1990). *Purposeful sampling. Qualitative Evaluation and Research Methods*. London: Sage, pp.169-183.

Pauchant, T.C., and Mitroff, I.I. (1992). *Transforming the crisis-prone organization: Preventing individual, organizational, and environmental tragedies*. Jossey-Bass.

Peirce, C. (1903). *The essential Peirce: Selected philosophical writings*, Vol. 2. Bloomington: Indiana University Press.

Perrow, C. (1984). *Normal accidents: Living with high risk technologies*. Princeton: Princeton University Press.

Rhodes, C., and Garrick, J. (2002). Economic metaphors and working knowledge: enter the 'cogito-economic' subject. *Human Resource Development International*, 5(1), 87-97.

Ryan, R.M., and Deci, E.L. (2000). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American Psychologist*, 55(1), 68-78.

Ryan, G.W., and Bernard, H.R. (2000). *Data management and analysis methods.* In N., Densin, and Y., Lincoln (Eds.), *Handbook of Qualitative Research*, pp.769-802. Thousand Oaks, CA: Sage Publications.

Shalley C.E., Zhou, J., and Oldham, G.R. (2004). The effects of personal and contextual characteristics on creativity: Where should we go from here? *Journal of Management*, 30(6), 933-958.

Schatzman, L., and Strauss, A. L. (1973). *Field research: Strategies for a natural sociology*. Englewood Cliffs, NJ: Prentice-Hall.

Schwaber, K. (1997). *Scrum development process*. In J.V., Sutherland, D., Patel, C. Casanave, J., Miller, and G. Hollowell (Eds.), *Business Object Design and Implementation*, pp.117-134. London: Springer.

Schwaber, K. (2004). *Agile Project Management with Scrum*. Microsoft Press.

Schwaber, K., and Beedle, M. (2002). *Agile Software Development with Scrum*. Upper Saddle River, NJ: Prentice-Hall.

Scott, W.G., and Hart, D.K. (1979). *Organizational America: Can individual freedom survive within the security it promises*. Boston, MA: Houghton-Mifflin.

Shin S.J., Kim, T., Lee, J., and Bian, L. (2012). Cognitive team diversity and individual team member creativity: A cross-level interaction. *Academy of Management Journal*, 55(1), 197-212.

Schön, D. (1983). *The reflective practitioner*. New York: Basic Books.

Simon, H.A. (1967). Motivational and emotional controls of cognition. *Psychological review*, 74(1), 29-39.

Simons, R. (1991). Strategic orientation and top management attention to control systems. *Strategic Management Journal*, 12(1), 49-62.

Simons, R. (1994). How new top managers use control systems as levers of strategic renewal. *Strategic Management Journal*, 15(3), 169-189.

Annosi M.C., Magusson M., Martini A., Appio F.P. (2015) "*Social conduct, learning and innovation: an abductive study of the dark side of agile software development*", Creativity and Innovation Management, forthcoming

Smart, C., and Vertinsky, I. (1977). Designs for crisis decision units. *Administrative Science Quarterly*, 22(4), 640-657.

Strauss, A.L., and Corbin, J. (1990). *Basics of qualitative research*, Vol. 15. Newbury Park, CA: Sage.

Suddaby, R. (2006). From the editors: What grounded theory is not. *Academy of Management Journal*, 49(4), 633-642.

Sutherland, J. (2010). The roots of Scrum. How the Japanese experience changed global software development. Online: http://www.gbcacm.org/sites/www.gbcacm.org/files/slides/5%20-%20Roots%20of%20Scrum.pdf (Accessed on 20 February 2014).

Takeuchi, H., and Nonaka, I. (1986). The new new product development game. *Harvard business review*, 64(1), 137-146.

Van Maanen, J., Sörensson, J., and Mitchell, T. (2007). The interplay between theory and method. *Academy of Management Review*, 32(4), 1145-1154.

Vidgen, R., and Wang, X. (2009). Coevolving Systems and the Organization of Agile Software Development. *Information Systems Research*, 20(3), 355-376.

Williams, L., and Cockburn, A. (2003). Guest Editors' Introduction: Agile Software Development: it's about feedback and change. *Computer*, 36(6), 39-43.

Woodside, A. (2010). Bridging the chasm between survey and case study research: Research methods for achieving generalization, accuracy and complexity. *Industrial Marketing Management*, 39(1), 64-75.

Yeo, R. (2002). From individual to team learning: practical perspectives on the learning organisation. *Team Performance Management: An International Journal*, 8(7/8), 157-170.

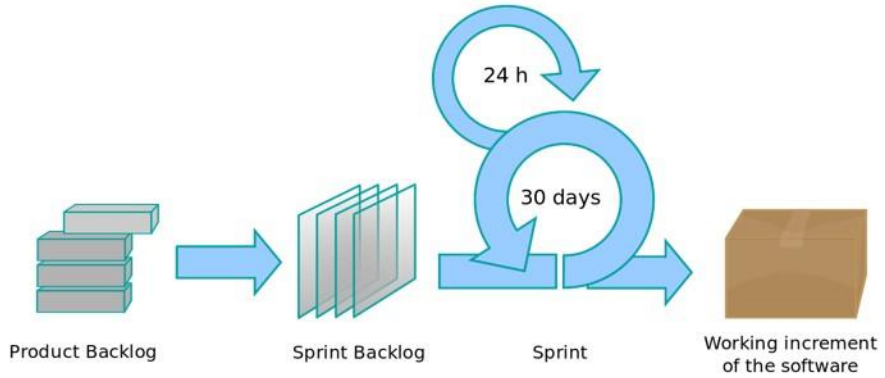
24 h
30 days
Product Backlog
Sprint Backlog
Sprint
Working increment of the software

Fig. 1

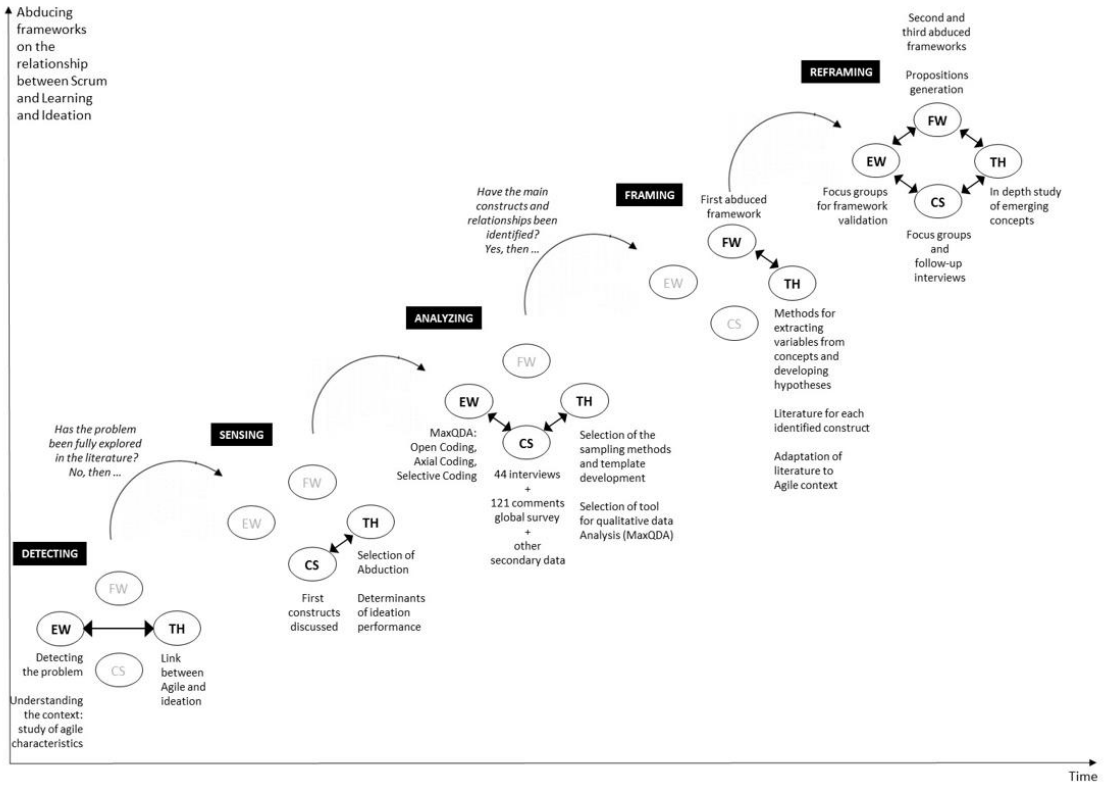
Abducing frameworks on the relationship between Scrum and Learning and Ideation
Time
DETECTING
Understanding the context: study of agile characteristics
EW
Detecting the problem
TH
Link between Agile and ideation
CS
FW
Has the problem been fully explored in the literature? No, then ...
SENSING
EW
FW
CS
First constructs discussed
TH
Selection of Abduction
Determinants of ideation performance
ANALYZING
Have the main constructs and relationships been identified? Yes, then ...
EW
MaxQDA: Open Coding, Axial Coding, Selective Coding
CS
44 interviews + 121 comments global survey + other secondary data
TH
Selection of the sampling methods and template development
Selection of tool for qualitative data Analysis (MaxQDA)
FW
FRAMING
EW
CS
FW
First abduced framework
TH
Methods for extracting variables from concepts and developing hypotheses
Literature for each identified construct
Adaptation of literature to Agile context
REFRAMING
Second and third abduced frameworks
Propositions generation
FW
EW
Focus groups for framework validation
TH
In depth study of emerging concepts
CS
Focus groups and follow-up interviews

Fig. 2

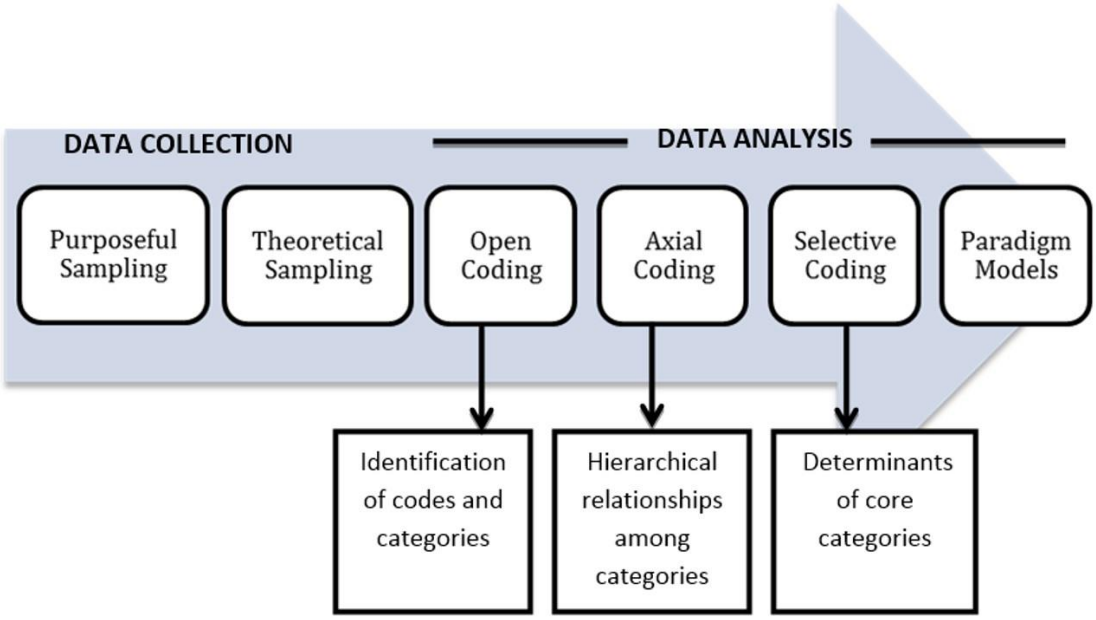
DATA COLLECTION
DATA ANALYSIS
Purposeful Sampling
Theoretical Sampling
Open Coding
Axial Coding
Selective Coding
Paradigm Models
Identification of codes and categories
Hierarchical relationships among categories
Determinants of core categories

Fig. 3

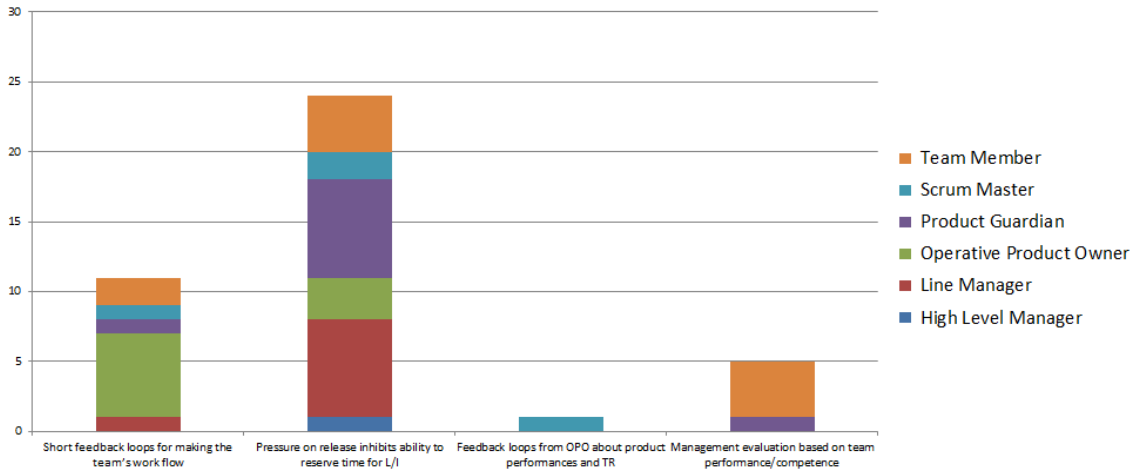
30
25
20
15
10
5
0
Short feedback loops for making the team's work flow
Pressure on release inhibits ability to reserve time for L/I
Feedback loops from OPO about product performances and TR
Management evaluation based on team performance/competence
Team Member
Scrum Master
Product Guardian
Operative Product Owner
Line Manager
High Level Manager

Fig. 4

Annosi M.C., Magusson M., Martini A., Appio F.P. (2015) "*Social conduct, learning and innovation: an abductive study of the dark side of agile software development*", Creativity and Innovation Management, forthcoming

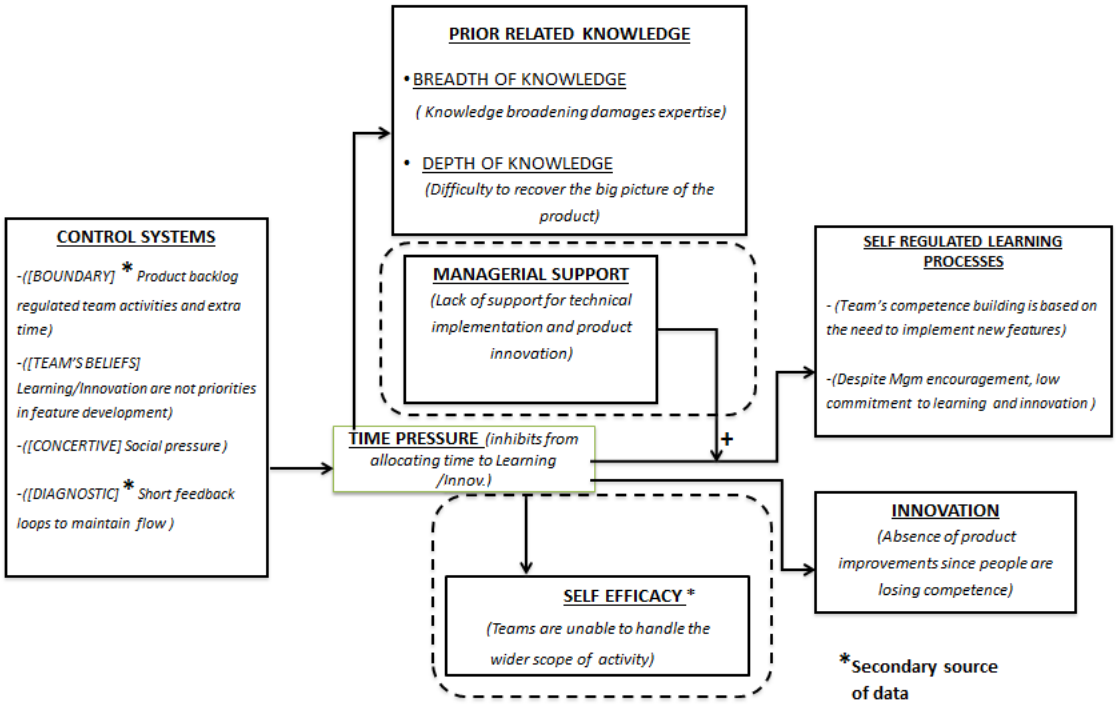

Fig. 5