Why Scrum works

A case study from an agile distributed project in Denmark and India

Lene Pries-Heje
Department of Informatics
Copenhagen Business School
Frederiksberg, Denmark
e-mail: lph.inf@cbs.dk

Jan Pries-Heje
Department of Communication, Business & IT (CBIT)
Roskilde University
Roskilde, Denmark
e-mail: janph@ruc.dk

***Abstract*—Scrum seems to work extremely well as an agile project management approach. An obvious question is why. To answer that question, we carried out a longitudinal case study of a distributed project using Scrum across Denmark and India. In our analysis of case study data we used three selected theoretical frameworks. We conclude that Scrum works so well because it provides communication, social integration, control, and coordination mechanisms that are especially useful for distributed and agile project management.**

Keywords – Distributed; Agile; Project management; Scrum; Boundary objects, Social capital, Articulation Theory

I. INTRODUCTION

Driven by increasing time-to-market pressure as well as cost saving incentives, more and more software projects are becoming global [cf. 1] with project participants distributed in different places. This means that work is being done by anyone who does it better, cheaper, or faster. It also means that a company may have many projects characterized by rapidly assembled project teams, geographically dispersed, but with highly specialized professionals who perform specific projects. Thus, distributed teams and distributed projects will be very common in the future.

Software development includes activities such as analysis, design, coding and testing. A project management approach prescribes processes for carrying out management of software development activities. In the mid 1990s software development approaches were effective in large-scale, long-term development efforts that employed stable and disciplined processes [2]. In contrast, many software projects involved rapid changes in requirements and unpredictable complexity. Software development approaches achieving a balance between flexibility and discipline were needed [3, 4] – and thus *agile approaches* were invented.

These approaches have received much attention from both the practitioner and researcher community over the last 10-15 years - first as a novelty and later as a development approach that has become widely used in practice [5]. Agile approaches are often referred to as high-speed development [6], an approach for dealing with change [7] characterized by iterative processes [8]. A prominent example of agile processes is Scrum [9].

Today, more and more companies are adopting Scrum for agile software project management. In Denmark, the authors have regularly been teaching professional IT project management courses, and recently – since 2008 – it has been obvious that more and more people know Scrum and are working in organizations using Scrum or at least parts of Scrum. Out of 100 project managers, approximately 1/4 were from organizations using Scrum in 2009, and in 2010 the number had risen to 1/3. These numbers are based on "raise-your-hand if you use it." Nevertheless, the authors believe that these numbers give a clear indication that Scrum has gained surprising momentum recently.

Based on the trend towards distributed and global software development and the fact that Scrum seems to work extremely well as an agile project management approach, we phrase the obvious question, *"Why does Scrum work?"* as our research question.

A. SCRUM

Scrum was first described as *The Rugby Approach* in Harvard Business Review [10], emphasizing that small cross-functional teams produce better results. Jeff Sutherland and Ken Schwaber used the Scrum approach in their companies in the mid-1990s, and they worked together to formalize the Scrum approach [11]. Jeff Sutherland continued to study distributed projects using Scrum [cf. 11, 12].

In short, Scrum is an iterative approach where an iteration lasting two to four weeks is called a *Sprint*. The wished-for functionality is written as *User Stories* and prioritized in a *Product Backlog* by a *Product Owner* representing the customer view. The highest prioritized functionality becomes the target functionality for a Sprint. The Sprint starts with a *Sprint Planning Meeting* where the targeted functionality is broken down and estimated (sized). Every day the project team meets in a *Daily Stand-up*

Meeting that takes place in front of a *Scrum Board*. The daily meeting is not supposed to take more than 15 minutes and is chaired by a *Scrum Master* responsible for the process. During the meeting every team member answers four questions. (1) What did you do yesterday? (2) What are you doing today? (3) Problems encountered? (4) Innovations? They work on four columns. The Scrum Board has estimated (sized) tasks in the first column, and the tasks in progress are in the second column – that is where team members move tasks that they have taken on. When a team member believes a task is finished it is moved to the third column called *Done*, and when another team member has quality assured the task it is moved to the fourth column called *Done Done*. Finally, when a task is finished it is registered on a *Burn-Down Chart* where you can see expected versus realized production.

After the two to four week iteration, the Sprint produces a deliverable of value to the customer(s) represented by the Product Owner. In the concrete, the produced functionality is demonstrated to the Product Owner who then recognizes the value of the deliverable. Last but not least, the team looks back and carries out a *Retrospective* where they try to learn from the Sprint just completed: What worked for us? What did not work? What changes will we implement in the next Sprint?

B. The challenges of distributed projects

We define a distributed project as one where the project team is separated by geography, time zones and/or culture, but nevertheless has to work together as a team.

Distributed projects will typically - as traditional projects – have a project manager. The majority of project management work will be the same in a traditional or a distributed project. But some things will be different or more challenging when managing distributed projects. The question is of course: What will be different?

Project management knowledge has recently been gathered and presented by two organizations worldwide. One is the Project Management Institute (PMI) [13]. They have published a book of knowledge with nine areas, each containing the processes that together make it possible to achieve effective project management. Analyzing these areas for a distributed project, we find that: (1) it becomes harder to obtain a common understanding of the Project Scope. (2) Human resource management becomes harder, as you cannot "pat your people on the shoulder" when they are doing a good job. (3) It is difficult to communicate at a distance. Building trust at a distance is a further issue. (4) More focus on early and fixed work breakdown structure (WBS) is needed to be able to distribute work to distributed team members. (5) As quality is the fulfillment of expectations and because expectations may be culturally influenced and therefore different, quality may be harder to manage.

The other organization is the International Project Management Association (IPMA) who published a so-called Competence Baseline [14] for project management. In addition to the challenges in distributed projects identified above, it is mainly the behavioral competences that need to be adapted. Especially: (1) Spreading energy and enthusiasm [14, p.90ff "Engagement and Motivation" competence] at a distance and possibly through some computer-supported media is more difficult. (2) Teambuilding, developing a team spirit [14, p.52], and building personal relations and networks can be a lot more difficult if people are not physically together. (3) "Help ensure effective communications with the project team …" [14, p. 106] will be more challenging when communication needs reach across distances and cultures.

In general, studies of distributed and global software development have shown that what you need to deal with are the three "C"s: communication, coordination and control [15-17]. We have answered the research question of why Scrum works by analyzing a longitudinal case study of a distributed software development project using Scrum to ensure in-depth knowledge.

The remainder of the paper is organized as follows. We describe the study and our analysis in section II. Then follows sections III - VII accounting for our findings in three areas: social team capital, boundary objects, and articulation theory, the three areas where our analysis reveals that Scrum works and provides value in distributed projects. Plus two sections on Control, social integration and motivation. Finally, our conclusion in section VII lists the nine answers we found that together explain why Scrum works.

II. RESEARCH METHOD

To obtain detailed qualitative data for analysis and to answer our research question of why Scrum works, we looked for a case of software development that we could follow in detail and over a period of time.

Danske Bank gave us the opportunity to follow a distributed project team using Scrum. Group IT of Danske Bank, has approximately 2500 IT people employed, of which 80% is located in Denmark and 20% is located in India. Danske Bank uses a staff augmentation strategy for outsourcing meaning in which most projects have team members from both Denmark and India. In India, the team members are employed by an Indian company, ITC Infotech, but they work for the Danske Bank account.

We found a project using the agile method Scrum with participants distributed across the two countries: Seven in a Scrum team in Denmark and eight in a Scrum team in India. For ease of reference, we call the project *DELHI*.

The DELHI project started in May 2010. We interviewed the Project Manager for the first time right from the beginning of the project when he was appointed project manager. Shortly after the first interview the project manager went to India and initiated a Scrum team there. The set-up in the project as a whole consisted of two Scrum-teams, one in Denmark and one in India, with daily *Scrum-of-Scrums* meetings where the Project Manager and the two Scrum Masters met virtually, either by a phone conference or using

a tool called e-meeting. As part of setting up the Indian Scrum team, the Danish Project Manager planned a one-week visit to India where the purpose was partly to diffuse knowledge and skills on Scrum, and partly to do some classic team building.

Our first interview guide - used in the first few interviews - was based on our conceptual analysis of project management knowledge areas [13] and competence [14] approaches. We used an open interview guide asking about the use of Scrum, communication, coordination and control [cf. 15]. We interviewed the project manager before and after his visit to India. The 'after' interview focused on whether his expectations and plans had been fulfilled.

At this point, we gathered our data and analyzed them applying some coding procedures from Grounded Theory - an inherently flexible methodology in which the researcher "should simply code and analyze categories and properties with theoretical codes which will emerge and generate their complex theory of a complex world"[18]. We used the Strauss and Corbin [19] school of thought where GT analysis is composed of three groups of coding procedures called open, axial and selective coding. Open coding reveals the core ideas found in the data by labeling phenomena and discovering categories. Axial coding develops a deeper understanding of how the identified categories are related. And finally, selective coding involves the integration of the categories that have been developed to form the initial theoretical framework.

When coding, we came up with categories such as: social integration, common vision, common language, networking, and building trust. We soon realized that a lot of what we were finding could be captured by the concept of *Social Capital* [20] which has three dimensions, namely, structural, relational, and cognitive. The *structural* dimension equals the relationships possessed by group members; the *relational* dimension is about nature and quality of the relations; and the *cognitive* dimension concerns shared language and meaning [20].

A. The second round of interviews and a visit to India

Based on our first round of coding, we decided to change our interview guide to include social capital. In doing this, we used a framework by Evans & Carson [21] that uses couple social capital as a moderator for "three common group processes for group effectiveness: communication, social integration, and coordination" – two of the three "C"s we had in our first interview guide. We interviewed the project manager three times: after the first two Sprints had been finished, after something had been released to the customer, and after the project was finished end of March 2011.

Furthermore, in November 2010, we went to the DELHI site in India for three weeks. Here, we interviewed the Indian part of the team (Scrum Master, Task Manager, Business Developer, IT Developer, and Professional Tester) for about an hour each. We also observed daily Scrum meetings, the use of the Scrum Board, a daily *Scrum-of-Scrums*, and a so-called '*All Hands*' meeting where all team members from both Denmark and India were present in a video conference towards the end of a Sprint.

For both Danish and Indian second round interviews, we used a semi-structured interview guide with the following content:

- Your background: Education / Experience?
- The project? Scope? / Organization? / Roles?
- Denmark – India? People on-site in Denmark? / You? / Use of Liaison Officers?
- Communication: What? / Daily? / Regularly? / Challenges? / Hindrances & obstacles?
- Teambuilding: How do you build team? Education / Experience
- Networking: Here? / Across to DK? / Community here at ITC? Bangalore?
- Project Management, Coordination and Control: Formal control and follow-up? / Common vision? / Informal? / Coordination Mechanisms?
- Examples: Surprises? Special & different about working here in general?

B. Coding and categorizing

The research methodology we adopted in analyzing our data after the second round of interviews can be described as a contextualized, interpretive one, using the technique of case study research [22, 23]. Our research can be characterized as being interpretive research in that we attempted to understand the distributed project using the agile Scrum "phenomena" and the problems therein through the meanings that people assigned to the issues we brought up in the interviews. Thus, our access to reality is through social constructions, such as language, consciousness, and shared meanings [24].

Our data analysis followed the interpretive tradition, using hermeneutics [25]. Interview minutes and observation documents were coded and analyzed. Careful qualitative data analysis [26] uncovered a number of underlying themes: one being social capital, but others were communication, coordination and control, as well as motivation and embedded quality assurance. In the following sections III – VII we give an account of our findings.

III. CASE ANALYSIS AND SOCIAL TEAM CAPITAL

The concept of Social capital is relatively new and is an attempt to bring together a number of concepts such as informal organization, trust, culture, social support, social exchange, social resources, embeddedness, rational contracts, social networks, and inter-firm networks [27]. Social capital has three dimensions, namely, a structural, relational, and cognitive dimension [21, 27].

A. *The structural dimension*

The structural dimension involves the network of ties and relationships possessed by individuals. When a new project is established all team members enter the project with a social network developed prior to the project, and they continue to develop their network as the project is progressing. Thus, when analyzing the DELHI project, the primary focus was on the development of social ties between participants within the project and secondarily on network ties to individuals outside the project team, but within the two organizations.

When the DELHI project started in May 2010 two completely new Scrum teams were established, one in Denmark and one in India. A Project Manager for the project and its two Scrum teams was appointed, namely, an experienced Project Manager with some prior experience managing Scrum projects.

As colleges of the same affiliation, the team members in Denmark had some prior knowledge about each other and some common work experience, although working in a Scrum team was new to them. Hence, weak network ties within the Danish team existed from the outset. The Indian team members, however, had very limited knowledge about each other and no experience working together, thus, virtually no network ties existed within the team prior to the project.

The Indian Scrum Master and the Project Manager knew each other beforehand, as the Indian Scrum Master had worked 1½ years in Danske Bank in Århus, part of the time working on a project with the Project Manager. From the outset, the only tie between the Danish and Indian Scrum teams was the relation between the Project Manager and the Indian Scrum Master.

A project kickoff was performed in India (partly as a Scrum sprint exercise) with participation of the Indian Scrum team, the Project Manager, and the Danish Scrum Master, which helped develop network ties within the Indian team, as well as ties between the Indian team members and the Project Manager and the Danish Scrum Master. During the interviews the participants explained that this event was very important for developing ties among the participants. As the project progress ties were strengthened, the ties between individuals in Denmark and India were developed. The Business developer in the Indian Scrum team explained in an interview "I [now] have a virtual network in Denmark, and I know that Morten [in Denmark] is a 'Viking' [winter bather] in his spare time." In the final interview with the Project Manager he commented, "Strong ties have been developed within each Scrum team and the ties between the teams are well developed." He further explained that an Indian software developer from the Indian Scrum team (who never met face-to-face with his Danish colleges) at a recent visit to Denmark walked around the organization with small chocolate gifts and talked to the people he knew – at a distance from the project.

As we see it, Scrum contributed to the development (virtual) network ties because it more or less prescribed an internal network within each of the distributed Scrum team, and ties between the Scrum teams related to roles. Thus, within each of the Scrum teams strong and redundant ties between all participants were emphasized, as well as strong ties between the Scrum Master for each team and the project manager. The All Hands meeting at the end of each sprint provided an opportunity to form at least some sort of (initial) tie between members of the two Scrum teams.

Hence, by prescribing and emphasizing ties within the Scrum teams and between people possessing specific roles within the overall project organization, Scrum created attention and an opportunity for ties to manifest themselves. The quality of any tie, however, depends on the individuals having trust in each other and being motivated to maintain the ties which is connected to the relational dimension of social-capital.

B. *Relational dimension and trust*

The relational dimension of social capital concerns the nature and quality of the relationship between individuals in the network. Trust is essential for developing strong ties, that is, ties of high quality. Four types of trust are suggested by Sabherwal [28]: calculus-based trust, knowledge-based trust, identification-based trust and performance trust, as being important for IT outsourcing projects. The four types of trust are discussed in greater detail below.

First, *calculus-based trust* relies on explicit or implicit perceived rewards and punishments associated with a particular contractual relationship. In the interviews we did not hear about any rewards or punishments that had or could have influenced the quality of the relations between individuals within in the project or between project members and the surrounding organization.

Knowledge-based trust relies on personal knowledge (experience) about the individual or second hand knowledge about prior merits. Knowledge based trust may exist prior to starting a project if participants have a history together. In the case organization, very limited knowledge-based trust existed when the project started. The relation between the Indian Scrum Master and the Danish Project Manager was, however, an exception. The Project Manager noted in an interview, "I have very high trust in the Indian Scrum Master, I have worked with him in Denmark and trust him, I trust he knows what he is doing." Observing the DELHI project work and conducting interviews in India in November 2011, it was clear to us (the researchers) that a high degree of trust had been developed between the different participants. The Indian team members expressed trust in each other, but the Indians also felt confident and comfortable when interacting with the Danes. The trust concerned both professional capabilities and trust as human beings.

Looking at the two Scrum teams in the DELHI project, we find that using Scrum supported developing knowledge

trust (knowledge about each bother) because the daily Scrum meetings allowed all team members to interact and get to know what the others were doing. The all-hands meetings provided knowledge about individuals across Scrum teams. This setup provided ample opportunity for the team members to get to know each other.

Identification trust depend on both parties effectively understanding and appreciating the others' wants. Especially in the relation between the two Scrum teams, a respect for the other party developed. Thus, during the interviews the interviewees expressed sympathy and understanding for the other party. When scheduling meetings, the times that fit both sides were agreed upon, and strong ties in both India and Denmark developed based on personal/private knowledge about each other.

Finally, *performance-based trust* was established very quickly in the DELHI project. In an interview shortly after the first release, the Project Manager told us that, in his opinion, the first release went very well and he expressed great confidence that the project would perform as intended. He also commented that the daily Scrum-of-Scrums allowed him to follow the performance closely. By using Scrum, performance trust was supported by frequent deliveries and short feedback loop realized due to the fact that tasks were relatively small and that someone else ensured quality (testing the product). Between the two Scrum teams, performance trust was supported by demos/deliveries at the end of each Sprint. Finally, performance trust was established relatively fast between the Scrum team and the project manager, based on the principle of daily Scrum-of-Scrums meetings, the burn-down chart, and deliveries as the result of each Sprint.

C. The Cognitive dimension

The cognitive dimension of social capital concerns shared vision and shared language and concepts. In the DELHI project, both the Project Manager and the team members expressed that a shared vision for the project as such was created quickly when using the product backlog as the cornerstone. Breaking the product backlog down into a release plan and sprint backlogs made the shared vision very operational. Especially the participants in the Indian team were enthusiastic about this way of expressing a shared vision. Using Scrum in the DELHI project also provided shared language and concepts concerning the developing process, as well as their roles and responsibilities.

Summing up, the analysis shows that the DELHI project was very effective developing social capital. Ties within the DELHI project's Scum teams were established quickly and developed into strong ties during the first couple of sprints. Developing ties between Scrum teams (sites) took a little longer but was quickly started by the visit to India by the Project Manager and the Danish Scrum Master. We found that Scrum support developed social capital by creating attention and opportunity for ties to manifest themselves.

IV. Articulation Theory for Coordination

In addition to social capital, our second round interview analysis had categories of data on the three "C"'s [15], the first being coordination.

Coordination was analyzed in the first team process. In order for multiple actors within a project to pursue a common goal, they have to perform activities, which single actors pursuing the same goals would not have to do; these extra ordinary activities we call coordination. Thus, coordination can be defined as "*the additional information processing performed when multiple, connected actors pursue goals that a single actor pursuing the same goals would not perform*"[29]. One way to understand how coordination comes about could be by using the notion of *articulation* [30]. Thus, in order for actors within a project to be able to collaborate and coordinate their effort, the tasks involved in pursuing the common goal need to be articulated, and it should be established who is doing what, when they do it, and how and when to coordinate/align their work.

Strauss [30] argues that, analytically, it is useful to be able to distinguish between two aspects of articulation: *the articulation process* and the *articulation work.* Articulation work is a part of the overall articulation process; thus, articulation process is the more inclusive set of actions. Splitting articulation into two different constructs, however, allows an important distinction. The *articulation process* focuses on how work is actually performed by actors, while the *articulation work* has a more descriptive or prescriptive nature, focusing on "*the specifics of putting together tasks, task sequences, task clusters.*" It even aligns larger units, such as lines of work and sub-projects, in the service of a workflow (ibid). The term articulation work has been adopted by a number of IS researchers and used in slightly different ways [31-34]. Here, we want to use Strauss' original definition of the constructs to analyze how Scrum, as a project management and systems development paradigm, influenced the articulation work and the articulation process performed in a software development project. Second, we use Strauss' work to discuss why Scrum also seems to be very well suited for virtual teams, although it was originally intended for collocated project teams.

When interviewing project managers and members of the Scrum team, they pointed out that what they liked about working with Scrum was that Scrum helped them to understand very clearly what work needed to be done within the whole project and the specific Sprint; what they were expected to do themselves, what others were doing, and how to coordinate work.

An analysis of the DELHI project with articulation theory indicates that articulation work, including coordination, is performed in a very constructive manner when using Scrum. When performing project work, it is important to understand how to break down project work into tasks, sequencing the tasks, assigning the tasks to specific individuals, deciding how to perform the tasks, and recognizing the need for coordinating/aligning tasks.

How is this done when using Scrum? First, abstract tasks (general user stories) are defined as part of the product backlog, without considering *who* or *how* to perform the tasks. Second, somewhat more detailed tasks are defined when establishing a Sprint backlog, and here the need for sequencing/coordinating tasks may also be addressed at an abstract level. Thus, establishing the product backlog and the sprint backlog allows reconciliation about the '*what*' part (tasks) of the project work (between the product owner and the project team) to take place without complicating it with the '*who*', '*when*' and '*how*' part. Third, tasks are defined in more detail when moving tasks from the sprint backlog into the *in progress* column, and, at the same time, it is established *who* (in the team) is actually doing the work. Thus, the need to coordinate with other tasks/people is addressed at a more detailed level. Finally, Daily Scrum meetings and the Scrum Board provide a simple structure, supporting the team and finalizing the articulation work.

In Danske Bank and the Delhi case it seems that Scrum allows articulation to take place just-in-time involving the actual participants in the work. Thus, using Scrum, the articulation work takes place at a more and more detailed level as the process progresses, and reconciliation of the articulation work takes place when the people actually performing the tasks get involved.

The principle of making tasks, their sequence, and their status visible through the Sprint Backlog and the Scrum board allows all team members to get an overview of the tasks to be performed and those already performed, as well as to understand who is working on what tasks. This knowledge enables each team member to approach other team members directly if a common issue has to be resolved. The daily Scrum meetings further allow team members to realize when coordination is needed. Thus, also in virtual projects Scrum relies on mechanisms that enable team members to realize when coordination is needed instead of using pre-designed coordination mechanisms related to specific tasks. On the project level, coordination between more Scrum teams is provided by using common sprints (common milestones) and all-hands meetings, and on the day-to-day basis, the Scrum-of-Scrums meetings allow ad-hoc coordination.

Hence, for Coordination, Scrum provides a framework that supports all parts of articulation work, and yet requires very little time trying to foresee and negotiate the work flow and coordination mechanisms prior to actually conducting the work. Four aspects of Scrum especially provide coordination: the Product Backlog, the Sprint Backlog, the Scrum Board and the Daily Scrum meetings.

V. Boundary Objects for Communication

Communication is the second of the "C"-categories we analyzed. The analysis has two parts: 1) general factors influencing the communication in the DELHI project, and 2) the theory of boundary objects to provide a deeper and better understanding of what is happening in relation to communication.

When focusing on communication, the principle of short 15-minutes Daily Scrum meetings ensures an open channel for communication, even if electronic media have to be used. In the DELHI project, separate daily Scrum meetings were conducted on location in India and Denmark, followed later in the day by a Scrum-of-Scrums meeting which was conducted as an e-meeting (telephone line plus a shared electronic document showing the status of the Scrum Board and the issues raised by the Scrum teams).

At the end of each Sprint an All Hands meeting was conducted as a video conference (live video of off-side participants), plus a shared presentation of documents, power points, product demo, and others.

Although meetings and boundary objects provide an opportunity to communicate, it may not result in open and honest communication, as the building-up of social capital obviously is more difficult to develop in virtual teams. But also here some of the principles used in Scrum seem to be very powerful and help develop trust more or less for free. Hence, performance trust may be established quickly due to frequent deliveries and quick feedback, knowledge trust developed due to actually meeting every day, and calculus-based trust formed from the Sprint Backlog which established clear expectations about each delivery

Apart from these observations on communication, in general, we found that we needed a more solid theoretical framework for explaining what went on when using Scrum for distributed projects. To provide that theoretical perspective, we choose *Boundary Object* Theory which can be used to provide a deeper understanding of communication between key stakeholders. Boundary Objects are defined as "… objects which are both plastic enough to *adapt to local needs and constraints* of the several parties employing them, yet robust enough to maintain a common identity across sites. They are weakly structured in common use, and become strongly structured in individual-site use. They may be abstract or concrete. They have different meanings in different social worlds, but their structure is common enough to more than one world to make them recognizable *means of translation*" [35].

In general a software development project will have four basic stakeholders. The intended users, developers, user management and development management. Basically communication is needed between all four basic stakeholders, and often things go wrong because of mis-communication exactly at the boundaries between the four basic stakeholders. For example a user may say "I need double entry bookkeeping", but a developer, without knowledge of the bookkeeping domain may say: "Let us make it simple; you can have single entry bookkeeping". This is a very bad idea, because all professional bookkeeping has been double, ever since it was invented several hundred years ago. Thus the developer clearly reveals (in the

statement on single bookkeeping) his or her lack of knowledge of the application domain.

However, lack of knowledge is common. Not only will developers lack knowledge of the user domain, users may also lack knowledge on what is technically possible. Further, user representatives – i.e. user management – may not have sufficient knowledge of the actual practice of the users, and sellers may promise more than developers can deliver.

Thus we need something plastic and adaptable to make the four basic stakeholders communicate, and that was exactly what we found in *Boundary Objects* [35].

In our analysis of the DELHI project, we found that Scrum offered three obvious boundary objects: (1) User stories, which bind together users (who can express their needs in everyday-stories on use) and developers (who can understand the story and transform it easily to design requirements). (2) Product Backlog, where the user representative can prioritize tasks and thereby easily communicate to developers what is needed first. (3) Visible Scrum Board and burn-down charts, where the developers can easily express what value has been delivered and where Danske Bank management can quickly see whether the project is on track.

Regarding communication, we also found that the two roles as Product Owner and Scrum Master in a way worked as boundary objects or 'bridge builders' - as they were called in the DELHI project. A closer look at the literature shows that exactly these two roles can be classified as so-called *Boundary Spanners* [36]. In the concrete, Scrum can be said to have pre-defined roles whereby individuals are nominated in the two boundary spanning roles: (1) the Product Owner role that provides knowledge from the user-world to the developer world, and (2) the Scrum Master role that ensures that the daily stand-up meeting runs smoothly and that knowledge is shared between developers and users (represented by product owner).

We are not alone in seeing Boundary Objects as a good theory for explaining what is going on in software design processes. A prominent example is Bergman et al. [37] who identify four essential features of boundary objects: (a) Shared representation, (b) Transform design knowledge, (c) Mobilize for action, and (d) Legitimizing design knowledge. In our analysis of DELHI, we found the first three in Scrum. User Stories and the Product Backlog are examples of (a) shared representations; the Sprint Planning Meeting is an example of (b), transformation of design knowledge, and shows – in our analysis – the closeness between articulation theory and boundary objects, similar to what we discussed in the section on Coordination above. The Scrum Board and the Burn-Down Chart clearly mobilize for action (c). Finally, the prioritization of the Backlog by the Product Owner brings "political" legitimacy (d) to the software development process, as it grants "a legitimate status to a boundary object through validation of its content as to align with the stakeholders' intent" [37, p. 551].

Thus, our conclusion on boundary object and spanners theory is that Scrum provides five different mechanisms that work as such, and seem to work extremely well – based on observations from the DELHI case.

VI. Control

The last "C" – Control - was also a category in our findings. Best practice in distributed teams takes in common milestones, frequent delivery, quick feedback, frequent meetings, and frequent progress reports [16]. All these practices can be found in Scrum as it was practiced in DELHI. Sprints serve as common milestones, and each Sprint (which is relatively short) will bring a delivery. Feedback on individual tasks is given when it is tested, and feedback on a unit of tasks at the end of a sprint, which is performed at the end of a sprint to provide group feedback/learning. Frequent meetings are implemented as daily Scrum meetings and Scrum of Scrum meetings. Progress can be read directly on the daily update of the scrum Board and the Burn-down chart.

VII. Social Integration, Quality and Motivation

Evans and Carson [21] found social capital to moderate coordination, communication and social integration. Thus, we had a category in our analysis of DELHI data on social integration.

Social integration concerns the sense of belonging to a team, identifying with the team. In the interviews with the DELHI project manager, he expressed satisfaction with the team members not only for their motivation, their dedication to the project, and their willingness to help each other, but also for helping the project to succeed and achieving their goal. The team members interviewed expressed great satisfaction working on this project, reporting that they were proud to be working on this project. We could not identify specific elements where Scrum supported social integration, but together they had a shared vision with clear and achievable goals. With much social capital, this provided a basis for social integration.

As for Quality, the Scrum Board division between *Done* and *Done Done* ensured quality of the deliverables produced by the team. In the DELHI project, however, there was a specific role as Professional Tester, both in Denmark and India, which ensured quality. We could see that Scrum was plastic enough to allow such roles to be part of the Scrum team.

Finally, when we asked the team participants how they found working with Scrum, the answer was three-fold. First, they pointed to the closer contact with, and immediate feedback from the customer. Second, they felt increased commitment and feelings of ownership. Third, they pointed to the energy released from being able to focus on and deliver quick results.

VIII. CONCLUSION

Based on the observation of Scrum gaining surprising momentum and being used for distributed projects, we phrased the research question: Why does Scrum work?

To answer this, we carried out a case study from which we obtained an in-depth understanding of the reasons for this. We analyzed the data gathered using first a grounded theory approach, followed by an interpretive hermeneutic approach. Our analysis shows why Scrum works in the following ways:

- Scrum can build up relations and networks within and for the project team;
- Scrum has mechanisms for building trust – even at a distance;
- Scrum gives the project team a common language and target to aim for;
- Scrum is very useful for coordinating work in the project team;
- Scrum makes a number of boundary objects and boundary spanner roles available;
- Scrum includes a certain meeting structure that works well for communication in the project team;
- Scrum includes simple but effective mechanisms for tracking project progress;
- Scrum has simple mechanisms built in for ensuring quality; and
- the use of Scrum gives energy and motivation to the team.

Together these nine answers form a comprehensive and profound explanation of the mechanisms in Scrum that affects how complex software projects are managed. This contribution – we believe – is both more generalizable and more useful than typical explanations which have just black-boxed the Scrum method and thereby explain the success.

A. Validity

Our findings were presented and discussed in a workshop in India in January 2011. Further results were presented to the Project Manager in January and in April 2011. The comments and critiques received have been worked into this version of the paper.

ACKNOWLEDGMENT

The authors wish to acknowledge the valuable help of Danske Bank Group IT, DCI, ITC Infotech, Linda Olsen, Søren Gildsig and the many project team members who allowed us access to their project.

IX. REFERENCES

[1] Hossain, E., M.A. Babar, and H.-y. Paik. Using Scrum in Global Software Development: A Systematic Literature Review. in Fourth IEEE International Conference on Global Software Engineering. 2009. Limerick, Ireland.

[2] Harter, D., M. Krishnan, and S. Slaughter, Effects of process maturity on quality, cycle time, and effort in software product development. Management Science, 2000. 46(4): p. 451-466.

[3] Cusumano, M.A. and D.B. Yoffie, Software development on Internet time. IEEE Computer, 1999. 32(10): p. 60-69.

[4] Iansiti, M. and A. MacCormack, Developing Products on Internet Time. Harvard Business Review, 1997(September-October): p. 108-117.

[5] Dybå, T. and T. Dingsøyr, Empirical studies of agile software development: A systematic review. Information & Software Technology, 2008. 50(9-10): p. 833-859.

[6] Ågerfalk, P.J., B. Fitzgerald, and S. Slaughter, State of the Art and Research Challenges. Information Systems Research, 2009. 20(3): p. 317-318.

[7] Conboy, K., Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development. Information Systems Research, 2009. 20(3): p. 329-354.

[8] Austin, R.D. and L. Devin, Research Commentary--Weighing the Benefits and Costs of Flexibility in Making Software: Toward a Contingency Theory of the Determinants of Development Process Design. Information Systems Research, 2009. 20(3): p. 462-a-477.

[9] Rising, L. and N.S. Janoff, The Scrum Software Development Process for Small Teams. IEEE Software, 2000(July/Aug): p. 26-32.

[10] Takeuchi, H. and I. Nonaka, The New New Product Development Game. Harvard Business Review, 1986(January-February).

[11] Sutherland, J. and K. Schwaber, The Scrum Papers: Nut, Bolts, and Origins of an Agile Framework. 2010, SCRUM Training Institute.

[12] Sutherland, J., et al. Fully Distributed Scrum: The Secret Sauce for Hyperproductive Offshored Development Teams. in Agile 2008. 2008.

[13] Project Management Institute, A Guide to the Project Management Body of Knowledge. 4 ed. 2008, Newtown Square, PA: Project Management Institute.

[14] Caupin, G., et al., eds. ICB IPMA Competence Baseline Version 3.0. 3.0 ed. 2006, International Project Management Association: Nijkerk, The Netherlands.

[15] Carmel, E. and R. Agarwal, Tactical Approaches for Alleviating Distance in Glowal Software Development. IEEE Software, 2001(March/April).

[16] Paasivaara, M. and C. Lassenius, Collaboration practices in global interorganizational software development projects. Software Process Improvement and Practice, 2003. 8(4): p. 183-199.

[17] Krishna, S., S. Sahay, and G. Walsham, Managing Cross-Cultural Issues in Global Software Outsourcing. Communications of ACM, 2004. 47(4): p. 62-66.

[18] Glaser, B., Basics of grounded theory analysis. 1992, Mill Valley, CA, USA: Sociology Press.

[19] Strauss, A. and J. Corbin, Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory. 2nd ed. 1998, Beverly Hills, CA, USA: Sage Publications.

[20] Nahapiet, J. and S. Ghoshal, Social capital, intellectual capital, and the organizational advantage. Academy of Management Review, 1998. 23: p. 242-266.

[21] Evans, W.R. and C.M. Carson, A social capital explanation of the relationship between functional diversity and group performance. Team Performance Management, 2005. 11(7/8): p. 302-315.

[22] Pettigrew, A.M., Longitudinal Field Research on Change: Theory and Practice. Organization Science, 1990. 1(3): p. 267-292.

[23] Walsham, G., Doing Interpretive Research. European Journal of Information Systems, 2006. 15(3): p. 320-330.

[24] Myers, M.D. and D. Avison, An Introduction to Qualitative Research in Information Systems, in Qualitative Research in Information Systems - A Reader, M.D. Myers and D. Avison, Editors. 2002, Sage Publications: London. p. 3-12.

[25] Myers, M.D., Qualitative Research in Business & Management. 2009, London: Sage Publications.

[26] Miles, M.B. and A.M. Huberman, Qualitative Data Analysis: An Expanded Sourcebook. 1994, Thousand Oaks: Sage Publications Inc.

[27] Adler, P.S. and S.-w. Kwon, Social Capital: prospects for a new concept. Academy of Management Review, 2002. 27(1): p. 17-40.

[28] Sabberwal, R., The role of trust in outsourced IS development projects. Communication of ACM, 1999. 42(2): p. 80-86.

[29] Malone, T.W., What is Coordination Theory? 1988, National Science Foundation Coordination Theory Workshop: Cambridge, Massachusetts.

[30] Strauss, A., The Articulation of project Work: An Organizational Process. The Sociological Quarterly, 1988. 29(2): p. 163-178.

[31] Gerson, E.M., Reach, bracket, and the limits of rationalized coordination: Some challenges for CSCW, in Resources, Co-Evolution and Artifacts. In: M.S. Ackerman, et al., Editors. 2008, Springer: London, UK. p. 193-220.

[32] Star, S.L., The sociology of the Invisible, in Organization and Social Process. 1991, Aldine De Gruyter: New York. p. 265-284.

[33] Strauss, A., Continual Permutations of action. 1993, New Your: Aldine De Gruyter.

[34] Suchman, L., Supporting Articulation work. Computerization and Controversy. 1991, San Diego: Academic Press. 407-423.

[35] Star, S.L. and J.R. Griesemer, Institutional Ecology, `Translations' and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. Social Studies of Science, 1989. 19: p. 387-420.

[36] Levina, N. and E. Vaast, The Emergence of Boundary Spanning Competence in Practice: Implications for Implementation and Use of Information Systems. MIS Quarterly, 2005. 29(2): p. 335-363.

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