Scrum master activities : process tailoring in large enterprise projects

Bass, JM

http://dx.doi.org/10.1109/ICGSE.2014.24

Scrum Master Activities: Process Tailoring in Large Enterprise Projects

Julian M. Bass
Robert Gordon University
Garthdee Road
Aberdeen, UK
Email: j.bass@computer.org

***Abstract*—This paper explores practitioner descriptions of agile method tailoring in large-scale offshore or outsourced enterprise projects. Specifically, tailoring of the scrum master role is investigated. The scrum master acts as a facilitator for software development teams, nurturing adherence to agile practices and removing impediments for team members. But in large projects, scrum masters often work together in geographically distributed teams. Scrum masters use sprint planning to avoid development tasks that overlap team boundaries, coordinate status and effort across teams, and integrate code bases. The study comprises 8 international companies in London, Bangalore and Delhi. Interviews with 46 practitioners were conducted between February 2010 and May 2012. A grounded theory research method was used to identify that the scrum master role comprises six activities: process anchor, stand-up facilitator, impediment remover, sprint planner, scrum of scrums facilitator, and integration anchor. This systematic description of activities in scrum master teams extends our understanding of practitioner perspectives on agile process tailoring in large enterprises. Understanding these activities will help coaches guide large scale agile teams.**

I. Introduction

This paper addresses the issues of development method tailoring by practitioners in large-scale enterprise software development projects. Managers and development team members are under intense pressure to successfully implement and deploy large-scale projects. Large development programs are challenging and suffer high risks of deadline slippage and cost overruns. For example, software development programs figure prominently in a May 2013 list of UK Government large projects "at risk" of failure [3]. Thus, implementation challenges with large software projects can impede government policy change goals. The term "Agile" is applied to a collection of software development methods including: Dynamic Systems Development Methods (DSDM) [41], Feature Driven Design [10], Crystal [11], Scrum [40], Extreme Programming [8] and more recently Lean Software Development [30].

Research shows that practitioners identify the three most important perceived agile principles as: (1) achievement of customer satisfaction through early and continuous delivery of valuable software, (2) business representative and development team members working together frequently throughout the project, and (3) face-to-face conversations are the most efficient way to convey information to, and within, the development team [14]. These practitioners are in broad agreement with proponents of Agile methods who argue that improved team morale, results in enhanced productivity and that improved responsiveness to customer needs, results in better software quality [1]. While empirical research suggests that agile methods do improve job satisfaction, productivity and customer satisfaction, there can also be challenges with adoption for large development programs [16].

The overall research question for the study is "how do practitioners describe the tailoring of agile method roles and practices in large-scale software development programs?" In particular, this paper focuses on the research question "how do practitioners describe the enhancement and expansion of the scrum master role to meet the needs of large-scale software development programs?" The central role of the scrum master in the development process has been recognised, for example "the scrum master plays the critical role of change agent. It is too risky to have the wrong scrum master" [9].

This research offers novel contributions in two specific areas. Firstly, the research presented here contributes to the literature on tailoring agile methods for use in large software development programs, including five CMMI maturity level 5 accredited offshore software development vendors. Secondly, the research contributes to the emerging use of grounded theory in software engineering. Replicated sampling has been conducted in different research sites. Perspectives on the scrum master role have been triangulated by research participants with different large project stakeholder roles. Qualitative data has been analysed using the grounded theory approach. Thus, this research further contributes an application of grounded theory in software engineering research.

A motivation for the research is to reduce burnout and high levels of staff turn-over by raising awareness of the complex range of scrum master activities. This understanding can be used to target training, clarify work allocation and provide management support for the activities scrum masters actually undertake.

The paper is structured as follows. Firstly, related work in agile software development methods, concentrating on scrum and the scrum master role in particular, is presented. Then the research methods adopted, including the research sites, data collection and data analysis techniques used are described. Next, is a presentation of the findings showing how, in practice, companies scale agile methods to large software development programs. In the next section, a discussion of the findings is presented. Finally, there are conclusions and suggestions for future work.

II. Related Work

Extreme Programming focuses on engineering practices such as test-driven development and pair programming [8]. Pair programming is where developers work together in a pilot/co-pilot configuration and has been studied extensively [6] [20] [29].

Scrum contrasts with the engineering focus of extreme programming, by focusing on the orchestration and management of agile development projects [39]. Scrum, which was adopted on projects by teams in seven of the companies investigated in this study, is briefly described here.

Managing requirements in large scale projects is complex and difficult [45]. User requirements for the software are often captured, analysed and prioritised in the form of user stories. User stories are brief textual, non-technical descriptions that are readily understood by all project stakeholders. A product owner prioritizes user stories, before the start of each sprint, by carefully considering the strategic needs of clients. Stakeholders, including the scrum master and the development team members, work together to create estimates of the work required to implement each user story often using a consensus-based scoring technique. In the planning phase at the start of each sprint, the development team members decompose each user story into the various technical tasks necessary for implementation. Scrum teams, comprising 7-12 developers, are said to be self-organizing, since they create work estimates and individuals often select user stories (from the prioritized list provided by the product owner) for implementation within that particular sprint [12] [22]. Scrum emphasizes incremental software development using a multidisciplinary "feature" team structure [40]. Feature team members holistically develop end-to-end user story functionality [10]. This contrasts with traditional approaches which hierarchically organize team members around specialist architectural components such as user interface, business logic or persistence layer sub-systems.

In the past, agile proponents argued that agile methods must be implemented in their entirety to achieve their full benefits (for example [8] p. 149). However, the findings presented here suggest this is not always possible in large software development programs.

A. Enterprise Agile

As already discussed, the challenges of scaling agile methods to large international projects have received attention from numerous practitioners [28], [27], [4]. The simultaneous use of agile methods and plan-based methods in large enterprises is also receiving interest from researchers [44]. Large team size, complex business contexts and demanding time constraints can converge to cause a range of threats to productivity in agile projects [20] as the co-existence of plan-based and agile methods increases complexity and impedes involvement of business stakeholders [44]. Large scale projects therefore require a more disciplined approach to software development [5]. There is also evidence that large scale projects can exacerbate communication problems [35].

A scrum of scrums approach has been advocated to accommodate large team size [28]. Several scrum teams are formed, each with a scrum master in the usual way, and each scrum team comprises 7-12 developers. Daily coordination meetings are held within each scrum team, and in addition, the scrum masters attend a coordination meeting across the teams (the scrum of scrums). The scrum of scrums is used to tactically manage and coordinate the progress of iterations through the various scrum teams.

During the scrum of scrums meeting, each scrum master will report: (1) "what my team has done since the last meeting", (2) "what my team will do between now and the next meeting" and (3) "what impediments that prevent progress my team has encountered or created for others." Scrum of scrum meetings with too many participants can lack focus and relevance [33] and communication is improved where scrum of scrum meetings are organised around common interests and participant needs.

A meta study of research papers has been conducted in the related area of global software development [25]. In global software development, geographical distribution is often, though not always, an indicator of large scale. The meta-study suggests the most researched agile practices are (1) continuous integration, (2) stand-up meetings, (3) pair programming, (4) retrospectives, (5) scrum of scrums, and (6) test-driven development [24]. Collaboration techniques use by scrum teams include: visits and periods of co-located working, unofficial meetings, training activities and distributed documentation support tools [21]. These communication and collaboration techniques help alleviate sociocultural distance within geographically distributed teams [15].

B. Scrum Master Role

Three roles are defined in the scrum agile process, the: self-organizing development team, scrum master and product owner [40]. The scrum master is the primary interface between the product owner and the software development team [39]. The scrum master is responsible for facilitating the development process, ensuring that the team uses the full range of appropriate agile values, practices and rules. The scrum master conducts daily coordination meetings and removes any impediments that the team encounters [40].

Conventional wisdom suggests that project managers use a command and control style of management, whereas scrum masters focus on leading and coaching [9]. Scrum masters are not line managers for their sprint team members. Further, scrum masters do not assign work items to the members of their team; the teams are self-organising.

Scrum masters facilitate daily coordination meetings. The coordination meetings are used to communicate status of development work within the team and to product owners. The efficiency of daily coordination meetings is often compromised because too many stakeholders attend, or because the meetings are held too frequently to be beneficial for attendees [42].

As agile methods scale to larger projects multiple scrum teams, each with their own scrum master, must be coordinated. This study investigates the scrum master role in large-scale development programs. The contribution of this paper is to systematically articulate the activities undertaken by scrum masters. It is argued that software development processes benefit from enhanced understanding of the range and complexity

TABLE I. PARTICIPATING COMPANIES, INDUSTRY SECTORS AND INTERVIEWEE JOB TITLES

Company	Company Sector	Interviewee Job Titles	Interview Dates	Interviewee Projects and Programs
Company A, Bangalore	IT Service Provider	Program Manager Senior Project Manager Team Member	January 2010	Customer Relationship Management
Company B, Bangalore	Internet	Engineering Manager Product Manager	January 2010 April 2011	Web Mail Web Calendar
Company C, Bangalore	Software Service Provider	Development Manager	January 2010	Rail Booking
Company D, Bangalore (Offshore Provider to Company E)	Software Service Provider	Project Manager Product Owner Scrum Master (3) QA Lead Team Member	January 2010	Marketing Campaign Management Customer Relationship Management
Company E, London	Enterprise CRM	Program Manager Project Manager Director of Engineering	February 2010	Banking Marketing Campaign Management Customer Relationship Management
Company F, Bangalore	Industrial Products	Scrum Master	April 2011	Healthcare Instruments
Company G, Bangalore	IT Service Provider	Engagement Manager	April 2011	Media Entertainment
Company H, Delhi	IT Service Provider	Chief Technology Officer Corporate Lead Architect General Manager Human Resources Delivery/Program Manager (3) Project/Senior Project Manager (3) Scrum Master (2) Technical Analyst/ Consultant/Specialist (6) Team Member (9) Business Analyst	May 2012	Airline Customer Service Flight Booking

of scrum master activities in global software development projects.

III. METHOD

Experimental [46] case study [38] and empirical [26] research methods have been advocated for software engineering research. Method selection depends upon the research question or hypothesis being tested. In studying real-world problems, that have a 'people' dimension [37], theories can be categorised as explanatory or predictive [36]. Explanatory theories have been used in diverse fields, such as earth science (plate tectonics) and cosmology (Mercury's orbit deviations). These fields share insurmountable practical difficulties in constructing experiments to falsify hypotheses. Explanatory theory has itself been categorised as descriptive or analytical [19]. Grounded theory is a method for deriving explanatory theories from empirical evidence [18] and is emerging as a method used in information systems [43] and software engineering [22]. Grounded theory has been used to investigate customers in agile projects using extreme programming [31] and activities in co-located self-organizing scrum teams in small and medium sized companies [23]. While Bass, investigates large-scale development programs but specifically focuses on product owner teams [7]. This research uses a qualitative, grounded theory, approach to investigate software engineering practice comprising 8 international companies and semi-structured interviews with 46 practitioners.

A. Research Sites

The companies, which were selected from a population of large enterprises, are engaged in (typically both) outsourced and off-shore software development projects. The selected companies have head offices in Germany, India and USA although the research sites were exclusively in UK and India due to researcher travel budget constraints. The two largest companies have turnover of almost €8 billion and over US $1.5 billion. The interviews were conducted in Bangalore, India (January 2010 and April 2011), London, UK (February 2010) and Delhi, India (May 2012). Altogether, there were 46 practitioner interviews at 8 international companies, as shown in Table 1. The companies investigated were involved in either off-shoring (companies B and F) or outsourcing (companies A, C, D, E, G and H). Off-shoring is typically motivated by a desire to access and cultivate specialist technical skills from around the world. Both off-shoring and outsourcing are thought to offer lower cost skills than in-house onshore staff members. All the projects shown in Table 1 are intended for revenue generation. None of the projects are internal IT infrastructure projects.

Company B, for example, is a well known Internet business. Company B retains an in-house development capability in California but has built a development team in India (as well as elsewhere) to reduce costs while also attracting a wide range of specialist skills. Company F, in contrast, has interests broadly in the industrial products space. Company F has headquarters in Europe but also has research and development centres in India and in other territories. These global enterprises allocate work, according to the concentration of expertise, to specialist groups within the enterprise. This helps to avoid duplication of competencies within the organization. The selected IT Services companies (companies A, C, D, G and H) are all well-known vendors in the world-wide software and/or IT service outsourcing sector.

The process used to select companies for the research study comprised two phases: (1) snowball sampling technique ([34] pg. 237; [32] pg. 37) followed by (2) intensity sampling ([34] pg. 234). Initially, former co-workers and other professional contacts provided access to the first study participants. Those participants then provided access to development teams in other companies. This early, more exploratory, phase of the study focused on replicated interviews from a broad range of Companies (companies A, B, C, F and G). Snowball sampling gave access to a range of project teams and stakeholders with different perspectives. Company C, for example, exclusively use agile software development methods. Companies E and G,

On the other hand, provided some agile sceptics with negative experiences to report.

Later in the study, intensity sampling was used to obtain greater richness and depth in the study by targeting a larger number of interview participants with different responsibilities in the same company or software development program. The interviews at Company H, and Companies D and E provided triangulated perspectives from developers, QAs (quality assurance, testers), project management, program management and corporate-level executives. These differing perspectives implement the intensity sampling technique ([34] pg. 234).

To summarize, research sites were selected to provide replication using snowball sampling in the early phase of the study. Then intensity sampling was used in the later phase to enhance both depth and richness; hence increasing data reliability through participant triangulation.

Using both snowball and intensity sampling is a combination sampling approach which provides methodological triangulation to the sample selection. Combination sampling was used to provide a snapshot of participant perspectives but also the underlying motivation for these software development practices. It is difficult to assess practitioners motivation for selecting software development practices using other research methods such as surveys for example.

B. Data Collection

Documentary sources were used to support the study. Access to certain commercially confidential corporate agile development method process guidelines was obtained. These corporate guidelines outline: agile practices, roles, policies and recommended techniques. Some documentation produced for specific software development programs, such as design and architecture documents, has also been investigated. Publicly available and web hosted marketing materials were also reviewed. These included white papers, technical reports, case studies and descriptions of vendor capabilities designed to inform potential customers. On-site visits enabled first-hand observation of working practices and work place environments. Some secure work environments were visited. Coordination meetings (stand-up meetings) were observed (at Companies C, D and H) for both co-located and distributed scrum teams. This enabled investigation of arrangements for distributed scrum coordination meetings using both video- and audio-conferencing technologies. Various informal, sometimes off-site, discussions with executives, project management and development team members were conducted.

The primary data used in the study were from face-to-face recorded interviews. Interviews were conducted with 46 practitioner interviewees, with recordings professionally transcribed and reviewed. The duration of interviews was between 40 and 75 minutes each and was typically 50 to 60 minutes. An open-ended interview guide approach was used to conduct and structure the interviews. An example of the semi-structured interview guide used in this research is presented in Appendix 1. Probing questions were used to encourage interviewees to provide more detail. The interviews were open-ended because respondents were given opportunities to raise any topics, issues and concerns they wished outside the scope of scripted interview questions. Interviews were typically conducted in small meeting rooms exclusively booked for interview purposes on the company premises.

C. Data Analysis

Initially, the audio interviews and corresponding verbatim transcripts were carefully reviewed to ensure consistency. The transcript text was then imported into a qualitative data analysis software tool, in this case Nvivo V9 [2].

Grounded theory is an approach for inductively generating novel theoretical ideas from data. The new theories arise from the data and are thus said to be grounded. The grounded theory analysis began with identification of concepts within the interview data [18]. The interview concepts were coded and then compared within and between interviewees. The constant comparison analysis technique is used to explore homogeneity and heterogeneity within the interview data. These interview concepts were then iteratively grouped and refined into selected categories. Thus, interview concepts were combined to create categories which were then themselves coded, listed and compared within and between interviewees.

1) Field Notes and Memo Writing: During data collection, a series of field notes were produced. These informal notes recorded interesting issues arising during the interviews, such as apparent contradictions, areas of possible uncertainty and striking examples of emergent topics. These field notes were extended during the data analysis with descriptions of selected categories. These notes describing categories are examples of memo writing during which categories are identified, refined and sharpened [17, Chapter 12]. The memos were used to keep track of the emerging theory; and they evolved and changed during the analysis as new transcript data was added.

2) Open Coding: Open coding was conducted on a sentence-by-sentence basis of the interview transcripts. Short descriptive phrases were used as codes and, during the early stages of analysis, were hand-written on to hard copies of the transcripts. This approach provided a quick and easy way to identify and collate the initial codes. The codes at this stage were tentative and evolved quickly. Subseqently, as the volume of interview transcript data increased, the coding process was formalised and the data analysis software tool, Nvivo, was employed [2].

3) Constant Comparison: Constant comparison was used to refine and sharpen the categories emerging from data in this research. The codes from each interview were compared with each other at two key levels: firstly, within the same organisation or project team; and secondly, with outside organisations and teams. The codes were honed over time using constant comparison. For example, "scrum of scrums master" was a coding category early in the analysis, but this was later refined into the two codes "scrum of scrum coordinator" and "integration anchor" as more detailed transcript data emerged from the intensity sampling stage.

In summary, data analysis emerged from a process of iteration involving memo writing, open coding and constant comparison. Early topics, identified using line-by-line analysis of the transcript data, were recorded in memos. These topics were subsequently refined and sharpened through constant comparison within and between interview transcripts. As the

volume of interview data increased the topics became discrete categories. The categories form the basis of the grounded theory which is described next.

IV. Findings

The main contribution of this paper is to develop a grounded theory from practitioner descriptions of scrum master role tailoring. Enterprise software development projects are characterized by large work volumes, short deadlines and entrenched organizational structures. These constraints lead to tailored agile approaches. The research identifies six scrum master activities used to scale-up agile methods to large international projects.

What does a scrum master actually do? The scrum method was used by 15 project teams in 7 of the companies in this study. The scrum master activities identified in this research are described next. The presentation of this data follows the grounded theory approach advocated in [43] and used in [22].

A. Process Anchor: the process anchor mentors team members in scrum method use

Perhaps the most important activity for a scrum master is the process anchor activity, which is to own and disseminate the scrum process within the development team. The scrum method comprises a set of roles (such as scrum master and product owner), policies (such as sprint duration and definition of 'done') and practices (such as customer demonstration and retrospective). The process anchor helps development teams make policy decisions about their use of scrum. In company H, for example, a "basic purpose of a scrum master is [to ensure] that the team follows agile principles and practices" (Scrum Master, Company H). In Company B, for example, scrum comprises "stand-up meetings, retrospection meetings, following backlogs closely and tracking things like that" (Product Manager, Company B). Scrum masters lead and mentor team members. A scrum master in company F said "the key actually is the mentor, [who] has to remove all the insecurity feelings from every individual, [and] has to ensure that it is not about individuals, it is about team" (Scrum Master, Company F).

A customer demonstration is conducted at the end of each sprint. The purpose of the customer demonstration is to make refinements to the product in the light of client feedback. For example, "when we are getting feedback [from the client], we have to change the product" (Scrum Master, Company H). The retrospective in scrum is used to reflect on the previous sprint and identify lessons learned. In company H, "we have a sprint retrospective after the end of every sprint" (Scrum Master, Company H) and during the retrospective "we just wanted to be sure what has been wrong, what was good, [and] what can be improved" (Scrum Master, Company H).

Agile methods advocate multidisciplinary teams comprising analysis, developers and testers. For example, at company H "we have a scrum master and we have four or five people team in one Agile team" (Project Manager, Company H) within the team are "a group of, let's say, five to six developers, a test analyst, and a scrum master going into a team" (Test Analyst, Company H). In another project in company H, "team size is eight, we have one scrum master, one or two C++ developers, depending upon the task and two or three Java developers, one test analyst and one business analyst" (Senior Developer, Company H).

Scrum masters accommodate agile method tailoring due to geographically distributed work allocation. This includes unusual scrum coordination meeting arrangements, such as "we had to do the scrum calls twice [a day] instead of once. One India versus the US, the other one India versus Europe." (Engagement Manager, Company G). An onshore client is typical in offshore development projects

> "onshore at the client site [we have a] proxy product owner representing our team. [The proxy product owner] is interacting with [the client's] project stream lead and technical product owner. Then we have the actual [client] product owner, so there is a hierarchy. Scrum master and team is in India; and project stream lead, technical product owner and proxy product owner are [onshore]" (Project Manager, Company H).

The process anchor scrum master activity is responsible for ensuring scrum practices and processes are adopted by team members. They facilitate sprints, contributing to policy decisions and selection of scrum practices. Scrum masters lead and mentor self-organising team members.

B. Stand-up Facilitator: the stand-up facilitator conducts coordination meetings within a team

In scrum, team members communicate their status and activities using a daily coordination meeting, the eponymous scrum. The coordination meetings are used to report to product owners (and perhaps other stakeholders outside the team) the current status of development work during the sprint. As one development manager said, "the day starts with a stand up, which happens at a pre-defined time, which the team agrees upon on…I would say it kind of kick-starts the day for everybody." (Development Manager, Company C).

The agenda of the coordination meeting is defined in scrum as three questions: 'what did you do yesterday?', 'what are you going to do today?' and 'are you facing any impediments?' For example, a scrum master said "you are supposed to say what I did yesterday, what I'm going to do today, [and] what are my blocking issues" (Scrum Master, Company F). The scrum coordination meeting is used to facilitate communication within the team, as a developer at company H said "I should know what others are doing" (Senior Software Developer, Company H).

The coordination meetings are often conducted standing up, to minimize the meeting duration. A common policy is to try to limit the scrum duration to 15 minutes. For example, "stand-ups are usually 15 minutes" (Scrum Master, Company D) and "15 minute stand-up, we do every day" (Project Manager, Company H).

Scrum practitioners keep track of user stories that are completed and tested. These completed tasks should be monitored on a burn down chart. For example, "every day we have stand-up during which we burn down the number of hours which has been estimated initially" (Scrum Master, Company H). When the development activity and testing is complete the

status of the user story is updated, for example in company H, "[when] the story is done from offshore perspective... there is a particular status which is called a sprint complete" (Scrum Master, Company H). Then customers can decide whether to release completed user stories to users. For example, "we move that story, which has a sprint complete status, to the client to decide whether this goes to production or not" (Scrum Master, Company H).

Sometimes, scrum masters complain that agile methods are being used to micro-manage team member effort. For example "I think there has to be a different way of measuring productivity or delivery. Do I really need to track an individuals activity on a daily basis, or is it enough that the team is able to deliver x number of stories for a given sprint?" (Scrum Master, Company F). Agile methods sometimes seem to be advocated so that onshore client product owners can participate in daily scrum meetings with offshore team members. The product owners hear first-hand status reports from team members.

Scrum masters facilitate the daily scrum coordination meetings. The agenda, meeting duration and attendance is facilitated by scrum masters. They nurture status reporting from all the team members and track progress through burn-down charts.

C. Impediment Remover: the impediment remover eliminates work blockages for team members

The scrum masters in this study spend time investigating and removing impediments for team members. A scrum master from company F said"we focus on the blocking issues and facilitate how to get rid of them" (Scrum Master, Company F) The coordination meetings are used to surface problems and not to resolve. them. In company D for example, "in the stand-ups we don't really talk about any issues [that arise]. Once the stand-up is complete, we discuss any issues" (Scrum Master, Company D) and in company H "the scrum master takes issues and queries offline after the [stand-up] meeting" (Project Manager, Company H). Outside the coordination meetings, scrum masters have to remove the impediments that have been identified.

In large teams, scrum masters must ensure team members have all the information they need to make progress. For example, "my main role is to make sure that the team members have enough information that they can work on" (Scrum Master, Company D). Developers then learn to raise impediments with their scrum master, "if we have any problem, we have a protocol to follow, we send a mail to our scrum master" (Senior Developer, Company H). The scrum master is then expected to find a solution to the issue.

Scrum masters spend time on activities that remove impediments for members of the development team.

D. Sprint Planner: the sprint planner helps select and estimate requirements for implementation

The product owner liaises with stakeholders to identify and select the most important requirements for inclusion in each sprint. Sprint planning is then conducted at the start of the each iteration. The scrum master provides technical support to the product owner, who will have business skills but may not have technical expertise. For example, at Company D "my product owner is completely a business analyst. He might not be aware of some technical aspects of the project" (Scrum Master, Company D). The scrum master plays an important role in reviewing the product backlog for technical dependencies. The technical dependencies might suggest a sequence for implementing user stories. Technical dependencies between tasks are identified "say task six is dependent on task five, so it cannot start before the other task can finish. So we specify exact dates for those tasks" (Product Manager, Company B). Thus, while product owners prioritise using business need as the main criterion, scrum masters look at prioritisation from a technical dependency perspective.

Requirements, in scrum, are usually managed in the form of user stories. Scrum masters coordinate the decomposition of requirements into technical tasks, during sprint planning. In turn, these tasks are decomposed in smaller technical activities, for example "we need to break any tasks into sub-tasks" (Product Manager, Company B).

Precise planning and estimation ensures that all user stories included in an iteration are fully implemented and tested. Scrum masters coordinate team members as they perform detailed estimates of the work required to implement user stories. For example, the "scrum master sits with the team to do the estimation of all the user stories" (Project Manager, Company H) and "we check whether we have the capacity to do those stories or not. We do the planning, we do the estimates" (Scrum Master, Company H). In company B, team members "have to come with effort estimates and say that, 'Yeah, I can comfortably take [that task] up, I understand the dependencies and I can finish it in three days or five days.' No manager goes back and tells the engineers that [their estimate is too long]" (Product Manager, Company B). Since the team are contributing to task estimation they feel a greater sense of commitment to their estimates. For example, "by choice people are committing to something, generally you are able to meet whatever you have signed up for" (Product Manager, Company B).

In large scale projects, some team members use time during the current sprint to plan for the next sprint. For example, in company H "the [scrum] team is developing the second sprint, the QA, BA, scrum master and product owner are discussing the third sprint" (Scrum Master, Company H).

However, sometimes some user stories are incomplete at the end of an iteration. For example, if "some part of [the sprint] is left, it goes again into the product backlog" (Scrum Master, Company H). This may happen because a team is inexperienced in the business domain or is using a new technology. Placing any incomplete user stories into the product backlog allows a re-prioritisation of both new and incomplete user stories.

In a scrum of scrums context, sprint planning is required to avoid interference between the development activities of different teams. An important tactic is to ensure different teams are given distinctive responsibilities. For example "we have a strategy team. Then, we have another, infrastructure team" (Scrum Master, Company H). The user stories are assigned to the corresponding scrum team, by asking if this user story is a strategic initiative or is it infrastructure? In this project, in

company H, for example "Then, the [user] story is assigned to the [appropriate] scrum master" (Scrum Master, Company H). The work of the scrum teams is differentiated in this situation. According to one project manager, "[the teams are] working on different areas. So they will not be hitting each other's area" (Project Manager, Company H). In another case, one team is focused on fixing bugs and issues with the existing code base. For example "there is one support team which actually works on problem requests and change requests" (Project Manager, Company H). Scrum masters check the assignment of user stories to minimise interference between different teams.

Co-located scrum teams are always considered desirable. Sometimes in large projects this is not possible. A manager at Company G said, "I worked with Europe, US and India together. So definitely it's a challenge" (Engagement Manager, Company G). One project, at Company H,

> "was based on two locations, the android application was being dealt with from Bangalore office and rest of the iOS, iPhone thing was being done from here [Delhi]. So there were two scrum masters involved" (Scrum Master, Company H).

In company G, "we were in a situation where we had the product people in a different time zone. Then we had the testing people in another different time zone" (Engagement Manager, Company G).

In summary, scrum masters contribute to sprint planning by providing technical support to product owners, ensuring user stories are correctly prioritized and assigned to teams.

E. Scrum of Scrums Facilitator: the scrum of scrums facilitator conducts coordination meetings between teams

Development team size increases in order to deliver larger software systems within required deadlines. For the purpose of this study, large is defined as at least 25 developers engaged for a duration of three months or more. However, the scrum method recommends small development teams. It follows that large development teams need to be divided into smaller teams for compliance with scrum. For example, "the entire [client] team is divided into two sub-teams. One is called the red team; the other is called the blue team. The red team has their own scrum master; the blue team has their own Scrum Master" (Developer, Company D). A similar appoach is followed in even larger projects, for example "say I have a big project, 100 members' project, where there are scrum teams with 10 members so we have 10 scrum masters" (Engagement Manager, Company G).

The 'scrum of scrums' approach is used to coordinate the activities of scrum teams working together on a large project. The term 'scrum of scrums' specifically refers to a coordination meeting comprising the scrum masters and product owners from each team working together on related projects. On one project in company H, for example, scrum of scrum coordination meetings are attended by "all the scrum masters and all the proxy product owners" (Project Manager, Company H).

Frequency and attendance tailoring of the scrum of scrums coordination meetings was found in this study. Teams did not always find it attractive to conduct the scrum of scrums daily. Two projects at company H, convened their scrum of scrums coordination meetings less often; either weekly or bi-weekly. In company B, selected team members from each scrum team are seconded to attend scrum coordination meetings in another team a few times each week. A product manager, says "[members of another team] become part of the sprint stand-ups at least a couple of days in a week" (Product Manager, Company B). Thus, coordination meeting frequency is adjusted to make good use of everyone's time.

Another tailoring approach is to reduce the number of coordination meeting attendees. In some projects, the work of many stakeholders needs to be coordinated because of the high level of coupling between requirements within the project. At company H "the internal scrum of scrums meetings, [comprised] scrum masters, onsite designers, and BAs [business analysts] as well" (Scrum Master, Company H). However, coordination between loosely couple aspects of a project is not an efficient use of resources. On a different project, at company H "there were 25 people on the same [scrum of scrums] status call. That was not making sense. So, we tailor [such] that [only] important stakeholders will be involved on a daily scrum [of scrums] call. So, we have a daily scrum call which is basically useful for everyone" (Scrum Master, Company H).

Scrum masters also have to facilitate unusual combinations of scrum team co-location and geographical distribution. Some enterprises find it attractive to adopt a mix of onshore and offshore development activity. The project team deployment reflects the technical skills available at different locations. On one project in Company H the client operated scrum teams both onshore and offshore. A team member explained, "we do scrum of scrums offshore, and we do scrum of scrums onshore as well" (Technical Analyst, Company H). In that project a hierarchy of staff members manage requirements. The scrum master said, "we have a product manager [onshore] who sends the backlog to the [onshore] scrum master. The [onshore] scrum master sends us across [the backlog]" (Scrum Master, Company H), and "here in offshore we have two teams. So we have basically two leads at offshore who will be coordinating with the scrum master at onshore" (Scrum Master, Company H). Thus, process tailoring in this project results in the two offshore scrum masters receiving requirements from an onshore scrum master. The onshore scrum master is coordinating onshore and offshore teams. The procedure is intended to reduce interference between the software under development by the different teams. However, co-located offshore scrum of scrum teams were found more frequently in this study.

To summarise, scrum masters facilitate large agile projects by supporting a scrum of scrums approach. In scrum of scrums coordination meetings, scrum masters ensure stakeholders are fully utilized by adjusting the composition and frequency of scrum of scrum meetings.

F. Integration Anchor: the integration anchor facilitates amalgamation of software elements

Code integration is seen as a challenging and important issue, for example in company H,

> integration is the big issue in these type of environments because most team members work on the

> same type of files. ... We have time slots [in each iteration] for integration because there would be a lot of merge issues. (Scrum Master, Company H)

This scrum master is explaining that they allocate time within their sprints to allow time to resolve conflicts between code modules that are being integrated. Similarly, "as a scrum master, the challenges were... code synchronizing problems" (Scrum Master, Company D). On large projects, scripts and software tools are used to manage integration. In company H,

> If a new story is being developed, we document the integration scripts which should be [included] to ensure the functionality of that particular piece is being covered. (Scrum Master, Company H)

It is this scrum master's responsibility to ensure these integration scripts are prepared. Scrum masters coordinate between teams when there are code conflicts between them. At company H for example,

> "there are a number of teams. So sometimes it happens that one of the teams is working on a particular piece of code and the other team is doing some other piece of code which impacts this one" (Technical Analyst, Company H).

Where possible these task conflicts are avoided by sprint planning. When task conflicts occur the scrum of scrums coordination meeting is used to surface the issues. Scrum masters then cooperate to minimise the interference.

In company D, version control tools are use to support integration,

> We have a repository where we put in all our code. There are two areas there, ones called the private branch; the others called the trunk. ... [when] you've done your development, you've done your testing, you've done your integration, everything is fine and the code is ready in the private branch, we move it to the trunk (Scrum Master, Company D)

The scrum masters need to ensure that the team members perform that activity consistently and reliably.The presence of multiple scrum teams (whether co-located or geographically distributed) leads to the problem of integrating the code bases under development within the team and by the different teams. One approach is to merge code bases for each release, for example "we take their code base merge into our one and give [a single release] to QA" (Product Manager, Company B). Another approach is to prevent updates ahead of customer demonstrations. A code freeze period allows a team time to handle merge issues, for example "no team members would be allowed to check in any code. The branch would be blocked by that particular team" (Project Manager, Company H). In summary, scum masters facilitate a code integration process. Code from multiple teams is integrated into a single code base prior to release.

V. Discussion

The project teams in this research viewed scrum masters as central to successful scrum project outcomes. This research confirms previous literature which advocates that scrum masters own the scrum process (process anchor) monitor team status (Stand-up facilitator) and remove impediments (impediment remover) [40], [39]. The sprint planner, scrum of scrums facilitator and integration anchor activities are mainly concerned with scaling scrum to large projects. The scrum of scrums has been advocated to scale scrum to large projects [28], but the scrum master role has not been articulated in detail. The detailed description of scrum master activities presented in this research has not been found elsewhere in the literature.

This research did not find evidence of the approach used in Nokia Siemens Networks where line manager and scrum master roles were combined [9]. In contrast, the projects in this study used separate line managers responsible for career objective setting and project managers for assigning staff to teams. This may be because the teams in this study were more mature, in terms of their adoption of the scrum method.

This research also found evidence supporting earlier findings that the efficiency of daily coordination meetings is often compromised [42]. Efficiency is compromised because too many stakeholders attend, or because the meetings are held too frequently to be beneficial for attendees. The teams in this study made efforts to reduce the frequency of scrum of scrums meetings to address this concern. Also, stakeholders were selected based on their interest in specific product feature themes to maximise meeting relevance for all attendees.

Teams in this study confirmed the findings of previous research that suggest senior management support is helpful for agile adoption. However, our findings did not find evidence of the approach used at Unisys Cloud Engineering where senior officers occupy scrum masters roles [13]. Rather, our findings show that scrum masters were selected from a pool of experienced and respected development team leaders.

VI. Limitations

Three tests have been identified for establishing the quality of descriptive empirical social research: construct validity, external validity and reliability [47]. Construct validity is ensured by using multiple sources of evidence and ensuring a chain of evidence. Specific perspectives on scrum master activities have been triangulated through interviews with both development team members and product owners as well as with scrum masters themselves. In fact, interviewees have also included corporate-level executives that can offer additional perspectives on the scrum master role. A chain of evidence has been ensured by including verbatim quotations from interviewees in the findings section of the paper. The grounded theory approach was used to analyse the transcript data and create the activity categories. However, transcript quotations maintain the chain of evidence to original sources; the interviewees themselves. The construct validity has also been tested by obtaining feedback on an early draft of this paper from Company H.

External validity can be achieved through study replication. Multiple sources of evidence have been achieved by conducting replicated studies at eight companies with large studies at Companies H and E (along with their offshore service provider Company D). However, the findings and conclusions presented here should not be generalized to small and medium sized companies. Smaller companies work under profoundly

different commercial pressures with different quality assurance responsibilities.

Reliability is achieved by minimizing errors and biases in the study. The study uses an open-ended semi-structured interview guide. The interview guide acts as a script guiding data collection and ensuring consistency between interviews. An example open-ended semi-structured interview guide is presented in Appendix 1.

VII. Conclusions and Further Work

This paper explores large-scale enterprise software development programs using practitioner descriptions of agile method tailoring. Specifically tailoring of the scrum master role has been systematically investigated using the grounded theory research method. Grounded theory is a technique, in software engineering research, for exploring complex real-world settings.

Six activities performed by scrum masters have been identified and described: process anchor, stand-up facilitator, impediment remover, sprint planner, scrum of scrums facilitator and integration anchor. The process anchor nurtures adherence to agile methods. The stand-up facilitator ensures that team members share status and impediment information during each sprint. The impediment remover ensures developers can make progress with their work. The sprint planner supports the user story triage and workload planning that occurs prior to development work starting in each sprint. The scrum of scrums facilitator coordinates work with the other scrum masters in the development program. The integration anchor facilitates the merging code bases developed by cooperating teams working in parallel.

The sprint planner, scrum of scrums facilitator and integration anchor scrum master activities help scale agile methods to large programs in the CMMI maturity level 5 accredited companies investigated. Further, scrum masters can specialize by assigning these activities within a scrum master group. The scrum master activities offer important resources for tailoring agile methods in large scale development programs. Knowledge of these activities can help management provide training, support and mentoring for scrum masters. Managers that understand the complex range of activities undertaken by scrum masters can mitigate burnout and high staff turnover by offering targeted training and support resources.

Further research would be desirable to investigate how scrum masters coordinate testing within self-organizing development teams. Preliminary results from this research show that scrum masters play a role in test planning and management. A follow-up study would be worthwhile to explore this aspect of agile process tailoring in large-scale offshore or outsourced enterprise projects.

VIII. Acknowledgments

I am grateful to the companies and interviewees who participated in this research. Thanks also to the students of the Executive MBA at the Indian Institute of Management, Bangalore who facilitated access to several participating companies. The International Institute for IT, Bangalore provided hospitality for several research visits. The research benefited in part from travel funding from the UK Deputy High Commission, Bangalore, Science and Innovation Network, and the Institute for Innovation, Design & Sustainability (IDEAS) at Robert Gordon University, UK. Accommodation and sustenance was provided by Company H during the data collection visit to Delhi, India.

References

[1] Agile alliance. http://www.agilealliance.org/.

[2] NVivo 9 help. http://help-nv9-en.qsrinternational.com/nv9_help.htm.

[3] Warnings over flagship projects. http://www.bbc.co.uk/news/uk-politics-22664672, May 2013.

[4] S. Ambler. Agile software development at scale. In B. Meyer, J. Nawrocki, and B. Walter, editors, *Balancing agility and formalism in software engineering*, volume 5082 of *lecture notes in computer science*, pages 1–12. Springer Berlin Heidelberg, 2008.

[5] S. W. Ambler and M. Lines. *Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise*. IBM Press, May 2012.

[6] V. Balijepally, R. Mahapatra, S. Nerur, and K. H. Price. Are two heads better than one for software development? the productivity paradox of pair programming. *MIS Quarterly*, 33(1):91 – 118, 2009.

[7] J. M. Bass. Agile method tailoring in distributed enterprises: Product owner teams. In *Proc. IEEE 8th Int. Conf. on Global Software Engineering*, pages 154–63, Bari, Italy, Aug. 2013.

[8] K. Beck and C. Andres. *Extreme Programming Explained*. Addison Wesley, 2nd edition, Nov. 2004.

[9] S. Berczuk and Y. Lv. We're all in this together. *Software, IEEE*, 27(6):12–15, Nov 2010.

[10] P. Coad, E. LeFebvre, and J. D. Luca. *Java Modeling in Color*. Prentice Hall, Englewood Cliffs, NJ, 1999.

[11] A. Cockburn. *Agile Software Development*. Addison Wesley, Reading, MA, 2001.

[12] M. Cohn. *Succeeding with Agile: Software Development Using Scrum*. Addison-Wesley Professional, Upper Saddle River, NJ, USA, 1st edition, Oct. 2009.

[13] C. Cowan. When the vp is a scrum master, you hit the ground running. In *Agile Conference (AGILE), 2011*, pages 279–283, Aug 2011.

[14] S. de Cesare, M. Lycett, R. D. Macredie, C. Patel, and R. Paul. Examining perceptions of agility in software development practice. *Commun. ACM*, 53(6):126–130, June 2010.

[15] C. de Souza and D. Redmiles. The awareness network, to whom should i display my actions? and, whose actions should i monitor? *Software Engineering, IEEE Transactions on*, 37(3):325–340, May 2011.

[16] T. Dyba and T. Dingsoyr. What do we know about agile software development? *IEEE Software*, 26(5):6–9, 2009.

[17] B. G. Glaser. *Doing Grounded Theory: Issues and Discussions*. Sociology Press, Mill Valley, 1998.

[18] B. G. Glaser and A. L. Strauss. *Discovery of Grounded Theory: Strategies for Qualitative Research*. Aldine, Chicago, IL., 1967.

[19] S. Gregor. The nature of theory in information systems. *MIS Quarterly*, 30(3):611–42, Sept. 2006.

[20] J. E. Hannay, E. Arisholm, H. Engvik, and D. I. K. Sjoberg. Effects of personality on pair programming. *IEEE Transactions on Software Engineering*, 36(1):61 – 80, 2010.

[21] R. Hoda, J. Noble, and S. Marshall. The impact of inadequate customer involvement on self-organizing agile teams. *Information and Software Technology*, 53(5):521–534, May 2011.

[22] R. Hoda, J. Noble, and S. Marshall. Developing a grounded theory to explain the practices of self-organizing agile teams. *Empirical Software Engineering*, 17(6):609–639, 2012.

[23] R. Hoda, J. Noble, and S. Marshall. Self-organizing roles on agile software development teams. *IEEE Transactions on Software Engineering*, 39(3):422–444, 2013.

[24] E. Hossain, M. A. Babar, and H. Paik. Using scrum in global software development: A systematic literature review. *2012 IEEE Seventh International Conference on Global Software Engineering*, 0:175–184, 2009.

[25] S. Jalali and C. Wohlin. Agile practices in global software engineering - a systematic map. *2012 IEEE Seventh International Conference on Global Software Engineering*, 0:45–54, 2010.

[26] B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. C. Hoaglin, K. E. Emam, and J. Rosenberg. Preliminary guidelines for empirical research in software engineering. *IEEE Transactions on Software Engineering*, 28(8):721–734, 2002.

[27] C. Larman and B. Vodde. *Scaling Lean and Agile Development: Thinking and Organizational Tools for Large-Scale Scrum: Successful Large, Multisite and Offshore Products with Large-scale Scrum*. Addison Wesley, Dec. 2008.

[28] D. Leffingwell. *Scaling software agility: Best practices for large enterprises*. Addison Wesley, Boston, MA, USA, 2007.

[29] K. M. Lui, K. C. C. Chan, and J. Nosek. The effect of pairs in program design tasks. *IEEE Transactions on Software Engineering*, 34(2):197–211, 2008.

[30] M. Poppendieck and T. Poppendieck. *Lean Software Development: An Agile Toolkit*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.

[31] A. Martin. *The Role of the Customer in Agile Projects*. PhD thesis, Victoria University of Wellington, New Zealand, 2009.

[32] M. B. Miles and A. M. Huberman. *Qualitative Data Analysis: An Expanded Sourcebook*. Sage Publications, Inc, 2nd edition, 1994.

[33] M. Paasivaara, C. Lassenius, and V. T. Heikkilä. Inter-team coordination in large-scale globally distributed scrum: Do scrum-of-scrums really work? In *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement*, ESEM '12, pages 235–238, New York, NY, USA, 2012. ACM.

[34] M. Q. Patton. *Qualitative Research & Evaluation Methods*. Sage Publications, Inc, Thousand Oaks, California, 3rd edition, Jan. 2002.

[35] M. Pikkarainen, J. Haikara, O. Salo, P. Abrahamsson, and J. Still. The impact of agile practices on communication in software development. *Empirical Software Engineering*, 13(3):303–337, 2008.

[36] R. W. Proctor and E. J. Capaldi. *Why Science Matters: Understanding the Methods of Psychological Research*. Blackwell, Malden, MA, 2006.

[37] C. Robson. *Real World Research*. John Wiley and Sons Ltd., Chichester, UK, 3rd edition, 2011.

[38] P. Runeson, M. Höst, A. Rainer, and B. Regnell. *Case Study Research in Software Engineering: Guidelines and Examples*. Wiley-Blackwell, Hoboken, NJ, 2012.

[39] K. Schwaber. *Agile Project Management With Scrum*. Microsoft Press, Redmond, WA, USA, 2004.

[40] K. Schwaber and M. Beedle. *Agile Software Development with Scrum*. Prentice Hall, Upper Saddle River, NJ, USA, 2002.

[41] J. Stapleton. *DSDM: Dynamic Systems Development Method*. Addison Wesley, Harlow, England, 1997.

[42] V. Stray, Y. Lindsjorn, and D. Sjoberg. Obstacles to efficient daily meetings in agile development projects: A case study. In *Empirical Software Engineering and Measurement, 2013 ACM / IEEE International Symposium on*, pages 95–102, Oct 2013.

[43] C. Urquhart, H. Lehmann, and M. D. Myers. Putting the theory back into grounded theory: guidelines for grounded theory studies in information systems. *Information Systems Journal*, 20(4):357381, 2010.

[44] G. van Waardenburg and H. van Vliet. When agile meets the enterprise. *Information and Software Technology*, 55(12):2154 – 2171, 2013.

[45] K. Vlaanderen, S. Jansen, S. Brinkkemper, and E. Jaspers. The agile requirements refinery: Applying SCRUM principles to software product management. *Inf. Softw. Technol.*, 53(1):58–70, Jan. 2011.

[46] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslèn. *Experimentation in Software Engineering*. Springer, Heidelberg, 2012.

[47] R. K. Yin. *Case Study Research: Design and Methods*. Sage Publications, Inc, Thousand Oaks, California, 4th edition, Dec. 2009.

APPENDIX 1

Interview Guide, Agile Method Tailoring, May 2012

Agile Processes

- What agile methods and practices are you using?
- Would you describe agile methods as being successful for you? In what ways?
- What challenges have you encountered with agile methods?

Scaling to Enterprise Projects

- Describe any software tools or technologies you use to support agile methods?
- Have you adapted agile methods because of the geographical distribution of the team?
- Have you adapted agile methods because the client organisation was geographically distributed?
- Have you adapted agile methods because of a particularly large team?
- Have you used agile methods in a context with demanding regulatory compliance? What adaptations did you make?
- Have you used agile methods in a particularly complex domain context? What adaptations did you make?
- Have you used agile methods on a particularly technically complex project? What adaptations did you make?
- Have you used agile methods with an especially complex range of stakeholder relationships?
- What adaptations did you make?
- Have you adapted agile methods for use on a strategically important enterprise architecture programme?

Future Perspectives

- What future trends do you forsee in your use of agile methods?
- If there was one thing you could change about the way agile methods are used at [Company H] what would it be?
- What advice would you give to improve transitioning to offshore agile?

Any other comments

- Do you have any further comments on agile methods?

About Your Project(s)

Now I want to ask some questions about you and your project. These details will be kept confidential.

- What project are you working on currently? How many projects?
- How is the project team structured (for management purposes)?
- How is the project team organised geographically?
- What is the project domain? What is the project purpose?
- How large is the project in terms of team size? In terms of value?