11-4-2013

Agile Method Tailoring in a CMMI Level 5 Organization

Julian M. Bass
Robert Gordon University

Ian K. Allison
Robert Gordon University

Udayan Banerjee
NIIT Technology Ltd. Bangalore

Follow this and additional works at: https://scholarworks.lib.csusb.edu/jitim

Agile Method Tailoring in a CMMI Level 5 Organization: Addressing the Paradox

Julian M. Bass
Ian K Allison
Robert Gordon University
UNITED KINGDOM

Udayan Banerjee
NIIT Technology Ltd. Bangalore,
INDIA

ABSTRACT

We investigate the use and tailoring of agile methods in a highly disciplined CMMI Level 5 organization. We explore gaps between traditional agile practices and those required for enhanced levels of governance required by CMMI appraisal. We conducted a case study with recorded interviews from practitioners at NIIT Technology Ltd. The interviews were iteratively analysed and coded. Our findings show that agile methods must be supplemented with pre-sprint planning, evidence recording, project metrics, team self-organization, process standardization, and process base-lining to support the CMMI appraisal. There are challenges around Agile compliance with CMMI Level 4 processes and misconceptions about agile methods precluding documentation of the rationale for design decisions. This research contributes data to the tailoring of agile methods in a high-maturity CMMI context and will be useful for agile teams conducting level 5 appraisals.

INTRODUCTION

Agile methods and the Capability Maturity Model Integrated (CMMI®) are seen as complimentary approaches in modern software development (Vinekar & Huntley, 2010). Companies are now utilising both these approaches to compete in the global market. With the use of agile methods growing (VersionOne, 2013), no longer can organizations consider it to be an independent facet of their development; therefore, understanding how organizations reconcile the two concepts is important for global IT organizations.

The CMMI model provides a means to assess the software development competencies of an organization. This model emphasizes adherence to defined processes. The underlying philosophy of the model is that software quality is determined by development process quality (Chrissis et al., 2003). Based on Humphrey's seminal work organizations that achieve greater adherence to the model and therefore rigour in their processes are considered to be more mature (Humphrey, 1989). The result is frequently shown to result in benefits through improved quality, reduced cost of failure and consistency of approach. Compliance with high maturity practices in CMMI provides an assurance of quality processes to clients and, therefore, it is seen as a market differentiator. Indian software companies have sought to achieve recognition for their high-maturity processes as a means of externally validating and hence marketing their capabilities. As

a result of these efforts, many Indian off-shore outsourcing companies have achieved CMMI level 5 status.

Agile development practices are sometimes considered to question the assumptions embedded in the CMMI. The tenets of agile are seen as challenging the traditional orthodoxy of software development (Nerur & Balijepally, 2007). A common view is that agile methods are intended for smaller scale projects; as Leithiser and Hamilton (2008, p.186) put it, the agile / CMMI trade-off often begs the question as to "at what point is a project too big or too critical for agile or too small for CMMI?"

The recent growth in the use of agile is not limited to small projects with 35% of organizations now using agile methods for distributed project development (VersionOne, 2013). In practice high-maturity Indian software companies, such as NIIT Technology Ltd. (NIIT) featured in this paper, are being mandated by clients to undertake large-scale projects using agile methods. Clients are looking for a faster return on investment, and in a global software development it helps to ensure quality through early visibility of the product.

So, what happens in high maturity organizations when clients insist on using agile methods for large-scale projects? When a small minority of projects were agile, it was possible to avoid emphasis on these activities during the CMMI appraisal process. As the number of agile projects approaches and exceeds 50% of the total number of projects, then it becomes necessary to include them as part of the appraisal portfolio.

Previous work has been undertaken to develop models that combine Scrum and CMMI. There are also prior case studies published where both CMMI and Scrum have been used. However these do not fully consider compliance (Lukasiewicz & Miler, 2012) or the nature of tailoring, and tend to be limited to lower maturity levels. This paper addresses challenges of using agile methods in a CMMI Level 5 organization.

This paper specifically focuses on the research question "How do practitioners explain the use and tailoring of agile methods whilst they maintain CMMI maturity at level 5?" This research question is being addressed through a focused study of software practitioners across one such high-maturity company, NIIT. This paper contributes to the research literature on how CMMI Level 5 companies are adapting agile practices on geographically distributed projects while ensuring compliance with high maturity software development processes.

The rest of the paper is structured as follows. We introduce related work on CMMI and agile methods below. This is followed by a more detailed review of literature on use of agile methods in high maturity organizations. We then discuss the research methods used in the study in terms of selection of the research site, data collection and data analysis. Next we present our case study findings. We describe the main challenges faced when using agile methods during CMMI appraisal in a high maturity organization. This is followed by a discussion of our findings in relation to previous research in the field. Finally we present our conclusions and some suggestions for further research.

BACKGROUND

Capability Maturity Model Integrated

The achievement of quality, productivity and predictability are important drivers in the software industry. Organizations are seeking to ensure and validate their process reliability to achieve high customer satisfaction. Organizations instigate software process improvement (SPI) initiatives to define and enhance their processes. One means organizations achieve this aim is by adopting the Software Engineering Institute's (SEI) CMMI (Chrissis et al., 2003).

The CMMI is a benchmark against which an organization's maturity can be developed and assessed. The model covers four disciplines: systems engineering (SE); software engineering (SW), integrated product and process development (IPPD) and supplier sourcing (SS). CMMI guides practitioners through software development process improvement. The detail in the model has now evolved to provide a guide for twenty-five process areas for SW, across both management and engineering activities. Each key process area comprises related practices defined to achieve improvement in that area. The CMMI also provides a set of supporting, or generic, processes in relation to the improvement initiative itself.

CMMI adopters can seek external appraisal of their capability maturity against the model. This appraisal can be made on a continuous basis for each independent process area or on a staged basis for groups of processes up to level five. The same process areas are present in both continuous and staged representations of the model. The continuous representation allows an organization to select the process areas that most benefit the organization as the focus of its improvement activity. The appraisal process assesses if the organization fulfils the goals at each level.

The capability levels range from 0 to 5:

> **0: Incomplete**: a process is not performed or only partially performed

> **1: Performed**: a process satisfies the specific goals of a process area enabling work products to be produced

> **2: Managed**: a basic infrastructure is in place to support the process so that is it planned and delivered in line with policy; people have the skills to do it; stakeholders are involved and it is monitored and evaluated against the process description

> **3: Defined**: this is a managed process that is tailored from the standard set of processes according to appropriate guidelines.

> **4: Quantitatively Managed**: defined processes are managed using statistical and other quantitative methods. Quantitative objectives are established to manage the quality of the product and the process.

> **5: Optimising**: common causes of variation are addressed to continuously improve the performance on an incremental and innovative basis.

A key feature of maturity is the standardization (or "institutionalization") of processes across organizational and business units as well as project teams. The implication is that there is a commitment to, and consistency in, the way the way the work is performed. At levels 4 and 5 this consistency aides the collection and utilization of metrics across projects. Chrissis et al (2003) draw out the distinction between the processes at these higher levels and those at 3 and below; they highlight the predictability of how the processes perform. This predictability is achieved through the use of statistical and other quantitative techniques. In a process that is optimized, process improvements are systematically managed to achieve quantitatively defined objectives.

Agile Methods

In a recent survey, 84% of organizations stated they were using agile methods, with half having begun in the last two years (VersionOne, 2013). Agile software development practices improve responsiveness to customer requirements and changes, resulting in better quality software products and improved productivity and morale of the development team. The three most important features of agile are (1) achievement of customer satisfaction through early and continuous delivery of software (2) co-operative working of client and development staff, and (3) the use of face-to-face conversations for communications in the team (Bass, 2013).

Agile software development methods comprise a range of approaches, such as: Dynamic Systems Development Methods (DSDM) (Stapleton, 1997), Feature Driven Design (Coad et al., 1999), Crystal (Cockburn, 2001), Scrum (Schwaber & Beedle, 2001), Extreme Programming (XP) (Beck and Andres, 2004) and more recently Lean Software Development (Poppendieck & Poppendieck, 2006). Scrum is the most widely used of these methods with 65% of organizations using this approach or a related hybrid (VersionOne, 2013).

Scrum focuses on the orchestration and management of agile development, in contrast with the engineering focus of XP (Schwaber, 2004; Schwaber & Beedle, 2001). A "feature" team structure is emphasized which focuses on satisfying all the implementation needs for an end-to-end user interaction. This is in contrast with traditional approaches that hierarchically decompose systems into technical subsystems such as user interface, business logic or database elements.

Scrum proposes time-boxed, focused, periods of development called sprints. Sprints typically last between two and four weeks. Sprints start with planning activities, include daily coordination meetings and end with a retrospective review. Each sprint produces potentially shippable code that is demonstrated to customers to gather feedback. User stories are brief textual, non-technical statements that are used to capture, analyze and prioritize software requirements.

Some agile proponents argue that agile methods must be holistically implemented in their entirety in order to achieve their full benefits (for example, Beck and Andres, 2004). However, our research suggests that enterprises tend to tailor agile methods so that they fit into entrenched organizational structures and meet regulatory and client needs. Consequently, agile methods adoption is not consistent or entirely as envisaged by proponents of the methods. Each organization or development team tailor the methods and techniques to suit their needs.

For example, Bass (2013) found that teams used sprints in Scrum that did not deliver code directly to clients. As a result therefore incremental demonstration was diminished as a quality control approach. Similarly, he found that two XP practices (coding standards and collective code ownership) were far more widely adopted than the others. The VersionOne (2013) survey also reflects these variances, with daily stand up almost ubiquitous whilst less than a quarter of developers adopting continuous deployment. In part, this variation is because Scrum teams are said to be self-organizing, since team members have the freedom to develop work estimates and can select a user story to work on during each sprint (Hoda et al., 2012). This self-organization means that teams can select techniques and approaches to suit their project.

Pilot projects exploring the adoption of XP in large organizations also use tailoring of agile practices to fit particular organizational contexts (Lindvall et al., 2004). Key challenges in larger projects relate to multiple teams and the need for an overarching technical architecture. In a distributed agile setting the geographical and potentially time-zone separation makes the communication more challenging. Minimization of the impact can be achieved through careful selection of synchronous and asynchronous methods and tools (Bass, 2012).

CMMI – Agile Alignment

Prior research has developed models that combine Scrum and CMMI. For example, Diaz et al (2009) map CMMI level 2 to Scrum practices; Paulk (2001) outlines the way in which XP addresses the key areas at levels 2 and 3. There are also a number of case studies that review how organisations have fared in adopting agile and CMMI (e.g. Pikkarainen & Mäntyniemi, 2006; Baker, 2005; Cohan & Glazer, 2009; Garzas & Paulk, 2013). From this prior work we can see in Table 1how the agile practices can be aligned to CMMI goals.

Table 1: How Agile Addresses the CMMI Practices.

CMMI Goal	Agile Practices
Requirement management	Stories; product & sprint backlog; sprint planning; sprint reviews; planning games; information radiator; Daily meetings; on-site customer; self-organizing teams; planning days; release days;
Estimates	Sprint planning; product backlog; Sprint backlog; Planning games; tasks and effort estimations 1-2 weeks ahead; information radiator; planning days
Project Planning	Planning games; task on information radiator; product backlog
Commitment to plan	Planning games; self-organizing teams; on-site customer; reflection workshops; planning days; task cards on information radiator; release days
Project tracking	Short planning cycles; Burn down charts; project velocity; visual control; daily meetings;
Configuration Management	Collective ownership; small releases; continuous integration;

However the prior work tends to be limited to levels 2 and 3 process areas in the CMMI. They do, though, show the difficulty in moving beyond level 3. We see from above that CMMI demands rigorous application of processes but agile does not. Similarly, the VersionOne (2013) survey shows that adoption of agile tends to fail where the culture is not open to an agile approach or where there is pressure to adopt a more traditional approach. For example, in the attempt to adopt agile and CMMI at DTE Energy, a source of contention was "the methodological rub between the waterfall-based approach used by the process improvement team and the increasingly agile-based approach used by the practitioners that were targeted for formal process improvement" (Baker, 2005, p.187).

So, the barriers to the integration of agile in a traditional engineering environment have been identified but "organizations can overcome them with diligence, patience, and work" (Boehm & Turner, 2005, p.38). Paulk (2001) supported this view when he looked at how XP can help organizations realize the then Software Capability Maturity Model (SW-CMM, since integrated into CMMI). Building on this understanding, the following section discusses the specific issues for high-maturity organizations in adopting agile methods and maintaining compliance with CMMI.

AGILE PRACTICES IN HIGH MATURITY ORGANIZATIONS

Here we discuss six challenge areas from the literature where the tailoring of agile practices has been highlighted as having the potential to ensure consistency with CMMI expectations, particularly at higher levels.

Firstly, in CMMI, project teams are required to record decisions, alternatives and evaluation criteria as part of the Level 3 process area Decision Analysis and Resolution. This is intended to demonstrate that project teams are following systematic decisions making processes. Decision making processes that consider alternatives establish criteria for decisions and make records. A convention has emerged on some project teams that this recording must be done using formal documents. Document templates are freely available to support such formal written documents. However, the Standard CMMI Appraisal Method for Process Improvement (SCAMPI) processes do not mandate the presentation format of evidence recording.

Secondly, it is well known that agile methods favour working code over documentation (Beck et al., 2001). When scaling agile methods to larger systems it necessary to get a good balance between sufficient documentation to record and disseminate decisions and the unhelpful diversion of resources towards bloated formal documents.

Thirdly, there is a common misconception, often by novice team members, that agile methods discourage all documentation. Agile methods argue that documentation sufficient for the purpose of recording and disseminating information is mandatory. This comes more naturally on larger scale projects where experienced practitioners understand the need for capturing decision-making rationale (Ambler & Lines, 2012).

The fourth challenge area is project planning and control, a key element of levels 2 and 3. Lina and Dan (2012) show that Scrum addresses most of the fundamentals for the CMMI project management areas at level 2 but not the key issues of the engineering processes at level 3.

However, it has been shown that even in high maturity environments, projects that augmented plan-driven processes with agile methods had significant performance improvements (Ramasubbu & Balan, 2009).

Similarly, the proponents of an engineered approach also argue for the early production and review of design artefacts to strengthen early decision-making on architectural issues to reduce the life-time cost of the product. The agile approach promotes the response to change and interaction with people to enable this to happen. Yet, whilst there is a perceived contrast, the CMMI is agnostic about the specific development methods adopted. The focus is on improvement and the ability to manage the processes to enable stability.

Fifthly, it has been argued that the main issue with XP is that it does not address the managerial and organizational aspects of the SW-CMM (as it was), especially in relation to the standardization, or institutionalization, of processes (Paulk, 2001). To some extent this is mitigated by the use of the Scrum method (Sutherland et al., 2007). Yet whilst it is feasible to achieve a unified approach, combining the ideas requires careful consideration and management. The SEI's concept of a Software Engineering Process Group (SEPG) is one way of addressing this question of process consistency. SEPG's are responsible for drawing together key practitioners from across the organization to agree the common methods.

Finally, the higher maturity levels target the quantification and optimization of process improvement. Analysis of this data supports the evaluation of methods and tools adopted. The key challenge then is for organizations wishing to utilize agile methods whilst being CMMI level 5 compliant. Yet, whilst agile methods do not address the process areas at level 4 and 5, high maturity CMMI can add a systematic approach to quantifiable management of agile projects/processes. Paulk states "the argument that CMM[I]'s ideal of a rigorous, statistically stable process is antithetical to XP is unconvincing" (Paulk, 2001, p26). So with enlightened management and appropriate structures these limitations can be addressed.

So, in summary, we identified 6 challenge areas from the literature that informed our analysis of this case, as follows:

- Need of recorded evidence,
- Need for project planning and control,
- Self-organizing vs. planned,
- Organization level standardization of process,
- Need for capturing project level metrics and using them to predict future performance, and
- Organization level data base-lining.

RESEARCH APPROACH

The research methodology was designed to meet the purpose of addressing the research question through an explorative study. The study is part of an overarching research programme exploring the adoption and tailoring of agile practices in global software organizations. To date this larger

programme has involved 45 practitioners from eight international software companies, with distributed global development teams.

Case studies are appropriate for this research project as they gather the knowledge and perspectives of practitioners. A single case study is adopted here as an initial exploration of the issues and themes from the literature (Yin, 2009). Interpretative research is useful in developing a deeper understanding of complex real-life situations, and case studies help in communicating conceptual developments to practitioners (Benbasat & Zmud, 1999). To ensure internal triangulation of data collection and interpretation, a variety of organizational perspectives is required. Here the qualitative study of software engineering practice comprised interviews with 38 practitioners in an international software service company.

Case Selection

The company, NIIT Technology Ltd. (NIIT), was selected from a population of large enterprises engaged in both on-shore and off-shore outsourced software development projects. The company is involved in off-shore out-sourcing, as a means for its clients to access highly skilled software development practitioners in lower cost territories. The company provides services in application development and maintenance, enterprise solutions and business process outsourcing. NIIT has over 220 clients across 16 countries, and employ about 8000 professionals. The company has its head office in Delhi, India and revenues of around US $ 430 million.

NIIT Technologies adheres to major global benchmarks and standards, having secured the ISO 9001:2000 certifications and the ISO:27001 Information Security Management accreditation and follows global standards of development. It has been assessed at level 5 of SEI CMMI version 1.2. They were identified as an example of a high-maturity Indian software company currently addressing the tensions of using agile methods in this context.

We view the different client project portfolios as embedded cases within the corporate-level single case design (Yin, 2009). Different clients have varying corporate cultures and off-shore development teams must recognize such nuances.

Data Collection

Various forms of documentary sources were used during the study. Internal and commercially confidential agile development process guidelines were investigated. These documents outline corporate agile practice roles, policies and project recommendations. Some design and architecture documents from specific projects have also been investigated. We have also reviewed marketing materials such as publicly available and web hosted white papers, technical reports, case studies and descriptions of the vendor capabilities designed to inform potential customers.

Site visits to the company enabled observations of working areas and working practices. Secure development team work environments for some of the projects were visited. The work environment arrangements and locations of coordination meetings (stand-up meetings) were investigated for both co-located and distributed scrum teams. The video- and audio-conferencing

technologies used were explored. The formal data collection instruments were complimented with a range of informal, off-site, discussions with executives.

A set of formal, semi-structured, face-to-face recorded interviews were conducted with the 38 practitioner interviewees (Table 2). An open-ended interview guide approach was used to structure interviews. In addition to the scripted interview questions, respondents were given opportunities to raise any topics, issues and concerns of their choosing. Standard interview questions ensured a systematic approach whilst the openness of the interview allowed the interviewer to respond to any gaps or misunderstandings. This interviewing approach is used to help uncover important topics and issues outside the assumptions of the researcher(s). Thus, the interviews can be considered to be "interpretively active" as ideas are pursued as the emerge from the responses (Holstein & Gubrium, 1997, p.114)

The interviews were conducted in Delhi, India in May 2012 and February 2013. Interviews were conducted in small meeting rooms exclusively booked for interview purposes and located on company sites away from normal working areas. Two of the authors conducted the interviews; using two interviewers offers observer triangulation (Robson, 2011).

The interviews at this company include developer, QA, project management and corporate-level perspectives. This in-depth phase of the study is an implementation of intensity sampling (Patton, 1990, p. 171). Thus, selecting research participants from within project teams as well as across project portfolios and at a corporate level provides a wide range of perspectives. This range of perspectives helps us triangulate our data sources (Robson, 2011). This exposes motivations for the use and tailoring of agile practices that would be difficult to obtain using large scale survey methods.

Table 2: Participant Job Titles.

Participant Job Titles	Number of Participants
Chief Technology Officer	1
Corporate Lead Architect	1
General Manager Human Resources	1
Practice Head	1
Delivery/Program Manager	11
Project/Senior Project Manager	3
Agile Coach/Process Champion	2
Scrum Master	2
Technical Analyst/Consultant/Specialist	6
Test Analyst	2
Business Analyst	1
Software/Senior Software Engineer	7

Data Analysis

After the interviews 25, of the recordings were carefully transcribed. All the interviews and transcripts were carefully reviewed. Transcripts were imported into a qualitative data analysis software tool (NVivo, 2012). Data analysis techniques followed those described by Miles and

Huberman (1984). This subset were selected as the most appropriate for the study, in line with data reduction processes. Data reduction is seen as part of the analysis process, whereby the researchers are deciding which data chunks to pull out from the total collection as part of the evolving story.

The six challenge areas were used to bound the data collection and analysis (Miles & Huberman, 1984). Key points were initially identified within the interview data and then coded and compared both within and between interviewees. We used an iterative data analysis approach to refine coding categories. Thus, the key points were progressively combined to create concepts which were then themselves coded, listed and compared within and between interviewees. This concept categorization was used to organize the large volume of audio and transcript data. By following this process we were able to conceptualize the key issues in this case through reflecting on the facts within the context of this case.

CASE STUDY FINDINGS

The company studied has witnessed an increase in interest in agile methods: "we started working in agile in 2002 and from 2002 until about 2009, it was primarily only ad hoc. So small projects here, small projects there... but after 2009 we have seen a huge uptake of agile... Some [projects] because the customer says; some because there were a lot of problems. So we recommended [agile]" (Corporate Lead Architect, May 2012). Along with an increase in the number of agile projects has been an increase in project size "the size of agile projects really increased. So instead of having 20 or 10 members on projects, now suddenly we have 140, you know large offshore development programs being run in agile" (Corporate Lead Architect, May 2012). This increase in interest in agile is changing the profile of projects across offshore software vendors "our agile project base has been increasing year on year and right now we have close to 600 people working on agile projects. That is something like, what, 20% of our total project staff members [are now working on agile projects]" (Corporate Lead Architect, May 2012). In line with other organizations, Scrum was the primary method adopted at NIIT for management of the development, supplemented by aspects of engineering practices and other management techniques.

Proponents argue this interest in agile methods is due to several benefits "the customer should get the feeling that "okay, [the development team] are working correctly and getting the product base [built]" So this is the main point; customer satisfaction is the most important thing in agile" and further "if I am a customer, I can see the user interface, the functionalities which are working [at the end of each sprint]" (Software Engineer, May 2012).

The iterative nature of agile methods provides early feedback to stakeholders on development effort results. Customer satisfaction is enhanced where this feedback can lead to process improvement "We have a retrospective session, regarding the previous sprint, so that we can improve the process" (Software Engineer, May 2012). Development teams themselves appreciate earlier visibility of working software "so the good thing about the agile is we get the result in a quite rapid manner, quite early. In Waterfall, obviously, it took months, depending on your cycle, four, five, six months. Maybe then you get to know, after integration, the complete application is working" (Technical Analyst, May 2012).

As discussed above, our research has identified six main areas of challenge. However, we have found mitigation strategies to ensure compatibility between CMMI and agile. We consider the case study findings related to each of these challenges in turn below.

Need of recorded evidence

The need for maintaining evidence is critical to the CMMI. The purpose of this evidence is to document the rationale for decisions at different stages of the project. Practitioners can refer to this information to review the decision, learn lessons for future projects or adapt an idea to solve a similar problem. From an assessment perspective, though, the evidence does not necessarily have to be a formal written document.

It is possible to have evidence in many other forms. It can be comments in the code, picture, voice recording, Power point presentation, diagram etc. An agile team can be creative about documentation media. The recording often takes the form of source code annotations (for lowest level decisions) or Intranet project content management systems. Wiki information repositories are often used, wikis that organically evolve over the project lifetime. "There is a wiki, we have an internet wiki. So generally, whatever project we do we have all the things posted [on the wiki], whatever the new clients, new technologies" (Technology Specialist, May 2012). We found that an email trail of discussions and other similar recording was done to help the distributed team.

Sufficient evidence is required so that "Senior Management Review (SMR) and Risk Management can be performed [so] there is no contradiction with agile" (Practice Head, Feb. 2013). The creative forms of documentation above are sufficient to meet the need of CMMI appraisal.

There are variations in practices between different project teams though. On some agile projects "people will say less documentation means no documentation but it is not that. So we have to find other ways by which we can reduce the effort going in [to formal] documentation but ensure the critical things we have documented... Suppose tomorrow I had to maintain that application. So without any documentation, without any design documents, without proper documentation in the code how will I be able to maintain that code? So these are the really big challenge" (Program Head, May 2012).

Need for project planning and control

High ceremony processes are more prescriptive in their approach. Rational Unified Process "RUP, you know normal Waterfall, tells you everything. In RUP you don't have to apply your brain, the methodology is telling you 'Do this; don't do this; do this; don't do this'" (Corporate Lead Architect, May 2012).

This highly prescriptive approach creates a false sense of security "if you follow [the approach] to the letter you will be successful. So whether you will be or not is a different matter" (Corporate Lead Architect, May 2012).

Agile approaches, in contrast, favor evolution and responding to change over upfront planning. In fact, "agile requires a more discipline than traditional methods. The leaner the method is, well the less prescriptive, you need more mature people to figure out 'I need this; I don't need this; I need this; I don't need this'." (Corporate Lead Architect, May 2012).

"So [agile] needs more discipline, you have to make builds, you have to do automation, you have to write unit test cases... agile is actually marshal law. You have no freedom. You have to follow and there is more discipline required in agile than Waterfall. So that is certainly a realization which teams get later" (Corporate Lead Architect, May 2012).

However, the project plan that is expected in CMMI need not be "what" but can focus on "how". As long as the project goal is clear, process is clearly defined, risk management plan is in place and plan for capture of metrics is in place the CMMI requirement is addressed. Changes related to any of these can always be incorporated by revising the plan.

Even in their agile projects, it was clear to NIIT that "every project needs to have a project goal and a defined process … so, preparing the initial project plan is not a problem" (Practice Head, Feb. 2013). In part the planning process emanates from need to communicate and agree ideas in the many geographically distributed development teams that NIIT are operating.

Self-organizing vs. planned

Some people assume that the CMMI mandates defined schedules, built from defined requirements, upfront design and known productivity rates. However, this assumes focus on upfront requirements gathering and design is not in fact a defined part of the CMMI.

The CMMI does though expect the process of requirements gathering and design to be consistent and measurable in order to ensure that the scope of the project is managed. Companies that apply a consistent and measurable set of processes are believed to be able to reduce variability between projects.

In agile the emphasis is on self-organization and emergence. The product features are expected to emerge as the development progresses. The release plan is flexible and expected to evolve based on perceived business value. The development team is expected to self-organize by providing estimates and selecting work tasks. This concept is not a constraint, "there is nothing in CMMI which states that teams cannot be self-organizing" (Practice Head, Feb. 2013).

For some organizations the planning is seen as a problem because of the lack of prior consideration of the task. But agile projects "in a distributed environment involving large team size, [then] some form of upfront design is essential" (Agile Coach, Feb. 2013).
So at NIIT the balance between planned and self-organizing is struck by emphasizing the common elements of goal-setting and enabling some upfront design to support the distributed context.

Organization level standardization of process

One of the key requirements of the CMMI is to have an organizational level operating procedure defined which individuals can tailor within the specified guideline. On the other hand, Agile provides basic guidelines and many alternatives which the team can adopt, modify and tailor. Flexibility is very much a feature of agile methods.

When a new project starts, "we generally assign a coach to that project" (Corporate Lead Architect, May 2012). To assess process compliance "there is a body called a SEPG which is the software engineering process group. So they will do monthly reviews of the projects. And they also do audits to see if the practices are being followed; are they following continuous integration or are they making daily builds" (Corporate Lead Architect, May 2012).

In Scrum it is possible to add engineering practices (such as those from XP). The set of roles, practices and ceremonies defined is rather small. In the CMMI, in contrast, a large super-set of processes are defined; in consequence, project teams must select practices from this large set of process definitions.

Project Metrics and Performance Prediction

High maturity organizations use metrics for decision making and software development process management. So, "we have to collect the data from different projects just to see what is my productivity? What is my quality? What is my achievement? So those type of things we have to do" (Program Head, May 2012). The metrics are used to compare between projects, spot trends, and anticipate problems as well as for managing an individual project. The data needs to be appropriate and part of the normal process to minimize the effort of capturing the data. Otherwise as NIIT found "we faced problems in collecting data required for level 4 KPAs [Key Process Areas]. The team felt that it was an overhead." (Delivery Head, Feb. 2013)

The purpose of the metrics is to be able to improve over time. So, "we have different audit in place. We have to do the analysis of the audit findings to see what are the weak area and what is the reason for that? After doing analysis we have to implement the correction plan" (Program Head, May 2012).

The problem in agile, however, is not the lack of data, as every agile project captures a certain amount of performance data, but the consistency of data between teams to enable comparisons and trends to be seen. What is captured may vary from project to project and may be different from what is captured in a traditional project. If an automation tool is used then the data capture process becomes simpler. The CMMI recommends the use of a quantities model to predict future performance. As long as the model is specific to the project there is no contradiction. "[The] most important aspect of complying with the metrics needed for CMMI is to understand that traditional metrics like, productivity and time slippage [are] not applicable … instead we have been capturing metrics on defect, resource utilization and on improvement." (Practice Head, Feb. 2013)

Organization level data base-lining

Clearly, some of the traditional parameters being base-lined like phase wise productivity, slippage, defect leak etc. are not relevant for agile projects. As outlined above, though, there are other metrics that can be captured that are highly relevant to managing agile processes in a more measured way.

For base-lining across projects the issue of consistency does need resolving. One example is how to size a user story so that estimation and productivity comparisons can be made across projects. The problem is that "some projects use story points and some projects do not use story points. They use function points" (Corporate Lead Architect, May 2012). To address this process specialists are "defining some guidelines which is called activity based estimation model. So for example you are saying this is a four story point for this feature... Okay so we are trying to define some guidelines on the basis of what you will do with your story point estimation. But that is against your agile concept. It should be spontaneous [within any given team]" (Program Head, May 2012).

Also, for organizations that are new to agile methods, it does take time to build up an understanding of the right data and a history of relevant data. NIIT found that the "initial problem with organizational metrics was that we did not have sufficient data points … with more agile projects getting executed that is getting resolved" (Practice Head, Feb. 2013).

NIIT have been capturing metrics on defect, resource utilization and on improvement. This cannot be done without having sufficient numbers of data points. With the increase in the number of agile projects the data point challenge is being resolved.

DISCUSSION AND RECOMMENDATIONS

In broad terms our findings are consistent with previous research that shows agile methods and the CMMI are compatible (Paulk, 2001; Sutherland et al., 2007; Jakobsen & Johnson, 2008). Our case study shows that CMMI appraisal requirements are a super-set of Scrum practices. This is similar to the approach taken by Jakobsen and Johnson (2008).

The perceived philosophical conflict between agile methods and the CMMI has been shown to be flawed in the literature (e.g. (Lina & Dan, 2012) and (Santana et al., 2009)). However, there remains some uncertainty in how organizations should tailor and extend agile methods to address the gaps. Our findings from NIIT show how one organization is addressing the complex issues associated with marrying these two approaches. Their recognition of the need to tailor the methods is seen as the means to move towards alignment (e.g. Lina & Dan, 2012)

This section provides a critical reflection of the ideas from that organization as an initial evaluation. Where appropriate ideas are compared to those other organizations have adopted or in the literature. From these discussions we highlight some recommendations.

Need of recorded evidence

Documentation of decision-making is important. Our findings show the CMMI requirement to maintain evidence of decisions and changes can be met using a variety of techniques. Agile methods generally lack the necessary guidance on documenting these key points. A significant number of practitioners find that the documentation created is insufficient for their needs (de Cesare et al., 2010).

 The Technical Data Package KPA in CMMI is intended to refer to a set of product-oriented documentation (Glazer et al., 2008). The purpose is to enable those responsible for future activities (e.g. maintenance) to have a clear technical description of the product. As we have already stated there is no definition of what this should contain. Indeed, not all projects produce the same documentation. However, many people unfamiliar with the "holistic purpose of CMMI" can misinterpret the requirements (Glazer et al., 2008, p.9).

So, organizations can define what is to be produced within their agile processes. We recommend that teams are made aware of the need to store evidence, in whatever form it is recorded, in line with the relevant key process areas. Whilst this does not have to be in the form of formal written documents, there will need to be sufficient audit trail and rationale of decisions to satisfy the appraisal process. Companies could create a framework to support a more consistent – or defined – approach across teams.

Need for project planning and control

There is a strong view in the literature that agile methods can address CMMI levels 2 and 3. These levels include the project planning and control elements. The agile planning and control processes will be different though.

The plan will not cover the whole of the project life, but each increment can be planned. The team is involved in this process, as XP highlights the need for the team to be estimating its effort every two weeks (Paulk, 2001). In Scrum, especially in large-scale development teams, the key issue will ensuring teams across the project are aligned and coordinated for the duration of the project (Lina and Dan, 2012).

Project control in agile is primarily through a high degree of collaboration between members of the team. Project tracking is done through visualization of progress, such as display boards (see Figure 1 as an example). The use of burn-down charts and an understanding of team velocity ensure a close progress feedback loop is maintained.

So a defined set of project planning and control mechanisms need to be in place for agile projects. The planning horizon is inevitably going to be shorter than the standard waterfall project, but nevertheless the process can comply with CMMI. The planning in agile environments can be undertaken through a defined cycle of short-term planning and longer-term decision making about the governance of project resources and spend.

Figure 1. Use of a Display Board for Work Control (Source: Ladas, 2007).
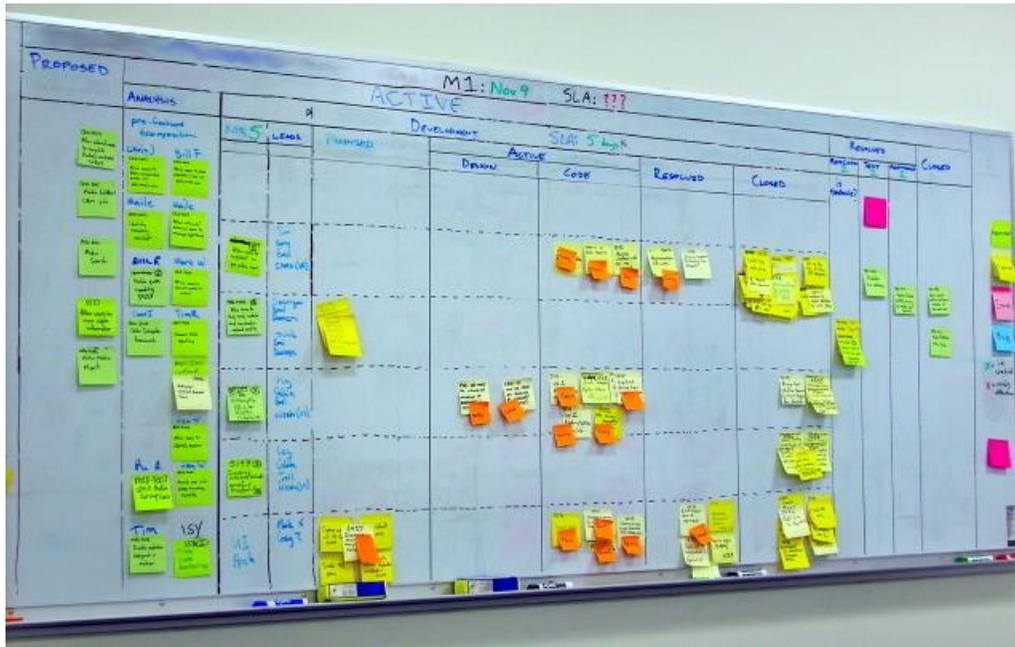
Self-organizing vs. planned

Across organizations undertaking agile projects (not only in CMMI based organizations) it appears that architecture and design are still viewed from a top-down perspective, rather than as items that will emerge through an iterative client-based dialogue. Some caution is expressed in relation to this finding but others also support upfront design in some types of projects, for example virtual teams or large systems – as we are considering here (for example, Paulk, 2001).

As discussed above, the balance between planned and self-organizing was achieved at NIIT by focusing on the common elements of goal-setting and enabling some upfront design to support the distributed context.

In these larger project settings we recommend an architecturally-driven agile approach to inform the communication across teams. Once the design is devolved to the development team then they can work in a self-organized fashion. The people-oriented, locally-devolved nature of agile development can work perfectly well within CMMI.

Organization level standardization of process

A level of standardization and conformity to procedures can be achieved through agile practices such as Pair programming (Paulk, 2001). But, whilst this will both share knowledge and assist in the application of agreed processes, it will not ensure that practices are consistent across project teams. Standardization is about ensuring that processes are defined and repeatable. From this consistency and reduced variability occur. However, high-maturity is "not about defining our

processes to the nth degree" (Cohan & Glazer, 2009, p.201). Rather ensuring that we understand our existing processes so well we know were the issues are.

That said, defining common processes is a key step. It is perhaps unusual in an agile environment to have an organizational level software quality assurance group or SEPG. An SEPG can define the processes and practices to be adopted, and the rules for tailoring them. An SEPG was used at DTE Energy to break down the barriers and assumptions between groups with preferences for agile and traditional methods (Baker, 2005). They created three practice groups (project management; core development; process management & measurement) alongside a project management and information office to build on the measurement aspects.   Through representation from different areas of practice the SEPG promoted a team approach to developing the software delivery process and resultant improvement.

We recommend that the SEPG takes control of defining the approaches for different types of projects in a way that enables consistent selection of approach. This way a variety of options are available to agile project teams, thus allowing for flexibility but without it being uncontrolled or in-defined. Variations from the method should be managed and reviewed, thereby updating the organizational level process and sharing knowledge and lessons from practice across teams. But in line with Glazer et al (2008) we should note that the tailoring guidelines should allow projects the necessary flexibility to adapt the standard processes to their needs and priorities.

Project Metrics and Performance Prediction

Agile methods do not address the process areas beyond level 3. So these structures and rigging will need to be imposed onto the agile projects. In particular, the collection of metrics does not feature high in agile methods. There is no problem in collecting data. As we found at NIIT it is not a lack of potential data, but ensuring the right data is collected and in a consistent manner. The company found that getting enough data points and having sufficient history was the problem.

We recommend that agile organizations develop a framework that enables the capture of data relevant to their agile projects. One example of an organization that has achieved this is at Pragmatics Inc., where they have reached level 4 and are working to identify the relevant, naturally occurring metrics from their agile processes that would help support the improvement to achieve level 5 (Cohan & Glazer, 2009). Ensuring the metrics are relevant to the business is important. In Garzas and Paulk's (2013) recent study of 12 SMEs in Spain they found it was useful to define the purpose of the metrics in a measurements objective document to be clear about the alignment.

Organization level data base-lining

Baselining data across the organization is one way of reducing variability between projects. This is achieved through statistical modelling that provides predictive capability to the lead developers and managers. Picking the right items to baseline is important. Agile estimation practices, such as story points, are used to build consensus within teams (Cohn, 2004). Our findings show that story points cannot be used as a benchmark across teams. So, therefore, a new set of baseline parameters needs to be evolved.

Organizations need to identify both the common metrics to be captured, and a consistent understanding of the means to do so. The framework discussed above will need to be flexible enough for local teams to define their activity but provide sufficient structure for rigorous statistical modelling between projects. One example is the definition of sizing of user stories to support consistent estimation of tasks in a Sprint.

The use of such data to aid the continuous improvement should be combined with frequent human reflection and sharing of practice across the community (Allison, 2005). DTE Energy (Baker, 2005) found that education and socialisation of the idea of CMMI 5 was important in managing the change and institutionalizing the processes. The tacit sharing of knowledge will be important given the minimized or embedded documentation discussed above. Knowledge sharing needs to be across teams as well as within the team, so concepts like Scrum of Scrums can help.

CONCLUSIONS AND FUTURE WORK

Our research suggests that there is no major contradiction between high-maturity processes and the use of agile methods. As Glazer and Dalton (2008, p.24) put it "it would be fair, then, to assert that the perceived problems between agile and CMMI stem not from agile and CMMI being inherently inconsistent, but rather a combination of misconceptions and negative experiences".

We view the needs of CMMI appraisals as a superset of text book agile methods. The NIIT experience of planning for a CMMI Level 5 reappraisal covers many projects executed in agile mode. This suggests that CMMI does not have any specific requirement that contradicts agile processes. CMMI has key performance areas (KPA) that are beyond the scope of agile methods. Some of these important differentiators are as discussed below.

There is a need to maintain management oversight of software development processes. In agile the approach is to assemble the right team, remove impediments from the path of the team and trust the team to deliver. Agile methods do not provide visibility to top management. In CMMI it is important to have a mechanism by which senior management monitor project status. In agile product owners are outside the team and are expected to review the project.

There is a need to perform risk mitigation on larger projects. Agile methods do not have explicit concept of risk mitigation. The development team is expected to perform all risk analysis and mitigation. There are no defined practices in this area. Risk mitigation is an important part of CMMI however.

When agile is deployed in an offshore context then there will be a natural tailoring from pure agile implementation. In an outsourced offshoring context, contractual obligations and geographical distribution comes into effect. The necessary extensions to the process pushes agile in a direction which brings it closer to the CMMI.

This article presented a set of embryonic guidelines for companies to ensure agile projects are run with sufficient discipline to meet the needs of CMMI level 5 appraisals. The guideline

identifies key areas where textbook agile methods must be tailored to meet appraisal governance requirements.

We found tensions between the requirements of enhanced governance imposed for appraisal purposes and team member conceptions of the light weight ceremony and documentation associated with agile methods. Thus, both agile development teams and appraisal authorities are required to adapt to a compromise position, that is both agile and subject to appraisal governance.

This study adds to our understanding of agile method tailoring. Agile methods are tailored for a variety of contextual reasons, in this case an offshore enterprise setting. The tailoring is required in areas such as process review and improvement and documentation.

The management of agile projects needs to be tailored to provide sufficient management oversight for CMMI Level 5 appraisal purposes. This enhanced level of governance need not undermine some of the key benefits of agile methods. However, there are increased costs associated with this enhanced agile governance approach.

Evidence recording is necessary for CMMI accreditation but also desirable to ensure dissemination of design decisions on projects. The recording does not need to be in the form of formal written reports. Indeed, formal reports are typically the least attractive way to collect such information. Team members should feel a sense of ownership of documentation to encourage information dissemination. Documentation that is embedded in to source code or on editable project documentation repositories has this advantage.

The limitations of this study are primarily the single case study. Whilst a single case can help to ensure a deeper understanding, the ability to generalize from that work is limited. We have overcome this through reflecting on a wider set of results. Further work is required though to fully develop the findings and recommendations.

Future work is planned in the form of additional studies to assist in the validation and development of the findings. We plan to collect data from other organizations involved in large-scale offshore agile development. This research will include studying activities at both the onshore client and offshore development group. From these studies a framework to assist companies in the management of large scale agile projects whilst ensuring their high maturity status is required.

REFERENCES

Ambler S. W. & Lines M. (2012). *Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise.* IBM Press.

Allison, I. (2005). Towards an Agile Approach to Software Process Improvement, *Communications of the IIMA, 5(1)*, 67-76.

Baker, S. W. (2005). Formalizing Agility: An Agile Organization's Journey toward CMMI Accreditation, In: *Proceedings of Agile Development Conference (ADC'05),* 24-29 July, Denver, Co. Los Alamitos, CA: IEEE Computer Society, 185-192.

Bass, J. M. (2013). Agile Method Tailoring in Distributed Enterprises: Product Owner Teams, in: Proc. IEEE 8th Int. Conf. on Global Software Engineering, Bari, Italy,. 154–63.

Beck K. et. al., (2001). *Manifesto for Agile Software Development*. [Online]. Available: http://agilemanifesto.org/. (Accessed: 24-Feb-2013).

Beck, K. & Andres, C.   (2004). *Extreme Programming Explained*, 2nd ed. Reading, MA: Addison Wesley.

Benbasat, I. & Zmud, R.W. (1999). Empirical Research in Information Systems: the Practice of Relevance, *MIS Quarterly, 23(1),* 3-16.

Boehm, B. & Turner, R. (2005). Management Challenges to Implementing Agile Processes in Traditional Development Organizations. *IEEE Software, 22(5)*, 30-39.

Chrissis, M. B. Konard M. & Shrum S. (2003). *CMMI: Guidelines for Process Integration and Product Improvement*. Boston, MA: Addison Wesley.

Coad P., LeFebvre E. & De Luca J. (1999). *Java Modeling in Color*. Englewood Cliffs, NJ: Prentice Hall.

Cockburn, A. (2001) *Agile Software Development*. Reading, MA: Addison Wesley.

Cohan S. & Glazer, H. (2009). An Agile Development Team's Quest for CMMI Maturity Level 5, In: *Proceedings of Agile Conference*, 24-28 August, Chicago, Il., Los Alamitos, CA: IEEE Computer Society, 201-206. B.

Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Boston, USA: Addison Wesley.

de Cesare, S. Lycett, M. Macredie, R. D. Patel, C.& Paul R. (2010). Examining Perceptions of Agility in Software Development Practice. *Communications of the ACM, 53(6),* 126-130.

Garzas, J. & Paulk, M.C. (2013). A case study of software process improvement with CMMI-DEV and Scrum in Spanish companies *Journal of Software: Evolution and Process* [Early view Online] Available at: DOI: 10.1002/smr.1605 (Accessed October 2013).

Glazer, H., Dalton, J., Anderson, D., Konrad, M., & Shrum, S. (2008). *CMMI or Agile: Why Not Embrace Both!, CMU/SEI-2008-TN-003*, Software Engineering Institute, Carnegie Mellon University, MA.

Hoda, R., Noble J. & Marshall S. (2012). Developing a Grounded Theory to Explain the Practices of Self-Organizing Agile Teams. *Empirical Software Engineering Journal (ESE)*, *17(6),* 609-639.

Holstein, J. A. & Gubrium, J. F. (1997).Active Interviewing. In: Silverman, D. (Ed.) Qualitative Research: Theory Method and Practice, London: Sage Publishing, 113-129.

Humphrey, W. S. (1989). *Managing the Software Process*, Reading, MA: Addison Wesley.

Jakobsen, C. R. & Johnson, K. A. (2008). Mature Agile with a Twist of CMMI, In: *Proceedings of Agile 2008. Conference*, 212 –217.

Ladas, C. (2007) *Kanban Bootstrap*, Lean Software Engineering (Online). Available at: http://leansoftwareengineering.com/2007/10/27/kanban-bootstrap/   (Accessed November 2013).

Leithiser, R., & Hamilton, D. (2008) Agile Versus CMMI − Process Template Selection and Integration with Microsoft Team Foundation Server, in Proceedings of the 46th Annual Southeast Regional Conference (ACMSE 2008), Auburn, Ala, 28 − 29 March, 186-191.

Lina Z., & Dan S., (2012). Research on Combining Scrum with CMMI in Small and Medium Organizations, In: *Proceedings of International Conference on Computer Science and Electronics Engineering*, 23-25 March, Hangzhou, China. Los Alamitos, CA: IEEE Computer Society, 554-557.

Lukasiewicz, K., & Miler,J. (2012). Improving Agility and Discipline of Software Development with Scrum and CMMI, *IET Software*, 6(5), 416-422.

Miles M. B., & Huberman, A. M. (1984). *Qualitative Data Analysis*, SAGE Publications, Newbury Park.

Nerur, S., & Balijepally, V. (2007). Theoretical Reflections on Agile Development Methodologies. *Communications of the ACM*, *50(3),* 79-83.

NVivo, Help, 2012, http://help-nv9-en.qsrinternational.com/nv9_help .htm

Patton, M. Q. (1990) *Qualitative Evaluation and Research Methods*, 2nd Edition, SAGE Publications, Newbury Park.

Paulk, M. C. (2001). Extreme Programming from a CMM Perspective. *IEEE Software*, *18(6),* 19-26.

Pikkarainen, M., & Mäntyniemi, A. (2006). An Approach for using CMMI in Agile Software Development Assessments: Experiences from Three Case Studies, In: *Proceedings of SPICE2006 Conference*, Luxemburg, 4-5 May.

Poppendieck M., & Poppendieck T. (2003). *Lean Software Development: An Agile Toolkit.* Reading, MA: Addison Wesley.

Ramasubbu, N., & Balan, R. K. (2009). The Impact of Process Choice in High Maturity Environments: An Empirical Analysis, In: *Proceedings of 31st International Conference of Software Engineering*, March 16-24, Vancouver, Canada, Washington, DC: IEEE Computer Society.

Robson, C. (2011) *Real World Research*, 3rd ed. Chichester, UK: John Wiley and Sons Ltd.

Santana, C., Gusmao, C., Soares, L., Pinheiro, C., Maciel, T., Vasconcelos, A. & Roullier, A. (2009). Agile Software Development and CMMI: What we do not know about Dancing with Elephant, In: Abrahamsson, P., Marchesi, M., & Maurer, F. (Eds.) *Proceedings of XP 2009*, LNBIP 31, 124-129.

Schwaber, K. (2004). *Agile Project Management with Scrum*. Redmond, WA, USA: Microsoft Press.

Schwaber K., & Beedle M. (2001) *Agile Software Development with Scrum*. Upper Saddle River, NJ, USA: Prentice Hall.

Stapleton J. (1997). *DSDM: Dynamic Systems Development Method*. Harlow, England: Addison Wesley.

Sutherland, J., Jakobsen, C. R. & Johnson K. (2007). Scrum and CMMI Level 5: The Magic Potion for Code Warriors, In: *Agile Conference 2007*, 272 –278.

VersionOne (2013). *7th Annual State of Agile Development Survey*, [Online] Available: http://www.versionone.com/state-of-agile-survey-results/ (Accessed October 2013).

Vinekar, V. & Huntley C. L. (2010) Agility versus Maturity: Is There Really a Trade-Off? *Computer*, *43(5)*, 87–89.

Yin, R. K. (2009). *Case Study Research: Design and Methods*, 4th ed. Thousand Oaks, California: Sage Publications, Inc.