

SPM-ICON 2009

4th Annual International Conference on Project Management

October 29-30, 2009 Bangalore

Agile, Offshoring and CMMI - Do They Mix?

by

Udayan Banerjee — Chief Technology Officer

NIIT Technologies Ltd

Copyright: Quality Solutions for Information Technology Pvt. Ltd.

Published with permission for restricted use in 'SPM-ICON 2009' in agreement with full copyrights from owner(s) / author(s) of material. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without the prior consent of the owner(s) / author(s). This edition is manufactured in India and is authorized for distribution only during 'SPM-ICON 2009' as per the applicable conditions.

Hosted By



White Paper Submitted for SPM-ICON 2009

Agile, Offshoring and CMMI – Do They Mix?

Author(s):

Name: Udayan Banerjee

Designation: Chief Technology Officer

Email: udayan.banerjee@niit-tech.com



NIIT Technologies Ltd
31/2 Rupen Agrahara
Hosur Road
Bangalore-560068

Abstract:

Introduction:

Organizations offering offshore software development services have traditionally focused on CMMI framework to deliver quality software. With agile methodologies gaining more wide spread acceptance, such organizations cannot stay away from them. There are concerns about combining agile methodologies with offshoring. There are also concerns about mismatch of agile methodologies with CMMI framework.

In this paper, we look at the problem from the perspective of an offshore software development organization and recount our experience in adopting agile methodology in executing a fixed price agile project. The challenges we faced come from three different dimensions. First was to address the issue of having an agile team split across two different countries. The second was to execute the agile project within the framework prescribed under CMMI. Here the challenge was not only to address the actual gap between CMMI and agile methodologies but also to address the perceived shortcomings of agile methodologies. The third dimension was to execute the project as a fixed price one where the requirement is not frozen at the beginning of the project.

We have listed the challenges that we faced. They can be categorized under requirements management, contract management, team management, distributed working, when to design, role of a specialist, testing challenges and CMMI adherence. The different solutions tried and the levels of success in overcoming them have also been included. Our experience indicates that agile methodologies can coexist with offshoring and CMMI framework.

Audience:

- Project Managers and Team Leaders planning to adopt Agile Methodologies
- CMMI process owners who are interested in Agile Methodologies

Area of Application:

- Agile project management
- Adopting agile practices within SEI-CMMI framework

Benefits:

- Understand the challenges faced by offshore development team to execute project following Agile Methodology
- Practical suggestions on how CMMI and Agile Methodologies can co-exist

Issues and Challenges:

- How to handle an agile project when, for various reasons, the team composition keeps changing
-

Table of Contents

ABSTRACT:	2
1.0 INTRODUCTION	4
1.1 Agile and Offshoring	5
1.2 Agile and CMMI	5
2.0 CONTENT	6
2.1 Cultural Fit	6
2.2 Requirement Management	6
2.3 Contract Management	7
2.4 Team Management	8
2.5 Distributed Working	8
2.6 Role of Specialist	9
2.7 Testing Challenges	9
2.8 CMMI adherence	10
3.0 CONCLUSION	10
4.0 DEFINITIONS, ABBREVIATION AND ACRONYMS	11
5.0 REFERENCES	11
6.0 ACKNOWLEDGEMENTS	12
BIOGRAPHY OF THE AUTHORS	12

1.0 Introduction

In the past, software was used in business to improve the internal efficiency of an organization. It has played a major role in increasing productivity through automation. However, in the last decade and since the advent of World Wide Web, software is becoming a significant component of every product and service offering. Organizations are inherently shifting their focus from using software to improve internal operations to using software to increase revenue. To do so in today's competitive world, every organization has to strive to stay ahead of the competition and come up with innovative products and services. This process involves uncertainty and it needs quick reaction from organizations to react to changes in external environment. Since software forms a key component in most offerings to customer, the software development methodology also has to keep pace with this changing scenario.

Traditional software development methodologies have been more heavy weight and had difficulties in adapting to situations where requirements either kept changing or were not clear. As an answer to the challenges of modern software development, different lightweight approaches have been established since the mid 1990s that can be subsumed under the brand Agile Methods [8-9]. They "allow for creativity and responsiveness to changing conditions" [10]. They also emphasize on customer participation, quick reaction to requirements' changes and continuous releases.

Business software development began as in house process done by people inside the organization. Since software development process is a specialized one, it is usually not the core activity of the organization. Hence, many organizations outsourced software development to others who specialized in the same. Over a period of time, advances in communication and networking technology made it feasible to outsource the development work to geographically distant location. This process could leverage the cost advantage offered by off-shored locations. Hence the term offshoring and moving development processes to countries outside became popular. Though the chief motive for this relocation is cost reduction through lower wage levels, there are other benefits viz., increased flexibility, concentration on a company's core business and the employment of qualified personnel not available in one's own country in sufficient quantities [11]. Not only does this mean reduced cost, but also chances of enhancing a product's functionality that could be developed for the same budget originally planned for. Offshoring also proved useful, as large numbers of trained manpower were readily available in the outsourced countries. Skill sets in newer and older technologies could be created in a relatively short period.

As off-shoring increased, concerns regarding the quality and integrity of the development process began to gain importance. This led to improving process rigor by standards and certifications. The concern regarding quality of software produced specially in a domain with confidential and sensitive data were addressed by rigors of process certification such as SEI-CMMI [5-6]. This multi level certification assured organizations on the quality of service expected.

Since both the trends, of agile adoption and offshoring, have different set of benefits, organizations would like to combine them and realize the benefit of both these trends. However, there are several challenges in marrying them and these challenges can be broadly classified into two categories.

1. Most agile methodologies assume co-located cross-functional team and this is not possible while offshoring software development.
2. Most organizations who undertake offshore engagement rely on SEI-CMMI process model and there are concerns about the compatibility between CMMI model and agile methodologies.

1.1 Agile and Offshoring

The feasibility of undertaking an agile off-shore development has been studied in depth. The analysis has shown that offshoring indeed poses special difficulties for development projects. Agile process models and practices seem to be appropriate for use in these contexts but have to be enhanced and adapted to work well. The direction of research points towards the conclusion that established practices and tools of software engineering can be employed to strengthen, formalize and structure agile offshoring without losing the flexibility of agile practices and falling back to a document-driven approach [11]. The focus on customer collaboration, continuous testing/integration, short iterations and test-first development seem to be the most important agile practices [12].

However, most of the study has been from the perspective of the organization which is offshoring the work and not from the perspective of the organization which is undertaking the engagement. Such organization faces many additional challenges. These challenges can be broadly categorized into:

1. **Team formation:** This includes how to quickly assemble a team and make them cohesive. It also includes how to change team composition midway either because of attrition, need for scale up or to bring in specialized skill.
2. **Heterogeneous environment:** This includes how team members can move between agile and waterfall projects and how management can have a uniform view of project status spanning different methodologies and measurement standards.

Some of these challenges can be attributed to perception and can be addressed through education and training. However, there are several real challenges where standard solutions don't exist and each organization has to formulate its own answer.

1.2 Agile and CMMI

The Capability Maturity Model for Software (CMM) [5-6] developed by the Software Engineering Institute (SEI) has had a major influence on software process and quality improvement around the world [7]. Organizations undertaking off-shored work have been in the forefront by adopting CMM practices and obtaining assessment for the same. This practice of obtaining CMM assessment has also acted as a stamp of quality software delivery. It was initially used as a differentiator but over a period of time has become a basic necessity. Therefore, any process change that can have an adverse impact on the assessment becomes a great source of risk. So, for an offshore software development organization, any contradiction between CMM and agile is a source of great concern.

SEI, the owner of the CMMI model, has realized the necessity of marrying CMMI framework and agile methodology. They have come up with an approach paper looking for CMMI & Agile synergy [3]. They come to the conclusion that agile methods and CMMI not only can co-exist, but can also be successfully integrated to bring substantial benefits to both Agile and traditional software development organizations.

However, there have been other studies that have looked at the compatibility and conflict between CMMI and agile. Some process areas, mainly those of the maturity levels 4 and 5, are in conflict with agile principles; agile methods can be applied without any major adaptation up to level 2 and with some minor changes to level 3 [4].

2.0 Content

In spite of the advances made in software engineering discipline, software development remains primarily to be a people oriented activity where automation plays a limited role. Tools, techniques and processes have reduced the effort involved in writing software and have made the process more predictable. However, software has become all pervasive and has increased in complexity. There is increasing pressure to complete software projects in a shorter and shorter time cycle. Therefore the dependence on people has remained.

2.1 Cultural Fit

Most people in our organization are used to following waterfall or a variant of waterfall methodology. Agile adoption requires a change in mindset, in the way people work and that has always been a big challenge. To keep switching between two different methods of working is a bigger challenge. It requires people to change, to modify the way they work and alter their thinking process. It not only affects the people who are directly engaged in writing the software but also it affects those involved in project management, customer interaction and those responsible for running the business. In short, there are multiple stakeholders in the organization who will look at this change from different perspectives and ask questions and raise concerns. Each of these questions and concerns may be real or perceived but need to be handled. Real concerns are directed either towards the process gaps not addressed by agile or towards contradiction between agile & CMMI. Perceived concerns are raised due to resistance to change.

Being a CMMI Level5 organization requires that establishing an organization wide agile process is consistent with the CMMI model. These processes need to be created, updated from time to time and adherence to the processes needs to be ensured. This is the primary responsibility of process owner and this responsibility includes ensuring all projects work within the laid down framework of CMMI. All the major questions and concerns raised by the stakeholders needs to be addressed.

2.2 Requirement Management

In any outsourced software development engagement, change in the scope of work can have cost and schedule implications. The impact of the change on the organization which has undertaken the outsourced development depends on how the contract is formulated. There are two dimensions to it:

1. Is the contract based on a fixed price or on time and material?
2. Is the offshore organization responsible for the schedule?

If the payment is on a time & material basis and the organization who has outsourced the work takes responsibility of managing the schedule, then the offshore organization has very limited concern about the development methodology followed and about the scope of work.

When the schedule management is shifted to offshored organization, there is concern about the scope of work. However, the concern is limited to the delivery commitment. If the contract is on a fixed price basis, then in addition to the concern about delivery commitment there is concern about managing the profitability. Following steps can be taken to minimize the risk of scope increase:

1. Focus on business value rather than a fixed set of requirements

Waterfall methodology focuses on the documented requirement where as agile methodology focuses on delivering business value. In waterfall it is very much possible to complete a project within budget and on schedule and not fulfill the business need for which the software was intended.

In agile methodology, each iteration delivers working software which can be validated by business users. The iteration planning process can take into account the business priority. The features can be fine tuned and it can reflect any change in business need. Therefore, usable software can be made available midway through the project. As a corollary it is also possible to predict project failure much early on, thereby minimizing wasted effort.

2. Exchange request rather than change request

Any change which does not impact the schedule or effort can easily be handled. Changes which impact either the schedule or the total effort needs to be handled using 'exchange requests'. The customer is free to add any new requirement provided he is able to remove any lower priority requirement of similar size which has not been worked on from the existing list. The customer is also free to decide the priority in which the features have to be worked on, before iteration. In case the customer has an additional requirement which is essential to them and cannot be exchanged with any other, then we follow the traditional change management process.

3. Feedback from working code rather than from extensive documentation

A common problem in waterfall projects is scope creep resulting from improper or ambiguous articulation of requirements. This leads to a situation where the customer's expectation from the final software differs from what the project team thinks it has to deliver. This either leads to disputes or the project team agreeing to the increased scope.

2.3 Contract Management

One of the four principles of agile manifesto is "Trust over Contract negotiation". This principle works fine when there is no major dispute. The point to remember is that you have trust between people and not between organizations. When two organizations are involved, there has to be some contractual obligation, about what software is to be delivered and how the work is to be compensated. In addition there has to be an agreement on what will happen when things do not go as planned.

Our experience shows that if both the organizations are clear about the principles of agile methodology then the process of contracting for a specific project can be significantly simplified. However, following item needs to be included.

- **Payment schedule:** We have found iteration based payment schedule to be most suitable
- **Termination clause:** It can happen when project is found to be nonviable mid way. It can also happen if the project has delivered enough business value before completion.
- **Handling scope increase:** The business goal needs to be clearly stated and the concept of exchange request needs to be included.

Dispute about the scope of work can happen when there is an improper understanding or elucidation of the requirement. It also arises when there is improper communication or there is a change in people involved in the project. One of the areas of concern is that since there is less emphasis on documentation, it will be difficult to establish who is right. However, short iterations and regular feedback on the working code, any disputes can be quickly identified and attempt can be made to reach an early consensus. However, maintaining customer trust is a very important factor in managing disputes and to prevent minor issues from turning into a major one.

2.4 Team Management

Between waterfall and agile, there is a clear difference on how the project teams are constituted and managed. In waterfall, the team composition may significantly vary from phase to phase. For example requirement analysis is expected to be handled by business analyst, design phase is to be handled by architects and designer, the coding phase is to be handled by developers and the testing phase by the testers. The team composition and size is expected to change from phase to phase. There is a clear handover between phases and in each phase the members are supposed to take over from where the other phase ended. During the construction the developers are expected to follow the design and code as per the given specifications. The process is also designed to allow for people interchangeability so that if people leave the project team then new people joining the team can gather the required knowledge from available documentation.

On the other hand, agile methodology assumes a stable and multi-skilled team. The team has had a flat structure with the same team retained as much as possible during the life cycle of the project. The team essentially consists of a fixed number of people who are preferably inducted from the start of the project. They carry a lot of implicit understanding and knowledge of what has to be delivered. Therefore, replacing a team member becomes more challenging. Such a situation can lead to a definite drop in productivity that can impact the project plan. The problem gets compounded because of the necessity of having to deliver working code in short cycles, which reduces the time buffer available for recovery.

Our experience shows that this is a real challenge. We have tried several measures to overcome this problem.

- Build about 10% redundancy in the team
- Use peer programming for all critical parts of the software
- Identify backup for each member of the team

As opposed to the typical waterfall team where the organization is hierarchical and managed top down, the agile team is expected to be self learning, self-managing, proactive and motivated. The agile coach is only expected to mentor and guide. It is imperative that the team members are comfortable with one another and have a good rapport. That makes induction of a new team member more challenging. So, apart from technical and functional knowledge transfer, the new member has to build a good rapport with the rest of the team.

The members of the agile team are also expected to be multi-skilled and be able to do analysis, design, development and testing. They are also expected to be mature enough to be self motivated and capable of interacting with customers. We have found this to be another challenge as the level of experience in the team can vary. The team can contain both developers with many years of experience and developers fresh out of college with no work experience. While we tried to staff the team with a set of mature developers, it took a few iterations for the team to settle down and achieve the required rapport.

2.5 Distributed Working

The challenge of having a distributed agile team has been well documented. The team distribution can happen in one of the two ways.

1. The development team is split across two different locations
2. The development team is co-located but the product owner is in a different location

In our project, the development team and the SCRUM master were located in India but the product owner, who belongs to the customer organization, was located in USA. For us the key challenge was in establishing a communication channel with the customer, such that it was

possible for any development team member to access them for quick query resolution and free exchange of ideas. We avoided the alternative of routing all queries through a single point of contact as it would have become a source of bottleneck and would run counter to the philosophy of having a self organizing team.

There was an initial meeting when the entire team from both the organizations spent one week at a single location. It established familiarity between all the members of the team and was a significant help for smooth interaction in the future. We used instant messaging software to enable any team member to communicate with the customer. Periodic Video conferencing and use of voice chats also kept the regular communication channel open. We also requested the customer to be a silent attendee to the daily stand up meetings over a voice service. Apart from this we also had weekly status reviews to monitor progress. These mechanisms helped us to increase the trust quotient as we were able to perceive ourselves as an extension of the customer's team rather than a vendor executing a project.

2.6 Role of Specialist

An agile team is supposed to be multi-skilled. However, a typical software organization has role specialization. There are specialist roles like architect, business analysts, user interface specialist and testers etc. These people have specialized skill set that are expected to be utilized by projects when needed. Such knowledge may not be required for the entire duration of the project. In agile methodology there is no formal mechanism to request such expertise and bring them in for a short duration. Even if such members are brought in to the project, they may have problems similar to a new team member about gaining an understanding of the requirement.

Though there may be a debate whether such roles are required, our experience suggests that specialist knowledge is essential irrespective of the methodology being followed. For example, an architect may join the project to create the reference implementation and set the technical direction for the project. The agile team members need to have a certain degree of technical understanding and maturity to take on from the architect once the base framework is in place. Some amount of formalism in form of documentation needs to be introduced to record the recommendations and decisions of the specialist.

The project in discussion did not require any specialist. However, in other projects there have been interventions from architects who created the reference architecture in the initial sprints. In such cases, the team would take over the architecture and one person from the team would become the custodian of the core architecture and this person would make incremental improvements based on the further need of the project. Where ever possible, we also ensured that the architect was available for consultation during the project tenure in case the team needed assistance. Similar approach can be followed when ever expertise is required and a specialist has to contribute to the project.

2.7 Testing Challenges

Following agile practices like SCRUM with short iterations meant that testing was a big challenge and there was a fear that the software was not adequately tested especially during the later iterations when the software size increased quite a bit. Automated Testing and Test Driven Development have been recommended as the way out by experts. However, we feel that these solutions only partially address the issue.

In some of the agile projects, we have tried to automate testing using tools like JUnit and during later sprints introduced regression testing tools to ensure that there was no regression bugs. However, the very nature of iterative and incremental development with evolutionary design meant that there were some design changes which involved a rework in previously released features. This required an end to end testing, not all of which could be

automated. We had to have sprints called stabilization sprints which did not have any new feature addition but were dedicated to testing and bug fixing.

In most cases, before the software could be released to production use, it had to undergo a series of additional tests like performance testing, stress testing and security testing etc. Since these tests required specialized knowledge and specialized testing environment they were kept outside the scope of the project. Typically it was handled by the customer where we provided the support to fix issues based on feedback received.

2.8 CMMI adherence

In our organization, the foundation of the software development process is the ETVX [1-2] model. This model has good synergy with the waterfall process and acts as a framework of how work can transition from one step to the next. It defines the verifications and validations that are needed to ensure proper flow. However, this model is in direct conflict with the agile way of working thereby creating a clear source of contradiction.

Since we are a Level 5 CMMI certified organization there was a need to audit our agile project according to the organization's prescribed process framework. The primary concern of the process group was that following agile practices meant cowboy style of programming with no processes in place. Our experience indicates that it is not the case. SCRUM requires a lot more discipline than waterfall as the team members have to be disciplined to deliver at the same velocity or higher for every sprint.

CMMI prescribed that there had to be documented evidence like minutes of the meeting, while SCRUM primarily relies on oral communication during the daily standup meetings. We were unable to address this issue as we did not want to add to the project overhead and only critical decisions taken in meetings were documented. We also saw that agile practices like SCRUM do not address the engineering aspects like configuration management, testing strategy, exploring design alternatives etc. So for these areas we adopted the CMMI practices recommended by our organization's Quality Management System (QMS).

We observed that we are able to meet most of the Level 3 KPA's in intent if not exactly by practicing as per the organization's recommended standards. But the CMMI Level 4 KPA's relating to measurement of detailed metrics was especially not addressed in SCRUM. The only metrics that we were capturing was the team's velocity with respect to Level 5 KPA's of continuous improvement. While SCRUM extensively talks about causal analysis and improvement at the project level, innovation at the organization level is not addressed. The causal analysis was practiced in the form of the sprint review meeting held after every sprint where we reviewed the sprint and identified areas for improvement.

To overcome this challenge we have created a separate process handbook for executing agile projects. While doing this, we noticed some gaps and conflict between agile methodology and CMMI framework. In both cases, CMMI prescription took precedence of agile recommendation.

3.0 Conclusion

The trend of adopting agile methodology is gaining momentum. Organizations providing offshore software development services cannot remain away from this trend. Though there are challenges, some real and some perceived, that needs to be overcome; we feel that it is possible to successfully adopt agile methodology for offshoring and it can coexist with CMMI processes. However, for successful coexistence, following points needs to be kept in mind.

- The customer and the execution team should believe in the agile way of working.
- Constant customer involvement is a must without which the project is doomed to fail.
- Developers not used to agile take some time to adjust to the agile way of working and this time needs to be planned.

Agile execution also means that teams have to be mature and well trained while in reality it is difficult to staff projects completely with such resources. So this is an area of concern for large scale adoption. Agile processes like SCRUM do not talk much about engineering practices and it is left to project teams to ensure that good engineering practices are followed. Release testing is an area where again it is left to the project teams to adopt suitable practices to ensure quality of deliverables. Stabilization sprints dedicated to converting 'potentially shippable' to 'shippable' deliveries are a reality. Contracting for an agile project still has to achieve the kind of standardization that a waterfall execution currently has.

Overall we find that following agile practices definitely delivers business value both to the customers as well as to the organization undertaking the offshored work.

4.0 Definitions, Abbreviation and Acronyms

Acronym	Description
CMM	Capability Maturity Model
SEI	Software Engineering Institute

5.0 References

Item	Description
1. R. A. Radice & R. W. Phillips. <i>"Software Engineering: An Industrial Approach"</i>	Prentice-Hall, 1998, ISBN 0138232202
2. R. Radice, N. Roth, A. O'Hara Jr, W. Ciarfella. <i>"A Programming Process Architecture"</i>	IBM Systems Journal 24(2): 79-90, 1985
3. Hillel Glazer, Jeff Dalton, David Anderson, Mike Konrad, Sandy Shrum. <i>"CMMI® or Agile: Why Not Embrace Both!"</i>	Technical Note, CMU/SEI-2008-TN-003, November 2008
4. M. Fritzsche, P. Keil. <i>"Agile Methods and CMMI: Compatibility or Conflict?"</i>	e-Informatica Software Engineering Journal, Vol. 1, Issue 1, 2007
5. S. E. Institute. Capability Maturity Model Integration (CMMI), Version 1.1 (CMMISE/SW/IPPD/SS, V1.1).	Technical report, Software Engineering Institute, Carnegie Mellon University, 2002
6. M. C. Paulk, B. Curtis, M. B. Chrissis, and C. V. Weber. <i>"Capability Maturity Model"</i>	Version 1.1. IEEE Software., 10(4):18-27, 1993
7. M. C. Paulk. <i>"Using the Software CMM With Good Judgment"</i>	ASQ Software Quality Professional, 1(3), 1999
8. K. Beck and C. Andres. <i>"Extreme Programming Explained: Embrace Change"</i>	Addison-Wesley, 2nd edition, 2004
9. A. Cockburn and J. Highsmith. <i>"Agile Software Development: The People Factor"</i>	IEEE Computer, 34(11):131-133, 2001
10. M. Doernhoefer. <i>"Surfing the Net for Software Engineering Notes"</i>	SIGSOFT Software. Engineering. Notes, 31(1):5-13, 2006
11. J Sauer. <i>"Agile Practices in Offshore Outsourcing - An Analysis of Published Experiences"</i>	IRIS 29, Helsingborg, Denmark, 2006
12. V Sachdev and K Iyengar. <i>"Will Agile Methodologies"</i>	SWDSI07 San Diego, USA, 2007

work in Offshore Outsourcing"

6.0 Acknowledgements

Name	Description
Kanakalata Narayanan	Architect, NIIT Technologies Ltd, Bangalore
Jyoti Venkatesh	Analyst, NIIT Technologies Ltd, Bangalore
Radha Rajgopal Tupil	Project Leader, NIIT Technologies Ltd, Bangalore

Biography of the author

Author:

Udayan Banerjee is the Vice President of NIIT Technologies and is responsible for exploring emerging technologies, assessing the impact and giving shape to the technology roadmap for the organisation. He is an engineering graduate from Indian Institute of Technology (IIT), Kanpur and has been working in Indian IT industry for over 30 years in different capacities. During this period, he has been associated with many global enterprises such as AstraZeneca, British Airways, Crédit Lyonnais, Hitachi, Indian Army and Misys. Udayan is the CTO and heads the Technology Innovation Centre (TIC) at NIIT Technologies where the centre is instrumental in creating several productivity tools, application frameworks and solution accelerators. The results from his involvement in the area of Model-Driven Development and Rich Internet Application have been made public in form of open source projects. He has published several papers at international conferences. The publications cover a diverse range of topics such as Service Oriented Architecture, Web2.0, Aspect Oriented Programming, Rich Internet Application, Knowledge Management, Code Comprehension & Transformation, Application of Agile Methodologies, Mobile Computing and Model Driven Software Engineering. He also maintains a blog <http://setandbma.wordpress.com> where he expresses his view on the impact of emerging technologies on enterprises and what strategy should enterprises use to adapt themselves to the emerging environment.