Journal of Computing and Information Technology - CIT 14, 2006, 3, 197–206
doi:10.2498/cit.2006.03.03

Challenges in Enterprise Adoption of Agile Methods – A Survey

Aniket Mahanti
Department of Computer Science, University of Calgary, Canada

Agile methods are a departure from plan-driven traditional approaches, where the focus is on generating early releases of working software using collaborative techniques, code refactoring, and on-site customer involvement. Research and surveys have shown that agile methodologies are an efficient way of producing software with significant advantages in production costs, time-to-market, complexity, and quality improvement over heavy-weight traditional methodologies. Even with such apparent advantages the information technology industry has not seen large-scale adoption of agile methods. In this survey paper, the major challenges in adopting agile practices by enterprises are addressed. Drawing information from the literature issues like framework for agile organizational change and adoption strategies are examined. Inputs from the industry suggest that most organizations are best suited in adopting a combination of traditional and agile methods. There is no agile methodology that can be universally applied and they all have to be tailored to integrate into existing processes.

Keywords: agile processes, chasm, enterprise, software development, technology adoption lifecycle.

1. Introduction

Agile software development methodologies have attracted considerable attention from the information technology (IT) industry in the past few years. As suggested by Hayes, the reason for this interest can be attributed to the following advantages over traditional plan-driven approaches [4]:

- *Improved return on investment (ROI)*: Customers provide frequent feedback on each iteration, which allows the development team to create better software, thus improving the ROI.
- *Early detection and cancellation of failing projects*: In traditional projects, optimistic reporting on abstract tasks such as analysis and design, delays problems from being identified in the early stages of the project. Agile methods execute design, analysis, and implementation tasks repeatedly in short iterations allowing greater visibility on the state of progress of the project. Sponsors can cancel the project early if they find it is not going as expected and thus loose minimal investment.
- *Higher software quality*: Test-driven development, short iterations, scoping, and frequent customer feedback improve the overall quality of the software.
- *Improved control over project*: Agile processes focus on people over processes. With less stress on documentation and more attention on delivering functionality at the end of each iteration, agile teams can improve their velocity. Short iterations, multi-disciplinary teams, knowledge sharing, continuous integration, and feedback allow better control over the project and increased visibility.
- *Reduced dependence on individuals and increased flexibility*: Agile practices emphasize on creating cohesive teams of developers that collectively analyze, design, and code software. This eliminates the possibility of any bottlenecks in the development process because of developers working individually on tasks.

Introducing agile practices into an organization can be a challenging task. For new adopters the case for agile methods has to be supplemented by results showing the high failure rates in traditional projects. Highlighting the inherent defects in the waterfall model and promoting the idea of iterative development could also serve as

useful strategies [19]. Adopters of agile methods usually face difficulties in assigning new roles to the management, bridging the move from legacy code to agile, and negotiating with the leadership on the issue [17]. After adopting agile methods, enterprises can collect hard data and compare them to industry benchmarks. Reifer reports that agile enterprises have seen 15-23% gain in productivity, 5-7 % reduction in costs, 25-50% reduction in time-to-market, and significantly reduced defect rates [11]. Citing success stories showing gains in the software development process can also create interest among sponsors in agility.

This survey paper takes an in-depth view of the challenges involved in introduction of agility into an enterprise. Drawing information from the literature, this paper provides a balanced view of agile software development experts as to how a blueprint for successful agile adoption can be created. Agile methods are not suited for every project. Military projects, where the requirements hardly change and are executed according to a rigorous pre-determined plan, may not benefit from agile methods. Agile practices seem to be best suited for business applications that are market-driven, where requirements change to cope with the ever changing demand. Researchers have recently focused on the limitations of agile methods and proposed new hybrid and scalable processes for use in large organizations. The paper studies some of the limitations of agile practices and explains how large organizations are using this methodology differently.

The remainder of the paper is organized as follows. Section 2 describes the technology adoption lifecycle and the agile chasm. Section 3 lists some of the common challenges faced by organizations in becoming agile. A roadmap for introduction of agility to enterprises is discussed in Section 4. Wholesale and incremental strategy for adopting agile practices along with an industrial case study is provided in Section 5. Section 6 illustrates the experiences of four large non-IT organizations in using agile methods and their future prospects. Section 7 presents a framework of organization change in large enterprises. Section 8 describes limitations of agile methodologies. Section 9 concludes the paper.

2. Agile Chasm

Adoption of new technology by businesses is governed by Moore's technology adoption lifecycle (TALC) model [16]. The TALC (see Figure 1) can be visualized as a bell curve divided into five phases through which every technology must pass during its lifecycle: *Innovators*, *Early Adopters*, *Early Majority*, *Late Majority*, and *Laggards* [16, 14].

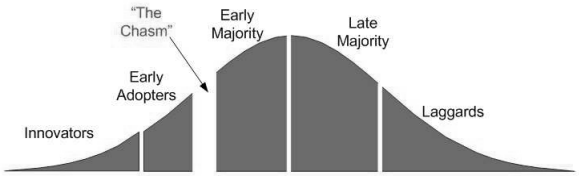

Fig. 1. Technology Adoption Lifecycle.

Innovators are the people who like technology for its own sake. They are the 'gatekeepers' who allow entry of any new discontinuous technology into the market. Approval of Innovators is critical for sustenance of a technology in the lifecycle. Early Adopters are an optimistic lot who will take the risk of trying out new technology for gaining competitive advantage and establishing market share. Early Majority are the pragmatists who are more concerned with using tried-and-tested technology for improving existing business processes. Late Majority are conservatives who dislike discontinuous innovation and prefer tradition over progress. They are least likely to buy high technology products and are not expected to like it. Laggards are traditionalists who out rightly reject new technology.

There is a substantial gap or chasm dividing the Early Adopters and Early Majority. Early Adopters will accept some of the risk associated with developing groundbreaking technologies and services. Early Majority want true and complete technologies that have proven to be useful. The difference between the two groups creates the chasm and establishes the need for markedly different business strategies. Early Adopters do not make good references for Early Majority because they are willing to take risk for a new untested technology. Because Early Majority are concerned with not disrupting their

organization, good references are critical for their buying decision [15]. Agile processes are currently in the Early Adopter zone. The next section discusses the challenges in introducing agile practices to mainstream Early Majority enterprises.

3. Challenges in Adopting Agile Methods

Common challenges in adopting agile methods are: serial thinking, closed mindedness, office politics, black and white mind-set, fear of change, specialized skills, outdated skills, documentation-heavy mind-set, and do-it-all-at-once attitude [1, 5].

Serial thinking: Many IT professionals have become accustomed to serial approaches and are unreceptive to new and evolutionary approaches. This can be attributed to the fact that the past 40 years have been dominated by software development methodologies using serial approaches. These people want to identify the complete requirements first, then design the system, and only after that start coding. Such people need to be given appropriate training, enough time, and targeted mentoring to learn the principles of agile development. At the same time, one should be vigilant to make sure that the serial mind-set does not hamper introduction and sustenance of agile practices in the enterprise.

Closed mindedness: There are many software professionals who have not invested time and energy to learn about upcoming and promising methodologies. These people belong to two camps; one group that perceives agile methods as nothing but code-and-fix in disguise, and the other group that has adopted an anti-agile stance. It is extremely important to actively educate these people about the advantages of agile methodologies. Some people would intentionally spread incorrect information about agility to prevent agile introduction into the organization. Such misinformation should be challenged and prevented from being spread. A minority of the population that can neither accept agility nor can be taught new approaches should be asked to look for new avenues. This would allow smooth introduction of agile practices.

Office politics: Office politics is an essential part of everyday life in enterprises. Some people will prevent the use of agile practices to save their power base. For example, a senior programmer who has a corner office may not like sharing space with junior programmers in a cubicle. A group of (politically savvy) professionals may at first accept agility. They would, however, wait for agile practices to fail and revert to old practices. One should become politically adept for achieving success in introducing agile methods.

Black and white mind-set: The black and white mind-set is prevalent among many IT professionals. People consider that they have to choose from a limited set of options and that there is no scope of selecting a middle path. For example, most large organizations will recognize the fact that a combination of agile and traditional practices is best suited for them over complete adoption of agile methods. Inflexible people could make agile introduction difficult. Education and training can serve as the most suitable counter-active measure for such people.

Fear of change: Doubts in people's minds that they would not be able to acquire the required agile skills and/or cope with the new agile environment may instil fear in their minds. Since fear of change is usually associated with a sense of loss, most people would protest agile introduction and try to undermine any related efforts.

Specialized skills: Thirty plus years of serial development has resulted in people being trained in specialized areas of software engineering. This causes professionals to have highly specialized skills in a particular area while having minimal or no knowledge in other aspects of development. For example, project managers without an understanding of the underlying technologies used by their teams, programmers with no analysis and design modelling skills cannot be effective in delivering high quality software. To solve this problem professionals should train to become generalizing specialists so that they have specialized skills in one or more areas as well as basic understanding of the technical and business aspects of software development. The advantage of comprising teams of generalizing specialists is that everyone can use artefacts that are easily understood by all team members.

Outdated skills: Many people face a steep learning curve because they have not brushed upon their skills for several years. These people are also unlikely to be up-to-date about the latest happenings in the development of new technologies and processes. Cuts in education and training budgets by enterprises have contributed to the problem furthermore. People should be given enough time and training to improve their skill set.

Documentation-heavy mind-set: Many professionals have the wrong notion about developing effective software as being based on producing comprehensive and detailed requirement and design documents. Agile methods focus on code development over creating UML design artefacts. People should be taught about the agile approach to documentation and retrained to adopt the 'document as needed' method.

Do-it-all-at-once attitude: It is a serious mistake trying to adopt all agile practices in one go. To make the evolution of the work environment and the transition to agile practices smoother, agile techniques should be implemented one at a time.

4. Introducing Agile Methods

Strategies for successful adoption of agile methods include: obtaining management buy-in, education and support to employees, integrating agile practices to external processes, starting agile pilot projects, reporting and adapting agile project success, and sustenance of agility [2, 3].

Obtain management buy-in: The upper and middle management should be educated about the benefits of agile approaches. For an organization, a direct move to agile practices could be difficult. It is thus advisable to promote agile practices as recovery tactics for failing projects. The buy-in process for such projects would be easier as the desire to rescue the project and try a new approach is high. Managers faced with the dilemma that by using agile methods they would not be able to make product commitments to customers should be told that any estimates about cost, time, and functionality of previous projects were probably wrong, heavily padded, or both. After convincing management about their history of incorrect estimates, they would be more susceptible to trying agile methods. If an organization has a record of on-time delivery of software, then the management should be advised that agile practices could result in early completion, lesser consumption of resources, or both. Managers concerned with progress tracking issues in agile projects could be given model status reports (e.g., burn graphs, percentage of tests passed, commentary on project state etc.) based on fictional data of an enterprise using agile processes. Most managers are uncomfortable with agile projects going on forever, because iterations will continue as customers keep on identifying high-priority tasks. To solve this problem, the management could be convinced to sponsor a project for a certain period of time based on estimates of when the project would be completed and any additional funding could be provided after discussion at the end of that period.

Education and support: The management and the development team should be familiar with the principles, practices, and framework of agile methods before actually implementing them. Training and support should be provided to eliminate any misconceptions and tackle fear of change. People should also be taught that agile development is a highly disciplined approach and a half-hearted approach would not produce the desired results. Overenthusiastic teams should understand that decisions made without foresight will lead to eminent project failure. Empowered teams are an essential feature of agile methods. Responsibilities associated with empowered teams should be gradually communicated to developers. Because many agile practices require daily stand-up meetings, people consider agile methods as professing micromanagement. To avoid the problem, managers should be ready to answer any questions posed by developers and remove obstacles promptly. Managers should also not complain if some tasks take longer than scheduled.

Integrate to external processes: It is important that agile practices are tailored to integrate with existing non-agile practices in the enterprise. This means being ready to produce lower-value deliverables and creating plans or status reports in traditional formats for obtaining the necessary approval. Once the benefits of agile processes are clearly visible, negotiations on the balance of agile and non-agile approaches can be started. When introducing an

agile process, the management should understand the impact on groups outside the development group. The performance of the development team could be negatively impacted by other groups using non-agile processes. Managers should find ways of resolving differences of opinions among these teams. The human resource (HR) department should be informed about any development team moving to agile practices. The HR department could share concerns on the imprecise deliverables and dynamic scope of the project. The HR department and the team could work proactively to define tasks and deadlines that satisfy the HR department's need of deterministic action plans and also allow the project to remain agile.

Start pilot projects: Enterprises should use pilot projects to prove to themselves that agile methods works for them, use pathfinder projects to determine how to integrate agile processes into existing processes, and then move to production projects [11]. By using project suitability filters (as in Dynamic Systems Development Method), one can determine the appropriate project for the pilot. The pilot project should have articulate business sponsors (to show that the project had good sponsorship), deliver high business value (to show that agile practices can produce high-quality working software in short amount of time), and be of short duration (to see quick results).

Report and Adapt: The success of the pilot project could be showcased to attract more sponsorship for adopting agile practices. Management should raise awareness and remain open to feedback. They should use the experiences of the team involved in the pilot project to plan for future (pathfinder) projects. This team could now act as mentors for the teams working on subsequent agile projects.

Sustain Agility: Having adopted agile processes, it is necessary to sustain the agile work environment. Constant encouragement and stimulation should be provided to prevent people from going back to old approaches. An atmosphere of discussion should be promoted through special interest groups, conferences, practice sessions, etc. Moreover, the management should take up a stewarding role instead of a policing one (because screaming at the development team does not increase productivity).

5. Agile Adoption Strategies – Experiences from the Industry

There are two strategies for adopting agile methods in an organization, namely, wholesale and incremental [6]. Wholesale adoption involves adopting a set of agile practices all together. Proponents of agile adoption say that this approach allows developers to understand the synergy of the practices and the pitfalls of excluding supporting practices. Incremental adoption involves adoption of specific agile practices. Advocates of incremental adoption cite the need for a smooth organizational change. The next section presents a case study of an enterprise employing the wholesale approach for agile introduction [6].

5.1. Case Study – Control Systems Manufacturer

The control system manufacturer (CSM) began a full-scale agile project to update their aging control software for maintenance systems in nuclear power facilities. The development team comprised of eight developers, two quality assurance personnel, and a product manager. The team had independently researched agile processes and decided not to undergo any initial training in agile practices. Also, the team did not share a common work space and several members of the team were later shared with other non-agile projects.

After starting their first two-week iteration the team realized the need for a common working area, workstation configurations, and an integration and build environment. The team would often stop for *ad lib* training sessions as each activity required some learning. The first iteration delivered no business value. The team thought that their initial (bad) experiences were restricted to the first few iterations. They hoped of witnessing a surge in project velocity once they overcame the overhead associated with initial adoption. Furthermore, they endorsed that much of the learning was behind them (ignoring their acceptance that their learning curve was longer).

After ten weeks, recognizing that the project was facing difficulties, the technical manger intervened. The project review showed that only 20% of the projected business value had been

achieved. A series of meetings followed next, which concluded that an agile process should no longer be pursued.

Final retrospection by the team found that the adoption effort was not supplemented with substantial preparation (such as training and environment factors). One can incrementally prepare for agile adoption in such a case; however, the team miscalculated the cost of concurrent work and its negative effect on delivery of product functionality.

6. Introducing Agile Practices in Large Organizations

This section summarizes the experiences and findings of four large organizations, namely, ABB (a power and automation technologies company), Daimler-Chrysler (an automobile manufacturer), Motorola (a global communications company), and Nokia (a mobile telecommunications company), in implementing agile (extreme programming or XP) pilot projects [7, 9]. Practitioners at ABB evaluated various software development methodologies and conducted several studies of pilot XP projects. On many occasions the development teams used in-house hybrid models that combined agile and traditional practices. Such hybrid models used selected XP practices and were applied in those projects where a complete agile implementation was not possible. DaimlerChrysler practitioners extended their development approaches by integrating selective XP practices. Several XP practice studies were conducted to understand the interaction of XP principles and the organization environment. Motorola practitioners used XP for developing large safety-critical systems with high quality requirements. Pilot projects were used to test the suitability of XP in creating embedded systems. Nokia developers often adopted XP practices for project. In many occasions, developers custom designed hybrid approaches suited for specific projects.

6.1. Challenges

The primary challenge in adopting agile practices in large organizations is integration of agile projects with the project environment's existing processes [9].

All organizations understood the need for tailoring XP to their particular requirements. Also, the organizations agreed that there was no one-size-fits-all software development methodology. Eight incompatibilities between the XP pilot projects and the environment were identified: customers, requirements, architecture, legacy systems, other teams, quality systems, change control board (CCB), and organizational software processes. To maximize efficiency in software development, these incompatibilities must be resolved.

Large organizations usually distribute teams across several physical locations. Motorola found that agile teams had difficulties interfacing with other teams using traditional practices. Facing similar problems, Nokia advocated minimizing cross-team communication. This might suit teams developing independent sub-systems, however, alignment and architecture consistency could be difficult to achieve. Cross-team communication facilitation workshop could serve as another solution to the problem. Nokia reported success in organizing workshops with people from different development teams coming together to perform specific tasks for solving communication problems.

Agile methods encourage continuous refactoring for enhancing the quality of the software. Refactoring is made easy by allowing collective ownership of code where any developer can change the code to implement the changed requirements without design breakage. However, this practice could clash with existing quality control systems in large organizations, such the CCB that oversees modifications to the source code. Motorola experienced a similar clash of interests. To allow refactoring and considering the issues raised by the CCB, the management asked team members to first communicate any refactoring ideas to the CCB and upon approval implement them.

Motorola's agile teams also faced difficulties when performing regular system integration. The CCB exerted its power in controlling which changes would be integrated into the next build. This reduced the project's agility as the rate of code and fix of the development team variedly differed from how often the CCB approved the fixes.

Large organizations follow well-defined software processes which could result in double work if the project applies new practices that

have not properly blended in with existing traditional processes. ABB experienced this problem when defining the scope and delivery times of an XP project. The company developed the project plans up-front, without consulting the development team and wanted that documentation be frozen before the design and implementation phases. This clashed with the principles of XP and impeded the project velocity. Customer requirements at ABB were collected in the traditional way. This meant that the requirements were produced without the participation of the development team and did not take the form of user stories. The team had to decompose the requirements twice; once to satisfy the traditional processes as required by the organization, and then into prioritized user stories for the iterations.

Formal reviews are important tool for quality management in large organizations. XP affirms that using pair programming can eliminate the need for formal reviews. Motorola, however, found that in complex and large projects a pair of developers can not consider all effects on the entire system. Motorola implemented formal review processes to go alongside pair programming.

6.2. Future Prospects

All four organizations agreed that agile processes can somehow be used in software development. ABB concluded that XP could be tried in organizations on a small scale. Implementing XP on a grand scale would require changes to ABB's work environment, culture, and quality management system.

DaimlerChrysler found that mature processes like formal reviews, quality management, and measurement are essential for large organizations. Agile methods can allow such organizations to become amenable to change. DaimlerChrysler dismissed concerns that agile practices could lead to chaos by noting that already existing mature processes would not allow this to happen.

Motorola found that agile practices could be used to develop safety critical software; however, integrating agile into mainstream practices could be difficult for those opposed to change. Appropriate planning and training can produce positive results.

Nokia observed that XP works best with small, independent, and collocated teams. They found that hybrid approaches to software development are a more favourable option.

7. Determinants of Change in Large Organizations

In 2002, Qwest Communications, a fortune 500 telecommunications company, decided to adopt XP for its 5000-strong employee IT department. The organization moved to a 3-month deployment lifecycle from a year's worth of development efforts. Spayd describes an organizational change framework used in the Qwest example [12]. Sponsorship and leadership play a central role in the change, which is surrounded by eight principles that can make the transition smoother: sponsorship, burning platform and vision, effective development team, following change methodology, engaging stakeholders, aligning organization, empowering employees, and long-term focus.

Obtain committed executive sponsorship: For large scale adoption of a new methodology, active sponsorship from the Chief Information Officer (CIO) or higher must be attained. Receiving this sponsorship is the single most important factor for a change initiative. Previous experience reports have shown that without ardent participation from the CIO a large organization is unlikely to embrace a complete change.

Articulate the 'burning platform' required for change: The concept of 'burning platform' postulates a scenario where someone is forced to make a decision because of unfavourable and rapid changes in the environment surrounding the person. Executives and employees must believe that they have met their 'burning platform' decision; there remains no choice but to adopt the new methodology. Without introducing this sense of urgency employees are likely to be less interested in a change. Possible burning platform scenarios that can be thought about are poor ROI, consistent failure in delivering software on-time and on-budget, unhappy customers, etc.

Form and maintain effective change team: Having effective project teams is essential for an organizational change. The team should be characterized by brilliance, cohesion, good communication skills, correct decision-making skills, and accountability. Defining clear purpose and challenging goals is also important.

Adhere to key project and change management practices: The organizational change should be followed according to a change methodology. By monitoring project metrics, managing project scope, and understanding dynamics of change one can design an effective change methodology.

Keep stakeholders engaged throughout the lifecycle: Stakeholder is a person who is affected by the change, such as developers, testers, analysts, customers, managers, executives, etc. Executives, middle management, and non-management opinion leaders are key stakeholders whose support is critical for adopting change. Executive stakeholders are the upper managers whose approval and sponsorship for change is important. Middle managers are significantly affected by change as they control key resources in an organization. Shifting to an agile environment could alter their power balance. Non-management opinion leaders like developers, testers, analysts, HR representatives, finance personnel, etc. should be frequently communicated with for receiving the necessary buy-in.

Align organization element: It must be made sure that existing HR practices, work design, organizational structure etc. do not undermine the organization change process. Any risks arising because of these issues should be mitigated at the earliest. Dealing with organizational politics is a difficult task as well. Devising an effective deployment strategy and working on the environment dynamics can be useful in this case.

Empower affected employees in adopting the new system: Providing training and support to affected employees can reduce confusion, fear, and speculation. Frequent multi-way communication is also essential for getting the desired support from employees. Developers have to become more user-centric. Also, newer portfolios can be created and professionals retrained to work with their new designation.

Maintain long term focus: To sustain agility in an organization sponsors should maintain long term focus. The change process can take several years and should be determined by the number of professionals trained or number of people talking about agile techniques. Key stakeholders should train and remain up-to-date for any new challenges ahead.

8. Limitations of Agile Software Processes

Agile methods have been projected as a mechanism for early and quick production of working software. Advocates of agile processes argue that the source code is the most important deliverable and that it should take preference over analysis, design, and documentation. On the other hand, critics refer to the notion of 'corporate memory loss' which could be caused if there is little emphasis on producing good design models and documents (especially in case of large complex software) based on empirical analysis, experience reports, and researching the agile manifesto, Turk et al. suggest the following areas of limitations in agile processes: distributed environments, building reusable artefacts, large teams, and developing safety-critical software [8, 13].

Limited support for distributed environments: Face-to-face communication is an essential characteristic of agile processes. However, the desire of organizations to outsource jobs to foreign markets to reduce production costs does not fit well into the scope of agile methods. Moore et al. report that offshore outsourcing can increase productivity by 44% and internal customer satisfaction by 20%, and reduce time-to-complete by 37% and labour costs by 40% [10]. Communication among distributed teams is a challenge and could be approximated through video conferencing etc., however, that would not be as effective as personal interaction. It is important that onshore and offshore teams are treated as one integrated team and that there are at least few hours of overlap among working hours of the teams. The offshore and onshore teams should visit each other in the first few iterations to reduce gaps in communication. Producing and maintaining good documentation can allow distributed teams to keep a single view of the

project. Even with these recommendations, it is extremely difficult to combine offshore and agile development.

Limited support for building reusable artefacts: A case against agile processes is that they target specific problems in the software and are deficient in producing generalized solutions. The development of generalized solutions and reusable software can be answered using the reuse-oriented framework called Experience Factory [12]. Reusable artefacts have a strong emphasis on product quality because of their use in numerous applications which can be impacted as a result of error-prone components. Agile processes can be used to produce high quality software, however, it is unknown how they can be applied to reusable artefacts.

Limited support for development involving large teams: Agile processes tend to fare well in small collocated teams. With more developers in the team, team members have to be located in different physical locations. In such cases (rather infrequent) informal face-to-face communication has to be supplemented by daily stand-up meetings (as in Scrum [18]) etc. Traditional approaches seem to be better fit for managing large teams. Having architecture-centric development, change control mechanism, and good documenting practices could serve as effective means of bridging the communication gaps and latency. However, these strategies can affect the agility of the team.

Limited support for developing safety-critical software: Agile quality control mechanisms like pair programming, informal reviews, etc. have been unable to assure users about the safety of the finished product. Existing agile practices like test-first and incremental development and pair programming contribute to the overall quality of the software. Agile processes can be combined with formal techniques, such as model checking and reliability testing, to produce life and business critical software.

9. Conclusions

Several challenges have inhibited the widespread adoption of agile practices by the software industry. Understanding the dynamics of organizational change is key to successful adoption of new methodologies. The success of agile adoption is directly related to how the new methodologies are introduced in the organization. Research efforts that evaluate agile practices have applied non-systematic and non-scientific processes for collection of data. Since new adopters need solid proof of this methodology being advantageous, any informal study is likely to be rejected.

The question of agile practices crossing the chasm is open. If the academia considers this methodology to be promising and introduces relevant courses at the university-level, then one can conjecture that the new generation of software professionals may be more accepting of the concept of agility. This transformation will take years and it is possible that during this period other promising technologies may arise and challenge the agility paradigm.

References

[1] S. AMBLER, Remaining Agile, *AmbySoft White Paper*, 2005, Available at: `http://www.agilemodeling.com/essays/remainingAgile.htm`

[2] M. COHN AND D. FORD, Introducing an Agile Process to an Organization, *IEEE Computer*, 2003, Vol. 36, No. 6, pp. 74–78.

[3] M. GRIFFITHS, Crossing the Agile Chasm: DSDM as an Enterprise Friendly Wrapper for Agile Development, *Quadrus Development White Paper*, available at: `http://www.dsdm.org/knowledgebase/download/91/crossing_the_agile_chasm.pdf`

[4] S. HAYES, Why use Agile Methods?, *ZDNet Australia Article*, 2003, available at: `http://www.builderau.com.au/program/development/0,39024626,20275975,00.htm`

[5] J. HIGHSMITH, Objections to Agile Development, *Cutter Consortium White Paper*, 2004, available at: `http://paginas.fe.up.pt/~aaguiar/mads/AgileObjections.pdf`

[6] P. HODGETTS, Refactoring the Development Process: Experiences with the Incremental Adoption of Agile Practices, *Proc. of Agile Development Conference*, 2004, pp. 106–113, Salt Lake City, USA.

[7] T. KAHKONEN, Agile Methods for Large Organizations – Building Communities of Practice, *Proc. of Agile Development Conference*, 2004, pp. 2–10, Salt Lake City, USA.

[8] G. KEEFER, Extreme Programming Considered Harmful for Reliable Software Development, *AVOCA GmbH White Paper*, 2002, available at: http://www.avoca-vsm.com/Dateien-Download/ExtremeProgramming.pdf

[9] M. LINDVALL ET. AL, Agile Software Development in Large Organizations, *IEEE Computer*, 2004, Vol. 37, No. 12, pp. 26–34.

[10] S. MOORE AND L. BARNETT, Offshore Outsourcing and Agile Development, *Forrester Research Report*, 2004, available at: http://www.forrester.com/Research/Document/Excerpt/0,7211,34978,00.html

[11] D. REIFER, How Good Are Agile Methods?, *IEEE Software*, 2002, Vol. 19, No. 4, pp. 16–18.

[12] M. SPAYD, Evolving Agile in the Enterprise: Implementing XP on a Grand Scale, *Proc. of Agile Development Conference*, 2003, pp. 6–70, Salt Lake City, USA.

[13] D. TURK AND R. FRANCE AND B. RUMPE, Limitations of Agile Software Processes, *Proc. of XP Conference*, 2002, pp. 43–46, Alghero, Italy.

[14] J. VECCHIO, The Tech Adoption Life Cycle, *Website*, 2000, available at: http://www.fool.com/research/2000/foolsden000825.htm

[15] Cisco Systems, The Five Phases of Successful Service Development, *Website*, 2000, available at: http://chimera.noanet.net/washington/members/ppp_PhasesofServiceDev/

[16] G. MOORE, Crossing the Chasm: Marketing and Selling High-Tech Products to Mainstream Customers, *HarperBusiness*, 1999.

[17] J. HIGHSMITH, Agile Project Management: Creating Innovative Products, *Addison-Wesley Professional*, 2004.

[18] K. SCHWABER, Agile Project Management with Scrum, *Microsoft Press*, 2004.

[19] W. ROYCE, Managing the Development of Large Software Systems: Concepts and Techniques, *Proc. International Conference on Software Engineering*, 1987, pp. 328–338, Monterey, USA.

Received: October, 2005
Accepted: November, 2005

Contact adress:
Aniket Mahanti
Department of Computer Science
University of Calgary
2500 University Drive N.W.
Calgary, Alberta T2N 1N4
Canada
e-mail: amahanti@cpsc.ucalgary.ca

ANIKET MAHANTI is a M.Sc. student in the Department of Computer Science, University of Calgary, Calgary, Canada. He graduated with a B.Sc. (Honours) in Computer Science from the University of New Brunswick, Saint John, Canada in May, 2004. He is the holder of post graduate scholarships awarded by the Natural Sciences and Engineering Council of Canada and Informatics Circle of Research of Alberta. His areas of interest include software engineering, data mining, and wireless networks.