**Communications of the Association for Information Systems**

Volume 27 | Number 1 Article 21

August 2010

# Balancing Agile and Structured Development Approaches to Successfully Manage Large Distributed Software Projects: A Case Study from the Cruise Line Industry

Dinesh Batra
*Decision Sciences and Information Systems, Florida International University*, batra@fiu.edu

Weidong Xia
*Decision Sciences and Information Systems, Florida International University*

Debra VanderMeer
*Decision Sciences and Information Systems, Florida International University*

Kaushik Dutta
*Decision Sciences and Information Systems, Florida International University*

Communications of the Association for Information Systems

CAIS

# Balancing Agile and Structured Development Approaches to Successfully Manage Large Distributed Software Projects: A Case Study from the Cruise Line Industry

Dinesh Batra

*Decision Sciences and Information Systems, Florida International University*

*batra@fiu.edu*

Weidong Xia

*Decision Sciences and Information Systems, Florida International University*

Debra VanderMeer

*Decision Sciences and Information Systems, Florida International University*

Kaushik Dutta

*Decision Sciences and Information Systems, Florida International University*

Abstract:

Agile methods and traditional structured approaches are often viewed as competing bi-polar choices. Agile methods such as Scrum and XP are recommended for small, co-located projects that involve changing requirements. The traditional structured plan-driven approaches, such as the Capability Maturity Model (CMM) and the waterfall lifecycle frameworks, are recommended for large projects with stable requirements. If a project is large, strategically important, distributed, and has dynamic user requirements and organizational changes, it presents unique challenges that neither the agile methods nor the traditional structured approaches can effectively deal with alone. Although there is an increasing call for a balanced approach, there is little empirical research that shows when and how the two approaches can complement each other. Based on a case study from the cruise line industry of a large distributed strategic project with unanticipated changes, we conclude that this balance is not only workable, but is essential to ensure that the project demonstrates both control and agility for achieving its challenging and dynamic goals. Agile without structure can cause chaos, particularly in large complex distributed projects where planning, control, and coordination are critical. Structure without agility can lead to rigidity, particularly when a project involves a great deal of learning, discovery, and changes.

**Keywords:** organization; project; development approach; stakeholder involvement; outsourcing; administrative functional system; transaction system; case study

Volume 27, Article 21, pp. 379-394, August 2010

## I. INTRODUCTION

Agile methods and traditional structured approaches are often viewed as competing bipolar choices [Boehm and Turner, 2004; Vidgen and Wang, 2009]. We can describe agility, characterized in part by agile methods such as eXtreme Programming (XP) [Beck, 2000], Scrum [Schwaber, 2004], Crystal [Cockburn, 2004], and Adaptive Software Development [Highsmith, 1999], as iterative and evolutionary in development, planning, and delivery to allow for rapid and flexible response to changes [Larman, 2004]. The Agile Manifesto articulates a common set of principles and beliefs underlying these methods [Cockburn, 2002]. In contrast, the structured approach is characterized by plan-driven-heavy methods usually characterized by waterfall lifecycle frameworks, and CMM-like standards and steps that are largely process-based with strict change control requirements [Ahern et al., 2003; Curtis et al., 2002; Deming, 1986].

There are different assumptions underlying the agile and traditional development approaches [Turk et al., 2005; Nerur and Balijepally, 2007]. As pointed out by Boehm and Turner [2004], traditional structured development approaches are suitable for large, critical, and complex projects with stable and predictable requirements. On the other hand, agile approaches are suitable for projects with high degrees of uncertainty and risk arising from unstable requirements and evolving project goals. In practice, while agile methods have been used successfully for small, co-located projects to quickly respond to changes [Lee et al., 2006], their applications in large, distributed projects have been challenging [Ramesh et al., 2006]. Chow and Cao [2008] studied 109 agile projects and found that nearly 80 percent of the project teams had fewer than twenty members. Bose [2008] studied twelve agile projects in the context of the application of agile principles in distributed projects, and noted that the majority of the projects were small in size. On the other hand, while structured approaches provide the organizational foundation needed for effectively planning, controlling, and coordinating large distributed projects [Kishore et al, 2003], they tend to be bureaucratic and rigid in embracing and responding to changes.

Projects that are large as well as distributed are becoming more prevalent [Herbsleb and Mockus, 2003]. Change is common in large projects as well; the case where the entirety of a project's complexity is understood in the early stages is quite rare [Benbya and McKelvey, 2006]. Large, distributed projects that involve evolving user requirements present a unique challenge that neither agile methods nor structured approaches alone can effectively address. In addition, organizations engage in a wide variety of systems development projects that are embedded in uniquely complex organizational contexts and contingencies. No one development approach can exclusively address all challenges in a particular project [Ramesh et al., 2007]. As Boehm and Turner [2004] suggest, there is a need to develop software development methods that balance agility and structure. However, simultaneously applying both agile and structured approaches in the same project is challenging for organizations because of the different, often conflicting, assumptions and principles that are inherent to the agile and structured approaches. Consequently, although there has been an increasing recognition for the need to balance agile and structured approaches [Fernandez and Fernandez, 2008; Fitzgerald et al., 2006; Nord and Tomayko, 2006; Sarker and Sarker, 2009; Vinekar et al., 2006], there is a general lack of understanding about how the two approaches can be effectively balanced in the successful management of large, complex projects with changing requirements and unstable environments.

In this article, we present evidence that such a balance is indeed both necessary and feasible by reporting the case of a project from the cruise line industry (we will refer to the company as the ABC Cruise Line) that faced several serious challenges to the possibility of successful completion. The project primarily involved building a new web presence for the cruise line. The project started as an important but limited venture, but grew into one of strategic importance. The project scope expanded considerably, and there were a significant number of changes in terms of requirements, executive sponsors, and the structure of the project. The project appeared to have moved into the "challenged" category [Masticola, 2007]. It became evident that the project would be late and considerably over budget; however, these overruns had to be controlled. Budget overrun was better tolerated than time delay, and there was considerable pressure to minimize the postponement of completion even as the size of the project increased considerably. Beyond the anticipated budget overrun and time delay, the project had to be "successful," that is, meet the enhanced requirement specifications and demonstrate business results in terms of increased web traffic and revenue.

The conventional wisdom is that as the size of a project increases, there should be more use of the structured approach, which can foster better monitoring of costs [Boehm and Turner, 2004]. However, there was pressure to complete the project as quickly as possible while managing the evolving scope, which can rarely be achieved by using the structured approach alone. The project leaders at the ABC Cruiseline tried a novel experiment: embedding Scrum, an agile approach, within the framework of the PMBOK framework, a structured approach. The conventional wisdom is that this is not desirable or even possible. Yet, the judicious blending of the Scrum-based agile approach with PMBOK checks and balances resulted in a balanced approach, and the project was eventually completed and deemed successful, despite the cost overrun and schedule delay caused by the enhanced scope. The project was considered successful because it met user requirements, provided a strategic advantage, and was replicated at other international sites of the company.

This article describes the case and its challenges, and illustrates the blending of agile and structured project management practices the project leaders followed to take control of the project while responding to changing requirements and project circumstances. The types of problems they faced are common to many large, distributed projects. Thus, this case provides strong support for the idea that it is possible to balance structure and agility in software development projects. In fact, contemporary research should focus on how best the two practices can be harmonized for large, distributed projects. The article is organized as follows. We first describe the characteristics of the project and the key challenges it faced. We then discuss how the project used an agile approach to deal with some of the challenges and the extent to which the agile principles applied or did not apply in this project. A similar discussion follows on how key structured factors were employed in the project. We then discuss how they used both agile and structured approaches to deal with each of the key project issues. We complete the article by presenting a general framework that illustrates the key project characteristics that favor a hybrid approach.

## II. THE SOFTWARE PROJECT AND ITS CHALLENGES

The purpose of the project was to create a completely new web-based customer booking engine to replace an outdated engine. The new website was envisioned to have an attractive interface, dynamic features such as online booking and web cameras from the ships, functionalities that were intended to provide customers with a pleasant shopping experience, brochure materials, and a portal for travel agents. Although it was not initially stated, the project was eventually deemed as one of strategic importance to the company to build and sustain competitive advantage. The starting budget for the project allocated was 3 million dollars and completion time was planned for fourteen months. These estimates were soon found to be unreasonably off-target as the true scope of the project emerged.

Table 1 summarizes the key project characteristics and the associated challenges to the project team. Out of the thirteen dimensions listed, nine were taken from and corresponded to the nine areas of the Project Management Body of Knowledge [PMBOK, 2004], which provides the knowledge areas for project management and evaluation. Four additional dimensions—Objective, Size, Changes in Top Management, and Outsourcing, were included to take into consideration the dynamic context of the project. *Objective* was included as a dimension to capture the evolving nature of the project goals. *Size* was included because it is a key project characteristic that determines whether the agile approach or the structured approach should be used for a particular project. *Changes in Top Management* was included as a dimension because it became a critical risk factor that required the project team to be agile. *Outsourcing* was included to capture the distributed nature of the project and the need to manage different, often conflicting, interests and perspectives between the client and the vendor.

Early in the project, turnover of business sponsors caused changes and expansion in the scope and turmoil in user requirements. A project manager recounted:

> *When the project was first defined, although we had gotten business sponsors involved, because of organizational changes … when we injected new players they had new ideas, and the scope was constantly evolving; they weren't willing to commit and to be accountable….*

While the business sponsors requested increasing functionalities, they were not cognizant of the impact these changes would have on the project budget or timeline, leading to significant tension across the project teams. There were also technical problems because they committed to a technology in which they were not particularly proficient. The internal IT department had skills mainly in the .NET development environment. However, they committed to J2EE implementation for this project and soon encountered difficulty because of the lack of in-house skills. They then brought in an outsourcing vendor that specialized in J2EE implementations as well as in agile development. The expertise made available through the outsourcing arrangements was invaluable in completion of the project.

| Table 1: Characteristics of the Project | | |
|---|---|---|
| **Key dimensions** | **Project Situation** | **Challenges** |
| Objective | Strategic—Enhance competitive advantage | – Meeting business needs and systems functionality requirements were key; however, these changed significantly; schedule was more important than meeting budget goal. |
| Size | Large project that was partly outsourced | – Internal business sponsors, users, project managers, analysts, external developers from the UK and India |
| Scope | Uncertain and ill-defined in the planning stage; evolved significantly over time | – Sponsors/users did not understand and could not articulate the precise scope in the planning stage.<br>– User requirements were evolving and emerging over time because of industry factors and changes in top management. |
| Changes in top management | Changes of CEO, CIO and CFO during the project | – Changed project scope and functionality requirements caused re-analyses and redesign of processes and systems requirements. |
| Cost | Initially estimated to be $3 million; completed with $15 million | – Cost had to be adjusted (increased) over time because of the increased scope and changing requirements.<br>– Initial cost was unrealistic and couldn't be used as measure for project progress and performance, but cost still had to be controlled. |
| Schedule | Initially estimated to be 14 months and completed in 28 months | – Schedule had to be adjusted (increased) over time.<br>– Initial schedule was unrealistic and couldn't be used as measure for project progress and performance.<br>– There was considerable pressure to minimize delays. |
| Outsourcing | The project team lacked internal resources skilled in the needed technology platform. | – Outsourced development to an external vendor with offices mainly in UK and India. |
| Quality | Uncertain and ill-defined in the planning stage; became critical because of changing requirements and outsourcing | – Quality specifications had to be documented and enforced with the vendors.<br>– The number of quality assurance and control personnel in the project was inadequate . |
| Coordination/ integration management | Both formal and informal structures were needed to plan and manage evolving goals and multiple stakeholders across different locations and skills. | – Conflicting goals between scope, cost, schedule, quality and agility (ability to discover, learn, and respond to changes quickly) |
| Communications management | Both formal and informal structures and documents were needed to plan and manage communications process. | – Formal communication can slow down development.<br>– Tacit knowledge was difficult to attain in such a large project.<br>– Informal means of communication were ineffective when the outsourcing vendor is involved. |
| Risk management | Risk management was not formally in place in the planning stage and gradually evolved over time. | – Lack of risk analysis and planning lead to significant difficulties in responding to unanticipated changes in scope, cost, schedule, and quality.<br>– Responsive change management became a critical success factor. |
| Procurement management | Critical development skills were acquired through outsourcing to vendors in UK and India. | – Formal documents and contracts were needed for all changes.<br>– Project cost was increased because of increasing scopes beyond original contract. |
| Human resources management | Long working hours (70–80 hours per week) | – High work-related stress<br>– Unsustainable development pace |

There were several instances of increased functionality. Some of the user requirements were not defined clearly with the needed specificity in the beginning. For example, the system was supposed to provide "a pleasant experience" for customers. However, as they delved into detailed analyses, managers came up with all sorts of requirements, such as allowing customers to choose their cabin, buy things onboard, and engage in pre- and post-hotel and shore excursions (e.g., a customer on a Caribbean cruise might book a windsurfing shore excursion at one of the ports of call). They had also underestimated the security and privacy needs of the system, such as protecting customer credit-card information. To give the reader some perspective on the increased scope involved in some of these expanded requirements, let us consider the requirement to allow customers to select specific cabins. This requirement expanded from allowing users to select, for example, "cabin #8553," rather than a cabin in the aggregated class of "balcony cabin on a high deck." This means that the system had to support tracking an individual cabin's availability and ensure that a "reserved" cabin was not offered to another customer. This was a significant change over the standard hospitality practice of booking aggregated classes of rooms and assigning individual rooms/cabins at check-in time.

User-requirements changes of 25–35 percent are normally considered high [Larman, 2004]. In this project, requirements changes were about 60 percent. Further, the initial scope had been considerably expanded, resulting in a large, distributed project that was dynamic in terms of user requirements. However, given that the project had evolved into one with a strategic nature, this escalation was not deemed to be a major problem. The main issue was always whether or not the potential benefits would exceed the projected costs. The functional features of the project were the topmost priority, and the lack of agility, the ability to respond to changes, and completing the project in a reasonable period presented a major challenge.

The initial outlay of the project was 3 million dollars; by the time the project was complete, the actual cost had gone up to 15 million dollars, and the schedule had doubled from fourteen to twenty-eight months. Given the cost and time escalation, it may sound somewhat odd that the project was considered a success by the sponsors; however, there was a sense that the project had met the expanded requirements, and there was evidence that the increased revenue justified the cost. As the scope expanded, the costs went up but the perceived benefits also increased, and as long as the perceived benefits, both tangible and intangible, were more than the costs, the project was deemed successful.

> *…. And then we actually pulled out the numbers of what the old website to the new one, …how much revenue was coming in … we actually discovered that the revenue was definitely a good stream.*
>
> *I would think they're definitely happier 'cause they're asking money to now internationalize the website for the European offices.*

## III. THE AGILE ASPECT OF THE PROJECT

Agile development methods subscribe to four essential values: individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to changes over following a plan. The two most popular methods are eXtreme Programming (XP) [Beck, 2000] and Scrum [Schwaber, 2004]. XP is based on values and factors such as communication, simplicity, feedback, timeboxing, pair programming, unit testing, flat management structure, and expecting changes in requirements. Scrum development is based on a timeboxed development period called a *sprint,* maintaining a backlog of user requirements, self-organizing teams, verbal communication, expecting changes in requirements, accepting that the user requirements cannot be clarified without actually developing software, and having a ScrumMaster, who protects the developers from interference caused by new user requirement changes during the sprint. The project discussed in this study used a Scrum-based method.

The project leaders realized early that the project was likely to encounter significant changes in the user requirements. To address this issue, they proposed the use of Scrum and brought in Craig Larman, a renowned expert in agile methodologies to help them with training and implementing Scrum. However, the project was also getting bigger. Conventional wisdom suggests that large projects need to follow the structured approach [Boehm and Turner, 2004]. There was a concern that the project would lose discipline if an agile approach were instituted. Nevertheless, the project proceeded with somewhat of a calculated experiment by harmonizing the two approaches.

The agile approach as employed by the project did not completely subscribe to the Agile Manifesto; Table 2 describes the extent to which the project team chose to follow the agile principles. In particular, principles that value individuals and their workload were not supported. Principles such as excellence in design were supported but were ascribed to structured development. The vendor was unwilling to accept late changes or even accept user requirements that had not been carefully deliberated by the client. The deliberate focus on requirements is a useful

practice in agile development [Orr, 2004], although the popular opinion is that requirements can frequently be modified as development proceeds.

Many researchers view agile development as lacking in discipline. Yet, Scrum brought discipline into the project, which was losing control because of constant intervention from the senior management who were the primary sponsors of the project. The project had a two-week iteration cycle. At the start of each cycle, the sponsors and project managers decided on what to do in the next cycle, based on a wish list that ranked items by priority, cost, and time estimates. During the two-week cycle, the developers were not to be interrupted as per a popular Scrum practice.

> *And that means that the business every 2 weeks had to give what was gonna happen for the next 2 weeks for the development team, and during that 2 week we locked them down that—only what they had given during that prior 2 weeks is what was going to go in there. We weren't gonna go back during that 2 weeks and revamp it.*

Discipline was also ushered by forcing the senior management, which was initially tentative, to work with the developers and come up with user requirements. Several Vice Presidents were pulled out of their daily work and were asked to work closely with the project team, and to participate in prioritizing requested functionalities to prevent the introduction of ad-hoc changes. However, business people and developers did not work together on a daily basis, although business people could generally be contacted as needed by the developers. The role of an on-site customer can be stressful and cannot be sustained for a long period [Dyba and Dingsoyr, 2008]. While the senior managers were generally available, they also had to attend to their regular responsibilities. Yet, agile development invariably enhances communication [Holmstrom et al., 2006], and the increased interaction between managers and developers helped the project make sustained progress.

One of the principles of agile development—sustainable development—was not supported, however. The agile method XP recommends a regular forty-hour week [Cockburn, 2002]. In this project, developers typically worked seventy to eighty hours per week. While many of these developers were contractors and were compensated for their billable time, the long hours created work-related stress. Although there were some stress-relieving practices, such as parties and taking time off to visit families, the project team was under tremendous pressure to complete the project quickly. Other principles that were not supported include, "Welcome changing requirements, even late in development," "Build projects around motivated individuals," and "The best architectures, requirements, and designs emerge from self-organizing teams." Some principles were only partially supported.

Paradoxically, the greatest benefit of the agile method was that it brought discipline into the project. The requirement log book served as a communication mechanism for sensing and prioritizing changing requirements, while the sprint period provided the stability that was needed to implement a select set of requirements. Project managers were willing to learn and compromise so that agile principles themselves did not become a straightjacket by forcing a "CMMI Level 6" kind of rigidity.

> *And for us it wasn't the book said some things, that the whole Agile Manifesto was a proper process if only you're like CMMI Level 6…. But that doesn't exist in the real world, so you have to actually then look at the Kaizen model of, lean and agile, and do what fits and what adds value and not just do it because some process says to do it.*

## IV. THE STRUCTURED ASPECT OF THE PROJECT

"As agility in responding to continual change in technological and business conditions has become critical to success, organizations must strive to create learning environments capable of rapidly adjusting to the changes engulfing them. A critical component of agility is a workforce with the knowledge and skills to make rapid adjustments and the willingness to acquire new competencies" [Curtis, Heffler, and Miller, 2002; page xi]. This quote is not from the proponents of the Agile Manifesto, but it is from the proponents of the People Capability Maturity Model (People CMM), which is a tool to help organizations successfully address the critical people issues. Even the structured approach intends to embrace agility.

The structured approach is based on CMMI [Aherns et al., 2003] and People CMM [Curtis et al., 2002] as well as planning-heavy methods that are usually built around waterfall lifecycle frameworks. CMMI refers to Capability Maturity Model Integration and helps integrate traditionally separate organizational functions, set process improvement goals and priorities, provide guidance for quality processes, and provide a point of reference for appraising current processes. Although there is no formal "structure manifesto," several researchers have outlined the key characteristics of the structured approach. Nerur et al. [2005] summarize the following characteristics of the structured approach: specifiable and predictable systems that can be built through meticulous planning, lifecycle

model, process-centric control, command-and-control management style, explicit documentation, specialized roles, formal communication, and important but not critical customer involvement. The high-level command-and-control management is achieved by the use of a steering committee. The People CMM guidelines advocate improvement in individual and workgroup processes; for this study, we interpreted these in terms of developer training and team management.

**Table 2: Evaluation of the Project Based on Agile Principles**

| Agile Manifesto principle | Support | Agile practices in the ABC case |
|---|---|---|
| 1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software. | Supported | – By following Scrum, the developers were able to continually provide working software. |
| 2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage. | Not Supported | – Requirements changed during early and middle stages; good design avoided late changes. The vendors had contracts that allowed only limited changes; however, the sponsors acknowledged and accepted cost implications in the case of changes. |
| 3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale. | Supported | – Two weeks was the norm for the project, governed by Scrum practices. (Some Scrum practices recommend a one-month time frame.) |
| 4. Business people and developers must work together daily throughout the project. | Partially Supported | – The stakeholders followed the principle when they understood that user requirements and changes were being handled in an ad-hoc fashion.<br>– Business people were forced to attend to developer queries, although the two groups did not always work together daily throughout the project. |
| 5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done. | Not Supported | – Outsourcing enabled them to obtain the needed technical capabilities but did not allow direct control over developer motivation.<br>– However, developers were competent, did work hard, and were compensated for the extra time, even though the compensation rate was ordinary. |
| 6. The most efficient and effective method of conveying information to, and within, a development team is face-to-face conversation. | Partially Supported | – The project involved distributed development. Some developers from the vendor were stationed at the client site while the others worked remotely. |
| 7. Working software is the primary measure of progress. | Supported | – Scrum practices enforced discipline in creating working code on a regular basis. |
| 8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely. | Not Supported | – Developers worked about 70–80 hours/week, and there were signs of burnout. |
| 9. Continuous attention to technical excellence and good design enhances agility. | Partially Supported | – They paid a lot of attention to design, but it is not clear if it enhanced agility.<br>– Good design ensured that requirements did not have to be modified at the implementation stage. |
| 10. Simplicity—the art of maximizing the amount of work not done—is essential. | Partially Supported | – They skirted scope and user requirements issues early, and this resulted in some bad decisions.<br>– Documentation was required because vendors needed it.<br>– However, for the most part, they kept things simple to avoid further escalation in costs. |
| 11. The best architectures, requirements, and designs emerge from self-organizing teams. | Not Supported: | – The ScrumMaster felt that many developers did not under-stand this principle, and sometimes abused the freedom.<br>– The relationship with the developers from the vendor side was reasonably formal. |
| 12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly. | Supported | – Failures in the earlier stages became a driver for the team to get into a practice of reflection. Initially, stakeholders were not involved, and the project was moving in an ad-hoc manner.<br>– The team changed its behavior and showed considerable propensity to adjust. |

Although the project essentially followed agile development, its size and outsourcing requirements necessitated that several structured principles work in harmony with agile principles. Some of the key structure-based practices are listed in Table 3. The structured principles were rarely applied without adaptations. For example, although the lifecycle notion was involved, it was implemented in small increments instead of a single waterfall lifecycle, and although there was a steering committee, their decisions were expected in a short timeframe. Although documentation was considered important, just-enough documentation was the norm.

The majority of the development was outsourced to a UK-based company with Java developers based mainly in India. Some of the developers were stationed at the client site. As mentioned earlier, the vendor was very experienced in agile practices, and this alignment helped the project. The outsourcing contract was on time-and-materials terms, and this arrangement required close supervision by the client. The need to manage outsourcing on an agile platform resulted in a hybrid approach. The structured principles acted as a discipline umbrella under which agile development could thrive.

**Table 3: Evaluation of the Project Based on Structured Principles**

| Structure Factors | Support | Structured Practices in the ABC Case |
|---|---|---|
| Lifecycle stages | Partially Supported | Although they were following Scrum, they did not proceed to code right away, as is the practice in a typical agile method. They went through user requirements, analysis, and design before embarking on coding, although this was done in small increments. The life cycle elements were embedded in a Rational Unified Process (RUP) kind of method. However, the waterfall process was adapted to fit the project needs and context. |
| Focus on careful planning, analysis, and design | Partially Supported | Analysis/Design brought discipline into the project. They were very deliberate on design and used a number of representations to communicate and discover. They felt that the user requirements were too nebulous and that analysis/design allowed them to think through the requirements. A number of requirement-change decisions were made at the design stage. But when a change was to be made, they made the decision quickly. Some requirements emerged as a result of periodic deliverables. |
| Use of Steering Committee for command and control | Partially Supported | Although the steering committee was established to make high-level decisions, the decisions were made in an agile rather than a structured manner. The steering committee was expected to be fully involved in the project. They made important decisions regarding scope and budget approval, but were expected to be agile in their response, sometimes in a 24-hour turnaround period. This brought both structure and agility to the project decision-making process. |
| Use of outsourcing | Supported | Outsourcing brought in much-needed skills, but the portions that were outsourced had to be thought through because of the specificity of user requirements as part of the contract. |
| Formal cost management and approvals | Partially Supported | Since the project was about developing and sustaining competitive advantage, cost implications resulting from changes in user requirements were accepted. A cost approval policy was in place, and although formal, it was quite nimble. Delays that are normally acceptable in a regular structured project were not tolerated. |
| Training | Partially Supported | Training on Scrum for developers and business personnel was conducted to primarily provide a shared understanding and a common framework and method for managing user requirements, team building, and delivering working software. Otherwise, training was not a priority. |
| Team management | Supported | Individuals on the client side were empowered, but teamwork was a mandate and management had no issues confronting ego-centric individuals. |
| Explicit documentation | Partially Supported | Maintaining just-enough documentation was the project norm. Senior managers were accessible and developers could contact them for decisions. |

To commence development, the vendor needed precise user requirements. However, the astute vendor, who was doing the logical design (or analysis) soon realized that the users, who were from sales and marketing areas, had not understood the requirements and needed more visual techniques to discover and define their requirements.

> *… all these marketing and sales peoples, we discovered, were more visual, so although we did story boards it wasn't until we actually started doing wire frames to actually—for them to visualize it.*

There was a significant emphasis on logical design, a practice more consistent with a structured approach. It was essential to get the logical design correct before beginning to code because a number of changes in the user requirements were discovered at this stage as a more detailed picture emerged.

> *On the logical design, you really need to take your time. So even though you want to do agile, you have to still think it through, that what you do now is not gonna bite you later on….*

The project leaders did not equate agile with coding. They felt that in a large project, they could use agile development, but they still needed to apply the basic project management principles. Interestingly, the project members found that agile development methods can usher in discipline, in a way like waterfall in small steps.

> *… but people are so sometimes misconstruing that agile is just quick and dirty, so they miss some of the steps. I would say agile's not so different from project management, from the PMBOK, just the waterfall methodology, and if you do not have the PMBOK, which a lot of people think is unnecessary in agile, they have nothing. They're completely misconstruing all the basic principles.*

Key functionalities of project management, such as change, cost, scope management, were achieved using a change control board, which was under the steering committee that was composed of senior managers. A key issue the project team faced was to make change management agile. That required a new mindset on the project stakeholders' parts; they needed to have the willingness to embrace changes and make decisions quickly. There was a formal cost approval policy in place, and the decision-makers could be convened quickly so that when decisions were required, they could be made quickly. Our finding in this case stud is consistent with past research which has shown that decision making time is a critical success factor in attaining agility [Misra et al., 2009].

## V. BALANCING AGILE AND STRUCTURED APPROACHES

The software project demonstrated that agile and structured approaches can work together. This is consistent with the recommendation that agile methods should be flexibly tailored to the particular development context to achieve maximum effect [Fitzgerald et al, 2006]. In Table 4, we consider each of the challenges introduced in Table 1 in the context of how (or whether) the agile and structured approaches were used in complementary ways to address them. The assessment shows that the project had a blend of the two approaches. In general, one approach could compensate where another was weak.

Neither the agile nor the structured approach alone can completely address all of the challenges. The agile approach is strong on some dimensions but weak on others. It provides a mindset that is open to change [Boehm and Turner, 2004; Vinekar et al., 2006], and is very customer- and goal-focused. Since the agile approach requires that users work closely with developers, user requirements can be obtained and corrected easily in a timely manner. The agile approach promotes and depend on teamwork, facilitate the deliveries of periodic working software, and emphasizes simplicity which in turn facilitates agility and at the same time controls costs.

In this project, the agile approach was found to be weak in other respects. For example, consistent with the findings of Erickson et al [2005], sustainable development (e.g., forty-hour or so workweek) was not being followed. Although it may be possible in theory to attain the dual maxims of customer satisfaction and sustainable development, in practice, there may be tradeoffs between the two approaches. Several agile principles were not supported or only partially supported. The agile approach is also limited in dealing with traditional project management issues. If the user requirements are not managed, cost and schedule implications are likely to be significant. Poorly defined user requirements in the early planning stage often cause costly rework. In addition, new requirement changes that user proposed in the later implementation stage may create new issues that are difficult to deal with. Such issues require top management decision making and support. The agile approach needs to be adapted to accommodate the need to document communication, coordination, and control issues in an outsourced project [Ågerfalk and Fitzgerald, 2006].

Similarly, the structured approaches have both pros and cons. The PMBOK's focus on schedule and costs ensures that these issues are appropriately planned and controlled, especially when the project scope is constantly evolving. Structures such as a steering committee and change overview board can provide decision making so that there are resources to carry on the project. Outsourcing contracts require a board that can ensure governance and implementation. The decision to outsource usually instills a discipline that results in detailed user requirements. Analysis and design are also important in a large project. Even if documentation is minimal, it helps construct a holistic picture of the project. Good documentation practices also enable the project team to effectively manage knowledge, which can be a critical success factor when there are team member turnovers.

| Table 4: Balancing Agile and Structured to Face Project Challenges | | | |
|---|---|---|---|
| **Key dimensions** | **Challenges** | **Agile in Response** | **Structure in Response** |
| Project Objective | Meeting business needs and systems functionality requirements were key; meeting schedule was more important than meeting budget. | The agile approach gave the team the freedom to make changes rather than blindly stick to the original plan. | The steering committee approved additional funds for additional scope, but only when the team was able to show strategic benefit. |
| Project Size | The project involved many internal business sponsors, users, project managers, analysts, and external developers from UK and India. | Scrum helped reduced the complexity caused by the project size by timeboxing development and by limiting ad hoc changes. | Contracts with the vendor provided needed resources with appropriate technology and methodology skills. |
| Scope | Users did not understand and could not articulate requirements in the planning stage; user requirements were evolving and emerging over time. | Agile allowed the team to take a step back, and realize that change was to be expected and managed; scrum cycles protected developers from ad hoc interruptions within each cycle. | In redesign, requirements were documented, and served as an agreement between the sponsors and the developers to describe expected functionality.<br>Contracts with the vendor imposed structure in requirement determination and logical design. |
| Changes in top management | Changed project scope and functionality requirements caused re-analyses and redesign of processes and systems requirements. | The use of Scrum helped reduce the impact caused by changes in top management. | Priority list based on Scrum backlog added structure to requirement determination and decision-making.<br>Contracts with the vendor imposed structure in requirements and logical design. |
| Cost | Cost had to be adjusted (increased) over time; initial cost estimate was unrealistic and couldn't be used as measure for project progress and performance. | Steering committee could be convened within 24 hours to make a decision. | Steering committee evaluated requests for funds. |
| Schedule | Schedule had to be adjusted (increased) over time; initial schedule was unrealistic and couldn't be used as measure for project progress and performance. | Each request for additional funding was accompanied by an estimate of the additional time required, keeping all stakeholders informed. | The team was expected to adhere to the modified timelines. |
| Quality | Quality of system design and implementation depended on the quality of requirement analyses and specifications.<br>Quality specifications had to be documented and enforced with the vendor. | Providing working software continually facilitated evaluation. | Internal processes and vendor requirements forced the team to document expectations to ensure that all parties (developers, sponsors, and vendor) were on the same page. |
| Coordination and integration Management | There were conflicting goals between scope, cost, schedule, quality and agility (ability to discover, learn and respond to changes quickly). | The project evolved to become a significant strategic priority for the company. Decision-makers made it a priority to attend to any required decisions quickly. | All decisions were documented at some level of detail to keep track of what was decided. |
| Communica-tions management | Formal structure can slow down development. | Scrum dictated a two-week cycle starting with a decision-making meeting, after which developers were allowed to work without interruption. Developers had access to and were able to communicate with client managers as needed. | Major decisions and user requirements were documented to keep track of the project, and to meet the vendor's requirements. |
| Risk management | Lack of risk analysis and planning led to significant difficulties in responding to unanticipated changes in scope, cost, schedule and quality; change management became a critical project factor. | Agile decision-making and learning allowed the team to reassess what wasn't working during the project, and make necessary adjustments to the process. | Outsourcing development to the vendor helped migrate the risk caused by lacking internal resources with the needed technology and methodology skills. |
| Procurement management | Formal documents and contracts were needed for all changes. Expanded scopes beyond the original contract caused cost overrun. | Agile principles were applied in procurement. | A formal, documented change management process was established to coordinate with the vendor, which added structure to the project. |
| Human resources management | High work-related stress; unsustainable development pace. | Agile principles were not applied in managing workload. | Structured methods were not applied in planning and controlling the workload. |

The structured approach, however, can be limiting due to its inherent command and control structure that can easily become very bureaucratic. The agile approach accommodates change and ensures that the project continues to make progress despite the challenges caused by changes in scope, management, and user requirements. Agile principles thus eliminate the weaknesses of the structured approach in dealing with project management maxims. This issue was summed up by an example given by a manager:

> *You know what, in Day 20 I discovered that what I said in Day 1 was not accurate, there's a verbal addendum there, agile allows me to do that. And I don't have to jump through 20 hoops to make that modification. So basically agile fills in the hole—a kind of a disadvantage that PM or more traditional methodology had….*

Given the significant schedule delay and increased cost, one may question whether the project was successful. The project leaders felt that it was a success because it fared well in a cost-benefit analysis despite the increased time and cost. The initial estimates were based on a much smaller scope, and the expanded scope generally resulted strategic value. One indicator of success was that the revenue stream had increased; another was that the online systems produced by the project was being expanded to European sites.

## VI. LESSONS FROM THE STUDY

The case shows that the agile and the structured approaches can complement each other, and are not necessarily bipolar choices as perceived by the popular literature [e.g., Vidgen and Wang, 2009]. Agile without structure can cause chaos, particularly in large complex distributed projects where planning, control, and coordination are critical. Structure without agile principles can lead to rigidity, particularly when a project involves a great deal of learning, discovery, and change.

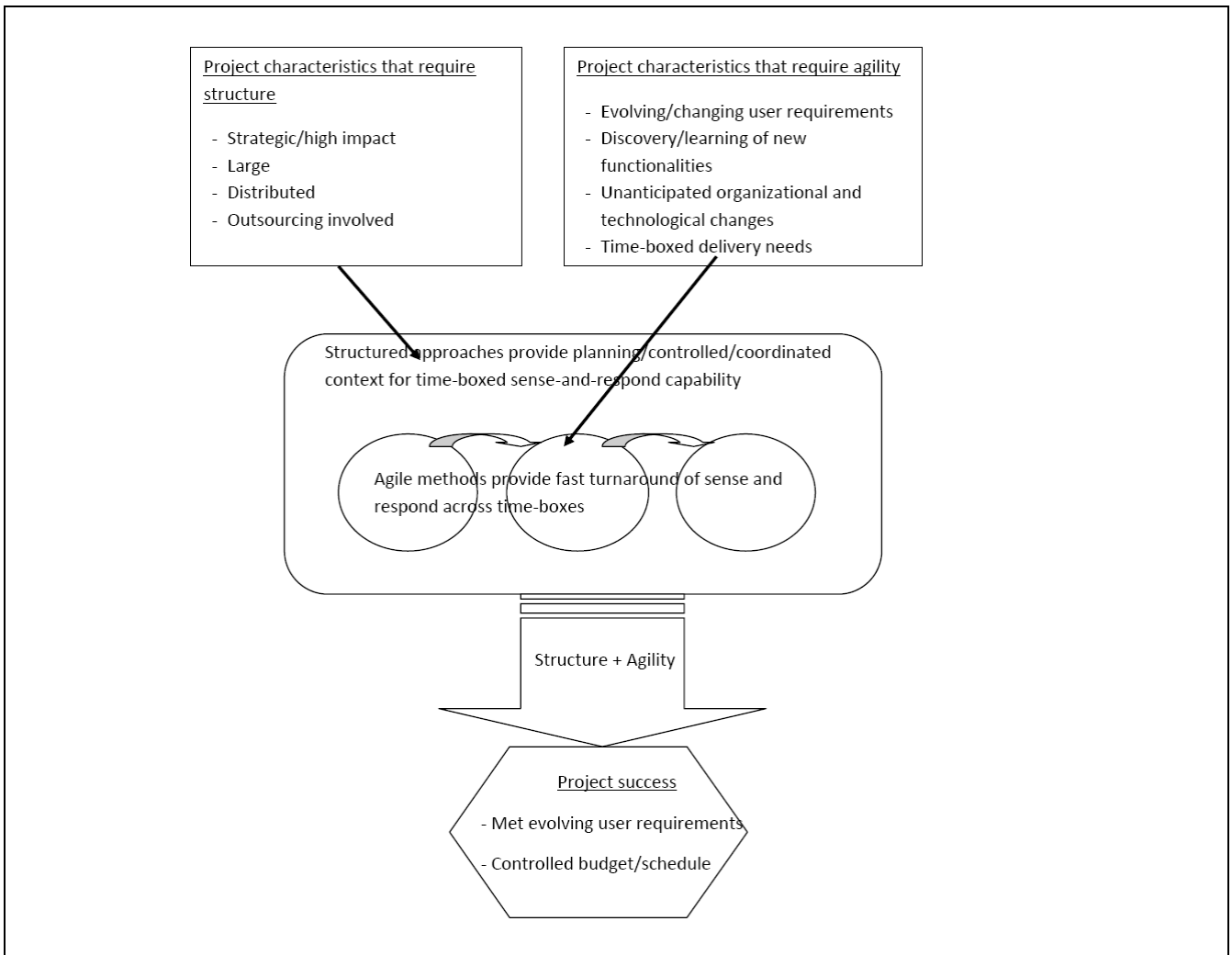


**Figure 1. Project Characteristics That Favor a Hybrid Approach**

Certain conditions amplify the need to harmonize the agile and the structured approaches in a software development project [Vinekar et al., 2006]. Figure 1 depicts the factors that we found in our case study that may come together to favor a hybrid approach. This is consistent with Boehm and Turner [2004], who point out that the choice of traditional methods for a given project is largely contingent on the size and strategic criticality of the projects. In our case study, the online reservation project had evolved into a large complex project that was deemed to be strategically important to the competitiveness of the company. Typically, such a project would have a clear overall objective but have ambiguity in scope and detailed requirements. Yet, getting the requirements right may be a critical success factor. Where competitive market forces are a driver, the schedule may be more of a priority then the cost. In addition, the distributed nature of the project required the discipline and control provided by the structured approach that was necessary for planning and coordination of the various aspects of the project across different locations. In addition, as a significant portion of the system development was outsourced, structured project management was necessary for contracting and control of the outsourcing activities [Sarker and Sarker, 2009].

While being large, strategically important, distributed and outsourced, the cruise line project was also dynamic and uncertain in various aspects. The business requirements were not stable, with increasing scopes and changing requirements. As the users and business sponsors became knowledgeable about the potential business opportunities that the technologies could provide, they discovered new functionalities that were not in the original set of requirements. As such, the project served as a learning and innovation mechanism [Nerur and Balijepally, 2007]. As a manager stated, the project team often felt that they were shooting at a moving target, and sometimes they felt there were no targets at all. Because of the learning- and innovation-oriented nature of the project, there was a requirement for time-boxed delivery of prototypes and working software. These challenges were further exacerbated by the unanticipated organizational changes. During the project, the top management team experienced dramatic turnover, including changes of such key leaders as CEO, CFO, and CIO. The turnover of key project sponsors created significant disruptions to the project's continuance of directions, policies, and resources.

An interesting aspect of the study was the relationship between ABC Cruiselines and the outsourcing vendor, who was carefully selected based on the need for JEEE expertise and subscription to Scrum method of development. For ABC Cruiselines, the use of Scrum meant that the stakeholders would have to communicate frequently, decisions would be made quickly, software development would be time-boxed, and teamwork would be paramount. Paradoxically, the vendor was even far more "agile," even to the point of rigidity in following the textbook Scrum method. Note that while some of the vendor developers were based at the customer site, a significant proportion of the development were based in India, and a few worked from the UK and Canada. The impedance mismatch in the "degree of agile" created some frictions between the client and the vendor. However, ABC Cruiselines' own ScrumMaster admitted that the vendor brought discipline into the project and was able to identify key problem issues. Further, it seemed that the vendor was flexible. For example, the vendor did not completely subscribe to YAGNI (You Aren't Going to Need It) principle, which is sometimes associated with agile development. This was evident because the vendor forced the senior managers to think through and commit to requirements that could be determined and did not have to emerge as the project progressed. In other words, YAGNI is not a substitute for deciphering requirements that can be determined based on deliberate and focused thinking.

As shown in Figure 1, while the large, strategic, distributed, and outsourced aspects required the planning and control capabilities provided by the structured approach, the uncertain and evolving nature of the project required the fast, iterative, and incremental learning and discovery capabilities provided by the agile approach. The structured approach serves as an overall architectural foundation for maintaining order and predictability throughout the whole project, while the agile approach serves as a time-boxed sensing and responding vehicle for dealing with the dynamic and uncertain requirements and project environments. Together, they complement each other in successfully managing the large cruise line project, one that was large and complex, with uncertain and evolving requirements and project circumstances. In essence, the company was able to respond to changes in an agile manner within a large, strategic, distributed, and outsourced project.

Our study opens avenues for future research. There is a need for research that would provide key concepts and frameworks as foundations for developing theories that can lead to practical guidelines for achieving agility by harmonizing the two approaches. Future research needs to study agility, beyond the specific agile principles/ methods, to include both structured and agile project management approaches. For example, Berger and Beynon-Davies [2009] have found that Rapid Application Development (RAD) can be deployed in large-scale, complex projects although the adoption can raise unique problems that need to be addressed. Further empirical research is needed to investigate conditions/areas under which agile and structured approaches can be reconciled, and to examine the antecedents and consequences of agility.

Some recent findings need further clarification. A recent survey of critical success factors in 109 agile software projects [Chow and Cao, 2008] considered numerous factors; however, only delivery strategy and team capability

were found to be significant. It is possible that such results mask the importance of factors needed to achieve agility under differing conditions. For example, the factors that are critical in achieving agility in a strategic project may be very different than those needed for a legacy improvement project. Executive support may be important in the former, but not in the latter. By treating all projects as having the same maxims, we may not be able to identify factors that are important only for a certain kind of project. It may, thus, be useful to conceptualize different kinds of agilities such as strategic, design, process, and outsourcing. Multiple case studies can help identify and develop taxonomies of agility. For each kind of agility, we need to outline the maxims, and then identify success factors by conducting questionnaire-based surveys and further in-depth case studies.

## VII. CONCLUSION

While the need for balancing agile and structured approaches has been widely recognized, making an agile approach work in large, strategic, distributed, and outsourced projects has been challenging [Lee et al., 2006; Ramesh et al, 2006]. Based on the case of a large complex distributed development project, we illustrate that agile and traditional structured approaches can and are indeed needed to complement each other. Structured planning, control, and coordination provided a disciplined organizational infrastructure that is necessary for agile to be effective. On the other hand, the agile approaches, with iterative and fast turnaround cycles, enabled the structured planning and control process to learn and adapt efficiently to changing conditions. The case challenges the generally accepted proverbs of the agile and structured approaches.

The project revealed several paradox-like phenomena that need further research and investigation. The agile method was found feasible in a large project; in fact, the use of the agile method actually brought more disciplines to the project. Although customer satisfaction was a paramount goal of the outsourcing vendor, which was well-versed in Scrum, they did not welcome late changes. The agile method was not just an exercise in coding and feedback; the vendor insisted on the best available requirements and devoted considerable time to design thinking. Agile principles that focus on quality of individual work life were not found to be prevalent given the time pressure of the project. Conversely, the structured approach did not inhibit agility. The steering committee could make resource decisions quickly. The hiring of an outsourcing vendor, well-versed in agile methods and information technologies, brought agility and quality in the project.

Past research has indicated that structured and agile development can coexist in ambidextrous organizations; however, this simply implies that different parts of the organizations have different structure, methods, and culture. Our study suggests structured and agile can be harmonized within the same project, especially when the project is large, strategic, time-sensitive, and distributed. We hope our study serves as a step stone for future research that further develop theories and insights about how the agile and the structured approaches can be effectively combined to achieve project success that each approach can't achieve alone.

## REFERENCES


Ahern, D.M., A. Clouse, and R. Turner (2003) *CMMI® Distilled: A Practical Introduction to Integrated Process Improvement,* The SEI Series in Software Engineering: Addison Wesley.

Ågerfalk, P.J. and B. Fitzgerald (2006) "Flexible and Distributed Software Processes: Old Petunias in New Bowls?" *Communications of the ACM* (49)10, pp. 26–34.

Beck, K. (2000) *Extreme Programming Explained: Embrace Change,* Reading, MA: Addison-Wesley.

Benbya, H. and B. McKelvey (2006) "Toward a Complexity Theory of Information Systems Development", *Information Technology and People* (19)1, pp. 12–34.

Berger, H. and P. Beynon-Davies (2009) "The Utility of Rapid Application Development in Large-Scale, Complex Projects", *Information Systems Journal* (19), pp. 549–570.

Boehm, B.W. and R. Turner (2004) *Balancing Agility and Discipline: A Guide for the Perplexed*, Boston: Addison-Wesley.

Bose, I. (2008) "Lessons Learned from Distributed Agile Software Projects: A Case-Based Analysis", *Communications of the Association for Information Systems* (23)15, pp. 619–632.

Chow, T. and D.B. Cao (2008) "A Survey Study of Critical Success Factors in Agile Software Projects", *Journal of Systems and Software* (81)6, pp. 961–971.

Cockburn, A. (2002) *Agile Software Development*, Boston: Addison-Wesley.

Cockburn, A. (2004) *Crystal Clear: A Human-Powered Methodology for Small Teams,* Agile Software Development Series.

Curtis, B., W.E. Hefley, and S. Miller (2002) *The People Capability Maturity Model: Guidelines for Improving the Workforce,* Upper Saddle River, NJ: Pearson Education.

Deming, W.E. (1986) *Out of the Crisis,* Cambridge, MA: MIT Center for Advanced Engineering.

Dyba, T., and T. Dingsoyr (2008) "Empirical Studies of Agile Software Development: A Systematic Review", *Information and Software Technology* (50)9–10, pp. 833–859.

Erickson, J., K. Lyytinen, and K. Siau (2005) "Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research", *Journal of Database Management* (16)4, pp. 88–100.

Fernandez, D.J., and J.D. Fernandez (2008) "Agile Project Management—Agilism versus Traditional Approaches", *The Journal of Computer Information Systems, 2008/2009* (49)2, pp. 10–17.

Fitzgerald, B., G. Hartnett, and K. Conboy (2006) "Customizing Agile Methods to Software Practices at Intel Shannon", *European Journal of Information Systems* (15)2, pp. 43-53.

Herbsleb, J.D., and A. Mockus (2003) "An Empirical Study of Speed and Communication in Globally Distributed Software Development", *IEEE Transactions on Software Engineering* (29)6, pp. 481–494.

Holmstrom, H., et al. (2006) "Agile Practices Reduce Distance in Global Software Development", *Information Systems Management* (25)3, pp. 7–18.

Highsmith, J. (1999) *Adaptive Software Development*, Cambridge, UK: Dorset House.

Kishore, R., et al. "A Relationship Perspective on IT Outsourcing", *Communications of the ACM* (46)12, pp. 87–92.

Larman, C. (2004) *Agile and Iterative Development: A Manager's Guide,* Indianapolis, IN: Addison-Wesley Professional.

Lee, G., W. Delone, and J.A. Espinosa (2006) "Ambidextrous Coping Strategies in Globally Distributed Software Development Projects", *Communications of the ACM* (49)10, pp. 35–40.

Masticola, S. (2007) "A Simple Estimate of the Cost of Software Project Failures and the Breakeven Effectiveness of Project Risk Management", First International Workshop on the Economics of Software and Computation (ESC'07) (May 20–26).

Misra, S.C., V. Kumar, and U. Kumar (2009) "Identifying Some Important Success Factors in Adopting Agile Software Development Practices", *The Journal of Systems and Software* (82), pp. 1869–1890.

Nerur, S., R. Mahapatra, and G. Mangalraj (2005) "Challenges of Migrating to Agile Methodologies," *Communications of the ACM* (48)5, pp. 73-78.

Nerur, S. and V. Balijapally (2007) "Theoretical Reflections on Agile Development Methodologies", *Communications of the ACM* (50)3, pp. 79–83.

Nord, R.L. and J.E. Tomayko (2006) *Software Architecture-Centric Methods and Agile Development*, IEEE Software, pp. 47–53.

Orr, K. (2004) *Agile Requirements: Opportunity or Oxymoron?* IEEE Software, pp. 71–73.

PMBOK, Project Management Institute (2004) *A Guide to the Project Management Body of Knowledge,* Newton Square, PA: Project Management Institute.

Ramesh, B., L. Cao, and R. Baskerville (2007) "Agile Requirements Engineering Practices and Challenges: An Empirical Study", *Information Systems Journal*, pp. 1–32.

Ramesh, B., et al. (2006) "Can Distributed Software Development be Agile?" *Communications of the ACM* (49)10, pp. 41–46.

Sarker, S. and S. Sarker (2009) "Exploring Agility in Distributed Information Systems Development Teams: An Interpretive Study in an Offshoring Context", *Information Systems Research* (20)3, pp. 440–461.

Schwaber, K. (2004) *Agile Software Development with Scrum*, Redmond, WA: Microsoft Press.

Turk, D., R. France, and B. Rumpe (2005) "Assumptions Underlying Agile Software-Development Processes", *Journal of Database Management* (16)4, pp. 62–87.

Vidgen, R. and X. Wang (2009) "Coevolving Systems and the Organization of Agile Software Development", *Information Systems Research* (20)3, pp. 355–376.

Vinekar, V., C.W. Slinkman, and S. Nerur (2006) "Can Agile and Traditional Systems Development Approaches Coexist? An Ambidextrous View", *Information Systems Management* (23)3, pp. 31–42.

## ABOUT THE AUTHORS

**Dinesh Batra** is a Professor in the Decision Sciences and Information Systems in the College of Business Administration at the Florida International University. His publications have appeared in *Management Science, Journal of MIS, Communications of the ACM, Communications of the Association for Information Systems, European Journal of Information Systems, International Journal of Human Computer Studies, Journal of Database Management, Computers and Operations Research, Data Base, Information and Management, Decision Support Systems, Requirements Engineering Journal, Information Systems Management,* and others. He is a co-author of the book *Object-Oriented Systems Analysis and Design* published by Pearson Prentice-Hall. He has served as the President of the AIS SIG on Systems Analysis and Design (SIGSAND).

**Weidong Xia** is an associate professor of Decision Sciences and Information Systems in the College of Business Administration at Florida International University. His research interests include IT-business alignment, strategy and governance; project management complexity and flexibility; innovation evaluation and adoption; and outsourcing management. He has published in such journals as *MIS Quarterly*, *Decision Sciences*, *Journal of Management Information Systems*, *Communications of the ACM*, *European Journal of Information Systems*, and *Journal of Information Technology Management.* He currently serves as an associate editor of *Information Systems Research*. He received his doctorate in Information Systems from the University of Pittsburgh.

**Debra VanderMeer** is an assistant professor in the College of Business at Florida International University. Her research interests focus on applying concepts developed in computer science and information systems to solve real-world problems. She is widely published in well-known journals, such as *Management Science, ACM Transactions on Database Systems,* and *IEEE Transactions on Knowledge and Data Engineering*, as well as prestigious conference proceedings, including the International Conference on Data Engineering, International Conference on Distributed Computing Systems, and the Very Large Database Conference. She also has significant professional experience in the software industry. She has served in software engineering and managerial roles in large companies, as well as early-stage venture-funded software enterprises. She received her doctorate from the Georgia Institute of Technology.

**Kaushik Dutta** is an associate professor of Decision Sciences and Information Systems in the College of Business at Florida International University. His research interest is enterprise IT Infrastructure and software development. He has published articles in journals such as *Management Science, Journal of Computing,* and *ACM Transactions on Database Systems*. Dr. Dutta also has several publications in various IEEE and ACM conference proceedings. He is the Area Editor (Database and Data Management) for Elsevier Journal of Systems and Software. Prior to joining FIU, Dr. Dutta was Director of Engineering for Chutney Technologies, funded by KPCB Venture Capital, that developed solutions to improve the scalability and performance of enterprise Web applications. He received his doctorate in Information Systems from the Georgia Institute of Technology.

Communications of the Association for Information Systems

ISSN: 1529-3181





Communications of the Association for Information Systems

Volume 27 Article 21 Balancing Agile and Structured Development Approaches to Successfully Manage Large Distributed Software Projects: A Case Study from the Cruise Line Industry Dinesh Batra Decision Sciences and Information Systems, FloridaÂ  Systems, Florida International University Agile methods and traditional structured approaches are often viewed as competing bi-polar choices. Agile methods such as Scrum and XP are recommended for small, co-located projects that involve changing requirements. The traditional structured plan-driven approaches, such as the Capability Maturity Model (CMM) and the waterfall lifecycle frameworks, are recommended for large projects with stable requirements. Balancing agile and structured development approaches to successfully manage large distributed software projects: A case study from the cruise line industry. Communications of the Association for Information Systems, 27(1), 21.Google Scholar. Battin, R. D., Crocker, R., Kreidler, J., & Subramanian, K. (2001).Â  Agile vs. structured distributed software development: A case study. Empirical Software Engineering, 19(5), 1197â€“1224.CrossRefGoogle Scholar. Fruhling, A., & Vreede, G.-J. D. (2006).