Business Horizons (2016) xxx, xxx—xxx

Available online at www.sciencedirect.com

ScienceDirect

KELLEY SCHOOL OF BUSINESS
INDIANA UNIVERSITY

www.elsevier.com/locate/bushor

EXECUTIVE DIGEST

Managing the transition to the new agile business and product development model: Lessons from Cisco Systems

Roger (Ronxin) Chen [a,*], Ramya Ravichandar [b], Don Proctor [b]

[a] School of Management, University of San Francisco, 2013 Fulton Street, San Francisco, CA 94117, U.S.A.
[b] Cisco Systems Inc., 170 West Tasman Drive, San Jose, CA 95134, U.S.A.

KEYWORDS
Agile development;
Innovation;
Product development;
Organizational change;
Cisco Systems;
Organizational agility

Abstract Through an in-depth case study of Cisco Systems, this Executive Digest finds that companies face two broad challenges when transitioning to the agile product development model. The first is identifying and helping business units and engineering teams adopt this method; the second is developing new management practices that are compatible with and can sustain the agile development practices. Although extant literature has conducted many analyses on these two challenges, there still exist gaps in the research of the agile development method. Herein, we explore how Cisco Systems addressed these two challenges followed by a discussion of the broad implications of adopting the agile development method. This research deepens our understanding of how to adopt and lead the agile development process.

1. Introduction and literature review

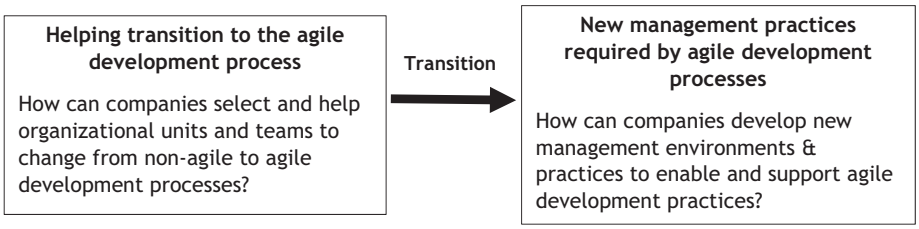
Since the publication of the *Manifesto for Agile Software Development* (Beck et al., 2001), the agile development model has been adopted by many companies and received increased attention in academic research. Organizations incorporating agile development face two major challenges: managing the transition for the company or business units, and developing new organizational environment and management practices that will sustain and support agile development practices once the transition to agile development practices is complete. Figure 1 depicts these two major challenges.

Existing research has reported on many studies on the agile development method. Regarding the first challenge, prior research has proposed various frameworks to help companies make the transition from a traditional to an agile development process. Boehm and Turner (2003a) defined five decision factors—size, criticality, personnel, dynamism, and cultural—to help companies decide whether they should adopt a traditional method, an agile development method, or some combination of the

* Corresponding author
E-mail addresses: chenr@usfca.edu (R. Chen), raravlch@cisco.com (R. Ravichandar), dproctor@cisco.com (D. Proctor)

0007-6813/$ — see front matter
http://dx.doi.org/10.1016/j.bushor.2016.06.005

Figure 1. Key management challenges for organizational transition to an agile development model



two. Qumer and Henderson-Sellers (2008) developed the agile adoption and improvement model (AAIM), which defines six levels of agile adoption—including agile infancy, agile initial, agile realization, agile value, agile smart, and agile progress. More recently, Gandomani and Nafchi (2015) used the grounded theory approach to develop an agile transition and adoption framework that includes five components: practice selection, adaption, assessment, retrospective, and adjustment.

Regarding the second challenge of developing new management practices to enable and sustain the agile development process, many studies discussed the impact of adopting agile development on management practices. Nerur, Mahapatra, and Mangalaraj (2005) analyzed the impact of the agile method on management style, organizational control, communication, and customer role. Hoda, Noble, and Marshall (2011) explained all required roles in an agile self-autonomy team. Moreover, previous studies argue that project managers, especially those who are experienced in traditional software development, need to transition from a traditional commander role to a leadership role. For example, Ambler (2005a) indicated that in agile teams, managers need to act as the team coach. Other studies analyzed the impact of the agile development method on additional management functions and practices, including planning (Ambler, 2005b; Boehm & Turner, 2003b), management coordination (Strode, Huff, Hope, & Link, 2012), and task design (James, 2010; Thomke & Reinersten, 1998). Scholars have also discussed characteristics of the customer in the agile method (Cohen, Lindvall, & Costa, 2004; Turner & Boehm, 2003), which has important implications for how agile teams operate.

In sum, many studies on the agile development method have examined topics pertaining to our two research questions or challenges; however, most of these studies focus on software companies or software products. It is unclear if the frameworks from these studies apply to companies offering system products that include both software and hardware components. Furthermore, the arguments and findings of the existing literature are drawn from different sources, ranging from mini cases, to theory papers, to professional opinion posts; few studies have examined the intricacies and complexities of how these factors and frameworks work holistically in the context of one company. In fact, scholars have called for a comprehensive and disciplined approach for companies to manage the transition to the agile method (Gandomani, Zulzalil, Ghani, & Sultan, 2013). In response, we conducted an in-depth case analysis of Cisco, a company that offers system products involving both software and hardware components, and the company in which some business units have transitioned to the agile development method while others have not. By focusing on the experience of one company, it allows us to validate some of the arguments, deepens our insights into the transition from traditional to agile development methods, and allows us to explore new management practices required to support and sustain agile development methods. Our analysis is organized as follows: we first introduce the case company, Cisco Systems, and our research method, and then present our findings and discussion of each research question.

2. Case company and research method

2.1. About the case company: Cisco Systems

We analyzed the two challenges depicted in Figure 1 through an in-depth case analysis of Cisco Systems Inc. Cisco is a leading global network equipment company that offers a wide range of products—such as routers, switches, and networking solutions—designed for enterprises and small businesses across a variety of industries. Cisco has traditionally used the waterfall method to develop new products. In the waterfall method, tasks and deliverables are clearly visible at each stage of the product development process; however, the entire development process can be lengthy. Waterfall methods typically start with various analysis reports (e.g., a business requirement document, product requirement

document, marketing requirement document) that aid in creating development tasks, which are then handed to various development teams (e.g., user experience, engineering). The teams go through a variety of product tests before launching the product to the market. From start to finish, the process can take up to 18 months or longer. In a fast-changing environment, such a lengthy development cycle can lead to an outdated product when the product is introduced to the market.

Under an agile development method, companies do not go through similarly lengthy and comprehensive analysis at the outset, and, as such, the planning period is much shorter. At Cisco, business units that use the agile method have major product releases every quarter. Within each quarter, product development is divided into sprints, with each sprint lasting approximately two weeks. After each two-week sprint, product managers demonstrate new features to customers and seek their feedback. Thus, agile development processes are closer to market and faster. But it also involves more change and uncertainty. We chose Cisco as our case study because some business units at Cisco have transitioned to an agile development method while others have not. Our analysis below first examines how Cisco selects and helps business units and engineering teams transition to the agile development method. We then analyze some of the new management practices Cisco adopted to support and lead the agile development practices.

2.2. Research method

In this research, one co-author is a senior executive at Cisco who has led various business units using the agile development method. Another co-author is directly responsible for helping business units at Cisco change from the waterfall to the agile development method and helping to establish new management practices to support agile development practices. In addition to their direct knowledge, we studied background information and various documents on Cisco's agile development practices. We also conducted a broad range of interviews with people who played different roles in the agile process, including engineering team managers, product and program managers, vice presidents of the business units that changed to the agile method, and trainers and coaches of the agile development method. All the interviews were semi-structured and recorded; we interviewed certain individuals multiple times to avoid misunderstanding. Most of the interviews were conducted by two interviewers to ensure reliability.

3. Findings

3.1. Challenge #1: Helping the transition to the agile development method

It is often difficult for business unit leaders and engineering teams to give up their old mindset and habits and adopt the new behaviors required by the agile development method. To address this challenge, Cisco established a dedicated corporate Agile Tiger Team (ATT) to assist business units and engineering teams making the transition. The ATT focuses on the following assessment and transition tasks.

3.1.1. Benefit assessment

When identifying business units for transition to an agile method, the ATT first works with business units to evaluate the potential benefits of adopting agile processes. Identifying benefits is a crucial step to get the buy-in and collaboration from business units to initiate and sustain the transition to the agile method. In general, the ATT observed that transition to agile development brings the greatest benefits in three areas: time to market, customer satisfaction, and employee engagement. Because of the value added in these areas, the ATT focused on three broad assessment questions: First, will the transition to an agile development process help the business unit reduce the time it takes to bring new products to market? Second, will the transition significantly help the business unit develop the right products to meet customer needs? And third, will the transition increase employee engagement and morale? Only business units that can fully benefit from the transition are good candidates for adopting the agile method. In addition, Cisco's experience also shows that it is not enough to clarify the potential benefits associated with transitioning to the agile method; it is also important to clearly communicate such benefits to mid- to low-level managers and to engineering teams.

3.1.2. Readiness assessment

The ATT also identifies three conditions to assess if a business unit, including its engineering teams, is ready for the transition. The first is leadership buy-in. Securing buy-in from engineering team managers is important because they are the frontline people who guide engineers in implementing the agile process. The commitment from mid- to senior-level managers in the business unit is also very important. During the transition to the agile development process, these managers will not only need to adjust their own mindset and behavior, but they will also need to support engineers and cultivate new culture and behavior in the engineering teams. In

addition, managers will need to help agile engineering teams cooperate with other teams or other business units during the agile development process. The transition to the agile method is not a simple one-off event; it is a long, gradual process, that requires continuous change, build up, and follow thorough to develop and reinforce new culture and behavior. Without continued support, people can easily fall back into old habits in their product development process. All this requires strong commitment and buy-in from the business unit leaders and managers. In general, the benefits of adopting the agile method are an important factor affecting the buy-in from business unit executives and managers.

The second condition for transition readiness is minimal task interdependence for the engineering team. Engineering teams adopting the agile method need to deliver new product features every two weeks. This requires very intense collaboration and coordination among engineers within the team and across the teams. If the tasks of an engineering team are highly intertwined with other teams, it will increase the complexity and difficulty of executing agile development projects. To assess this, the ATT examines if an engineering team's task can be self-contained or if it is highly intertwined with other teams or units. For example, the ATT examines the architecture of the product that an engineering team is working on. ATT favors teams that work on modular product architecture because such architecture allows engineering teams to work on self-contained and autonomous tasks, which is important for the agile method.

The third condition is early-stage product development. If an engineering team is working on a development project using the waterfall process and the project is in mid to late stages, it will be costly for the team to change its development process mid-stream. In contrast, teams at the beginning phase of a project can make the transition with minimal disruption.

3.1.3. Supporting the transition

For the business units and their engineering teams that are ready—or almost ready—to transition to the agile method, the ATT typically goes through the following process to help the transition. First, the ATT examines various areas or conditions that the engineering teams need to change to successfully adopt the agile method. For example, the ATT checks to see if the engineering teams have appropriate roles required by the agile method, including scrum team master, product owner, and product manager. If such roles are missing, the ATT helps the teams put them in place. The ATT also looks at other areas such as task interdependence and even seating layout of engineers within a team. The ATT then makes suggestions or works with the business units and their engineering teams to modify or to fix conditions or issues that are unfavorable for adopting the agile method.

Next, the ATT provides training on the fundamentals of the agile method to engineering teams. After the training, the ATT embeds a coach in the engineering team for several months to help the team get the agile process started. After the engineering team starts the agile method, the coach will leave for several weeks, occasionally returning to review the team's progress and to check on any issues or problems. For example, the coach may evaluate the percentage of product features the team planned but failed to deliver after each sprint. During the coaching process, it is important for the coach to be flexible and, more importantly, to observe and learn the operating culture of the engineering team and its business unit. It is also vital to understand the motivation or the areas the engineering team or the business unit looks to gain through the transition to the agile method. This knowledge helps the coach find effective ways to help the transition to the agile method that will be better received by the engineering team and the business unit.

Early at Cisco, people who delivered agile training were separate from those who did the coaching. The company found that this leads to a problem where coaches did not know what engineers learned in the training, and engineers did not know how to apply the training content to real agile processes; the trainers were not on-site to help the engineers connect the training content to real practices. To address this problem, the ATT recently changed to make the same person do the agile training and coaching.

In addition, since most trainers/coaches are contracted outside agile experts, the ATT made efforts to ensure that their training content is consistent and tailored to Cisco. To achieve this, the ATT analyzed the lessons and learnings from the engineering teams at Cisco that went through the transition, especially the successful ones, and made sure these learnings are incorporated in the agile training. To objectively assess how well engineering teams adopted the agile development method, the ATT used different performance metrics. These included customer satisfaction in collaborating with Cisco agile teams, how well engineering teams deliver pre-planned new product features in each two-week sprint, net promoter scores (how likely people who went through the agile transition will be to recommend the agile method to others), and employee engagement. These metrics also help

the ATT learn in which area(s) an engineering team succeeded or failed in its transition and why, and how to help future engineering teams leverage the learning to avoid similar mistakes.

Cisco also learned that when a coach joins an engineering team to help the transition, the coach should not focus solely on the changes in the product development process. Many times, the transition to the agile method involves a broad range of issues and parties. For example, the agile development method requires frequent decision making on developing and adjusting new product features based on the feedback from collaborative customers. Many times these decisions go beyond engineering teams. They involve other functional roles such as product manager and product marketing or require the cooperation and coordination with other business units or external partners. Typically these different parties have their own interests, working styles, and culture. Some departments or partners may work well with the transition to the agile method while others may not. Due to this complexity, effective coaches need to pay attention to these broad issues and involve relevant stakeholders to remove barriers for engineering teams' successful transition to the agile practice. In addition, Cisco's experience shows that coaches should let engineering teams and business units know that there is no such thing as best practices in the agile development method—there are only better practices, and it is okay if the teams and business units do not achieve the best goals during the transition to the agile method. The key is to derive valuable learning from the practices and to build a culture of continuous improvement. Besides training and coaching, Cisco also developed standardized transition documents to help teams and business units change to the agile development method. For example, the ATT developed CPDM (Cisco Product Development Method), a standard operating procedure to guide engineering teams. In addition, Cisco set up an internal website to illustrate how to adopt agile processes that meet industry quality standard policies like ISO-9000. The website includes tools and effective practices from which engineering teams can learn and use in their agile process. Additionally, Cisco conducts community building through internal conferences and workplace presentations to promote and educate employees on the agile method.

3.2. Challenge #2: New management practices for the agile development method

Once business units and engineering teams change to the agile development method, companies need to develop new management practices to support and lead the agile process. Our analysis below focuses on four aspects of management practices developed at Cisco: leading agile development teams, planning and forecasting, coordinating tasks, and recruiting early collaborative customers.

3.2.1. Managing and leading agile engineering teams

Cisco's practice shows that the transition from the waterfall to the agile development method has a fundamental impact on the roles and behavior of mid- to senior-level executives as well as engineering team managers. In the waterfall methodology, engineering team managers typically run meetings, manage the task schedule, and are responsible for delivering tasks. They are heavily involved in planning, monitoring, and delivering individual work products.

In contrast, management's role under the agile method becomes decentralized as engineers take on more authority to organize their work. Because agile development teams need to deliver new product features every two weeks, the interaction and interdependency among team members becomes extremely important. Thus, it is more efficient for engineers to interact with each other rather than to go through team managers to resolve issues, especially technical problems. As a result, both first-line engineering managers and mid-level managers must let go of traditional command-and-control management styles in order to give more autonomy to their teams. As one team manager indicated about the waterfall process:

> My boss used to come and tell me to get my team to do this or do that. Now [after the transition to the agile method], I tell him (the boss) that I cannot tell my team to do this or that; I can suggest (my boss's idea) to them, but they will discuss and decide if that is the right thing to do.

In leading agile development teams, managers also need to develop new behaviors. For example, in some business units at Cisco, mid-level managers and first-line engineering managers attend engineering teams' biweekly product demo meetings. By participating in these meetings, the leaders/managers can see the progress, make suggestions, and ensure the team is moving in the right direction.

The transition to autonomous teams in the agile method also opens up the possibility for managers to work in more flat organizations. For example, in one business unit at Cisco, after the unit successfully adopted the agile development process, a director in the unit had 45 engineers directly reporting to

him. Such breadth of oversight is only possible because the director empowers engineering teams and gives them a lot of autonomy.

In addition, managers leading agile teams need to learn to play supporting and servant leadership roles (Ambler, 2005a). For example, they need to focus more on helping their agile engineering teams communicate and coordinate with other departments to resolve obstacles or to protect their engineering teams from excessive external requests. Some teams at Cisco developed very efficient processes so mid-level managers could remove obstacles for their engineering teams or escalate the obstacle issues to upper-level management. Managers and leaders in this context can also play other supporting roles, including helping their teams learn, reflect, and improve their agile working process; motivating engineers; and helping them in their professional development.

3.2.2. Planning and forecasting in the agile development process

Transition to the agile method also significantly affects management planning. Under the traditional waterfall process, Cisco follows a multi-phase process to plan and arrange resources. The first phase is the concept commit, followed by execution commit. Once a project enters an execution commit phase, business unit and engineering teams define what they plan to deliver, what resources they need, the scope of the tasks, and product features. When project execution begins, managers expect monthly status reports and expect engineering team deliveries based on pre-planned milestones. The typical waterfall planning cycle is 12 to 18 months. Such a long planning cycle gives companies predictability in planning resources and arranging work.

But in the agile development process, such long-term predictability can be difficult. Engineering teams, for example, seek customer feedback after every two-week sprint. Yet, because customer reactions may change, establishing stable and predictable plans can be problematic. To overcome this challenge, scholars have analyzed the types of projects or circumstances in which either a traditional planning method or the agile method is suitable (Boehm & Turner, 2003b). Some business consultants suggest that in the agile method, companies only "schedule in detail several weeks ahead but not several months, and only do a high-level scheduling for future iterations" (Ambler, 2005b). These analyses assume that planning for several months or longer would only be possible in the agile method if the plan was at a high level and not detailed. If companies did want to develop detailed planning, the plan would have to be short term—several weeks at most.

The practice at Cisco, however, illustrates that it is possible to forecast and plan several quarters ahead under the agile development process. One business unit at Cisco developed the following process to improve the accuracy and predictability of its planning. This business unit adopted the agile method by talking to early collaborative customers about new product features every sprint (two weeks) and releasing new product features to market every quarter (three months). Before an engineering team starts developing a new product, the product manager, product architecture manager, and other relevant managers carefully identify use cases. These managers then translate the use cases into engineering tasks, prioritizing tasks and slotting them into future quarters, with an emphasis on the first two to four quarters (i.e., the first six to 12 months). Because these plans look at detailed engineering tasks, they are very specific. And while the task plans will be adjusted, they give the engineering team strong guidelines to organize and arrange future work. As the business unit executive summarized: "This is a bottom-up (planning) approach and you get into many details···. You can gain about six months' predictability that is very accurate, though not perfect."

This bottom-up approach rests on a two-team methodology that works as follows. When an engineering team starts to develop the new product, the business unit establishes another team to continuously forecast and adjust the future plan. For a given product, as an engineering team ('execution team') finishes the new product development of the first quarter and begins second quarter work, another team ('planning team') is established to plan the development tasks for the future two quarters (in this case, third and fourth quarters) with an emphasis on the upcoming one (in this case, the third quarter). This planning team looks at very specific task items for the future two quarters, assessing which engineering works are needed and whether it makes sense to move certain development tasks to earlier or to later quarters, or to take the tasks out of the plan. They plan these tasks by relying on customer feedback and engineering execution information from the first quarter. These two teams continue this parallel working process as new product development continues.

In order to make specific task arrangements for the subsequent two quarters, the planning team meets with the execution team every other week to seek feedback and comments on the proposed plans. Because the execution team not only knows their own ability and development speed but also

has direct knowledge of customer requirements, the execution team can provide valuable insights in these meetings. Sometimes, the execution team will tell the planning team they do not have enough manpower to deliver the proposed plan over the next two quarters or they cannot move certain features to other time slots because of prior commitments to customers. At other times, the discussion between the planning and execution teams can result in disagreements. Any disagreements between product and engineering managers about engineering capacity and planning that cannot be resolved are sent to senior level managers for resolution.

The planning team is a dedicated team usually consisting of senior people—such as senior product manager, user experience manager, and product architecture manager—who have overall platform knowledge and a wide range of expertise on the product. It is critical that the business unit places their best people in the planning team. Because the planning team has one and a half months to develop each quarter's plan, it is under great time constraints. Yet any mistakes by the planning team can greatly affect the ultimate execution. As the executive of the business unit indicated: "For the execution team, it is common that you have A or B or C level players, but in this [planning] team, you better have all A players."

3.2.3. Coordination in the agile development process

At Cisco, when engineering teams change to the agile development method, new coordination complexities arise at three levels: within agile development teams (intrateam coordination); across different teams or departments (interteam/interunit coordination); and between the company and its external partners.

At the intrateam coordination level, when engineering teams finish developing features at each two-week sprint, they will immediately let customers try the features. This requires that within each agile sprint, engineering teams not only finish new feature development, but also complete other tasks such as unit testing and code review. Accomplishing these many tasks within each two-week cycle demands intense and effective coordination among engineers within the agile team. To achieve this, Cisco co-locates engineers of the same team, reduces or avoids multitasking of the engineering teams, and optimizes team size (about seven to nine people). In addition, Cisco created open space to allow engineers to collaborate and interact, and built a strong suite of collaboration tools to enable engineers in multiple locations to collaborate.

For interteam/interunit coordination, some agile teams at Cisco work on products that involve multiple subproducts, such as hardware and software, which belong to different units or departments. Because the agile process is commonly used in software development while hardware development follows a more traditional process, cross-department/unit coordination challenges are created when an engineering team working with other units switches to the agile method. As a result, Cisco developed several solutions to mitigate cross-unit/department coordination challenges. One is through product design. When a product involves both hardware and software components, overall product architecture design must be well established from the outset. The design provides a framework and flexibility for software teams to use the agile method to build and adjust new product features. It also reduces the possibility of drastic changes later in the development phases, which can be costly. Another solution is to modularize the interfaces of different components (Thomke & Reinersten, 1998). Cisco also established weekly meetings where agile teams give other teams/units updates on new product features. In addition, the company created boundary-spanning roles such as program managers, who coordinate all the constituencies or subprojects.

The agile method also creates new challenges in coordinating with external development partners. In the agile method, engineering teams need to constantly work with collaborative customers to test features, get customer feedback, and adjust future development work. If the teams rely upon external partners for their development work, it will require continuous and very close collaboration with the external partners to synchronize their efforts. Problems can quickly occur if the Cisco agile development teams and their external partners are out of sync or miscommunicate. In one agile development project, Cisco had to rework about 80% of the software code provided by an external partner due to lack of communication and coordination, which caused significant stress and problems to the project. Because of the risks posed by situations like this, it is very important to build closer coordination and monitoring mechanisms with third parties on agile development projects.

3.2.4. Recruiting collaborative customers

Cisco's experience also illustrates the potential challenges in recruiting collaborative customers who provide feedback to Cisco's agile development teams. These customers are different from ordinary customers. Collaborative customers are typically early adopters of new products; they need to know

how to effectively work with external agile engineering teams, be able to use unfinished products that are under development, and be willing and able to adjust their use of the products. Sometimes, multiple departments within the collaborative customer group will be involved in using and testing the products under agile development. Not all customers are qualified for such tasks. As a result, if agile development teams do not give enough consideration to these unique characteristics when recruiting collaborative customers, the teams may recruit wrong customers that could undermine the agile development projects.

In addition, agile engineering teams at Cisco typically rely on other departments at Cisco to recommend and recruit early collaborative customers. Sometimes the departments recommend such customers based largely on their own interests, including the desire to please customers or to influence certain sale deals, even though these customers might not be appropriate to serve as agile collaborative partners. For example, when such customers' needs are incompatible with the directions or market positions of the developing products, the customers' feedback can steer the product development in the wrong direction.

This misdirection occurred at Cisco. An agile engineering team at Cisco planned to develop a new product that was positioned as a horizontal platform for companies from a broad range of sectors. Another department recommended a financial institution as a collaborative customer. However, the financial institution asked the agile development team to put extra layers of security and privacy protections on the product. Although common among financial institutions, these requests were not appropriate for a horizontal platform where open access is important. Yet this financial institution was an important customer and had a lot of clout in influencing the product direction. It took the product development unit over a year to stop the misdirection. It also forced the agile team to redo much of its development work and caused delay in the development process.

Based on Cisco's experience, one solution to address the above problem is to work with multiple collaborative customers, which allows engineering teams to compare and validate different customers' feedback. Another solution is to clarify the requirements for early collaborative customers from the onset. One executive at Cisco indicated that before selecting early collaborative customers, related managers—such as product and engineering managers—needed to get together to identify the characteristics and qualifications of the collaborative customers based on the market position and nature of the new products. Managers should then use the criteria to guide the selection and recruiting of the collaborative customers.

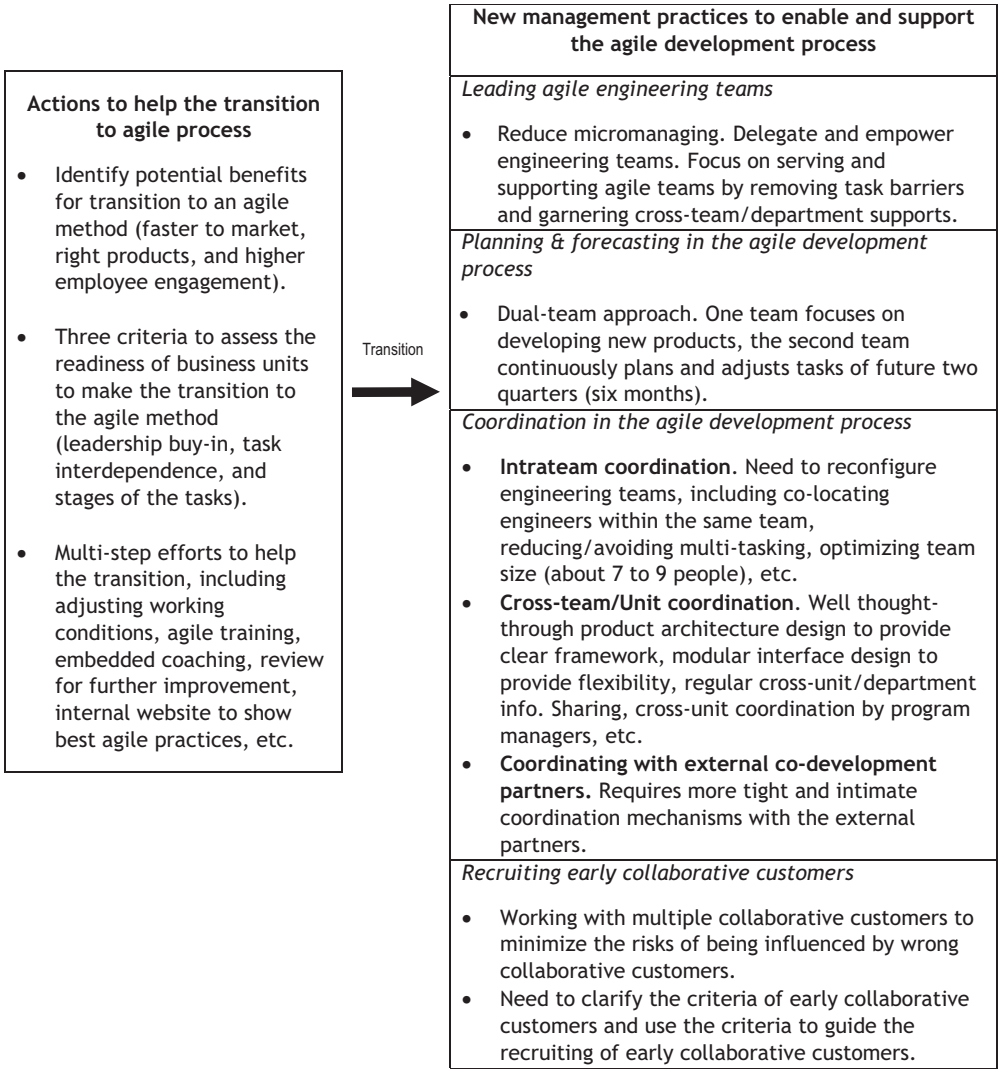
4. Conclusion

The Cisco experience suggests that when companies change to the agile development process, they need to tackle two major challenges. The first is to help teams or business units within the company to make the transition, and the second is to develop new management practices that can support agile development method. We summarize these lessons in Figure 2.

Besides the key learnings summarized in Figure 2, we next highlight a few other implications of adopting the agile development method. Our study shows that companies need to take a holistic, systematic approach to handling the transition to the agile development method. Although this research focuses on the experience of Cisco, a large company, it also sheds light on the challenges of adopting agile development practices by small to medium-size companies. For example, as discussed earlier, companies need to recruit the right customers to help steer their agile development process; talking to the wrong collaborative customers can lead to costly detours for the company. On the other hand, only a small subset of customers is qualified to serve as early collaborative customers. This dilemma presents challenges for small to medium-size companies. Small customer pools and limited resources will reduce their ability to find the right early collaborative customers. This type of challenge could be especially true for small to medium-size companies in B2B space where the customer base tends to be more limited and more relationships would be required to recruit collaborative customers.

This study also sheds light on the challenges that the agile development process poses for global operations. Our study illustrates that the agile method demands close and intense coordination with customers and requires organizational units and engineering teams to be self-contained and autonomous. These characteristics create new challenges for global integration and global operation where employees and organizational units from different countries work together on certain projects. Such multi-location operations lead to more coordination complexity, which can be detrimental to the agile development process. As more companies adopt the agile development method and as globalization and integration further develop, companies will have to find new global operating models

Figure 2. A summary of management tasks to cope with the organizational transition to the agile development method



to reconcile the potential conflict between fast-paced agile development processes and multi-country global operations. In sum, although this study only analyzes a small set of management practices associated with the adoption of the agile method. We hope our work will invite more research analyzing new management practices and systems to enable and support the implementation of the agile development method.

References

Ambler, S. (2005a). Roles on agile teams: From small to large teams. *Ambysoft*. Retrieved February 9, 2016, from http://www.ambysoft.com/essays/agileRoles.html

Ambler, S. (2005b). Agile project planning tips. *Ambysoft*. Retrieved February 9, 2016, from http://www.ambysoft.com/essays/agileProjectPlanning.html

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., et al. (2001). *Manifesto for agile software development*. Retrieved February 8, 2016, from http://agilemanifesto.org/

Boehm, B., & Turner, R. (2003a). Balancing agility and discipline. In F. Maurer & D. Wells (Eds.), *XP/Agile universe* (pp. 1—8). Berlin: Heidelberg.

Boehm, B., & Turner, R. (2003b). Using risk to balance agile and plan-driven methods. *Computer, 36*(3), 57—66.

Cohen, D., Lindvall, M., & Costa, P. (2004). An introduction to agile methods. In M. V. Zelkowitz (Ed.), *Advances in computers vol. 62: Advances in software engineering*. Cambridge, UK: Academic Press.

Gandomani, T. J., & Nafchi, M. Z. (2015). An empirically-developed framework for agile transition and adoption: A grounded theory approach. *Journal of Systems and Software, 107*, 204—219.

Gandomani, T. J., Zulzalil, H., Ghani, A. A. A., & Sultan, A. B. M. (2013). Towards comprehensive and disciplined change management strategy in agile transformation process. *Research Journal of Applied Sciences, Engineering, and Technology, 6*(13), 2345—2351.

Hoda, R., Noble, J., & Marshall, S. (2011). Developing a grounded theory to explain the practices of self-organizing agile teams. *Empirical Software Engineering, 17*(6), 609—639.

James, M. (2010, March 15). Obstacles to enterprise agility. *Collabnet*. Retrieved February 9, 2016, from https://www.open.collab.net/media/pdfs/How_to_Steer_Large_Scale_Development.pdf

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM, 48*(5), 72—78.

Qumer, A., & Henderson-Sellers, B. (2008). A framework to support the evaluation, adoption, and improvement of agile methods in practice. *The Journal of Systems and Software, 81*(11), 1899—1919.

Strode, D. E., Huff, S. L., Hope, H., & Link, S. (2012). Coordination in co-located agile software development projects. *Journal of Systems and Software, 85*(6), 1222—1238.

Thomke, S., & Reinersten, D. (1998). Agile product development: Managing development flexibility in uncertain environments. *California Management Review, 40*(1), 8—30.

Turner, R., & Boehm, B. (2003). *Balancing agility and discipline: A guide for the perplexed*. Boston: Addison-Wesley/Pearson Education.