CHAPTER

20 Emerging Development Practices[1]

Poor development productivity has been a perennial problem for IT (Brooks 1975; McKeen and Smith 2003; Oman and Ayers 1988). "IT takes too long to deliver" is a common complaint among today's business leaders (Luftman and Zadeh 2011; Overby 2005). Over the past three decades (or more), a considerable number of panaceas have been proposed for helping organizations to get the systems and IT functionality they need better, faster, and cheaper. Structured approaches to programming and design and the introduction of systems development life cycle methodologies were first. Then came automated systems development tools, attempts to measure productivity (e.g., function points), and new development approaches such as rapid application development (RAD). More recently, organizations have sought to buy off-the-shelf software, use middleware to integrate it, introduce enterprise resource planning systems (ERPs), or adopt software-as-a-service in order to deliver more functionality at a lower cost. Companies have also realized that the processes *around* systems development, such as system prioritization and enterprise architecture, can have a significant impact on development timelines and most now have procedures in place to manage these activities. Finally, many organizations have turned to contract or outsourced staff (often in other countries) to help them with extra resources during high-demand periods or to provide a large group of qualified development personnel at a lower overall cost (Han and Mithas 2014; Lacity and Willcocks 2001).

Nevertheless, over the past decade the situation has gotten worse in many ways. Changes in technology, connectivity and collaboration, and the introduction of open standards has meant that the IT function is sitting at the intersection of two powerful and rapidly changing forces—technological innovation and globalization—and IT has become absolutely critical to effective business strategy. Furthermore, development teams are becoming increasingly complex to manage, incorporating people and partners from different companies and locations. And development activities are more challenging, involving many regulatory, architectural, business, financial, HR, security, and risk

[1] This chapter is based on the authors' previously published article, Smith, H.A., J.D. McKeen and W.A. Cram, "Enhancing Development Productivity," *Journal of Information Technology Management*, XXXIII, no. 3, September 2012. Reproduced by permission of the Association of Management.

management hoops that have little to do with the traditional design and coding of the past but that need to be orchestrated to deliver a coherent, viable service. Unfortunately, new systems development techniques have not always kept pace with these changes. Many that have promised, such as service-oriented architecture (SOA), software-as-a-service, and agile development, still have not displaced traditional approaches. At the same time, the new technical and managerial practices needed to support them have not been fully introduced. In short, improved development productivity is still long on promises and short on delivery.

This chapter explores improving development productivity from a number of perspectives. It begins by examining the problem of IT development productivity and how system development practices are changing. It then explores the key obstacles involved in improving development productivity and outlines practices that are proven to work. It concludes with recommendations for managers about how to create an improved environment for systems development productivity.

THE PROBLEM WITH SYSTEM DEVELOPMENT

In the past, the focus group explained that "system development" largely meant creating customized software applications for an individual organization. Today, it still means custom building but development also includes selecting, implementing and integrating packaged software solutions, and increasingly, integrating smaller, reusable software components with existing legacy applications across a variety of platforms with a variety of development tools. However, although systems development has changed over time, many of the problems associated with it have not changed; that is, there are still very high failure rates with development projects and they are still perceived to take too long, cost too much, and deliver limited business value (Korzaan 2009).

Research has not been particularly helpful in providing ways to improve on any of these fronts. There have been few studies of actual development practices to determine what works and under what circumstances and there is thus very little on which to base guidelines for different types and sizes of development (Dyba and Dingsoyr 2009). In short, "we need to know more about what we know and don't know about software development" (Adams 2009). One study noted that improvement in software development models and best practices has been a "long slog" since the 1980s and using the traditional "waterfall" model of systems development[2] has "continued to fail in delivering acceptable measures of software development performance" (Royce 2009). The Standish Group's ongoing study of software development success rates shows that in 2009 only 32 percent were considered successful (that is, on time, on budget and with the required features and functions), while 24 percent were considered failures (i.e., they were cancelled or never used). The remaining 44 percent either finished late, were over-budget, or had fewer than required features or functions (Levinson 2009). While these measures have improved somewhat since 1994, progress has been agonizingly slow.

Although IT practitioners and consultants have worked hard to define a strict set of rules to guide and govern software development, and have seen some modest gains

[2] By this we mean a system development life cycle (SDLC) approach in which all requirements are first defined, and then an application is designed, developed, tested, and implemented with few changes.

from such factors as improved governance, project management offices, and better methodologies, many believe that "rules don't work and haven't since 1967" (Berinato 2001). These ongoing problems have meant that system development has long "suffered from way too many management fads and silver bullets *du jour*…and [left managers prey to] consultants and sellers of 'software oil'" (Adams 2009).

Finally, system development continues to be plagued by the difficulty of measuring "productivity." What exactly is a successful systems development project? Many companies define it as meeting schedules and budgets and by the functionality delivered (Levinson 2008). Yet, these common metrics typically "do more harm than good" (Cardin et al. 2008). While they are easy for business people to understand, they perpetuate a "myth" that these are the only three factors that make a project successful. Furthermore, they take no account of some major elements that are often responsible for project failure, such as changes in requirements or scope, unreasonable deadlines, project dependencies, and lack of business accountability (Levinson 2008). "We still have no formal productivity metrics," said one IT manager, "and it's not a priority for us." Nevertheless, said another, summarizing the challenge faced by everyone in the focus group, "we are still expected to deliver business value with increasing speed and efficiency."

TRENDS IN SYSTEM DEVELOPMENT

For many years, system development has been conceptually seen as a functional, engineering project, similar in nature to building a bridge (Chatterjee et al. 2009). Unfortunately, efforts to develop methodologies that embody software engineering principles designed to lead to consistent performance outcomes, while resulting in some improvements, have not been as successful as predicted (Chatterjee et al. 2009; Royce 2009). Therefore, in the past two decades, numerous efforts have been made to address system development productivity shortcomings in other ways, including the following:

1. ***Adopting new development approaches.*** There are a significant number of new development approaches that their proponents believe address some or all of the problems with the traditional waterfall development method. While a comprehensive assessment of these approaches is beyond the scope of this chapter, they can be classified into three major types:
   - ***Agile.*** Introduced in the 1990s, this approach encompasses a variety of "anti-waterfall" methods of system development, such as spiral, incremental, evolutionary, iterative, and RAD. They stress the need to incorporate flexibility into system development by breaking up a large project into smaller pieces that can be developed in overlapping, concurrent phases to rapidly deliver business value in a series of short increments. Speed and agility are achieved by collapsing or compressing one or more phases of the waterfall method and by incorporating staged delivery or incremental implementation (Jain and Chandrasekaran 2009).
   - ***Composition.*** This approach models and develops generic components comprising data, processes, and services that can be reused in different development efforts (Plummer and Hill 2009). Based on detailed analysis and architecture, components (e.g., acquire customer name and address) can be plugged into any system without being reprogrammed. Initially called "object

oriented programming" in the 1990s (McKeen and Smith 1996), and "service oriented architecture" (SOA) more recently, composition has been difficult to achieve because of the intensive modeling and architecting required and the IT organizational changes require to adapt to them (Blechar and Norton 2009; Plummer and Hill 2009). With this approach, system development becomes process orchestration, combining various software components into an "application container" (Blechar 2010).

- ***Integration.*** The 1990s also saw the widespread introduction of packaged software to the marketplace that could be purchased and implemented rather than developed in-house. As a result, many companies, including most of those in the focus group, adopted a "buy don't build wherever possible" philosophy for their generic applications, such as accounting, human resources, or customer relationship management. More recently, this marketplace has begun to evolve so that companies can purchase software-as-a-service from the cloud, rather than implementing it within their own organizations. Although preprogrammed, such services or packages still require various amounts of effort to select and then integrate them into an organization's existing processes, platforms, and data (Mahoney and Kitzis 2009; Plummer and Hill 2009).

However, for most companies, adopting new development approaches still involves using them only selectively and change has been agonizingly slow as a result.

2. ***Enhancing the waterfall methodology.*** Although new development approaches are gaining ground in organizations, the waterfall remains the predominant system development process for large-scale, industrial strength projects (Royce 2009; Schindler 2008). The waterfall method is still considered most practical for large system development projects because the engineering principles implicit in it involve formal coordination strategies, centralized decision making, formal communication, and prescribed controls, which help to offset the challenges caused by the increased complexity and interdependencies and reduced communications opportunities on large projects (Xu 2009). The focus group's presentations concurred with this assessment. "While we are trying to introduce new and more flexible approaches to development, our senior management is not committed to them and are resisting them," said one manager. "We're doing lots of experimentation with different development approaches but these are done within our standard methodology," said another. Improving the waterfall development process is therefore still a high priority for most companies. In recent years, organizations have attempted to improve the "maturity" of their traditional software development processes using Capability Maturity Model Integration (CMMI) to move them from ad hoc activities to more managed, better defined, quantifiable processes, so they yield standardized, replicable results (Chatterjee et al. 2009; Hanford 2008). For example, one focus group company has created an enhanced delivery framework complete with a process map, detailed activities, templates, inputs, outputs, entry and exit criteria, artifacts, roles, and links to standards. Another manager stated, "We have well-defined SDLC methodologies and standards and procedures are enforced . . . [But] we are always looking for applications development best practices to improve them."

3. ***Improved governance.*** It has also been accepted that there are a number of factors other than the development process itself that will affect the quality and the effectiveness of systems development. Today, in spite of a persistent engineering mind-set that permeates system development practices, there is also growing acceptance that building systems can be more of an art than a science. "Systems are a unique and complex web of intellectual property bounded only by vision and human creativity…They are more similar to movie production [than bridge-building] where no laws of physics or materials apply…most quality is subjective [and] anything can change" (Royce 2009). To deal with these conditions, some organizations are beginning to adopt governance mechanisms based on economic disciplines that accept the uncertainties involved in systems development—especially at the beginning—and adapt and steer projects through the risks, variances, and moving targets involved (Royce 2009). Thus, many focus group companies have adopted different governance practices for different stages of the development life cycle, such as staged estimates of cost and time, "gating reviews," and quality assessments at different life cycle phases. Other governance mechanisms, such as those used in Sweden, also consider the social and cultural implications involved (Chatterjee et al. 2009). Still others govern by a set of software outcomes, including flexibility, responsiveness, operational efficiency, quality of interaction, learning, product performance, and benefits achieved (Liu et al. 2009; Smith et al. 2010). In the focus group, most managers stressed that compliance with all legislation and regulations has become a further significant governance issue for all their systems initiatives. Some also stressed the need for better governance of the processes that "touch" and impact systems development activities, such as quality assurance, architecture, security, and testing. In short, governance at a variety of levels is becoming more important to ensure productivity in systems development (Plummer and Hill 2009).
4. ***Changing resourcing strategies.*** One trend in systems development that is very clear is the widespread use of contractors and outsourced developers to supplement in-house development staff. A major driver behind improved governance, methodologies, standards, and componentization of software is the desire to use cheaper development labor, often located in other countries. This globally dispersed development, however, increases the need for new internal business and technical skills. New resourcing strategies increase the need for better business, technical and data architecture, improved business analysis, IT strategy that is more closely linked to business, and project managers who can coordinate and leverage the efforts of a diverse group of internal and external, IT and business staff to deliver consistent and effective IT products (Blechar 2010; Plummer and Hill 2009). At present, only 28 percent of CIOs believe that they have the right skills in their IT organizations to support these changes (Mahoney and Kitzis 2009). The group agreed that development skills are changing. "Our focus is on improving project management, business analysis and quality assurance staff," said one manager. "We're stressing the development of relationship management, analysis and consulting skills," said another. "Improved resource allocation is also essential," said a third, "because there are only so many staff with the necessary skills. In the past, each business unit had dedicated resources; now they all work for the enterprise."

OBSTACLES TO IMPROVING SYSTEM DEVELOPMENT PRODUCTIVITY

It is clear from the earlier mentioned trends that systems development *is* changing and has changed to address complaints of poor productivity. However, it is also clear that these changes are still not adequately addressing the problem. There are several reasons why improvements in development productivity have been difficult to achieve. While many of them may not be surprising to long-time IT managers, they bear repeating since they pose significant barriers to success in this area.

First, there is still a need for a more holistic understanding of system development, both within IT and within the business. As already noted, development is a much more complex and uncertain process than was first understood. Too often, our mental models of development appear to be dated—locked into a time in the past when the problem being addressed was straightforward and the programming effort significant. Today, the programming is straightforward, while the problems are highly complex, typically involving many parts of the business and many IT functions and requiring significant business knowledge, technical skill, relationship and communications abilities, and conceptual understanding (Chakraborty et al. 2010). In an earlier look at this subject we noted that *all* activities impacting system development should be considered when trying to improve productivity. "There is a need to ensure that everything works together to further the overall goal. It makes no sense to improve one part of the process if it doesn't accomplish this" (McKeen and Smith 1996). Members of the focus group identified three primary areas where there are currently significant bottlenecks in the development process:

- ***Business involvement.*** This can be an obstacle to development success at several levels. At the highest level, it is well-known that business sponsorship is essential to ensure that the right projects are developed (Hanford 2008). While many organizations have addressed this problem through their governance processes, the focus group stressed that many business leaders still pay only lip service to their responsibilities. This impacts the system development process in several ways. "Our business users take forever to complete their parts, such as agreeing to a proposed solution or signing off on key phases of a project," said a manager. "They don't see how this affects our work, which can't proceed without it." The focus group felt strongly that business users needed more education about their roles in (and impact on) every level of the system development process, including governance, analysis, testing, and change management, in order to make development more productive.
- ***Analysis.*** "We were very surprised to find that analysis takes about 30 percent of the elapsed time of development," said one manager. "Business analysis is not at the same level of maturity as other parts of development," said another. Analysis can be an obstacle to productivity and effectiveness in many ways, in addition to the time it takes. Significant problems can be caused by failing to clearly define the scope of a project, to understand the dependencies between projects, to identify the changes that will need to be made to business processes when a system is implemented, or to recognize and incorporate the needs of multiple stakeholders in system requirements (Lemmergaard 2008; Levinson 2008).
- ***Testing.*** Several companies are focusing on testing, which they have found takes between 20 and 40 percent of development effort and resources. "We are spending

increasing amounts of money on testing; it's a growing job," said one manager. "It's extremely complex and expensive to set up and maintain test environments," said another. In system development, testing is typically done by three groups—the development team itself; quality assurance; and business users. Delays often occur with the last two groups, who focus on their own needs and optimize their own processes with little regard for their impact on the progress of an individual project or the business as a whole.

Second, the systems development process itself continues to be problematic. Today, many organizations try to force fit all projects to a single development approach, often with disastrous results (Norton and Hotle 2010). If there's one thing that practitioners and managers agree on, it's that whatever development approach is used, it should be appropriate for the project being undertaken. Typically, small projects suffer from too much process when a full-scale, CMMI-style methodology is used, while large projects cannot coordinate all their variables using an agile development approach (Adams 2009). Agile approaches are useful when requirements are not fully known or in rapidly changing business conditions. Yet, "for most organizations, [agile development] should be known by the acronym BDSF (delivering bad software fast)" (Norton and Hotle 2010). Conversely, too much process makes a project inflexible and adds layers of red tape that causes a project to bog down (Levinson 2009). Not using a methodology is not the answer as this can increase the risk that important tasks will fall through the cracks or that a project won't be completed on time (Levinson 2008). Members of the focus group were finding resistance to an overabundance of methodology from within IT as well as from the business. Thus, the ongoing challenge for IT managers is to find the right balance between structure and consistency and speed and flexibility.

Third, poor communication on the part of both IT and business tends to create misunderstandings and conflicts that can inhibit projects. One of the major goals of a good development methodology is to mediate between all stakeholders to prevent the changes in requirements and scope that result from problematic communication. But communications issues cannot be fully dealt with by a methodology (Liu et al. 2009). "Most of the project management mistakes IT departments make boil down to either a lack of adequate planning or breakdowns in communication (either among the project team or between the project team and the project sponsors. These mistakes can be fatal" (Levinson 2008). While much of the blame for ineffective communication tends to be placed on IT (Smith and McKeen 2010), there is considerable evidence that business people do not take the time or make the effort to understand what is being said to them (Liu et al. 2009). "Our business doesn't want to hear about what we must do," said a focus group manager. Too often, executives rely on simplistic metrics, such as progress against schedule and budget, because they are easy to understand. These in turn perpetuate the perception of poor development productivity (Royce 2009). "Project sponsors latch on to initial estimates…and because [they] don't understand project complexity and other factors influencing cost and timelines…they may see a project as a failure…even if changes resulted in improved value…" (Levinson 2008). Improved communication about changes in requirements, cost estimates, and schedules is therefore critical to improving perceptions of development productivity and success (Cardin et al. 2008).

IMPROVING SYSTEM DEVELOPMENT PRODUCTIVITY: WHAT WE KNOW THAT WORKS

There is still a lot that we don't know about improving system development productivity and members of the focus group were actively experimenting with a wide variety of initiatives in this regard, which may or may not be successful. However, they identified five sets of practices that they believed clearly made a significant difference:

1. ***Optimize the bigger picture.*** System development should be seen as only one part of an overall business and technical effort to deliver value to the enterprise. This starts at the top with a clearer understanding of the IT value proposition: delivering strategic insight and leadership; understanding business needs and designing solutions; and sourcing solutions implementation (Mahoney and Kitzis 2009). This bigger picture has a number of implications for both business and IT. First, IT and business strategy must be closely aligned to ensure IT is working on the right things and in the right order, said the focus group. Business and technology architecture functions, combined strategic governance, roadmaps, and improved business and IT relationships should all be designed to deliver *enterprise* value (not IT or business unit value). Second, all aspects of the earlier stages of development need to be reassessed and streamlined, including governance activities around project approvals, prioritization and funding; managing demand; educating business people in their roles and responsibilities in system development and holding them accountable; improving business casing; information and solutions architecture; use of proofs-of-concept, prototypes, and use cases; and developing strong project managers with excellent communications skills. Finally, resource management and sourcing strategies must be developed to ensure staff with the right skills are available when needed; applications development best practices need to be monitored and implemented; and testing and quality assurance should be centralized to eliminate duplication of effort.

   However, although each of these activities is important, none should be optimized at the expense of delivering overall value. All too often, individual functions seek to do the best job possible but forget how their work affects the overall goal. It is therefore important for senior IT leaders to ensure that this goal is kept in mind by all groups involved in delivering solutions to the enterprise. One company has had significant success—reducing cycle time by 30 percent—through such holistic process improvements. Another noted, "Becoming more outcome-focused, optimizing the whole development process and developing a shared business/IT agenda has led to substantial productivity improvements for us."
2. ***Adopt more flexible processes.*** While not all companies are willing to give up on the waterfall development methodology, they all recognize that "just enough" process should be the goal. Ideally, a development approach should be matched with the deliverables involved and the level of compliance required (Hotle 2009). Focus group companies were actively exploring ways to accomplish this goal. One company has developed a methodology tailoring tool that helps determine the levels of oversight and control that are needed by outside groups (i.e., security, architecture, operations) according to the level of risk involved. Another company ranks its development projects into three tiers. "Tier 1 is very visible and requires a higher level of formality and governance; Tier 3 projects are encouraged to adopt more

agile approaches," said the manager. A third is encouraging "smarter execution choices" from a full range of development approaches by enabling teams to choose from a variety of methodologies depending on business needs. Finally, one manager noted that his organization uses a little bit of everything when it comes to its efforts to improve its productivity. "We have adopted a 'buy vs. build' approach and have packaged ERP systems in several divisions; we use composition services for data capture, transformation, and delivery between systems—to take the burden away from the system developers; and we use a combination of agile and waterfall methods for new development."

3. ***Reduce complexity.*** It is widely accepted that complexity is a major cause of slow system development (Chakraborty et al. 2010). Standardization wherever possible therefore reduces complexity and makes development more straightforward (Royce 2009). While aiming for flexibility, the focus group was therefore also trying to reduce complexity in a number of ways. One organization has cut back on the reporting it requires, for example limiting the paperwork for its Project Management Office to just a few short questions. "This has helped us a lot," said the manager involved. Standards are a key way most companies are using to limit technological complexity. "Multiple technologies, platforms, languages and tools mean more complex software engineering," said a manager. Finally, several companies are trying to increase reuse of software components. "We're actually tracking the amount of reuse in each system; doing this has led to a 50% increase in reuse and a corresponding 20% reduction in defects," said a manager, noting that making reuse a performance metric for systems has been an important factor in its success.
4. ***Enhance success metrics.*** Success is a multidimensional concept depending as much on perceptions as on objective reality. While, as noted earlier, metrics of progress against schedule and budget are too simplistic for the current development environment, it is also true that IT can overdo the metrics it provides (Levinson 2008). Metrics for system development should be designed to accomplish four goals and used selectively for different audiences:
   - ***Increase buy-in.*** System development is a team activity, with business and other parts of IT playing key roles on the team. It is therefore essential that all team members be committed to achieving the same goals. In fact, the more people are committed to a goal, the more likely they are to contribute toward its outcomes (Korzaan 2009). Thus, metrics that clearly link a project and its component parts (e.g., architecture, testing, change management) with delivering well-articulated strategic business value are most likely to ensure a coherent and consistent effort to deliver. Such metrics are usually developed in a business case but may also be part of an overall business or technical roadmap and should be kept front and center throughout system development (Smith and McKeen 2010).
   - ***Promote desired behavior.*** Measuring something is an important way to promote behavioral change (Kaplan and Norton 1996). Members of the focus group had therefore developed scorecards to track desirable new development behaviors, such as reuse, quality, and collaboration. These metrics are often designed to change perceptions within IT, regarding what management values in systems development.
   - ***Educate perceptions.*** Perceptions can be "educated, trained and controlled" (Gladwell 2005) and business perceptions of system development productivity

need management, transparency, and clear communication. Metrics therefore need to be interpreted for them by IT in light of business conditions and individual situations (Levinson 2008; McKeen and Smith 2009).

- ***Monitor performance.*** Finally, system development performance should be tracked to determine the actual results delivered rather than the progress of the various activities of the software development process (Royce 2009). "We need to become more outcome-oriented so that we don't get bogged down in process," agreed a focus group manager. "This is a fundamental change in IT's mind-set." Such a new mind-set also supports the shift to newer development approaches, such as agile, package implementation, reuse, and delivery of software-as-a-service.

5. ***Create a smarter development environment.*** Getting "smarter" about development involves improving collaboration, knowledge sharing and capabilities, and finding new opportunities for leveraging the work that is done. With the boundaries between business and IT becoming increasingly blurred and larger numbers of stakeholders involved in the process (both within IT and in business), development has become both a much more social and multidisciplinary process, while at the same time teams are becoming increasingly dispersed geographically (Chakraborty et al. 2010; Mahoney and Kitzis 2009). Collaboration and knowledge sharing initiatives can enhance traditional forms of communication, facilitate relationship building, and ensure that there is a single version of the "truth" available to everyone on a project team. Several companies in the group have implemented collaboration and document sharing tools with considerable success. "Our top priority is promoting collaboration with the business," said one manager. Another is implementing knowledge repositories and document-sharing software to enable better access to work that has already been done. Improved search capabilities are also a top priority for companies seeking to improve reuse. Another focus group company is stressing improving its capabilities by creating communities of practice around its four main technology disciplines (i.e., project management, business analysis, development, and quality assurance) to create thought leadership that is "more than the sum of its parts" and drive change throughout the IT organization. One has identified the key gaps in capabilities for its major functional areas and is developing learning paths to close them. Finally, companies are becoming smarter about how they handle requests for compliance projects, for example, gathering all compliance requirements together in planning to ensure that they are dealt with "once for all."

NEXT STEPS TO IMPROVING SYSTEM DEVELOPMENT PRODUCTIVITY

Although these five general trends in systems development are working well in the focus group companies, their breadth and the integration and behavior change required is daunting. While keeping these "big picture" initiatives in mind, the managers in the group identified five "quicker fixes" that were likely to have an immediate impact on productivity, while furthering these larger goals:

- ***Look for and address bottlenecks.*** Assessing the entire system development process for bottlenecks in an organization can yield surprising results. One company had no

idea how long it took business sponsors to complete sign-offs; another found that cumbersome governance processes took inordinate amounts of time to resolve simple conflicts. With time pressures extreme these days, it makes sense to identify and speed up such bottlenecks first rather than increasing pressure on the core members of the development team.

- ***Focus on outcomes.*** As already noted, IT metrics have typically measured elements of the process, such as consumption of resources, rather than value delivered. With the development world changing rapidly due to the advent of software services and application assembly, it is essential to refocus both business and IT on *what* functionality is being delivered, not *how* it is delivered. Making the shift to a more dynamic, innovative, and effective IT organization means changing what is measured. One firm now undertakes a quarterly assessment across its entire IT organization of the seven key capabilities it wants to develop: community participation, collaboration, transparency, innovation, agility (i.e., time to value), component-based development, and asset management and reuse. It believes encouraging these behaviors will promote faster time to market for all its development initiatives.
- ***Clarify roles and responsibilities.*** Several firms have seen commitment to development projects increase, both from internal IT groups and from business sponsors and users when their roles and responsibilities were clarified. For example, one company clearly explains where IT architecture is accountable in system development, when it should be consulted, and when it should merely be informed. Another provides clarity about who is responsible for resolving development problems. "This has helped us to stop churning and increase motivation," said the manager. Another manager, who had overseen a transition from a traditional waterfall IT development organization to an SOA function, stated, "Making change is *all* about clarity of roles and responsibilities."
- ***Simplify the development environment.*** All companies in the focus group had some initiatives to decommission or replace end-of-life or duplicate technologies and applications. Some are attacking this type of complexity more vigorously than others. One firm had slashed its legacy applications by one-third over the past three years. The benefits of a simpler environment are numerous—speed of implementation, flexibility, more investment dollars, and easier new technology deployment. In particular, one firm that had mandated a single desktop and common infrastructure found it dramatically increased its time to market for new development initiatives.
- ***Simplify testing.*** Testing has long been seen as a system development bottleneck (McKeen and Smith 1996), and with the addition of more complex technological environments and more stringent compliance regulations, requiring separate groups to perform different types of testing, the situation has become much worse in recent years, said the focus group. Therefore, they have each put much effort into streamlining and automating this activity. Many companies have created a centralized test environment with automated scripts and standard tests that dramatically increase throughput. "With these you are not starting from scratch each time," said a manager. Testing tools and methods, including automated regression testing, risk assessments, and analysis of defects have helped both to speed up the process and provide the necessary documentation of results.

Conclusion

Much has improved in the practice of system development over the past two decades and if the development environment had stayed static, it is likely that productivity would also have been perceived to have improved dramatically. Instead, systems have become increasingly complex at every level so process improvements have barely made a dent in the dilemma of development productivity. This chapter has addressed the ongoing nature of the productivity problems facing IT managers in systems development and how the field is changing. It has examined some of the serious systemic barriers to fundamental change in how systems are developed and documented best practices for dealing with them. There is unfortunately no silver bullet when it comes to improving system development productivity, in spite of much effort to find one. While a few organizations are "pushing the envelope" in an attempt to radically change how systems are delivered, for most, improvements are more likely to come as a result of persistent and iterative analysis of what works and what doesn't in their particular organizational context.

References

Adams, W. "An Agile Cure for All Ills?" *IEEE Software* 26, no. 6 (November/December 2009): 8.

Berinato, S. "The Secret to Software Success." *CIO Magazine,* July 1, 2001.

Blechar, M. "Why You Should Coordinate Your SODA, MDM and Business Process Improvement Initiatives." Gartner Group, ID Number: G00175167, March 24, 2010.

Blechar, M., and D. Norton. "Trends in Model-Driven Development, 4Q09-3Q10." Gartner Group, ID Number: G00169442, August 27, 2009.

Brooks, F. *The Mythical Manmonth: Essays on Software Engineering.* Reading, MA: Addison-Wesley Publishing, 1975.

Cardin, L., A. Cullen, and T. DeGennaro. *Debunking IT Project Failure Myths*. Cambridge, MA: Forrester Research, July 28, 2008.

Chakraborty, S., S. Sarker, and S. Sprateek. "An Exploration into the Process of Requirements Elicitation: A Grounded Approach." *Journal of the Association for Information Systems* 11, no. 4 (April 2010): 212–49.

Chatterjee, S., S. Sarker, and M. Fuller. "Ethical Information System Development: A Baumanian Postmodernist Perspective." *Journal of the Association for Information Systems* 10, no. 11 (November 2009): 787–815.

Dyba, T., and T. Dingsoyr. "What Do We Know about Agile Software Development?" *IEEE Software* 26, no. 5 (September/October 2009): 6–9.

Gladwell, M. *Blink: The Power of Thinking without Thinking,* New York: Little Brown and Company, 2005.

Han, K., and S. Mithrs. "The Real Saving from IT Outsourcing." *MIT Sloan Management Review* 55, no. 2 (Winter 2014).

Hanford, M. "The CMMI for Development Value Proposition for the PMO." Gartner Group, ID Number: G00158078, May 21, 2008.

Hotle, M. "'Just Enough Process' Is Built on Deliverables." Gartner Group, ID Number: G00168230, September 22, 2009.

Jain, R., and A. Chandrasekaran. "Rapid System Development (RSD) Methodologies: Proposing a Selection Framework." *Engineering Management Journal* 21, no. 4 (December 2009): 30–35.

Kaplan, R., and D. Norton. *The Balanced Scorecard,* Boston, MA: Harvard University Press, 1996.

Korsten, P. "The Essential CIO." *IBM Institute for Business Value,* Somers, NY: IBM Global Business Services, 2011.

Korzaan, M. "The Influence of Commitment to Project Objectives in Information Technology (IT) Projects." *Review of Business Information Systems* 13, no. 4 (Fourth Quarter 2009): 89–97.

Lacity, M., and L. Willcocks. *Global Information Technology Outsourcing: In Search of Business Advantage,* Chichester, England: John Wiley and Sons, 2001.

Lemmergaard, J. "Roles in the ISD Process: A Collaborative Approach." *Journal of Enterprise Information Management* 21, no. 5 (2008): 543–56.

Levinson, M. "Common Project Management Metrics Doom IT Departments to Failure." *CIO Magazine,* August 1, 2008.

Levinson, M. "Recession Causes Rising IT Project Failure Rates." *CIO Magazine,* June 18, 2009.

Liu, J, G. Klein, J. Chen, and J. Jiang. "The Negative Impact of Conflict on the Information System Development Process, Product and Project." *Journal of Computer Information Systems* 49, no. 4 (Summer 2009): 98–104.

Luftman, J., and H. S. Zadeh. "Key Information Technology and Management Issues 2010–11: An International Study." *Journal of Information Technology* 26, no. 3 (2011): 193–204.

Mahoney, J., and E. Kitzis. "Integrating the Transformation of Business and IT." Gartner Group, ID Number: G00167927, May 15, 2009.

McKeen, J., and H. Smith. *Management Challenges in IS: Successful Strategies and Appropriate Action.* Chichester, England: John Wiley and Sons, 1996.

McKeen, J., and H. Smith. *Making IT Happen: Critical Issues in IT Management.* Chichester, England: John Wiley and Sons, 2003.

McKeen, J., and H. Smith. *IT Strategy in Action.* Upper Saddle River, New Jersey: Pearson Education, 2009.

Norton, D., and M. Hotle. "Best Development Methods: A Scenario View." Gartner Group, ID Number: G00171772, March 18, 2010.

Oman, R., and Ayers, T. "Productivity and Benefit-Cost Analysis for Information Technology Decisions." *Information Management Review* 3, no. 3 (Winter 1988): 31–41.

Overby, S. "Turning IT Doubters into True Believers: IT Value." *CIO Magazine*, June 1, 2005.

Plummer, D., and J. Hill. "Composition and BPM Will Change the Game for Business System Design." Gartner Group, ID Number: G00173105, December 21, 2009.

Royce, W. "Improving Software Economics: Top 10 Principles of Achieving Agility of Scale." *IBM White paper,* May 2009.

Schindler, E. "Getting Clueful: 7 Things CIOs should Know About Agile Development." *CIO Magazine,* February 6, 2008.

Smith, H. A., and J. D. McKeen, "How to Talk so Business Will Listen…And Listen so Business Can Talk." *Communications of the Association of Information Systems* 27, no. 13 (August 2010): 207–16.

Smith, H., J. D. McKeen, C. Cranston, and M. Benson. "Investment Spend Optimization: A New Approach to IT Investment at BMO Financial Group." *MIS Quarterly Executive* 9, no. 2 (2010): 65–81.

Xu, P. "Coordination in Large Agile Projects." *Review of Business Information Systems* 13, no. 4 (Fourth Quarter 2009): 29–43.