Critical Decisions in Software Development: Updating the State of the Practice

Michael A. Cusumano and Alan MacCormack, *Massachusetts Institute of Technology*

Chris F. Kemerer, *University of Pittsburgh*

William Crandall, *Hewlett-Packard*

This article discusses choosing a development process, structuring global design chains, managing the interaction of project structure and software design, and balancing innovation and efficiency.

Our 2003 article "Software Development Worldwide: The State of the Practice"[1] demonstrated how far software development has come since *IEEE Software* debuted in 1984. At that time, waterfall-style development processes led by star programmers with small, colocated, single-company teams were common. Our global survey of 104 projects showed the spread of iterative development practices (such as the use of early prototypes, multiple development subcycles, and daily builds) as well as generally accepted good software engineering practices, such as design and code reviews. Yet, the survey also revealed wide variations across countries and firms in both development practices and performance (productivity and quality).

In the subsequent six years, our collective research has continued to explore this variation to better guide the most critical decisions that software managers make. These decisions include how to choose the "right" software development process, how to structure global software design chains, how to manage the interaction of project structure and software design, and how to balance innovation and efficiency in a software business.

In January 2009, *Software*'s editorial and advisory boards selected Michael Cusumano, Alan MacCormack, Chris Kemerer, and William (Bill) Crandall's 2003 article "Software Development Worldwide: The State of the Practice" (Nov./Dec., pp. 28–34) as one of the magazine's all-time best peer-reviewed articles ("*IEEE Software*'s 25th-Anniversary Top Picks," pp. 9–11). We asked the authors to reflect on what has changed since 2003 and provide an update on their insights. In the following article, Michael, Alan, Chris, and Bill compress years of research and experience into four contemporary questions central to succeeding in the software development business. Their to-the-point summary stands out in an environment in which we're bombarded with many different success recipes covering a multiplicity of angles, which can cause us to lose sight easily. *—Hakan Erdogmus*

The State of the Practice: 2003 Revisited

Regarding process variation, approximately 80 percent of European and Indian projects used multiple development subcycles, compared to about 55 percent in the US and 44 percent in Japan. US firms were more likely to use daily builds at a project's beginning (35 percent, compared to 9 percent in Europe). However, only 71 percent of the US projects ran a regression test on each build, compared to nearly all the Indian and Japanese projects. Indian and Japanese projects reported 100 percent use of design reviews, compared to 77 percent of the US and European projects. In sum, most Japanese projects adopted some form of waterfall process, US firms favored a more flexible process, and Indian firms adopted many practices associated with both conventional and more flexible models.

0740-7459/09/$26.00 © 2009 IEEE

Regarding performance, Japanese projects reported by far the highest quality, about one-tenth the defect level reported for European and Indian projects and one-twentieth the number for US projects. Japanese projects also reported the highest LOC productivity, with Europe a close second and the US and India lagging significantly. Although Japanese firms developed more software for mainframes and classified more projects as requiring higher reliability, these differences' magnitude was still surprising.

We developed our global survey as part of a parallel study conducted at Hewlett-Packard and Agilent, reported in the September/October issue of *IEEE Software*.[2] Here, our major observation was that iterative practices, particularly the use of early prototypes, daily builds, and regression testing, along with a conventional good practice—design reviews—were best used as part of a "coherent system of practices." Projects that combined all these methods performed equally well in terms of quality, compared to projects using a waterfall approach. These projects could also accommodate late design changes with no apparent trade-off in quality. In contrast, this result didn't hold for projects using these practices on an ad hoc basis or inconsistently.

How to Choose the Right Software Development Process

Given the wide range of processes available, how can managers confidently choose the right one to deliver high productivity and quality? Our research indicates that managers must select a combination of practices and integrate them into a coherent process that's aligned to their business context.

Compelling evidence for this view comes from Alan MacCormack and Roberto Verganti, who found that the level of uncertainty a project faces affects development practices' impact.[3] In particular, in projects facing high market uncertainty, releasing an early beta version to customers was critical. This practice wasn't associated with performance in projects where customer needs were more predictable. Furthermore, projects facing high platform uncertainty required greater investments in architectural design. In contrast, this practice had no impact on projects facing little platform uncertainty.

These findings help explain why we find diverse practices across industries and firms, as well as within the same firm: this can be a natural response to differing project contexts. No one "best practice" fits all projects and situations. Instead, managers should adopt a contingent view of software development, evaluating a range of contextual factors before choosing the most appropriate process to adopt in a specific project. Paradoxically, this insight suggests that a project's first stage shouldn't deal with product design but rather with designing the development process itself.

Choosing the right process is critical because managers have a wide variety of choices, including agile methods such as Extreme Programming, Scrum, Crystal, and Evo. Although each prescribes its own seemingly complete and coherent set of practices, there's little explanation of when to use each. Managers must avoid falling for the latest development fads, just as they must overcome the inertia that comes from sticking to what they know best—whether a standard process or the one they happened to use last. MacCormack and William Crandall and their colleagues recently described how Hewlett-Packard achieved this objective by using a small number of process "templates" and defining criteria for when to use each one.[4]

A project's first stage shouldn't deal with product design but rather with designing the development process itself.

How to Structure Global Software Design Chains

Offshore software development firms now play a key role for their Fortune 1,000 clients and increasingly take part in joint product development ventures. Apart from well-documented cost arbitrage benefits, an important factor attributed to the increase in offshore development is the dramatic improvement in offshore service providers' process quality and project management capabilities. Our 2003 study demonstrated Indian firms' progress, highlighting their adoption of many sophisticated software engineering techniques. Indeed, India possesses the largest pool of software firms assessed at level 5 of the widely adopted CMM.

Given software development's globalization, how can software managers succeed as outsourcers (or outsourcees)? Our ongoing research indicates that outsourcers can orchestrate global software design chains to lower their costs without sacrificing quality, productivity, or innovation. At the same time, outsourcees might struggle to profit in these design chains if they don't offer a sufficiently differentiated value proposition.

To probe the challenges inherent in offshore development, another research project involving Chris Kemerer investigated 42 projects at a large Indian firm with a level-5 CMM rating.[5] They aimed to examine whether structured software

Process improvement must extend beyond the software development life cycle.

processes can effectively mitigate task dispersion's potential negative effects. They found that, for every 1 percent increase in the dispersion of tasks to outsourced locations, productivity decreased by about 0.7 percent and quality by about 1.5 percent. However, their results also revealed that you can use key CMM process areas to create a platform for learning, making offshore development process improvement initiatives more effective. These results show that investments in structured processes, and the learning that comes from their adoption, can mitigate declines in productivity and quality.

This research has two important implications. First, placing some development tasks offshore does reduce project performance, even in a high-process-maturity environment. So, managers shouldn't discount these costs in estimating offshore development's impact. They must weigh the savings from distributing work to lower-cost locations against the possible loss of productivity and quality. Second, managers should recognize that process improvement must extend beyond the software development life cycle. Designing and implementing complementary routines, particularly geared to learning and adapting the process to better fit the context, are critical to realizing process improvement initiatives' benefits.

How to Manage Project Structure's Impact on Software Design

Historically, software development projects have focused on colocated teams, but now we also see many distributed team structures with multiple external partners. At the extreme, some software is being developed by large, global, self-organizing communities. Yet we still understand little about how these structural choices impact the resulting software design.

MacCormack and his colleagues recently explored this topic by analyzing a sample of functionally similar software products developed using different organizational modes—specifically, open source versus closed-source development.[6] The results revealed dramatic design differences. The open source products were significantly more modular than their closed-source counterparts. In essence, software products appear to mirror the structure of the organizations developing them.

Two main implications follow from this research. First, managers must recognize that the choice of project structure will have a distinct impact on the nature of the resulting design. A distributed team will naturally develop different solutions than would a team working in a single room. Second, attempts to change how development is organized will likely fail unless managers consider how to adapt the legacy design to support the new approach. In general, products developed by a colocated internal team aren't ready to move to a distributed team of external partners. To create the right "architecture for participation" will require a redesign.

How to Balance Innovation and Efficiency

Finally, our 2003 study raised several questions regarding why development strengths observed in India and Japan haven't yet led to greater success in innovation. Although large, profitable players exist in each market, Western firms such as Microsoft, Oracle, SAP, and IBM continue to dominate the software systems, applications, and product spaces.

Recently, Michael Cusumano observed that Indian firms continue to treat software mainly as a service business and, like Japan, haven't translated their considerable software engineering skills into products or reusable intellectual property that they might sell in India or globally.[7] Whereas Japan boasts the second-largest IT market in the world, Indian software companies rely almost entirely on foreign customers and face competition from China, Southeast Asia, Eastern Europe, and Russia. Moreover, as software product revenues have declined, firms such as Oracle and SAP have built up their service and consulting businesses to the point at which more than half their revenues consist of service and maintenance revenues, not product license fees.[8] So, we see former partners—Indian IT companies and Western product companies—competing for the same services pie. This change, along with rising wages, doesn't bode well for the future of Indian outsourcing and might require Indian firms to develop more of their own technology and development processes.

With Japan, the puzzle remains as to why high productivity and quality in software development haven't translated into a more prominent role in the global software industry. The problem might well stem from the development culture in large Japanese firms such as those that answered our survey. This culture has its roots in techniques from the 1980s, when software factories were popular. Although Japanese projects demonstrated by far the highest quality levels, their low bug rates might suggest an overly rigid development style and a preoccupation with zero defects, rather than attempts to experiment and innovate.[9] This problem, on top of language differences and a market focused on custom software, are likely the main reasons so few Japanese firms are creating innovative software products and selling them globally. (An important exception is video games, where the market benefits from a positive cultural tradition.)

So, on the basis of our research since the 2003 article, we can make four observations.

First, a project's context should dictate the development strategy it employs. Both iterative and waterfall methods have their place and can be effective in different project contexts, depending on how much uncertainty exists and how much flexibility a team requires.

Second, strong process capabilities can overcome the disadvantages of distributing development work, letting firms take advantage of lower costs or superior skills available offshore.

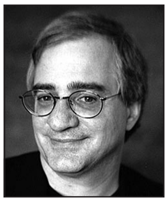
Third, because the choice of project structure will affect the resulting software design, managers must evaluate potential choices in terms of both project effectiveness and product performance.

Finally, although strong processes can help a firm improve quality and productivity in software development, potential trade-offs exist with respect to creativity and innovation. Overemphasis on process over, for example, novel products or technologies can be a disadvantage—as we appear to see in Japan and might well see in India.

References

1. M. Cusumano et al., "Software Development Worldwide: The State of the Practice," *IEEE Software*, Nov.–Dec. 2003, pp. 28–34.
2. A. MacCormack et al., "Trade-offs between Productivity and Quality in Selecting Software Development Practices," *IEEE Software*, Sept.–Oct. 2003, pp. 78–85.
3. A. MacCormack and R. Verganti, "Managing the Sources of Uncertainty: Matching Process and Context in Software Development," *J. Product Innovation Management*, vol. 20, 2003, pp. 217–232.
4. A. MacCormack et al., "Is Your Product Development Strategy Broken?" working paper, MIT Sloan School of Management, Feb. 2009.
5. N. Ramasubbu et al., "Work Dispersion, Process-Based Learning, and Offshore Software Development Performance," *MIS Quarterly*, vol. 32, no. 2, 2008, pp. 437–458.
6. A. MacCormack, J. Rusnak, and C. Baldwin, "Exploring the Duality between Product and Organizational Architectures: A Test of the Mirroring Hypothesis," working paper 08-039, Harvard Business School, 2008.
7. M. Cusumano, "Envisioning the Future of India's Software Services Business," *Comm. ACM*, vol. 49, no. 10, 2006, pp. 15–17.
8. M. Cusumano, "The Changing Software Business: Moving from Products to Services," *Computer*, vol. 41, no. 1, Jan. 2008, pp. 20–27.
9. M. Cusumano, "The Puzzle of Japanese Software," *Comm. ACM*, vol. 48, no. 7, July 2005, pp. 26–27.

About the Authors


Michael A. Cusumano is the Sloan Management Review Distinguished Professor at the Massachusetts Institute of Technology's Sloan School of Management, with a joint appointment in MIT's Engineering Systems Division. His research focuses on technology management and strategy, especially in the software business. Cusumano has a PhD from Harvard University in Japanese studies and completed a postdoctoral fellowship in production and operations management at the Harvard Business School. He's a director of Patni Computer Systems and the Eliza Corporation, and the author or coauthor of eight books, including *The Business of Software* (Free Press, 2004), *Platform Leadership* (Harvard Business School Press, 2002), and *Microsoft Secrets* (Free Press, 1995). Contact him at cusumano@mit.edu.

Alan MacCormack is a visiting associate professor at the Massachusetts Institute of Technology's Sloan School of Management. His research examines the management of innovation and new product development in high-technology industries, focusing on the software sector. MacCormack has a doctorate from the Harvard Business School. Contact him at alanmac@mit.edu.

Chris F. Kemerer is the David M. Roderick Professor in Information Systems at the University of Pittsburgh's Katz Graduate School of Business. His research interests include management and measurement issues in information systems and software engineering, and the adoption and diffusion of information technologies. He has been the departmental editor for *Management Science*, the editor in chief of *Information Systems Research*, and associate editor of *IEEE Transactions on Software Engineering*. Kemerer has a PhD in systems sciences from Carnegie Mellon University. He's a member of the IEEE Computer Society. Contact him at ckemerer@katz.pitt.edu.

William (Bill) Crandall is a former employee of Hewlett-Packard, where he was the senior director and general manager of HP Global Engineering Services. His research interests include product development, architecture, and innovation. Crandall has an MS in computer science and an MS in management from the Massachusetts Institute of Technology, where he was a fellow in the Leaders for Global Operations program. He's a member of the ACM. Contact him at bill_crandall@alum.mit.edu.