22 REFLECTIONS ON SOFTWARE AGILITY AND AGILE METHODS: Challenges, Dilemmas, and the Way Ahead

Linda Levine
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA U.S.A.

Abstract *What are the drivers for the burgeoning interest in agile methods? Have these drivers stimulated a similar rethinking on other fronts? What have we discovered? In this paper, I take a reflective stance in order to look at these larger issues and patterns. This stepping back is informed primarily by involvement in a multi-year research project on Quality Software Development @ Internet Speed and ongoing research on diffusion theory and the practices of technology adoption. I suggest the shift toward agile models and methods signals a larger transformation in the workplace toward the organization of the 21st century. This transition state is "between paradigms" and turbulent, marked by relentless change and volatility. The transition is a work in progress and by no means complete.*

Keywords Agile, agile methods, organizational dynamics

1 INTRODUCTION

Agility and agile methods have been popularized through the proponents of the Agile Alliance, their Agile Manifesto, and related writings (Agile Manifesto 2001). The concept of agility also has a longer history in manufacturing. More recently, Grover and Malhotra (1999) studied the interface between operations and Information Systems and Kathuria et al. (1999) linked information systems choices to manufacturing operations in order to understand how information systems support manufacturing operations and competitive strategy. Dove (2001) claims that agility requires an "ability to manage and apply knowledge effectively, so that an organization has the potential to thrive in a con-

tinuously changing and unpredictable business environment"(p. 9). Initially, he characterized agility as having two key elements: response ability and knowledge management. Subsequently, Dove (2005) added a third dimension of value propositioning.

For agile approaches to be fully understood—to mature and to gain ground—we would be wise to consider what agility means as part of a larger landscape, and what kind of shift it marks in technology development and in organizational behavior and change. This is the concern of this paper: to reflect upon the current preoccupation with agility, describe some of what we have learned about Internet-speed software development, and characterize challenges for the future.

What are the drivers for the burgeoning interest in agile methods? What have we discovered? In this paper, I take a reflective stance to look at such larger issues and patterns. Primarily, my stepping back is informed by two efforts: (1) involvement in a multi-year research project on Quality Software Development @ Internet Speed and (2) ongoing research on diffusion theory and the practices of technology adoption.

Agility in software development has implications for organizational agility. I will suggest that the shift to agile methods and models signals a larger transformation in the workplace toward the organization of the 21st century. This transition state is turbulent, marked by continuous change and volatility. Experimentation in this time of turbulence has attempted to break down and speed up old models, disrupting traditional approaches and turning conventional concepts and methods on their heads. No clear or easy solutions have resulted. The transformation is a work in progress, one that is by no means complete. To be realized, it will require a melding of inquiry across a wide range of disciplines and initiatives, including organizational development, diffusion of innovations, process improvement, knowledge management, complex adaptive systems, chaos theory, systems thinking, software engineering, and information systems.

We begin by looking briefly at definitions of agility, considering connotations and metaphors for agile behavior. Then, I discuss the current state of agility and Internet-speed software development, as informed by our research findings. Finally, I speculate on a desired state—and on challenges that the future holds for a next generation of agile approaches. Discussion of the future also involves consideration of conundrums and dilemmas.

2 DEFINING AGILE

What do we mean by agile? Is it simply fast? Are agile and fast one and the same? Agility implies speed, although something that is fast is not necessarily agile. Developers and customers alike appreciate speed, through being "first to market" and in terms of responsiveness. We know that developers are invested in how the use of agile methods emphasizes discovery, improvisation, and patterns.

Members of the Agile Alliance have expressed the following preferences and values (Agile Manifesto 2001):

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Do customers support agile methods? Perhaps, but not in precisely the same way that developers do—rather, they care as the use of such methods translates into the results, benefits, and profits that they seek. Thus, customer interest is indirect. This trend is evident even in large Department of Defense acquisitions, which are notoriously late and over budget, and where acquisition program managers are expressing interest in whether agile methods can better satisfy their goals and result in the delivery of quality systems in a more timely fashion. Some are actively advocating for agile methods. Unfortunately, these same customers are often at a loss when it comes to identifying an appropriate means for governance—for oversight and monitoring agile development efforts. Development efforts that embrace the left-side values presented above do not lend themselves well or easily to program monitoring. We will discuss this further under the topic of challenges.

Agility, by definition, exists in relief against a norm or opposite. In this regard, agility is relative; we know that a behavior is agile because we can compare it, if only in our own heads, with a visible or invisible state that is slower, clumsy, brittle, or inflexible. Who, we might ask, displays agility? Acrobats, ballerinas, and racing car drivers may all be agile. Gazelles, deer, and big cats may be agile. Elephants and hippopotami are not agile, or so we believe.

Merriam Webster (2004) defines agile from the Middle French and from the Latin *agilis,* from *agere* to drive, act as "**1:** marked by ready ability to move with quick easy grace [and] **2:** having a quick resourceful and adaptable character <an *agile* mind>." Agility is defined as "the quality or state of being agile**: NIMBLENESS, DEXTERITY** <played with increasing *agility*>."

Agility, then, for purposes of our discussion is made up of several attributes. We can liken it to a table which stands on four legs:

- *speed*: quick, fast.
- *nimble*: able to improvise, and use patterns creatively to construct new solutions on the fly, flexible.
- *adaptable*: responsive (sense and respond), dynamic and interactive in response to a customer, or to changing circumstances.
- *resourceful*: thoughtful or exhibiting some discipline. This, however, is not the same as a traditional "command and control" approach with defined, formal procedures.

This definition will be useful, especially in later discussion, where we discuss controversies between process-based and agile approaches. This has special implications for the role of discipline in agility.

3 WHAT WE HAVE LEARNED SO FAR: THE CURRENT STATE

In this section, I will briefly summarize key research findings from a multi-year study (2000–2003) on Quality Software Development @ Internet Speed. Detailed findings are available elsewhere. This is not a survey; rather, this is intended to serve as

a catalyst for discussing a future state and the challenges ahead. Passing references are made to related research on agility and fast-paced development to a limited extent.

Our three-part study on Internet-speed software development used a mixed-methods research design involving the collection of multiple kinds of data (Tashakkori and Teddlie 1998). Case studies of Internet-speed software development in Phase 1 were complemented with a Discovery Colloquium held in Phase 2. Phase 3 continued the original case studies.

3.1 Phase 1: Case Studies of Internet Software Development

During the first phase, in Fall 2000, we conducted detailed case studies of Internet software development at 10 companies in two major metropolitan areas. The firms ranged in size from 10 employees to more than 300,000 employees, in different industries in the private and public sectors including financial services, insurance, business and consulting services, courier services, travel, media, utilities, and government services. Some of the firms were new Internet application start-up companies while others were brick-and-mortar companies with new Internet application development units.

Our objective was to understand how and why Internet-speed software development differs from traditional software development. We collected data through open-ended interviews and analyzed it using grounded theory (Strauss and Corbin 1990). With this methodology, we were able to develop a theory for a problem under investigation without prior hypotheses. The analysis identified core categories and their inter-relationships, explaining how and why Internet-speed software development differs from traditional approaches. In essence, we uncovered three major causal factors.

- A desperate rush-to-market
- A new and unique software market environment
- A lack of experience developing software under the conditions this environment imposed

As a result, a new development process that depends on new software development cultures evolved. In this process, software product quality becomes negotiable. Eight identifiable practices (see Figure 1) characterizing the Internet-speed software development process emerged from Phase 1 (Baskerville et al. 2001).

3.2 Phase 2: Discovery Colloquium

Our Phase 2 objectives were to synthesize knowledge on best practices for quality and agility in Internet-speed software development. We held a one-day Discovery Colloquium on Innovative Practices for Speed and Agility in Internet Software Development using innovative open-forum search techniques to enable what has been called *creative abrasion* (Leonard 1999).

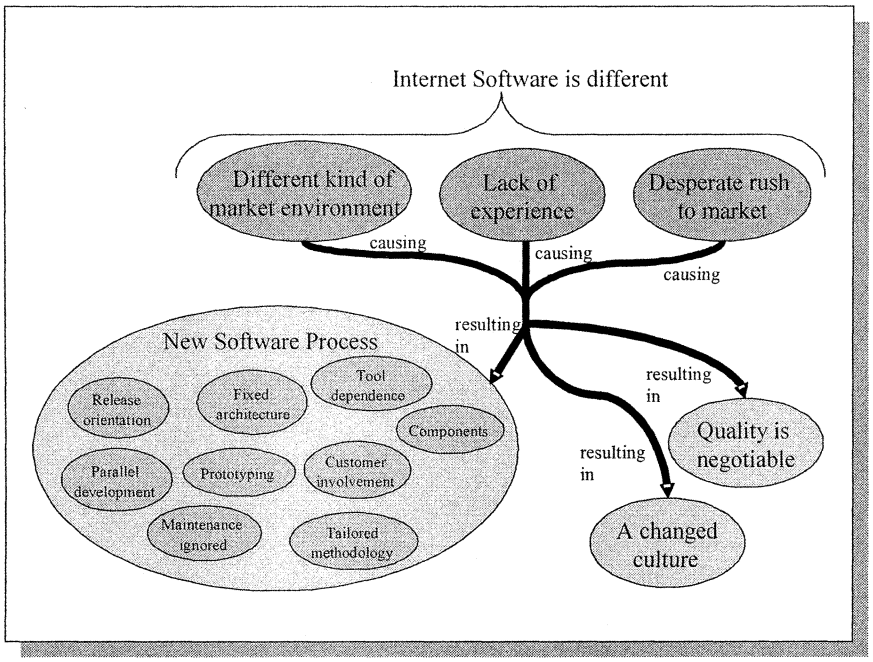

Figure 1. Results from Phase 1
(Figure 1 from B. Ramesh, J. Pries-Heje, and R. Baskerville, "Internet Software Engineering: A Different Class of Processes," *Annals of Software Engineering* (14), December 2002; © Kluwer Academic Publishers; with kind permission of Springer Science and Business Media.)

The colloquium benefited from the Phase 1 findings and included participants from Phase 1 companies as well as selected experts. Software practitioners from entrepreneurial small companies and large brick-and-mortar companies, Internet business strategists, and leading software development experts also participated.

Participants joined one of several breakout groups dedicated to exploring a core issue. The groups first identified observations relating to their core issue, and then developed hypotheses about possible associated factors. The groups tested the hypotheses, identifying linkages, contradictions, and interdependencies among them. They identified principles, promising practices, and other dynamics (Levine et al. 2002). Although the findings from the colloquium distinguished Internet speed as a set of practices, it denoted the underlying principles as principles of agility.

Subsequently, also as part of Phase 2, we set about to compare and analyze differences between seemingly *traditional* and *agile* principles and related practices. We adopted a set of principles, rigorously developed in a workshop on software development standards held in Montreal using a multistage Delphi study involving well-respected researchers and practitioners (Bourque et al. 2002). It exemplifies the best attempt to date to define general metaprinciples for traditional software development.

Based upon our analysis (Baskerville et al. 2003) we concluded that many Internet-speed development practices look deceptively similar to long-standing software development practices. However, a close examination of how Internet-speed development practices unfold, and the agile principles to which these practices respond, reveals that Internet-speed software development is a fundamentally new way to develop software. Each Internet-speed development practice can also be found in traditional software development. What distinguishes the practices is how Internet-speed developers combine and apply them—sometimes to extreme.

Our results yielded at least four implications for software management:

- Cost and quality do not drive Internet-speed software development. Rather, development speed is paramount. Quality becomes negotiable, a moving target in play with functionality and product availability.
- Project management in Internet-speed development differs from project management in traditional development. Projects do not begin or end, but are ongoing operations more akin to operations management. Development problems are chunked into small jobs that can be rolled out as small, tailor-made products.
- Maintenance in Internet-speed development is sometimes merged into the specification–build–release cycle along with new functionality, or maintenance cycles become small project cycles interspersed with larger project cycles.
- Human resource management differs in Internet-speed development. Team members are less interchangeable, and teams require people with initiative, creativity, and courage as well as technical knowledge, experience, and drive.

3.3 Phase 3: Case Study Continues

In 2002, we returned to study our original 10 companies which were developing application software for the Internet. At the time of the interviews, only five of the original nine companies remained in business or were available to participate in the study. Only one of the small Internet software houses had survived. To maintain the representative nature of the selection of companies, we added an additional company—a small innovative Internet software house. In all, six companies participated in Phase 3.

In 2002 (as in 2000), we used semi-structured interviews as a forum for collecting data, following the same study guide. Again, the data were analyzed using grounded theory techniques to develop a central story line or core category.

We traced trends and changes and observed new circumstances. A comparison of the 2000 and 2002 data shows how major factors, such as market environment and lack of experience, emerged to change the software process and the attitude toward quality. The interrelationship between the core factors of speed and quality, together with the other major factors, unfolded in a decision process wrought with trade-offs and balancing decisions at multiple levels in the software organization. These trade-offs and balancing decisions—a high-speed balancing game—were taking place at three different levels: the market, the portfolio, and the project.

Two major changes had taken place from 2000 to 2002. First, quality was no longer being treated as a disadvantaged stepchild. Speed and quality must be balanced for

companies to survive in the newer market. Second, related monetary factors have been reversed: the unending supply of money characteristic of the boom has dried up; and good people are no longer scarce resources.

At the market level, during the two time periods, we saw values shift from a fever-pitched struggle for first-mover advantage to a slower, less intense consolidation of best practices. Notably, the changing market and IT economy slowed the interest in IT products, while at the same time easing the intense competition for human resources necessary for wide-scale software development. In 2002, with the market focused on a narrowed scope of Internet applications, competition remained intense but concentrated.

At the portfolio level, from 2000 to 2002, we detected a shift from a resource-rich, build-everything blast to a resource-constrained, tightly managed, and well-organized stable of ideal jobs. As a result of the changing market, the companies began to make major adjustments to their project portfolios. The *business case* became the primary vehicle for apportioning resources and selecting projects for inclusion or continuation within the portfolio. With falling resources, managers began to "cherry pick" the most ideal projects to meet their customers' needs.

At the project level, from 2000 to 2002, project values moved from speed-at-all-costs to an economized scope. Denied the resources to build products without a clear economic justification, project managers began to consolidate the product development to embrace construction of fewer products. The major Internet speed development values persisted, such as parallel development, limited maintenance and documentation, frequent releases, etc. These factors are still necessary to maintain customer satisfaction and compete in the (more focused) marketplace. The factors are also noted for enabling quick, economical products.

The study suggests that the nature of the balancing game has evolved with the shifting of the market and organizational environments over recent years. The peak of the dot-com boom was characterized by few constraints on financial resources, but severe constraints on availability of qualified personnel and very tight deadlines. At this peak, the balancing game was focused more toward achieving speed, often at increased project costs and lower levels of quality. This situation later evolved into market conditions that expect higher levels of product quality and lower costs while still demanding product development agility. As a result of market changes, the balancing games at the organizational and portfolio levels have grown in importance compared to the dominance achieved by the project balancing game in 2000.

4 WHERE ARE WE GOING: THE FUTURE STATE

Use of agile methods and agility is consistently associated with software development techniques. But more recently, we have seen fledgling signs of expansion. Ironically, the contracting of the market and the tightening of resources has contributed to an enlarged scope and increased complexity in enacting the balancing games at the portfolio and organization levels. This may spur further growth for agile approaches in atypical areas.

That said, the current state for agile methods is still isolated and limited. We have a partial understanding of what agility means for software development activities. For example, we know that agile methods work well with small teams (especially those that

are colocated), where requirements are emergent, and in a turbulent environment of constant change. Agile methods are not recommended in the development of life critical systems; and its use in developing embedded software remains unclear (Ambler 2004). We have little understanding of the consequences of agile approaches for technology adoption and implementation activities. Within the development and adoption arenas, we have yet to fully grapple with the implications of agility for people, process, and new technology.

Our best insights into agility are still achieved through discrete activities—through projects which exist like islands in our organizations. From the development perspective, we have information on different agile methods, where they apply, particular emphases, and some acknowledged limitations. From an adoption perspective, we can speculate that an agile approach would favor pilots, trials, and demonstration projects; and from a knowledge transfer perspective, an agile approach would favor high customer involvement through face-to-face interaction or "body contact."

The challenge for the future is two-part. First, we must optimize the current state with vertical coupling to loosely integrate and propagate agile approaches for development, deployment, and knowledge transfer. This lightweight alignment would allow us to leverage what we know, and to reinforce these otherwise discrete areas of success. Second, and more radically, we must tackle the issue of scaling to investigate options for agile approaches and opportunities that can span organizations. On its face, this might seem contradictory since use of agile methods favors small teams with high contact. But to realize the potential for agile, we must ask how such methods adapt and scale. Perhaps they will do so in entirely new ways.

Austin and Devin (2003) speculate that old production models for software development are no longer useful. Rather, agile software development has the potential to be artful making. They write:

> Artful making (which includes agile software development, theater rehearsal, some business strategy creation, and much of other knowledge work) is a process for creating form out of disorganized materials. Collaborating artists, using the human brain as their principal technology and ideas as their principal material, work with a very low cost of iteration. They try something and then try it again a different way, constantly reconceiving ambiguous circumstances and variable materials into coherent and valuable outputs (pp. xxv-xxvi).

Whereas industrial making places a premium on detailed planning, closely specified objectives, processes, and products, artful making is different, fusing iteration and experimentation.

Austin and Devin point out that, "if you think and talk about iteration as experimentation, low cost of iteration seems to make business more like science. Its broader effect, though, is to make business more like art" (p. xxv). The authors go on to build an artful framework employing the analogies of theatrical production, extending beyond surface collaboration to the on-cue innovation that theater companies routinely achieve. In a similar vein, Stefan Thomke (2003) investigates experimentation in innovation, as it "encompasses success and failure; it is an iterative process of understanding what doesn't work and what does" (p. 2). He reminds us that both results are equally important for learning.

Finally, on a related topic, Dee Hock (1999) has characterized the organization of the 21st century organization as a *chaord*. The term chaord was formed out of combining the first three letters of the word chaos, with the first three letters from the word order. Hock and other leading scientists believe that the primary science of the next century will be the study of complex, self-organizing, nonlinear, adaptive systems, often referred to as complexity theory or chaos theory (De Geus 1997; Wheatley 2001). They assert that living systems arise and thrive on the edge of chaos with just enough order to give them pattern, but not so much to slow their adaptation and learning. This is not unlike the challenge for agility. We ask: Does this represent the larger paradigm shift of which agile methods are a part?

5 CHALLENGES, DILEMMAS, AND CONUNDRUMS

Achieving the future state is a challenge in itself—enhancing, adapting, applying, and scaling agile approaches is no easy feat. In addition, several dilemmas or conundrums have become evident. I will single out three to discuss briefly relating to process, discipline, and oversight.

The first controversy surrounds the role of process in agile methods. Typical views pit agility against process, and agile methods against process-intensive or *monumental* models like the software Capability Maturity Model (SW-CMM®) (Highsmith 2000). Paulk (2001) takes a closer look at how such approaches are not entirely at odds and illustrates how a development group following extreme programming might simultaneously embrace CMM, at least up until level 3. At level 3, the approaches diverge. Boehm (2002) and Boehm and Turner (2003) argue that agile and plan-driven methods each have a "home ground." They emphasize balance and attempt to make a case for hybrid strategies. Nevertheless, this split between process and agility has become a lightning rod, reinforcing entrenched positions and a strict drawing of lines.

For example, Steven Rakitin (2001) offers the following pointed and skeptical view in response to the values of agile developers. He argues that the values on the right (below) are essential, while those on the left serve as easy excuses for hackers to keep on irresponsibly, throwing code together with no regard for engineering discipline. He provides "hacker interpretations" that turn agile value statements such as "responding to change over following a plan" into chaos generators. Rakitin's hacker interpretation of "responding to change over following a plan" is roughly "Great! Now I have a reason to avoid planning and to just code up whatever comes next." He offers the following translations:

- **Individuals and interactions over processes and tools**
  Translation: Talking to people gives us the flexibility to do whatever we want in whatever way we want to do it. Of course, it's understood that we know what you want—even if you don't.

- **Working software over comprehensive documentation**
  Translation: We want to spend all our time coding. Real programmers don't write documentation.

- **Customer collaboration over contract negotiation**
  Translation: Let's not spend time haggling over the details, it only interferes with our ability to spend all our time coding. We'll work out the kinks once we deliver something.

- **Responding to change over following a plan**
  Translation: Following a plan implies we would have to spend time thinking about the problem and how we might actually solve it. Why would we want to do that when we could be coding?

While we might find reassuring appeal in the "sensible middle ground" (DeMarco and Boehm 2002), I suggest we closely examine our assumptions and first principles for agility and stability—to ensure that we do not fall into an easy trap of compromise. DeMarco and Boehm reassure us that "the leaders in both the agile and plan-driven camps occupy various places in the responsible middle. It's only the overenthusiastic followers who over interpret discipline and agility to unhealthy degrees" (p. 90). Alas, the challenges of agility and agile methods are only beginning to emerge; and innovation rarely comes from the responsible middle.

A second controversy is related to process issues and it concerns the role of discipline. Agile proponents tend to see CMM as engendering bureaucratic, prescriptive processes, fostering a command and control environment. Process and discipline are viewed as of whole cloth in this reductive manner. Unfortunately, more subtle definitions of discipline (for example, self organizing, nonlinear, or adaptive modes) have not yet been brought to bear in this argument.

Where does discipline fit in the context of agility? Under the auspices of agility, there must be some structure, order, and organization. We know that, in actuality, it takes time to speed up, unless you are simply cutting things out (Smith and Reinertsen 1998). By extension, it takes discipline to be agile. What kind of discipline, albeit adaptive and self organizing, is at play in the agile environment? Here, new approaches to experimentation (Thomke 2003) and frameworks such as artful making (Austin and Devin 2003)—where the emphasis is on a method of control that accepts wide variation within known parameters—will help us arrive at new understanding. If we are to embrace agile methods and move forward, we must begin such inquiry.

The final dilemma concerns the matter of governance. At present, we are faced with two conflicting models—one for development which can be agile, but no equivalent for project management, for oversight and monitoring. As I have indicated already, acquisition program managers have expressed interest in their development teams using agile methods. However, they are entirely at a loss to identify appropriate mechanisms that could be employed for monitoring and oversight of systems development. It is naïve to assume that oversight is antithetical to agile approaches, and thus once again we are challenged to reach beyond comfortable and convenient walls to explore new territory.

6 CONCLUSION

For agile approaches to be fully understood—to mature and to gain ground—we must consider what agility means as part of a larger landscape, and what type of shift

it marks in technology development and in organizational behavior and change. What are the drivers for the burgeoning interest in agile methods? What have we discovered?

In this paper, I step back to consider these questions, as informed by my involvement in a multi-year research project on Quality Software Development @ Internet Speed and ongoing research on diffusion theory.

I begin with a brief look at definitions of agility, and conclude that agility is more than speed, extending beyond to encompass nimbleness, adaptability, and resourcefulness. Then I discuss the current state of agility and Internet-speed software development, using case study findings from 2000 and 2002.

Our case study suggests that a balancing game has evolved with the shifting of the market and organizational environments over recent years. In 2000, the peak of the dot-com boom was characterized by free flow of financial resources, severe constraints on availability of qualified personnel, and very tight deadlines. Project activities formed the focus for the balancing game and speed was to be achieved almost at all costs. A new development process that depended on new software development cultures emerged. We were also able to identify eight distinct practices characterizing an Internet-speed software development process.

In 2002, we detected a shift from this build-everything gold rush to a resource-constrained, carefully managed stable of jobs. As a result of the changing market, the companies were making major adjustments to their project portfolios. The business case became the primary vehicle for selecting projects for inclusion or continuation within the portfolio. Managers were cherry picking the best projects to meet their customers' needs. Denied the resources to build products without a clear economic justification, project managers were consolidating product development to embrace construction of fewer products. The major Internet speed development values persisted, such as parallel development, limited maintenance and documentation, frequent releases, etc. These factors were, and still remain, necessary to maintain customer satisfaction and compete in the more focused marketplace.

The future holds key challenges for a next generation of agile approaches. Of particular note are the need to (1) loosely integrate and propagate agile approaches for development, deployment, and knowledge transfer, and (2) tackle the issue of scaling to investigate options for agile approaches and opportunities that can span organizations.

Finally, I conclude with a short discussion of conundrums and dilemmas. The first of these controversies surrounds the role of process in agile methods. Typical views pit agility against process, and agile methods against process-intensive or monumental models. These are views that we must get past, at the same time as we resist the trap of too-easy compromise.

The second controversy also relates to process issues and concerns the role of discipline. Agile proponents tend to see CMM as engendering bureaucratic, prescriptive processes, fostering a command and control environment. Process and discipline are viewed as cut from whole cloth in this limited manner. Unfortunately, more subtle definitions of discipline (for example, self organizing, nonlinear, or adaptive modes) have not yet been brought to bear in this dialogue. Where does discipline fit in the context of agility? If we are to embrace agile methods and move forward, we must begin this inquiry.

The third and final controversy relates to governance. We must investigate appropriate and meaningful mechanisms that can be employed for monitoring and

oversight of projects using agile methods. If we do not do so, agile methods will never come of age in large programs.

Agility in software development has implications for organizational agility—and the shift to agile methods and models signals a larger transformation in the workplace and the organization of the 21st century. As we have noted, this transition state is turbulent, marked by continuous change. No clear or easy solutions have resulted. The transformation is a work in progress, one that is by no means complete. To be realized, it invites investigation across a range of disciplines and initiatives, including organizational development, diffusion of innovations, process improvement, knowledge management, complex adaptive systems, chaos theory, systems thinking, software engineering, and information systems.

REFERENCES

Agile Manifesto. "Manifesto for Agile Software Development," 2001 (available online at http://agilemanifesto.org/).

Ambler, S. W. *The Object Primer* (3rd ed.), Cambridge, England: Cambridge University Press, 2004.

Austin, R., and Devin, L. *Artful Making: What Managers Need to Know about How Artists Work*, Upper Saddle River, NJ: Pearson Education Inc. Publishing as Financial Times Prentice Hall, 2003.

Baskerville, R., Levine, L., Pries-Heje, J., Ramesh, B., and Slaughter, S. "How Internet Software Companies Negotiate Quality," *Computer* (34:5), May 2001, pp. 51-57.

Baskerville, R., Levine, L., Pries-Heje, J., Ramesh, B., and Slaughter, S. "Is Internet-Speed Software Development Different?," *IEEE Software* (20:6), November-December 2003, pp. 70-77.

Boehm, B. "Get Ready for Agile Methods, with Care," *IEEE Computer* (35:1), January 2002, pp. 64-69.

Boehm, B., and Turner, R. *Balancing Agility and Discipline: A Guide for the Perplexed.* Reading, MA, Addison-Wesley, 2003.

Bourque, P., Dupuis, R., Abran, A., Moore, J. W., Tripp, L., and Wolff, S. "Fundamental Principles of Software Engineering: A Journey," *Journal of Systems and Software* (62:1), May 2002, pp. 59-70.

De Geus, A. *The Living Company*, Boston, MA: Harvard Business School Press, 1997

DeMarco, T., and Boehm, B. "The Agile Methods Fray," *Computer* (35:6), June 2002, pp. 90-93. .

Dove, R. *Response Ability—The Language, Structure, and Culture of the Agile Enterprise,* New York: Wiley, 2001.

Dove, R. *Value Propositioning—Perception and Misperception in Decision Making,* Tuscon, AZ: Inceni Books, 2005.

Grover, V., and Malhotra, M. "A Framework for Examining the Interface Between Operations and Information Systems: Implications for Research in the New Millennium," *Decision Sciences* (30:4), Fall 1999, pp. 901-920.

Highsmith, J. *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems,* New York: Dorset House Publishing, 2000.

Hock, D. *Birth of the Chaordic Age*, San Francisco: Berrett-Koehler Publishers, 1999.

Kathuria, R., Anandarajan, M., and Igbaria, M. "Linking IT Applications with Manufacturing Strategy," *Decision Sciences* (30:4), 1999, pp. 959-991.

Leonard, D. *When Sparks Fly: Igniting Creativity in Groups*, Boston: Harvard Business School Press, 1999.

Levine, L., Baskerville, R., Loveland Link, J. L., Pries-Heje, J., Ramesh, B., and Slaughter, S. "Discovery Colloquium: Quality Software Development @ Internet Speed," SEI Technical Report CMU/SEI-2002-TR-020, Pittsburgh, PA: Software Engineering Institute, 2002.

Merriam-Webster Incorporated. *Merriam-Webster Online Dictionary*, 2004 (available at http://www.m-w.com/).

Paulk, M. C. "Extreme Programming from a CMM Perspective," *IEEE Software* (18:6), November-December 2001, pp. 19-26.

Paulk, M. C., Weber, C. V., Curtis, B., and and Crissis, M. B. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Reading, MA: Addison-Wesley, 2001.

Rakitin, S. "Manifesto Elicits Cynicism," Computer (4:12), December 2001, p. 4.

Ramesh, B., Pries-Heje, J., and Baskerville, R. "Internet Software Engineering: A Different Class of Processes," *Annals of Software Engineering* (14), December 2002, pp. 169-195.

Smith, P. G., and Reinertsen, D. *Developing Products in Half the Time*, New York: John Wiley & Sons, 1998.

Strauss, A., and Corbin, J. *Basics of Qualitative Research: Grounded Theory Procedures and Techniques*, Newbury Park, CA: Sage Publications, 1990.

Tashakkori, A., and Teddlie, C. *Mixed Methodology: Combining Qualitative and Quantitative Approaches,* Thousand Oaks, CA: Sage Publications, 1998.

Thomke, S. *Experimentation Matters: Unlocking the Potential on New Technologies for Innovation,* Boston: Harvard Business School Press, 2003.

Wheatley, M. J. *Leadership and the New Science: Discovering Order in a Chaotic World* (revised ed.), San Francisco: Berrett-Koehler Publishers, 2001.

ABOUT THE AUTHOR

Linda Levine is a senior member of the technical staff at Carnegie Mellon University's Software Engineering Institute. Her research focuses on acquisition of software intensive systems, agile software development, systems interoperability, diffusion of innovations, and knowledge integration and transfer. She holds a Ph.D. from Carnegie Mellon University. She is a member of IEEE Computer Society, Association for Information Systems, National Communication Association, and cofounder and vice chair of IFIP Working Group 8.6 on Diffusion, Transfer, and Implementation of Information Technology. Contact Linda at ll@sei.cmul.edu.