CUTTER CONSORTIUM

TIME IS OF THE ESSENCE

WHITE PAPER

BY JIM HIGHSMITH
Director, Agile Project Management Practice

CUTTER CONSORTIUM, 37 BROADWAY, SUITE 1, ARLINGTON, MA 02474-5552, USA;
Tel: +1 781 648 8700; Fax: +1 781 648 8707;
E-mail: sales@cutter.com; Web site: www.cutter.com/

CUTTER CONSORTIUM

CUTTER CONSORTIUM

ABOUT CUTTER CONSORTIUM

Cutter Consortium's mission is to help organizations leverage IT for competitive advantage and business success. Through our products (subscription-based advisories and journals, reports, and training tools) and services (consulting, training, and mentoring), we assist enterprises in creating and implementing IT strategies and maintaining best-in-class IT organizations that support their business needs.

At Cutter, we understand that an organization's IT strategy must align with its business objectives. Whether it's seizing e-business opportunities, aligning business and IT goals, profiting from business intelligence technologies, or refining relationship management skills to improve relationships with outsourcing partners, Cutter looks at the business side of the task you face. And we help you create the solution that fits your organization's culture and requirements.

To accomplish our mission, we have assembled the world's preeminent IT consultants — a distinguished group of internationally recognized experts committed to delivering top-level, critical, objective advice. Each Consortium practice area features a team of Senior Consultants whose credentials are unmatched by any other service provider. This group of experts provides all the consulting, performs all the research and writing, and fields all the inquiries from Cutter clients.

This is what differentiates Cutter from other analyst and consulting firms and why we say Cutter gives you *access to the experts*. All of Cutter's products and services are provided by the top thinkers in business and IT today. Cutter's clients tap into this brain trust and are the beneficiaries of the dialogue that takes place — at the annual Cutter *Summit*, in the pages of the ***Cutter IT Journal***, in the collaborative forecasting the Cutter Business Technology Council provides, and in our many reports and advisories.

The Consortium takes a unique view of the business-technology landscape, looking beyond the one-dimensional "technology" fix approach so common today. We know there are no "silver bullets" in IT and that successful implementation and deployment of a technology is as crucial as the selection of that technology.

Cutter Consortium's menu of products and services can be customized to fit your organization's budget. Most importantly, Cutter offers objectivity. Unlike so many information providers, the Consortium has no special ties to vendors and can therefore be completely forthright and critical. That's why enterprises rely on Cutter for the no-holds-barred advice they need to gain, and to maintain, a competitive edge.

Time Is of the Essence is published by Cutter Consortium, 37 Broadway, Suite 1, Arlington, MA 02474-5552 USA; Tel: +1 781 648 8700; Fax: +1 781 648 8707; E-mail: sales@cutter.com; Web site: www.cutter.com/.

FOR MORE INFORMATION

Cutter Consortium's Agile Project Management practice promotes the use of flexible, responsive, and adaptive practices for successful project management and software development. Directed by Jim Highsmith, a leader in the Agile Methodology movement, the practice provides strategic advice that will help you implement agile processes and methodologies, resulting in higher-quality software, greater flexibility, faster return on investment, and greater predictability for your software projects.

Cutter's Agile Project Management practice offers products and services that will help your organization implement Agile software development. Whether it's custom consulting, inhouse training, public workshops, or the subscription-based Agile Project Management Advisory Service, the practice provides expert advice and guidance directly from the founders of today's hottest Agile Methods — Kent Beck on Extreme Programming, Jim Highsmith on Adaptive Software Development, Alistair Cockburn on Crystal Light Methods, Rob Thomsett on Extreme Project Management, and more.

For more information on the Agile Project Management practice, please visit www.cutter.com/project/practice.html, or contact David Gijsbers by phone at +1 781 641 5104, or by e-mail at david@cutter.com.

Time Is of the Essence

One-third the time, one-third the budget, one-third the defect rate: these are the goals of lean development (LD), a software development management approach originated by Dr. Bob Charette. Based on the lean manufacturing work of James Womack and others, lean development is as much a management challenge as a set of practices. Charette, best know for several books and articles on risk management and risk entrepreneurialism, comments, "You have to set the bar high enough to force rethinking traditional practices."

Lean development is one of the initiatives focused on accelerating the speed of delivering software applications, but not at the expense of higher cost or defect rates. As Charette says, "These three objectives need to be achieved concurrently or it isn't LD." But what does increasing speed really mean? Rapid application development (RAD) techniques sprang into the lexicon within the past 10 years. The phrase "Internet time" was poised to redefine software development speed until the market informed both Netscape and Microsoft that a new browser every three to four months caused too much pain.

There are two key, interwoven questions: How fast do we need to go? And how do we go that fast? Many application development groups, having no concrete answer to the first, have no solid basis on which to pursue the second. Assaulted by waves of change, and the constant mantra to increase speed, we fail to either ask or answer these two basic questions. Speed is expensive. However, the extra costs may not be financial, but organizational. The cost may be in changing our thinking about managing software delivery.

The other set of practices covered in this article is complementary to lean development. It also has roots in manufacturing. Preston Smith and Don Reinertsen have worked with companies from a wide number of industries to help them cut product development time in half. Their principles and practices are management oriented. Since Smith and Reinertsen don't have a tag line like "lean development," I will use an acronym based on their book title, *Developing Products in Half the Time*, (*DPHT*) when quoting or referencing information from their book (see sidebar, page 3).

As a good friend once said, "There are only a few ways to get things right, but an infinite variety of ways to get things wrong." It is not by accident that these two approaches have much in common.

LD and *DPHT* provide some new ideas about how to create application development strategies geared to today's marketplace. But they are more than just ideas.

Accelerating Development

There are a few general ways to increase product development

speed, such as:

- Eliminate inactivity
- Eliminate activity
- Organize activities
- Organize staff
- Improve effort productivity

The mantra in many organizations is to increase productivity — for example, to move from 10 function points (FPs) to 12 FPs per staff-month by training and skill building. Improving effort productivity is hard, costly work. It is more efficient to eliminate an activity than to improve its productivity. Even better, speed delivery by eliminating nonactivity.

Many of the practices encountered in *DPHT* are, not surprisingly, similar to good software engineering management practices. The following discussion, therefore, concentrates on ideas that may be new to software development managers or that reinforce better-known software practices. At the end is a discussion with author Preston Smith.

Eliminate Inactivity

Brilliant. Smith and Reinertsen espouse such simple, yet such profound ideas. Their research shows that for many companies, the "fuzzy front end" of a project can consume as much as half the total elapsed time from identification of need to delivery. A measurement problem in many organizations is that they measure "schedule" from the inception of the actual project development work to delivery. Some projects languish in the dreaded backlog for years, then, once launched, become rushed. The time spent waiting is ignored — at least most development organizations ignore it; customers, however, know exactly how long the project has bobbed around in the priority queue waiting for its starting time.

What about all the time spent waiting? Think about the cost of cutting a month off the schedule of a 50-person project toward the project's end. Compare that with the cost of cutting a month off time not working on the project at the beginning. "It is an extremely cheap place to shop for cycle time" (*DPHT*). As Smith and Reinertsen point out, most project portfolio analysis methods concentrate on careful selection of the right project on some valuation (return on investment, payback) method. They don't focus on the speed of the selection process. Compounding the problem is the budgeting cycle. Between portfolio analysis and budgeting delays, time-critical projects can easily be held up for six months to a year in larger organizations.

Large IT projects are processed similarly to capital expenditures. They grind through procedures, usually tightly controlled by the accountants. While this works reliably for many projects, it puts an onerous burden on those that are truly time critical. The key to action is drawing attention to the cost of front-end delay: "You must think clearly about it ... The calculated cost of delay is often 500 to 5,000 times higher than the visible costs of assigned personnel" (*DPHT*). One-quarter of a person working on the capital budget, or even a feasibility study, for three months might cost $6,000 or $7,000. Delaying a revenue-generating project for the same three months could result in the loss of $1 million in revenue. It is unfair to saddle the project team with constant overtime and intense psychological pressure when the organization whiled away so much time up front.

As shown in Figure 1, improving the front-end process speeds up projects and costs very little, a seemingly unbeatable combination. But it is often difficult politically. It is hard to sell exceptions to standard business processes, and it is politically hard to anoint some projects with special status. It is much easier to just admonish everyone to work harder and faster.

As implied earlier in this section, reducing the fuzzy front end is first dependent on recognizing that it exists. What is the schedule productivity of a four-month RAD project that remained in the backlog for eight months before it was started"? For a 500 FP project, the schedule productivity with no delay would be 125 FP/team-month (500 FPs per four months' team effort). With the eight-month front-end delay, it would be 42 FP/TM. Maybe it

seems unfair to tag projects with wait time, but in the high speed of today's markets, "concept-to-cash" may mean the difference between success and failure of a business initiative. Calculating the actual cost of delay is a key part of measurement.

You should also screen projects early. Most projects will proceed using the regular business-approval process. A "shunt valve" is needed to divert time-critical projects to a special "fast-track" process. This process should be optimized and staffed for speed. It should have enough capacity (staff) and management attention that projects continue rapidly through the process or are kicked out. In, out — fast. There also needs to be a restriction on the process, e.g., no more than 15% of all projects can be considered for fast tracking. This sends a message that if a project is really time critical, there is a mechanism for speedy attention; however, it needs strong justification. If too many projects get shunted onto the fast track, it will either slow down to the speed of the normal process, require inordinate resource commitments, or both.

Hidden in the last paragraph is a paradox. If projects are really time critical, then speed up their fuzzy front end. However, to select those projects, a strong justification is required, which indicates a justification process that then itself slows progress. A justification process to select projects to bypass another justification process? The secret is to modify traditional accounting measures and establish criteria based on the need for speed.

Projects usually go through three generic phases prior to traditional "startup." There is an initial concept development where the project requestor prepares an initial proposal, followed by a feasibility study, and finally detailed project planning. In many instances, detailed planning is followed by staffing and, finally, a project "kick-off." There is normally significant inactivity that can be weeded out of the process. Some of the activities can be overlapped. Approval time between proposal and feasibility can be reduced. In particular, Smith and Reinertsen recommend assigning the core team in time for them to participate in the detail planning phase. Not only does it speed planning, but team members accelerate their project learning curve.

One major concern about fast tracking is usually the perception of increased risk. To address this issue, risk needs to be viewed from two perspectives. First, technical risk is the risk that the project will not meet its mission profile — schedule, resource costs, scope (functionality and performance), and defect levels. It is the risk that the software runs too slowly or the features had to be cut too far to make the schedule deadline.

Market risk is the opposite of the "field of dreams." Market risk is "build it and they don't come." Market risk is having the wrong product or the wrong timing. Startup companies survive or perish on very thin time margins. Their burn rate on capital is so high that slight miscalculations mean the difference between being millionaires or looking for another job.

If a project is eligible for the fast track, the market risk probably completely outshadows the technical risk. And in most situations,

Developing Products in Half the Time

Great ideas have a wide range of applicability. In 1993 when I was looking for ideas to complement RAD development, I stumbled across Preston Smith and Don Reinertsen's book, *Developing Products in Half the Time* (*DPHT*). I incorporated some of their ideas into a RAD approach I was developing with a colleague, Sam Bayer. *DPHT* has gone on to be a very successful book (more than 60,000 copies sold), and a second edition has just been published. Titles and names such as Ph.D. in engineering from Stanford, McKinsey & Company, MBA from Harvard Business School, Bell Labs, GM, and more dot the authors' resumes. They have done it, consulted on it, and written about it: "it" being reducing product development time.

In the introduction to *DPHT*, Neil Hagglund, vice president and director of Corporate Technology Planning for Motorola, Inc. offered the following: "At Motorola we achieve rapid development the same way we achieved breakthroughs in quality — with old-fashioned hard work and constant management attention … many readers may wonder if pursuing development speed requires a company to compromise quality. At Motorola we have firmly rejected this option. We often find that faster development actually provides higher quality to our customers."

"At Motorola, we found the original edition [of *DPHT*] to be far and away the most useful book of its kind."

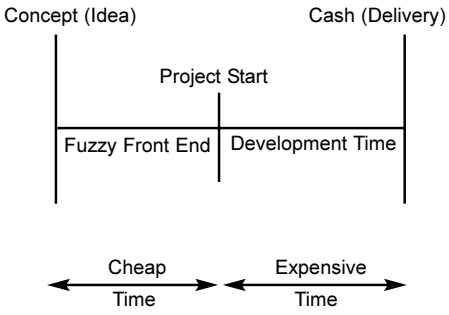

Figure 1 — Concept to Cash

market risk is dominated by schedule risk. In these situations, a slight increase in technical risk is more than justified.

Another area to reduce inactivity is the management decisionmaking process. There is a difference between delaying decisions because more information is required and delaying them because of a poorly constructed decisionmaking process.

There are also sources of inactivity to be eliminated during a project. Joint application development (JAD) sessions help reduce the inactivity of waiting around for users to have time for interview sessions. Almost all RAD approaches recommend JAD sessions as a tool for time compression.

Eliminate Activities

Eliminating activities is a process of making hard tradeoff decisions. Tradeoffs require good information. "Many companies believe that fast development is good, and slow product development is evil. This is a dangerously sloppy way of thinking about rapid product development" (*DPHT*). The object is not speed; the object is to make money! Smith and Reinertsen recommend a careful analysis of four key objectives and the tradeoffs between them: market introduction date, product performance, development project expense, and product unit cost (not as important in software development).

Making tradeoffs is not easy; in fact, it is easier to postpone these decisions. One of the benefits of timeboxing development — defined, periodic development cycles — is that it helps reduce marginal application features, whereas long development cycles create excess room for marginal features to flourish.

Making tradeoffs means understanding business strategy. For example, Wal-Mart's competitive strategy is low price; Nordstrom's is service and selection. Each of these retailers defines its market niche explicitly. While Wal-Mart excels at low price, it has to maintain "good-enough" service. Poor service loses customers. Great service costs too much. Good-enough is just right. Similarly, Nordstrom must maintain "good-enough" prices. Excessive prices lose customers. Low pricing reduces the revenue needed to support great service. Each company understands it can only excel in one dimension. Muddling the focus leads to mediocre performance in all dimensions.

Software projects are no different. A project focused on delivery speed may have to accept "good-enough" defect levels or "good-enough" costs. Trying to achieve all three adds significant activity, which in the long run makes achieving excellence in any one area impossible. Client, project, and IT managers who don't understand tradeoffs, who don't understand the difference between perfection, excellence, and good enough, will have a difficult time eliminating activities and, therefore, accelerating schedules.

Organize Activities

Developing manufactured products and software products is alike in many ways. For example, the ideas behind iterative software and what Smith and Reinertsen call incremental innovation are very similar. In *DPHT*, they state, "We will argue that the incremental innovators are in fact the unsung heroes of product development." The authors discuss the advantages and disadvantages of incremental innovation from four perspectives: financial, marketing, engineering, and rapid learning. The pluses and minuses of these perspectives are detailed below:

- Incremental innovation increases revenue and profits much faster, and cash flow is better; however, some fixed costs are duplicated for each iteration.
- Short iterations combat very uncertain planning horizons, and

successes and mistakes show up quickly; however, frequent delivery can clog distribution channels (how fast can customers implement applications?).

- Short iterations force early integration, motivate team members, get products into the field early, and spread technical commitments; however, they make it difficult to adopt a whole new technology, and they are atypical of the way many engineers prefer to work.
- Incremental innovation accelerates the learning process (there isn't a "however" for this one!).

It is interesting to note that Smith and Reinertsen advocate a lifecycle approach very similar to the phase and gate, or adaptive lifecycle. They discuss moving beyond the "phases" mentality and advocate a looser "stage gate" in contrast to the more traditional "toll gate." "Toll gates send an interesting message to developers that it is more important to stop traffic and collect the toll than it is to keep the flow moving. Checkpoints are rarely used to kill projects, but the cost of delay associated with them is substantial" (*DPHT*).

The final key point is that to accelerate development, overlapping (working concurrently on multiple activities) is a basic tool. To do this effectively requires the teams to become effective using partial information. *DPHT* also strongly recommends using small, co-located teams.

Organize Staff

The organizing concepts of *DPHT* mirror software management best practices — co-locate, push decisionmaking to the team level, work on communications and collaboration, reduce fragmentation. The authors provide some interesting data points from research that show two projects per person is somewhat more productive, while one person per project increases speed. Their research shows having one person per project causes a slight penalty in efficiency, but at the same time, provides a large savings in cycle time — confirming again the negative impact of fragmentation on speed.

An Interview with Preston Smith

Although *DPHT* discusses manufacturing products, its message has broader applications. Author Preston Smith took time away from his busy schedule to discuss his work and its applicability to software development.

JH: Your book deals primarily with manufactured products. Have you worked with clients who are developing software, and how applicable have you found the ideas to software development?

Smith: We have worked with many clients where software is a major or dominant part of their hardware product (products such as medical computers) and some where software is the only product. For example, last week I led a two-day workshop with a software company, and they filled the walls with flipcharts listing applicable tools (from the book). The book is gradually getting more software examples. However, we tend to avoid software examples for the same reason that we avoid small-company examples and unfamiliar products; it is easier for the reader to relate to a familiar, tangible product.

Although some software developers claim there is little connection between software and hardware development, this is usually a smoke screen to avoid looking at their work in new ways. With a couple of exceptions, most of the techniques are broad enough that they do apply well to software with some adaptation. One obvious exception is that the manufacturing phase is trivial with software. However, clients have pointed out that even here there are comparable activities, such as product documentation and technical support, that require close communication and coordination with the design effort to avoid delays late in the cycle.

The other apparent exception is in physical co-location of the development team. As powerful as this technique is, it receives a great deal of resistance in high-tech and especially software companies. Programmers do

need uninterrupted periods of concentration. But due to this isolation, they run an even higher risk than hardware designers of being disconnected from the nuances of what is really going on with the customer, the marketplace, or the rest of the team. To me, this simply means that we have to work harder to find effective ways to co-locate software teams.

JH: Have you found companies who seem to be successful in accelerating a pilot project or two, but can't seem to sustain the improvements over time? What seem to be the main barriers to sustainability?

Smith: Unfortunately, too many companies encounter this sustainability problem. There are three causes.

First, they do a pilot and see how wonderful the results are. They obviously want to do more of a good thing, so they expand it to the rest of the company at a rate beyond what they can sustain or even justify. For example, there are real limits to how fast one can train heavyweight team leaders.

Second, since a pilot, by nature, can be done without everyone being involved, all too often it does not have the full support or understanding of senior management. Then, when senior management changes, the pilot becomes an orphan.

Third, they get the cart before the horse and forget that the main reason for a pilot is to learn from it, rather than to get a specific product to market faster. They get the latter, but without the learning, there is no basis for repeating their success. I emphasize this learning heavily because I have observed the misplaced emphasis too often. The other difficult part here, though, is that the learning takes effort, which is essentially deducted from the effort that could go into the next project. Many managers either don't recognize or are unwilling to accept the need to invest in improving a core business capability.

JH: In the book you discuss the preference for dedicated, co-located, cross-functional teams, but also the trend toward more dispersed, virtual teams. The number of virtual teams, armed with new groupware tools, is increasing. Are companies really understanding the additional challenges of these virtual teams? What are some ways leading-edge practitioners are meeting the challenge?

Smith: These virtual teams are growing, in part due to improvements in technology and in part due to the globalization and decentralization of business. These communication tools, which include video conferencing, e-mail, voice mail, and others, can improve team communication. But they can also degrade and delay communication. What has been the net effect of voice mail and its associated phone tag on decisionmaking speed? The trick is to realize that fast, reliable communication is the objective, and the technologies are some of the tools at our disposal, each with its own pluses and minuses. The highest-bandwidth solution is physical co-location, so it is preferred and is worth paying a premium for. Other technologies can fill in, but we would be wise to recognize their limitations. This important topic receives quite a bit of attention in the new edition of our book because virtual teams are being used increasingly.

JH: Most IT organizations are a level removed from manufacturing product development since they support the product development process; however, since IT develops products also, do you think it understands the process for other goods any better? Do other product development groups feel IT understands them? Are relationships with IT improving?

Smith: Frankly, I have not seen any improvement in relationships with IT groups. IT groups should have an advantage over most product developers because their customers are inside the company and thus, in principle, are more reachable than external customers. However, IT is usually a staff function that is quite isolated from product development and is measured and rewarded by different criteria. Consequently, IT seldom feels the pressure to get the manufactured product out quickly. Instead, the permanence and universal applica-

bility of their work becomes more important.

JH: You make a great distinction between faster development and increasing productivity. Is this new for many people? Do you think most companies put proportionally too much effort into productivity improvement? If so, why?

Smith: Even though people talk about cycle time, productivity seems to be a bigger issue these days. It seems that all employees are being asked to do more with less, and, to some managers, productivity and cycle time appear to be the same thing. In many ways they do drive toward the same ends, but in a couple of important ways they don't. One problem is that if management uses cycle time as a guise to squeeze more work out of their people, when the workers see through it, they will rebel. More basically, raising productivity, on a sustainable basis, will require an initial investment to make the organizational change required. If you view productivity or cycle time as a cost-cutting measure, it will be difficult indeed to invest in these changes, and you will reap only temporary advantage.

I see no problem in companies placing more emphasis on productivity than on cycle time. Provided that they have analyzed what is driving their profitability and know that it is productivity, this is exactly the right thing to do. My concern is with those who haven't run their numbers and are just hoping that some combination of speed and productivity is the right answer.

JH: There has always been a push in certain software engineering circles for carefully engineered, repeatable processes and development lifecycles (and an underlying assumption that the "megaproject" approach is better). Alternatives, such as the iterative cycle approach, are often resisted. How have you dealt with this issue in manufacturing engineering?

Smith: The trick is to know what is the right amount of structure and process, neither too much nor too little. Because software is a relatively young, fast-growing field populated by relatively unseasoned troops, usually the amount of structure in place lags the need for it. However, there seems to be a fairly recent recognition that the Software Engineering Institute's model is not the magic bullet either.

I believe that the real issue is recognizing when one should make something first (RAD in software) versus designing and analyzing it first (a waterfall approach). This applies in hardware and software development. I wish I had more guidance here, but, for example, RAD is effective in areas where subjective user factors prevail (such as user interfaces), and analyze-first is best in more objective areas where there is a derivable "right answer," such as in developing database updating algorithms.

This said, I believe managers usually would be wise to emphasize the build-first approach, simply because it reveals the big problems earlier, which is critical for speed. Because engineers usually aren't too keen on having their conceptual flaws revealed, the normal bias for management should be to encourage this early revelation while they have some freedom of action in addressing the problems discovered.

JH: In software development, the term RAD has been used to tag acceleration efforts. However, many organizations have also tagged RAD as an excuse for poor engineering and/or only useful on small projects. Do you see these same kinds of comments? How do you move organizations beyond these issues?

Smith: RAD, which is called rapid prototyping in mechanical development, can be a haven for those who act before they think and thus wander around trying to converge on a solution. RAD/prototyping is also limited in its applicability for the overall design of a big project. But it is nevertheless a valuable arrow to have in one's quiver under the right circumstances, and the fastest companies make sure they have arrows for any game they may encounter.

One way to keep these iterative approaches on track is to have a clear goal for each iteration, so that each step drives toward the global answers you need in a methodical

way. This puts some rigor in the process and focuses it better on rapid discovery of the preferred design.

JH: One of the underlying issues, particularly relevant to reducing the fuzzy front end, seems to be organizations' inability to deal with uncertainty. They want predictability where there is none or control where there is none. Do you find this to be a major stumbling block? If so, how have you helped them get by this?

Smith: You are right. We want this predictability and control where it just can't exist. Our approach, as with much of what we do, is to demonstrate through the organization's own projects that many of the questions just can't be answered until some design and testing is done. To wait for this assurance, which is impossible to obtain before doing some design, simply delays the project.

However, it is just as important to recognize that the fuzzy front end is a bootstrapping process. Many companies get caught in the fuzzy front end because they have no resources — human or financial — to do the work needed to get some answers in order to approve the project so that they can work on it legitimately.

JH: Why is fragmentation so hard to fix? When I discuss fragmentation with clients, they always agree it is a major problem, but there always seems to be a long list of reasons why they can't fix it. Awareness doesn't seem to breed good solutions. Do you run into the same thing?

Smith: There is a revealing figure in our book [Figure 11-1] that shows the fallacy of trying to gain efficiency by splitting people between projects. The original data, from Professors Wheelwright and Clark at Harvard Business School, shows that two projects per engineer is most productive. We replotted the data to show that if cycle time rather than productivity is what drives profitability, then one project per engineer, not two, is the right number.

This is difficult for managers to change because, basically, it requires them to say "no" when the request comes in to take on one more task, and then the manager may not be viewed as a "team player." It is far easier to go along with one's boss or with the customer.

However, another difficulty in dedicating a person full time to one project is that we have trained a workforce of specialists, so that it takes an assortment of bit-part players to get anything done. In place of this group of specialists, it is often far more effective to assign a generalist to a project full-time, but our recruiting, training, and compensation systems work against us in encouraging generalists.

This generalist issue is an illustration of the many situations we encounter in shrinking cycle time in which we know what must be done to solve the problem. It isn't exotic, nor is it difficult to understand. It just requires a certain commitment to improving our product development.

ABOUT THE AUTHOR

Jim Highsmith is a Fellow with the Cutter Business Technology Council, directs Cutter Consortium's Agile Project Management practice, and is considered a leader of the Agile Methodology movement. He is a frequent keynoter at Cutter Summits and symposia. Mr. Highsmith has 20-plus years' experience as an IT manager, product manager, project manager, consultant, and software developer. He consults with IT and product development organizations and software companies worldwide to help them adapt to the accelerated pace of development in increasingly complex, uncertain environments. Jim Highsmith is the author of *Agile Software Development Ecosystems*, which portrays the principles, problem domains, key industry leaders, and project success stories associated with Agile Methodologies. His first book, *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems*, won the prestigious Jolt award for product excellence. Jim Highsmith can be reached at jhighsmith@cutter.com.

CUTTER CONSORTIUM

get Agile

Get Your Free Trial to the Agile Project Management E-mail Advisor!

Agile Project Management Director, Jim Highsmith

There's no risk and no obligation!

Current pressures to deliver software quickly, change quickly, and change often are compelling reasons to rethink traditional software engineering practices. Cutter Consortium's weekly e-mail bulletin, the **Agile Project Management E-Mail Advisor**, can help.

Written by Practice Director Jim Highsmith, celebrated project management expert Rob Thomsett, and other Cutter Consortium Senior Consultants, the **Agile Project Management E-Mail Advisor** helps you implement project management and software development practices that support responsiveness, adaptability, and innovation.

Whether the topic is methodology design principles, exploratory project examples, project leadership, problem domains, project-level refactoring, feature-driven planning, or the architect's role in agile development, you'll get practical, real-world advice that will help you improve your organization's approach to project management.

Register Today and receive your free, four-week trial to the **Agile Project Management E-Mail Advisor**! Get strategic advice and guidance that will help you:

- Implement Agile Methodologies
- Benefit from Extreme Programming practices
- Achieve customer-focused development
- Apply the most effective software practices
- Improve your decision making skills
- Employ Extreme Project Management

It's yours free for four weeks!

Two ways to register:

1. Online at **www.cutter.com/project/email.html**
2. Complete the order form below and fax it to Cutter Consortium at +1 781 648 1950. Your trial subscription to the Agile Project Management E-mail Advisor will begin the Thursday after you subscribe.

Begin benefiting from the insight of Jim Highsmith, Rob Thomsett, and the rest of the Cutter project management team!

☐ **YES! Please register me for a FREE, four-week trial to the *Agile Project Management E-mail Advisor*!**

Name	Title
Company	Dept.
Address	Mailstop/Suite
City	State/Province
Zip/Postal Code	Country
Telephone	Fax
E-mail (Be sure to include for *Advisor* delivery.)	

CUTTER CONSORTIUM

Mail to: Cutter Consortium
37 Broadway, Suite 1
Arlington, MA 02474-5552, USA

Phone: +1 781 648 8700

Fax: +1 781 648 1950

E-mail: service@cutter.com

Web site: www.cutter.com/

FAX THIS COMPLETED FORM TO +1 781 648 1950