THE DORSET HOUSE QUARTERLY VOL. XIV NO. 1 SUMMER 2004
DORSET HOUSE PUBLISHING 353 WEST 12TH STREET NEW YORK, NEW YORK 10014 USA

DHQ

DHQ EXCERPT

Agility and Largeness

by **Jutta Eckstein**

Adapted from Chapter 1 of

Agile Software Development in the Large: Diving Into the Deep

ISBN: 0-932633-57-9 ©2004
248 pages softcover $39.95 postpaid

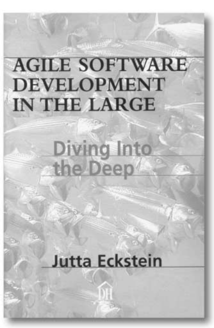
Agile processes promise to react flexibly to continuously changing requirements. That is why agile processes are currently treated as a panacea for successful software development. However, agile processes are almost always recommended for small projects and small teams only—bad news for those large teams that have to deal with speedy requirements changes.

Software engineers tend to question the feasibility of agile software development in the large, not only because most agile processes claim to work mainly for small teams, but also because most of the projects that fail are large. The reason most (large) projects fail is a lack of communication: among teammates, between team and manager, between team and customer, and so on.

The Importance of Communication

Communication is one of the focal points of agile processes. But can effective communication ever be established successfully for large teams? The popular opinion is that it can't, leading to the idea that if you have a hundred people on a development team and get rid of all but the top twenty or (preferably) fewer, the chances for project success will rise significantly.

However, you can't generally avoid large projects. Sometimes, you will face constraints that force you to run a large project with a large team. For instance, some projects have such a large scope that it is not possible to realize it with a small team in the defined time frame.



If you want to take advantage of agile processes, several questions arise: Are agile processes able to scale? That is, Can they be amplified in order to support large projects? And, moreover, are they able to support large projects? And what kind of problems occur when an enterprise decides to use an agile process for a large, perhaps even mission-critical, project? My book tries to answer these and many questions relating to agile software development. Here, though, I discuss some aspects of what I mean by *large* projects.

The Dimensions of Largeness

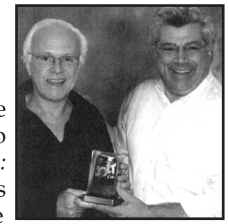
In my experience, I have found that a project can be considered large in many dimensions. For example, the money, scope, amount of people, and risks involved can be large. These different "dimensions" of largeness are mostly interrelated.

(continued on page 2)

Greetings

As we approach our 20th anniversary in December, we take pride in what's shaping up to be our most prolific year ever. As we describe below and on page 6, we've added three books to our forthcoming titles list. Be sure to subscribe to *e-DHQ* for updates as we release these titles in the coming months—just e-mail info@dorsethouse.com.

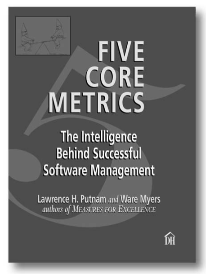
DeMarco and Lister *Waltz* Away with a Jolt Award for Best Book



Tom DeMarco and Tim Lister celebrate their Jolt Award for best book.

In a ceremony at the SD West conference in Santa Clara, California, Tom DeMarco and Timothy Lister's *Waltzing with Bears: Managing Risk on Software Projects* was awarded the Jolt Product Excellence Award for best general-interest book.

The Jolt Awards are presented every year to products that have boosted the productivity of software professionals. Tim Lister was present to receive the award, a can of Jolt cola ensconced in a lucite cube. Out of just six finalists for the award, another Dorset House release was honored: *Five Core Metrics: The Intelligence Behind Successful Software Management*, by Lawrence H. Putnam and Ware Myers. To celebrate these honors and to jolt as many readers as possible, we are offering a 20% discount off *Waltzing with Bears* and *Five Core Metrics* when you order before August 31, 2004. Call (800)-342-6657 and mention "JOLT 20" or order online at www.dorsethouse.com/jolt/. See page 3 for an excerpt from *Waltzing with Bears*.



Large Teams Go Agile, Too, with New Book

At XP2004, Jutta Eckstein celebrates her brand-new book with Linda Rising and Mary Lynn Manns.

HOT OFF THE PRESSES: *Agile Software Development in the Large: Diving Into the Deep*, by Jutta Eckstein, explores ways to adapt agile methods for use on large projects. A board member of the Agile Alliance (www.agilealliance.org) and an early supporter of the Agile Manifesto, Jutta (pronounced *U-tah*) shows large teams how to scale-up the processes and value system of lightweight, agile methods, which have generally been considered limited to small teams. Jutta presented the first printed copies of her book while serving as the program chair at XP2004, held in Garmisch-Partenkirchen, Germany, June 6–10. See this page for an excerpt from Jutta's book and page 7 for ordering information.

A *DHQ* Debut for Three Titles

Three titles make their first appearance in this publication: *Endgame*, by Robert Galen • *Just Enough Requirements Management*, by Alan Davis • *Object-Oriented Computation in C++ and Java*, by Conrad Weisert. You'll save 20% when you reserve by credit card prior to publication. See page 6.

INSIDE *DHQ*

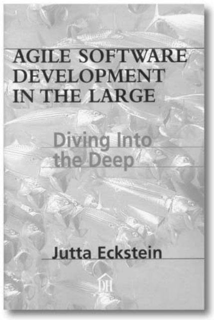
Is this your first *DHQ*?

Send your postal address for a free subscription, but also send your e-mail address for **e-DHQ**, our monthly e-mail newsletter. Since we only mail ***DHQ*** to subsets of our list, **e-DHQ** is the most reliable way to get ***DHQ***'s news and features. We maintain our list in-house, and we do not rent, sell, or share information about our customers with any third-parties.

Call 1-800-342-6657 or (212) 620-4053, fax (212) 727-1044, e-mail info@dorsethouse.com, or visit us at www.dorsethouse.com/comments.html.

Editor: David McClintock
Editorial Assts.: Nuno Andrade and Vincent Au

NOW IN STOCK



Agile Software Development in the Large:

Diving Into the Deep

by Jutta Eckstein

ISBN: 0-932633-57-9 ©2004
248 pages softcover $39.95 postpaid

Agile or "lightweight" processes have revolutionized the software development industry. They're faster and more efficient than traditional software development processes. They enable developers to embrace requirement changes during the project, deliver working software in frequent iterations, and focus on the human factor in software development.

Unfortunately, most agile processes are designed for small or mid-sized software development projects—bad news for large teams that have to deal with rapid changes to requirements. That means all large teams!

With *Agile Software Development in the Large*, Jutta Eckstein—a leading speaker and consultant in the agile community—shows how to scale agile processes to teams of 1 to 200. In fact, the same techniques are also relevant to teams of ten or more developers, especially within large organizations.

Topics include

- the agile value system as used in large teams
- the impact of a switch to agile processes
- the agile coordination of several subteams
- the way project size and team size influence the underlying architecture

Stop getting frustrated with inflexible processes that cripple your large projects! Use this book to harness the efficiency and adaptability of agile software development.

Read More About This Book at
www.dorsethouse.com/books/agile.html

DHQ Excerpt *(continued from page 1)*

Some dimensions—scope and people—exist as a first-order consequence of the requirements and constraints. Others are derived from these first-order dimensions.

You can definitely run a large-scope project with a small team. But large-scope projects are almost always developed by a large team—especially in large companies.

Typically, if a project is large in terms of people, all its other dimensions are probably just as large. For example, you will hardly ever find a large team working on a project with a narrow scope, a schedule of only three months, or a budget of only a few hundred thousand dollars. The project itself might not carry any extraordinary risk, but scaling all the project's dimensions implies a risk of its own. For instance, if a lot of money is involved, there is a high risk that a lot of money will be lost. Or, if the time frame is extremely large, the risk that the project will never be finished at all increases.

The Impact of Largeness

Of course, *large* is not a well-defined magnitude, and neither is the largeness of a team. Will a team be considered large if it contains 2, 10, 100, 1,000, or even more people? And what impact does every additional order of magnitude in staff number have on the process? For example, let's look at its influence on communication:

- **2 people and more:** If a project is developed by only one person, that person should have the big picture of the project in mind. He or she knows the whole system in terms of code and design. As soon as another person is added to the project, these two people will have to communicate with each other. Communication is the only thing that will enable both developers to understand what is going on and to further coordinate their efforts. For example, it would be annoying if they both worked on the same programming task unknowingly, only to find out once they began to integrate the code.
- **10 people or more:** With teams of this size, you have to start coordinating members' efforts and their communication. You have to explicitly establish communication channels in order to discuss topics with the whole group.
- **100 people or more:** Even if you have an open-plan office available, teams of this size will not fit in a single room. Therefore, across the entire team, you have to strategically foster the kind of "natural" communication that would take place inside a single room.
- **1,000 people or more:** Chances are high that this team will not only be distributed over several rooms, but also over several buildings, perhaps over several locations. Consequently, the people on the team are unlikely to know all their teammates.

This example shows not only that large is relative, but also that scaling can lead to different consequences.

Detecting the Agile Process for Scaling

A large team is typically split into many smaller teams. Because a lot has been written elsewhere about agile processes in small teams, I do not focus on the processes these subteams are using. Instead, I concentrate on the process that brings them all together and enables them—despite the large number of people—to work together with agility. Therefore, rather than focus on every aspect of agile processes, I concentrate only on those that work differently in large projects developed by large teams. ■

". . . a thought leader in agile processes and patterns. . . . she has a lot to say and the industry will be the better for her guidance and advice."

—KEN SCHWABER
Founder and director, Agile Alliance
Co-developer of the Scrum Agile Process

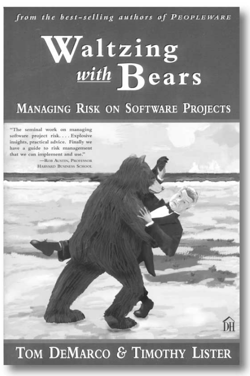
About the Author

Jutta Eckstein is an independent consultant and trainer. She has unique experience in applying agile processes within medium-sized to large mission-critical projects. She is a member of the board of the Agile Alliance and a member of the program committee of several European and American conferences in the area of agile development, object-orientation, and patterns. For more information, visit www.jeckstein.com.

DHQ Excerpt

Shocked, Disappointed, and Dismayed

by **Tom DeMarco** *and* **Timothy Lister**



Adapted from Chapter 7 of

Waltzing with Bears:

Managing Risk on Software Projects

ISBN: 0-932633-60-9 ©2003
208 pages softcover $33.95 postpaid

"Without knowing anything at all about your current project, I'll bet even money that you'll be late. After all, well over half of all projects deliver late or deliver less than was promised by the original deadline. It's far worse when a project is on an admittedly aggressive schedule. Project people seem disconcerted when I proclaim that I'm willing to bet against them. They try so hard to believe that they'll buck the odds. What usually happens is that everyone agrees that the deadline is very tight; everyone works very hard; and then, when people see that they won't make it, they are shocked, disappointed, and deeply dismayed."

—Tim Lister

Somehow, the tactic of professing to be shocked, disappointed, and dismayed when you don't catch all the lucky breaks makes it okay to have followed a plan that depended on catching those breaks. But depending on luck for success is not okay; it's real kid stuff.

The Indy 500 Analogy

Enough about software projects, for the moment. For now, you're going to be an Indy Racing League driver.

There you are behind the wheel of the Panther Racing Penzoil Dallara machine with its huge Aurora engine roaring. This is the maximo racing experience. You downshift going into the third turn and skid slightly, but you come out of it nicely, shifting up and accelerating. Your speed on the straightaway is maybe 220 or 225. You pass one, two cars in a blur, and by golly, you're leading the pack. This is your dream and it's coming true.

Take an instant to get perspective: You've been driving for two hours and fourteen minutes, and it's no wonder you're tired. This is lap 198. There are fewer than five miles between you and the checkered flag. Whatever you do, don't let up. Keep applying the heat, but play it safe because this race is yours to lose. In fact, the only real threat is Team Green. They're still behind you, but not very close. You place yourself tight on the rail and concentrate. Only one little piece of your mind is focused on anything but driving: It's the piece that's listening to the gas alarm. You glance down and see that the needle is on empty. But there are only a few miles left. Your pit crew is waving you in, but a pit stop now means losing. The engine has never sounded better. You bear down, holding your position exactly between Team Green and the finish. The last lap. This is it—you're going to win! But wait, the engine is sputtering. It's coughing; you're starting to lose momentum. Hold on, baby. You urge it on as best you can, but there is no engine now. What the hell, you think, being first across is still a win, even if you're coasting. You coast, nearer and nearer and nearer the line . . . but then you stop, just a few feet short. Team Green goes roaring past.

What just happened? You made a calculated decision to skip the pit stop in order to have any chance at all of winning. You willingly took a chance of not finishing at all in order to hang on to even a remote hope of finishing first.

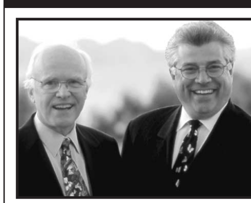
That makes good sense if you're an Indy 500 racer. But you aren't. (Sorry.) You're a software project manager. The same mind-set on a software project is a disaster. When you take every chance in order to win, you may raise the consequences of losing, far beyond where they need to be.

It's a strange calculus but true: Limiting the extent of your losses in software project work is more important, on average, than doing anything about your wins. Every organization suffers defeats in this business. The ones that get hurt most by their defeats are the losers, no matter that they win a few others.

When you challenge your subordinates to pull out the stops and bring the project home on time (even though the schedule is ludicrous), you need to understand that you're staffing your key positions with NASCAR racers. They will take every chance, ignoring every imaginable downside, in order to preserve—at least for the longest time possible—any thin, little chance of winning.

Call that what you will, but it ain't risk management. ■

About the Authors



Tom DeMarco and Timothy Lister are longtime colleagues as principals of the Atlantic Systems Guild (www.systemsguild.com). Other Dorset House collaborations of theirs include *Peopleware*, *Software State-of-the-Art*, and the video *Productive Teams*.

DHQ Interview

About the Book

with

Rodger D. Drabick

author of

Best Practices for the Formal Software Testing Process:
A Menu of Testing Tasks

ISBN: 0-932633-58-7 ©2004
312 pages softcover $41.95 postpaid

DHQ: ***You describe your new book,* Best Practices for the Formal Software Testing Process, *as one you wish you'd had at the start of your testing career. What's your main message to today's testers?***

RDD: This book is intended to provide a process-related look at the software and system testing life cycle.

I wrote this book because when I first began software testing, I had previous experience in testing complex hardware systems. Had I just followed that model, I wouldn't have realized the criticality of beginning test planning during requirements definition.

Also, I initially didn't realize the importance of test design (for example, identifying test cases at a high level, prior to developing the cases and writing test procedures). Fortunately, I had been introduced to Software Quality Engineering's Systematic Test and Evaluation Process at the very start of my software testing career, as well as being introduced to the IEEE Std. 829-1991, the IEEE Standard for Software Test Documentation.

Many people don't start out with this background. When I began my software testing career in the early 1980's, software QA and testing were seen as policemen and adversaries of the development team. Now, we realize that we should be a single team interested in providing products to our customers with a minimum number of defects while satisfying the user's requirements.

My main message is that we test engineers and test managers need to optimize our processes and the way we work so that we can help our organizations deliver quality products. I believe we can only do this if we understand our testing process and are continually working to improve it.

DHQ: ***In his foreword to your book, William E. Perry describes how your focus changed over the years, until you began concentrating on the testing process. What drew you to the testing process?***

RDD: The answer to this one is simple. After attending many conferences, I realized that process is critical. As a poster on the wall of our work area states, "Action without process is perilous." Such a situation is also chaotic. I had also been exposed to the importance of process by coordinating the first SEI-style self-assessment in Eastman Kodak Company's Research and Engineering Division in 1987. After working on a couple of process definition and improvement initiatives at Eastman Kodak, I realized that understanding a process was the first step to effective process improvement. What solidified my interest in the testing process was the need to create a solid Basis of Estimate for a large program Kodak and IBM were bidding on for the IRS. That's what led to the development of this model. By the way, we won the contract, but that's another story.

DHQ: ***One especially memorable phrase from the book is that "There is no 'one true way' to test software." What misconception is out there—and where does that one true way mislead testers?***

RDD: Like anything, the way you test depends on the environment you're in. If you're in a DoD, NASA, FDA, or other highly regulated environment, you need a rigorous testing process and approach to testing, because errors can cost people's lives. If you're on a program working a spiral or other iterative life cycle, you need a defined process for each iteration. In the dot-com environment, you have to develop and test quickly to hit a very limited window of market opportunity. In an XP environment, you're coding and testing together. Each environment demands a different approach to testing. So, a testing process model has to be tailorable. One size *doesn't* fit all; one process doesn't apply to all projects. In some environments, an "exploratory testing" strategy may be the best approach, as compared to a more documented, systematic approach.

DHQ: ***You also assert that testing is not a phase—that testing should start during requirements elucidation and extend through system testing. How is this scope of testing different from common practice, and what problems does it solve?***

RDD: There are so many visions of the scope of testing that this question is difficult to answer without writing a whole other book. I believe in what I like to call "life cycle testing."

I'd like to hope that with all the work that's been done by good testing consultants and practitioners over the years, my assertion isn't unusual. However, in a number of companies in the commercial world, the development environment is still one where the developers "throw products over the wall" to test engineers, who in some cases don't even know there's a new product being worked on. (I worked at a small company like this a few years ago. Surprising the testers with a delivery doesn't say much for management, the test engineer, or the development engineer's ability to communicate, but that's another issue.)

If test engineers have a customer's interests and perspective in mind, they can ask significant questions during requirements analysis. Test engineers should also be planning their tests at that time, so that when coding and debugging is complete, the test team has its test procedures ready to run and the test environment in place to properly evaluate the quality of the product.

> **Each environment demands a different approach to testing. So, a testing process model has to be tailorable. One size *doesn't* fit all; one process doesn't apply to all projects.**

But let there be no mistake; the developers are responsible for the quality of the product. Test and QA engineers measure the quality of the product, but you don't achieve zero-defect code through the application of testing alone.

Note that I consider testing and SQA to be two different disciplines. This philosophy came from the fact that IEEE has separate standards for SQA plans (IEEE-Std-728) and test plans (IEEE-Std-629). I happen to have a similar process model for SQA, if anyone's interested.

DHQ: ***Thanks, Rodger!***

There's More to This Interview
Visit **www.dorsethouse.com/books/bpf.html**.

DHQ Excerpt

What Is ECSAM?

by **Jonah Z. Lavi** *and* **Joseph Kudish**

Adapted from the Preface of
Systems Modeling & Requirements Specification Using ECSAM:
An Analysis Method for Embedded and Computer-Based Systems
ISBN: 0-932633-45-5 ©2004
est. 408 pages softcover $53.95 postpaid

This book introduces ECSAM, an Embedded Computer Systems Analysis and Modeling method. The ECSAM method allows systematic analysis of the interaction of a system with its operational environment.

Although the method was originally developed to help in the analysis of embedded systems—which typically consist of hardware and software—it has shown its effectiveness over the years in the analysis of software-only systems, hardware-software systems, as well as in the analysis of the operation of organizations and teams.

The E-Level Model

The analysis process focuses initially on the E-level model, an extended scope of the system. The E-level model defines the dynamic properties involved in the interaction of the system with its environment. The scope of the E-level model includes the system—presented as a "black box" whose inner workings are not of concern at this stage—and the environmental systems with which it interacts. The E-level model defines the system boundary, the external ("environmental") systems with which the system interacts, the interfaces between the system and the external systems, and the known interfaces between the external systems. The dynamic view of the E-level model provides the basis for the systematic derivation of use cases and of the resulting operational scenarios that define the required dynamic behavior of the system, as well as the basis for the derivation of operational and test requirements.

The S-Level Model

The static and dynamic properties of the system discovered by analysis of the E-level model serve as a starting point for the second phase of analysis, in which the inner workings of the system to be built are modeled and analyzed as a "white box." Transition between black-box and white-box models is carried out by limiting the scope of analysis to the system and its interfaces, creating a system model that is called the "S-level model."

The S-level model is a hierarchical conceptual model of the system, allowing separate analysis of conceptual subsystems to a level of refinement deemed appropriate by the analyst. The analysis of the S-level model addresses its static and dynamic properties in a manner similar to the one employed in the analysis of the E-level model. During successive iterations of modeling and analysis, the system is broken down into lower-level conceptual (logical) subsystems.

Verification and Flow-Down

The analysis process verifies the correctness of the decomposition and the consistency of the properties of the system and its subsystems (at any level). Correctness is verified by demonstrating that the static and dynamic properties of the system (or any of its subsystems) can be expressed in terms of the properties of its conceptual components.

Requirements originally allocated to the system are iteratively allocated to progressively lower-level subsystems. This flow-down process drives the refinement of high-level requirements and determines the mandatory characteristics of the individual subsystems.

Views and Semantics

The ECSAM method uses several complementary and interrelated views, all of which utilize graphical representations. ECSAM also uses formal (graphical and mathematical) semantics, wherever applicable, in the specification of the E-level and S-level models, producing executable specifications that allow the analysts to test the system's conceptual, static specification and simulate its dynamic behavior.

To the beginning ECSAM modeler, a graphical modeling language that has rigorous, well-defined semantics and that imposes strict modeling rules may seem an unnecessary burden, particularly when the diagrams are compared to freehand drawings that may initially appear similar. However, insistence on the correctness and consistency of the model—from its earliest stages of development—proves to be one of the best investments a project can make, sparing the budget the ever-escalating cost of rework necessitated by ambiguous diagrams and error correction during later phases of the project. ■

Early Praise

"I like this book a lot. It is one of very few books on systems and software engineering that introduces a solid and comprehensive methodology, which has been carefully worked out, has been extensively used, and has also been meticulously taught to a large number of students and engineers. . . . A truly valuable contribution to the field!"

—DAVID HAREL
Professor, Faculty of Mathematics and Computer Science, The Weizmann Institute of Science, Israel

About the Authors

Jonah Z. Lavi, the lead developer of ECSAM, consults and teaches industrial and university courses in the modeling and requirements specification of computer-based systems.

Joseph Kudish is an independent consultant specializing in systems and software engineering and in the appraisal and improvement of technical and management processes.

Author News

Jutta Eckstein: In Salt Lake City (6/23–25), serves on the program committee for the Agile Development Conference (www.agiledevelopmentconference.com). In Irsee, Germany (7/7–11), serves on the program committee for EuroPlop (www.hillside.net/europlop). In Calgary (8/15–19), serves on the program committee for XP Agile Universe (www.xpuniverse.com/home).

Visit www.jeckstein.com.

Naomi Karten: In Boston (9/20–23), gives tutorial, "Introverts and Extroverts in the Workplace," and presents, "Changing How You Communicate During Change," at the Software Development Best Practices Conference (www.sdexpo.com). In Phoenix (11/7–10), cohosts Amplifying Your Effectiveness (AYE) conference (www.ayeconference.com).

Visit www.nkarten.com.

Gerald M. Weinberg: In Phoenix (11/7–10), cohosts Amplifying Your Effectiveness (AYE) conference (www.ayeconference.com).

Visit www.geraldmweinberg.com.

Recent Reviews

Best Practices for the Formal Software Testing Process
by **Rodger D. Drabick**
ISBN: 0-932633-58-7 ©2004 312 pages
softcover $41.95 (includes $6 for UPS in US)

"When faced with impossible tasks, something that software testing has now become, the best that you can do is examine a subset composed of the most likely scenarios. By applying the models in this book, it is possible to raise the level of your testing quality to the point where you can be confident in your software."

—**Charles Ashbacher**
Ashbacher Technologies, on Amazon.com

Five Core Metrics
by **Lawrence H. Putnam** and **Ware Myers**
ISBN: 0-932633-55-2 ©2003 328 pages
softcover $49.95 (includes $6 for UPS in US)

"The book does a very good job explaining software process and discussing different approaches to improving software development. I especially like the opinions about the Capability Maturity Model." —**Tim Kelleher**
Software Quality Professional

Waltzing with Bears
by **Tom DeMarco** and **Timothy Lister**
ISBN: 0-932633-60-9 ©2003 208 pages
softcover $33.95 (includes $6 for UPS in US)

"The authors provide a fresh approach to risk management. . . .

"A valuable contribution of the book is a set of questions that evaluate whether risk management is actually taking place. Readers can use this test to guide their conclusions concerning whether additional investment in risk management is warranted."

—**Ralph Young**
Software Quality Professional

ORDER EARLY & SAVE*

Endgame: *DHQ* Debut
Eliminating Defects, Controlling Change, and the Countdown to On-Time Delivery
by ROBERT GALEN ISBN: 0-932633-62-5 ©2004 est. 288 pages softcover $39.95 ppd.

In software development, projects are won or lost during the endgame—that final stage between the release for testing and the release to customers. Focusing solely on the endgame, experienced software project manager Robert Galen provides hands-on practices for defect triage, defect repair decisions, CCB meetings, defect tracking, and more.

Hiring Technical People:
The Artful Science of Getting the Right Person for the Job
by JOHANNA ROTHMAN ISBN: 0-932633-59-5 ©2004 est. 424 pages softcover $33.95 ppd.

Hiring technical people is one of the most critical and difficult processes a manager can undertake. This book takes the guesswork out of hiring, and diminishes the risk of costly hiring mistakes. With the aid of step-by-step descriptions and detailed examples, you'll learn how to • write an effective job description • source desirable candidates • develop intelligent ads • analyze résumés • interview efficiently • create phone-screens • check references • construct an offer • and more.

Just Enough Requirements Management: *DHQ* Debut
Where Software Development Meets Marketing
by ALAN DAVIS ISBN: 0-932633-64-1 ©2004 est. 240 pages softcover $39.95 ppd.

Requirements specify the needs of our users, customers, and markets. Without a thorough understanding of those needs, we are wasting our time. Professor, author, and consultant Alan Davis, former editor of *IEEE Software,* offers a pragmatic approach to identifying how many requirements are "just enough" to satisfy our customers and to meet our goals for schedule, budget, and resources.

Object-Oriented Computation in C++ and Java *DHQ* Debut
by CONRAD WEISERT ISBN: 0-932633-63-3 ©2004 est. 200 pages softcover $39.95 ppd.

Virtually all business, scientific, and engineering applications are heavily reliant on numeric data items. However, relatively few programmers are trained to exploit the full computational power of C++ and Java. Professor and consultant Conrad Weisert offers new techniques for manipulating numeric data types, reinforcing the concepts with helpful exercises throughout. The text is suitable for self-study or use in an undergraduate course.

System Testing with an Attitude
by NATHAN PETSCHENIK ISBN: 0-932633-46-3 ©2004 est. 306 pages softcover $45.95 ppd.

Developers striving for high quality and rapid time-to-market need to adopt an attitude: It is unacceptable for software that does not meet requirements to even reach the system test phase. This book explains how to cultivate productive relationships between developers and system testers and stresses the importance of identifying and delineating the responsibilities of each group. The book also provides detailed technical and procedural solutions for achieving excellence in system testing, including a fourteen-step methodology for architecting, designing, and implementing well-documented, repeatable, data-dimensional system tests.

Systems Modeling & Requirements Specification Using ECSAM:
An Analysis Method for Embedded and Computer-Based Systems
by JONAH Z. LAVI and JOSEPH KUDISH ISBN: 0-932633-45-5 ©2004 est. 408 pages softcover $53.95 ppd.

Discover ECSAM, a method for requirements engineering and the modeling of computer-based systems (CBS). Practiced since 1980 in evolving versions by large numbers of systems and software engineers worldwide, ECSAM was developed in part at Israel Aircraft Industries for the analysis and design of complex reactive embedded systems and software. The method guides engineers in modeling operational, functional, and design requirements, considering both static and dynamic aspects of systems.

Testing Dirty Systems
by WILLIAM E. PERRY and RANDALL W. RICE ISBN: 0-932633-56-0 ©2004 est. 368 pages softcover $41.95 ppd.

Some systems are more difficult to test than others. Software testers contend with undefined or partially defined requirements; outdated, incomplete, or nonexistent documentation; complex logic; a mixture of languages; or worse. All of these factors make a system dirty, or virtually untestable. In *Testing Dirty Systems,* William Perry and Randall Rice—authors of *Surviving the Top Ten Challenges of Software Testing*—teach testers a six-step process for approaching such systems: system diagnosis • test planning • test execution • test analysis • report development • dirty system repair.

Order Before August 31, 2004 and Save 20% On These and Other Titles—See Page 7 for Details.

Order Early... Save 20%

Reserve with a credit card before the indicated expiration date or in-stock date.

Endgame

by Robert Galen
ISBN: 0-932633-62-5 ©2004
est. 288 pp. softcover
Reg.: $39.95 postpaid

Pay Only $33.16 Before August 31, 2004

Hiring Technical People

by Johanna Rothman
ISBN: 0-932633-59-5 ©2004
est. 424 pp. softcover
Reg.: $33.95 postpaid

Pay Only $28.36 Before August 31, 2004

Just Enough Requirements Management

by Alan Davis
ISBN: 0-932633-64-1 ©2004
est. 240 pp. softcover
Reg.: $39.95 postpaid

Pay Only $33.16 Before August 31, 2004

Object-Oriented Computation in C++ and Java

by Conrad Weisert
ISBN: 0-932633-63-3 ©2004
est. 200 pp. softcover
Reg.: $39.95 postpaid

Pay Only $33.16 Before August 31, 2004

System Testing with an Attitude

by Nathan Petschenik
ISBN: 0-932633-46-3 ©2004
est. 306 pp. softcover
Reg.: $45.95 postpaid

Pay Only $37.96 Before August 31, 2004

Systems Modeling & Requirements Specification Using ECSAM

by Jonah Z. Lavi and Joseph Kudish
ISBN: 0-932633-45-5 ©2004
est. 408 pp. softcover
Reg.: $53.95 postpaid

Pay Only $44.36 Before August 31, 2004

Testing Dirty Systems

by William E. Perry and Randall W. Rice
ISBN: 0-932633-56-0 ©2004
est. 368 pp. softcover
Reg.: $41.95 postpaid

Pay Only $34.76 Before August 31, 2004

Your Credit Card Won't Be Charged Until We're Ready to Process Your Shipment.

Call 1-800-342-6657
or (212) 620-4053
Weekdays 9AM-6PM EST.
VISA MC AMEX

Order Form

Call 1-800-342-6657 or (212) 620-4053. *Mention "DHQ V14N1."*
Fax to (212) 727-1044 *or* **Mail** *(see address below)*
Questions? Call or fax, **e-mail** info@dorsethouse.com, or visit **www.dorsethouse.com**.

TITLE ✓***SAVE 20% ON UNDERLINED TITLES—CALL FOR DETAILS***	#	*PRICE*	*TOTAL*
Adaptive Software Development (HIGHSMITH)		$44.95	
Agile Software Development in the Large (ECKSTEIN) ***NEW***		$33.95	
Amplifying Your Effectiveness (WEINBERG, BACH & KARTEN)		$24.95	
Are Your Lights On? (GAUSE & WEINBERG)		$13.95	
Becoming a Technical Leader (WEINBERG)		$29.95	
Best Practices for the Formal Software Testing Process (DRABICK) ***NEW***		$35.95	
Communication Gaps and How to Close Them (KARTEN)		$33.95	
Complete Systems Analysis (J. & S. ROBERTSON)		$57.95	
Creating a Software Engineering Culture (WIEGERS)		$39.95	
Data Model Patterns (HAY)		$39.95	
The Deadline: A Novel About Project Management (DeMARCO)		$24.95	
Designing Quality Databases with IDEF1X Information Models (BRUCE)		$57.95	
Dr. Peeling's Principles of Management (PEELING) ***NEW***		$29.95	
Endgame: Eliminating Defects, Controlling Change (GALEN) ***FORTHCOMING***		$27.16†	
Exploring Requirements (GAUSE & WEINBERG)		$44.95	
Five Core Metrics (PUTNAM & MYERS) ***NEW***		$43.95	
Fundamentals of Object-Oriented Design in UML (PAGE-JONES)		$39.95*	
General Principles of Systems Design (D. & G. WEINBERG)		$27.95	
Handbook of Walkthroughs, Inspections, and Technical Rev. (FREEDMAN & WEINBERG)		$49.95	
Hiring Technical People (ROTHMAN) ***FORTHCOMING***		$22.36†	
How to Plan, Develop & Use Information Systems (VAN STEENIS)		$34.95✓	
An Intro. to General Systems Thinking: Silver Anniv. Edition (WEINBERG)		$33.95	
Just Enough Requirements Management (DAVIS) **FORTHCOMING**		$27.16†	
Managing Expectations (KARTEN)		$27.95	
Measuring and Managing Performance in Organizations (AUSTIN)		$24.95	
More Secrets of Consulting: The Consultant's Tool Kit (WEINBERG)		$33.95	
The One Minute Methodology (ORR)		$12.95✓	
Object-Oriented Computation in C++ and Java (WEISERT) **FORTHCOMING**		$27.16†	
Peopleware, 2nd ed. (DeMARCO & LISTER)		$33.95	
Practical Guide to Business Process Reengineering Using IDEF0 (FELDMANN)		$34.95	
Practical Project Management (PAGE-JONES)		$34.95✓	
Process for System Arch. and Requirements Eng. (HATLEY, HRUSCHKA & PIRBHAI)		$59.95	
Productive Teams: A Video (DeMARCO & LISTER)		$95.00✓	
Project Retrospectives (KERTH)		$33.95	
The Psychology of Computer Programming: Silver Anniversary Edition (WEINBERG)		$44.95	
Quality Software Management, Vol. 1: Systems Thinking (WEINBERG)		$41.95	
Quality Software Management, Vol. 2: First-Order Measurement (WEINBERG)		$43.95	
Quality Software Management, Vol. 3: Congruent Action (WEINBERG)		$39.95	
Quality Software Management, Vol. 4: Anticipating Change (WEINBERG)		$44.95	
Rethinking Systems Analysis & Design (WEINBERG)		$27.95	
Roundtable on Project Management (BULLOCK, WEINBERG & BENESH)		$15.95	
Roundtable on Technical Leadership (WEINBERG, BENESH & BULLOCK)		$15.95	
The Secrets of Consulting (WEINBERG)		$29.95	
Slack (DeMARCO) ***NEW***		$19.95	
Software Productivity (MILLS)		$25.95✓	
Strategies for Real-Time System Specification (HATLEY & PIRBHAI)		$49.95	
Surviving the Top Ten Challenges of Software Testing (PERRY & RICE)		$27.95	
System Testing with an Attitude (PETSCHENIK) ***FORTHCOMING***		$31.96†	
Systems Modeling & Req. Spec. Using ECSAM (LAVI & KUDISH) ***FORTHCOMING***		$38.36†	
Testing Dirty Systems (PERRY & RICE) ***FORTHCOMING***		$28.76†	
Understanding the Professional Programmer (WEINBERG)		$24.95	
Waltzing with Bears (DeMARCO & LISTER) ***NEW***		$27.95	
What Every Programmer Should Know About Object-Oriented Design (PAGE-JONES)		$44.95▲	
Why Does Software Cost So Much? (DeMARCO)		$29.95	

†Discounted price valid on orders received before the indicated expiration date (see left column). ✓Call for details: Save 20% when you order with a full-priced title. *No qty. discount. ▲Qty. limited. See Dorsethouse.com for offer.

SHIPPING AND HANDLING POLICY: First book or video, add $6.00 for UPS shipping (first class, air, and all shipments outside continental USA are additional: please call, fax, or e-mail with quantities desired, shipping address, and your contact information (phone, fax, e-mail); we will provide shipping costs estimate). Each additional book, volume, or video up to 5, add $1.25 per item. For 6 or more items, call us for a quote. We ship UPS, unless otherwise requested. For UPS, please give complete street address, no postal boxes. PAYMENT MUST ACCOMPANY ORDER in US funds; NYS residents, please add sales tax. Prices are effective Jan. 1, 2002 and are subject to change without notice.

SUBTOTAL:
SHIPPING:
NYS TAX:
TOTAL:

❑ *Enclosed is my check or money order.* ❑ *Charge the total to my credit card:*

❑ VISA ❑ MC ❑ AMEX Card # ____________

Exp. date ________ Signature ____________

Daytime phone ____________ e-mail address ____________

❑ *E-mail me a shipping confirmation.* ❑ *Subscribe me to* **e-DHQ**.

NAME ____________ TITLE ____________

COMPANY ____________

❑ HOME ❑ BUSINESS STREET (NO P.O. BOXES FOR UPS DELIVERY)

CITY ____________ STATE ______ ZIP ______ *Callers, please mention* **DHQ V14N1**

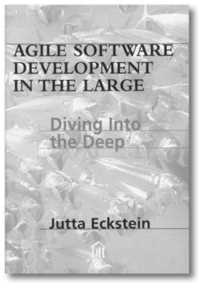
DORSET HOUSE PUBLISHING 353 WEST 12TH STREET NEW YORK, NEW YORK 10014 USA www.dorsethouse.com

DHQ

THE DORSET HOUSE QUARTERLY VOL. XIV NO. 1 SUMMER 2004

DORSET HOUSE PUBLISHING 353 WEST 12TH STREET NEW YORK, NEW YORK 10014 USA

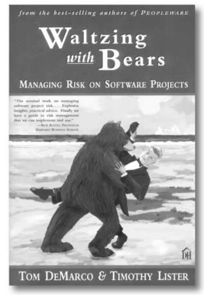
See Page 7 to Save 20% On Selected Titles



Inside, read an excerpt from Jutta Eckstein's **Agile Software Development in the Large: Diving Into the Deep**.

ISBN: 0-932633-57-9 ©2004
248 pages $39.95 postpaid

See page 1.



Inside, read an excerpt from Tom DeMarco and Timothy Lister's **Waltzing with Bears: Managing Risk on Software Projects**.

ISBN: 0-932633-60-9 ©2003
208 pages $33.95 postpaid

See page 3.

software development 14th annual product excellence award JOLT

Waltzing Wins the **JOLT**! See Page 1.

Greetings

This issue of *The Dorset House Quarterly* offers an interview, excerpts, news about forthcoming books, reviews, and Author News.

If this is your first issue of *DHQ*, or if you'd like to receive *e-DHQ*, the e-mail edition, call us at 1-800-342-6657 to sign up for a free subscription.

Or, send us your address and phone number via our Website, **www.dorsethouse.com**, or e-mail **info@dorsethouse.com**.

We welcome your comments, requests, and questions.

Toll Free! 1-800-DH-BOOKS

ADDRESS SERVICE REQUESTED

Please mail, fax, or e-mail address corrections/removals

www.dorsethouse.com info@dorsethouse.com
(212) 620-4053 FAX (212) 727-1044
New York, NY 10014
353 West 12th Street
THE DORSET HOUSE QUARTERLY

Presort Standard
U.S. Postage
PAID
Waldorf MD
Permit 144

December 4, 2004, Is Our 20TH Anniversary!

ISSUE V14N1