October 2008 $9.95 www.StickyMinds.com

BETTER SOFTWARE

The Print Companion to StickyMinds.com

CODERS GONE WILD
Three common archetypes and how to handle them

GOOGLE WEB TOOLKIT
Your key to simplifying Ajax app builds

SERIOUS SOURCECODE ANALYSIS

Klocwork helps software developers create better code.

When developing mission-critical software, developers must quickly and accurately identify, assess and fix critical software bugs and security vulnerabilities right at their desktop before they impact anyone else.

Klocwork's leading static source code analysis tools provide powerful, collaborative analysis of your C, C++ and Java code at the earliest point in the software development process – before code check-in – when detected issues are easiest and less costly to fix, freeing developers to do what they do best – innovate.

Take the first step towards better code. Visit www.klocwork.com/freetrialsignup for a free product trial.

Klocwork®

www.klocwork.com

Visit the Klocwork booth at **STARWEST 2008** and attend our session **Source Code Analysis: The Basics and Beyond** with Gwyn Fisher, CTO on October 1 at 1:45pm.

ADVERTISEMENT

Seapine Software

Five Issues that Prevent Software Quality

Software development organizations often fail to recognize what keeps them from achieving quality.

Software quality and reliability are lifelines to customer loyalty and profitability. Yet, many organizations overlook quality in the rush to develop and release products. Consider the following statistics from the Cutter Consortium, an IT advisory firm.

- 32 percent of organizations say they release software with too many defects
- 38 percent of organizations believe they lack an adequate software quality assurance program
- 27 percent of organizations do not conduct any formal quality reviews

Following are some ways poor quality impacts a company's overall revenue and profitability. Poor quality software:

- Creates avoidable labor costs
- Increases baseline costs
- Causes customers to delay purchases
- Diminishes your reputation and market share
- Exposes you to regulatory non-compliance and higher risk of litigation

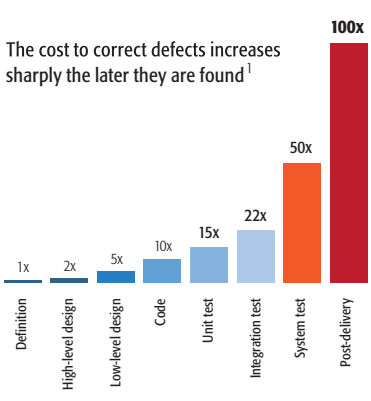
If poor quality can cause all these problems, why do most companies fail to make changes? Five issues come into play.

1 Testing will find the problems

In most organizations, software development moves through the following stages:

- Requirements collection/analysis
- Software design
- Software development
- Testing
- Deployment
- Post-implementation support

In this arrangement, you focus on quality in the testing stage, long after many of the decisions that impact software quality are made.



Organizations wrongly assume that quality can be tested into software at the end of the development process. This delay allows preventable bugs to enter the code and leaves too little time to find them before the release date. Consider that most forms of testing are less than 30 percent efficient[2], and you can see why the traditional approach to quality produces buggy applications.

2 Time to market is the key measurement for success

Many software companies believe they lose market share if they do not deliver frequent releases. As a result, time to market becomes a critical measurement. Being able to deliver a product every few months on or near budget becomes the primary goal, and quality is a minor consideration.

When they focus on an arbitrary deadline first, companies ask whether the software is good enough to release rather than asking if it is truly high quality. If the development phase takes longer, the release date remains fixed. To hit the deadline, the organization cuts the testing schedule. Because the majority of the attention to quality is placed in the final stages of development, it is shortchanged as well.

3 Management perceives little ROI from QA

Determining the ROI for quality assurance is not as clear cut as figuring out the return on a traditional product where you spend three dollars on raw materials and labor and sell the final product for ten. The effects of quality, both good and bad, are extremely subtle and may take a long time to manifest. As a result, it is easy to view the QA department as a cost center or an institutionalized expense.

1 Software Reliability: Achievement and Assessment, B. Littlewood, ed. (Henley-on-Thames, England: Alfred Waller, Ltd., November, 1987)
2 Capers Jones, *Software Quality: Analysis and Guidelines for Success*, International Thomson Computer Press, June 14, 2000

Seapine Software

Companies often feel the temptation to cut or offshore quality assurance. However, moving the quality function outside the organization can cause employees to conclude that quality is not part of their responsibilities. They assume quality resides outside of their control and someone else will take care of it later.

Poor quality affects a company's reputation and ultimately its sales. This is the hidden ROI of QA.

4 QA is not involved from the beginning

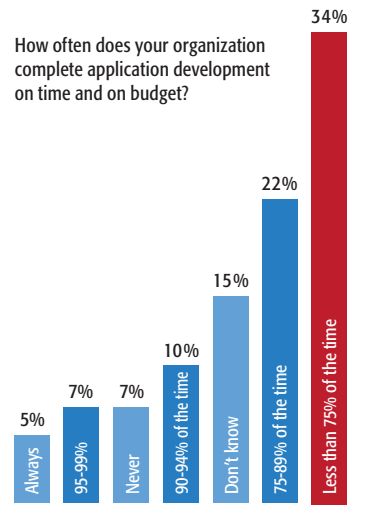
In many organizations, the development department undertakes the requirements collection and analysis, designs the application, and sets the project schedule. As they complete development and move into the alpha and beta stages, they notify QA that the application is ready to be tested. QA must then work to both understand the requirements and test them in the compressed final stages of the project, reducing their efficiency and adding stress to the process.

5 Developers only want to work on new features

The exciting part of programming for most developers is creating new functionality and seeing the lines of code become concrete features. Troubleshooting and fixing bugs is monotonous and often frustrating. As a result, antagonism grows between developers and testers.

Developers may believe that testers only want to find fault and do not appreciate the amount of effort that goes into developing the software. Testers, on the other hand, may think developers are unresponsive to what they view as an attempt to improve the product.

Placing QA too late in the development process exacerbates this strained relationship. Developers think their job is finished when they deliver the code to QA. Testers struggle to maintain the developers' attention as they work to resolve issues in the limited time before the release date.



Recommendations

Organizations cannot point to a single stage of their development process and say that is where quality resides. Quality results from the thousands of decisions made by the members of an organization. By providing guidelines and metrics throughout the process, you can improve your ability to release high quality software.

If you have expressed a commitment to quality but cannot seem to reach your goals, consider the following process enhancements.

1. **Implement testing early.** Testing should begin before the alpha stage of development. Implementing peer code reviews and unit testing reduces the number of defect found by QA late in the process. The earlier you discover defects, the easier and cheaper they are to fix.

2. **Refine your quality metrics.** Frequent releases help maintain mindshare. However, you damage your reputation if these are low quality releases. Make sure you include quality measurements at each process milestone. Measuring quality at each stage of development helps mitigate risk. Do not move to the next stage of development if the software does not pass the quality metrics.

3. **Measure the impact of quality.** Implement a new quality metric and track its impact on the number and type of defects reported. The ROI of quality is often subtle. Be sure to gauge the changes in customer satisfaction as a result of these metrics as well.

4. **Involve QA earlier.** Invite QA team members to requirements and design meetings. This allows them to point out testing issues before you write a single line of code. They can also begin writing test cases long before they receive a build, which improves testing efficiency.

5. **Communicate your dedication to quality.** Ensure everyone understands that quality is part of their job. Implement quality measurements as part of performance appraisals for developers as well as QA analysts. With more people considering quality, you will improve your ability to release on time and on budget.

Discover how serious your company is about quality.
Take Seapine Software's Quality-Ready Assessment at www.seapine.com/qualitybsw.

Copyright © 2008 Seapine Software, Inc. All rights reserved worldwide. TestTrack Pro, TestTrack, Surround SCM, QA Wizard, and the Seapine logo are trademarks of Seapine Software, Inc. This document is the result of research performed by Seapine Software, Inc. Seapine Software believes its findings are objective and represent the best analysis available at the time of publication.

MAKE YOUR PRODUCTIVITY SOAR

"OptiFrame has become a world-class software development organization thanks to Seapine Software."

David Grange
Requirements & Process Auditor, OptiFrame Software

Today's competitive business environment requires a complete toolset to get the job done on time and within budget. Utilizing efficient processes and Seapine Software's integrated ALM software tools, OptiFrame has become a world-class software development organization. Seapine Software's easy-to-use, award-winning tools help companies achieve success by streamlining communications, improving traceability, and making development and QA teams more productive.

Get the details on OptiFrame's success and learn how Seapine Software can help your productivity soar by integrating quality into every stage of software development:

www.seapine.com/optiframe

Seapine Software
Satisfy Your Quality Obsession

©2008 Seapine Software, Inc. Seapine and the Seapine logo are trademarks of Seapine Software, Inc. All Rights Reserved. All other trademarks are the property of their respective owners.

SOFTWARE TESTING
ANALYSIS & REVIEW

The Greatest Software Testing Conference on Earth

STAR EAST

MAY 4–8, 2009

The Rosen Centre Hotel

Concurrent Sessions Covering:

- Test Management
- Test Techniques
- Test Automation
- Agile Testing
- Test Processes
- Exploratory Testing
- Metrics
- Outsourcing
- Performance Testing
- Security Testing
- Web Services

Over 98% of 2008 Attendees Recommend *STAREAST* to Others in the Industry

www.sqe.com/stareast

REGISTER EARLY AND SAVE $200!

SOFTWARE QUALITY ENGINEERING CONFERENCES

www.sqe.com

BETTER SOFTWARE

October 2008
Volume 10, Issue 8

Cover Story

THE ONE-HOUR REGRESSION TEST 24
What if a customer asked you to demonstrate to him, within an hour, that your newest software is ready for use? What tests would you run? Are these the same tests that you are now performing in your first hour of regression testing? They should be.
by Steven Woody

Features

HELD HOSTAGE 30
Software developers are not typically at the top of the organizational chart. Yet in some cases, developers are able to wield their knowledge and control of the code to hold management hostage to the developers' own agenda. How can you avoid being taken hostage and losing control of your company and its software?
by Linda Hayes

GOOGLE WEB TOOLKIT 36
The Google Web Toolkit (GWT) is a tool designed to aid Ajax developers. GWT compiles Java source code into browser-friendly JavaScript and has many features that let you focus on writing your application instead of worrying about browser compatibility problems.
by Daniel Wellman

Columns & Departments

TECHNICALLY SPEAKING 15
A Gram of Prevention • *by Chuck Allison*
Good code can and should evolve from clear, up-front descriptions of the solution to the problem at hand.

CODE CRAFT 16
Passing the Buck • *by Allen I. Holub*
The basic rule for defining good interfaces is to have clients ask for help, not information. In other words, clients of an object shouldn't be dependent on its inner workings—they should only have to know how to talk to it.

TEST CONNECTION 20
Got You Covered • *by Michael Bolton*
Our evaluation of the quantity and quality of our test coverage starts with asking who our client is and what he wants to know about the system. Yet, if our client doesn't have that information, where do we start?

MANAGEMENT CHRONICLES 22
Tale of a Yo-Yo Manager • *by Esther Derby*
There is much more to empowering your team than simply stating "You're empowered." Consider the three Ws of empowerment: what, when, and why.

THE LAST WORD 47
The Tester Who Came In from the Cold • *by Lisa Crispin*
Traditionally, relationships between testing and coding teams often have bordered on frosty. But the wall has started to come down, especially in organizations that have embraced agile principles, values, and practices.

In Every Issue

Mark Your Calendar 4
Contributors 6
eLightenment 8
Product Announcements 42
10 Things You Might Not Know About ... 46
Ad Index 48

***Better Software* magazine**—The print companion to StickyMinds.com brings you the hands-on, knowledge-building information you need to run smarter projects and deliver better products that win in the marketplace and positively affect the bottom line. **Subscribe today to get ten issues.**

Visit www.BetterSoftware.com or call 800.450.7854.

StickyMinds.com We invite you to visit StickyMinds.com, the online companion to *Better Software* magazine. StickyMinds.com covers the same pertinent topics as the magazine, putting the power of information at the click of your mouse. Weekly columns, headline-making bugs, hundreds of technical papers, an online tools guide, discussion boards, and so much more make StickyMinds.com your site for 24/7 brainfood to help you build better software.

MARK YOUR CALENDAR

Training Weeks

www.sqetraining.com/Public

Testing
October 20–24, 2008
San Francisco, CA

November 17–21, 2008
Tampa, FL

Agile Software Development
October 20–24, 2008
Seattle, WA

November 3–7, 2008
San Diego, CA

Software Testing Certification

www.sqetraining.com/certification

October 21–23, 2008
Sacramento, CA and Pittsburgh, PA

October 28–30, 2008
Omaha, NE; Charlotte, NC; and Cincinnati, OH

November 4–6, 2008
Ft. Lauderdale, FL and Bethesda, MD

Conferences

Agile Development Practices 2008
www.sqe.com/agiledevpractices

November 10–14, 2008
Shingle Creek Resort
Orlando, FL

STAR*EAST* 2009
Software Testing Analysis & Review
www.sqe.com/stareast

May 4–8, 2009
Rosen Centre Hotel
Orlando, FL

BETTER SOFTWARE

Publisher
Wayne Middleton

Vice President of Publishing
Holly N. Bourquin

Editor-in-Chief
Heather Shanholtzer

Editorial

Managing Technical Editor
Lee Copeland

Technical Editors
Chuck Allison
Jonathan Kohl
Antony Marcano

Editor, StickyMinds.com
Francesca Matteu

Managing Editor, Multimedia
Joseph McAllister

Production Coordinator
Cheryl M. Burke

Design

Creative Director
Catherine J. Clinger

Advertising

Senior Advertising Sales Manager
Shae Young

Advertising Sales Manager
Joe Anderson

Production Coordinator
April Evans

Circulation and Marketing

Marketing Coordinator
Megan Harbison

Circulation Coordinator
Jamie Green-Gago

A PUBLICATION OF SOFTWARE QUALITY ENGINEERING

CONTACT US
Editors: editors@bettersoftware.com
Subscriber Services: info@bettersoftware.com
Phone: 904.278.0524, 888.268.8770
Fax: 904.278.4380
Address:
Better Software magazine
Software Quality Engineering, Inc.
330 Corporate Way, Suite 300
Orange Park, FL 32073

DevTest Studio

The integrated solution for defect tracking, test management and automated testing

DevTrack

Use DevTrack to track defects/issues

- Track each issue through a definable workflow
- SCM integration – track fixes against their source code deliverables
- Deploy a resolution across multiple releases, versions and products
- Reporting and metrics to illustrate the entire defect lifecycle

DevTest

Use DevTest to manage your testing

- Create a central repository for your test cases, knowledge items and automation scripts
- Schedule releases and test cycles using a wizard-driven interface
- Execute test assignments and submit defects from the same interface
- Track results with real-time dashboards and reports

TestLink

Use TestLink to automate your testing

- Add automated tests to the DevTest test library
- Schedule automated tests along with manual tests
- Launch automated tests from the DevTest interface
- Track automation results with real-time dashboards and reports

TechExcel

www.techexcel.com | 1-800-439-7782

Contributors

Chuck Allison developed software for more than twenty years before becoming a professor of computer science at Utah Valley University. He is a technical editor for *Better Software* magazine and founding and current editor of the online journal *The C++ Source*. He spent most of the 1990s as an active member of the C++ Standards Committee and is author of two C++ books, including *Thinking In C++, Volume 2: Practical Programming*, with Bruce Eckel. His company, Fresh Sources, Inc., gives onsite training in C++, Python, and design patterns. His current top technical interest is the resurgence of functional programming. Whenever he finds a little down time he plays classical guitar or bikes the country roads of central Utah. Contact Chuck at chuck@freshsources.com.

Michael Bolton lives in Toronto and teaches heuristics and exploratory testing in Canada, the United States, and other countries. He is co-author, with James Bach, of *Rapid Software Testing* and a regular contributor to *Better Software* magazine. Contact Michael at mb@developsense.com.

STAR WEST SPEAKER

BETTER SOFTWARE Agile Development Practices SPEAKER

Steven Woody has more than seventeen years of experience testing embedded software for networking, telecommunications, Global Positioning System receivers, and satellite telemetry and command systems. He has worked at companies ranging from the Fortune 500 to startups. Steven has presented at peer workshops and has contributed to books on software testing. You can contact Steven at slwoody@earthlink.net.

Lisa Crispin is co-author with Janet Gregory of *Agile Testing: A Practical Guide for Testers and Agile Teams*, which will be published in January 2009. Since joining her first agile team in 2000, Lisa has helped many testers and their teams enjoy adopting agile principles and practices. She frequently leads tutorials, classes, and workshops on agile testing and is a regular contributor to *Better Software* magazine, *IEEE Software*, and *Methods and Tools*. Lisa co-authored with Tip House *Testing Extreme Programming*. Find out more about Lisa's work (and see pictures of her miniature donkey friends) at lisa.crispin.home.att.net.

STAR WEST SPEAKER

Esther Derby is a regular StickyMinds.com and *Better Software* magazine contributor and has presented at STAR*EAST* and the Better Software Conference & EXPO. Esther is one of the rare breed of consultants who blends the technical issues and managerial issues with the people-side issues. Recognized as one of the world's leaders in retrospective facilitation, she often receives requests to work with struggling teams. Esther is one of the founders of the AYE Conference. She is the co-author of *Agile Retrospectives: Making Good Teams Great*. You can read more of Esther's articles on the wonderful world of software at www.estherderby.com and on her Weblog at www.estherderby.com/weblog/blogger.html. Her email is derby@estherderby.com.

Linda G. Hayes is a founder of Worksoft, Inc., developer of next-generation test automation solutions. Linda is a frequent industry speaker and award-winning author on software quality. She has been named one of *Fortune* magazine's People to Watch and one of the Top 40 Under 40 by Dallas Business Journal. Linda is a regular columnist and contributor to StickyMinds.com and *Better Software* magazine, as well as a columnist for *Computerworld* and *Datamation*, author of *The Automated Testing Handbook*, and co-editor with Alka Jarvis of *Dare to be Excellent*. You can contact Linda at linda@worksoft.com.

Allen I. Holub has worked in the computer industry since 1979. Focusing on object-oriented technology and agile process, he excels at helping companies fix broken software development processes and adopt new ones by providing CEO/CTO coaching, staff training, project management, and programming services. Allen has authored nine books, most recently *Holub on Patterns*, and more than 150 magazine articles, writing as contributing editor for *Dr. Dobb's Journal*, *JavaWorld*, and *SD Times* among others. Allen has worked on projects ranging from operating systems and compilers to Ajax Web applications, and he teaches for the University of California, Berkeley, Extension. Allen is the security-track chair for the Software Development conference and sits on the board of advisors of Ascenium Corp. and Ontometrics Corp. Contact Allen at www.holub.com/allen.html.

Daniel Wellman is a technical lead at Cyrus Innovation (www.cyrusinnovation.com), a leading New York agile consultancy where he leads development projects and coaches teams on adopting agile software development practices. Daniel has more than ten years of experience building software systems and is an expert in agile methodologies, object-oriented design, and test-driven development. Contact Daniel at dan@danielwellman.com.

Ed Weller is an SEI certified High Maturity Appraiser for CMMI appraisals, with nearly forty years of experience in hardware and software engineering. Ed is the principal of Integrated Productivity Solutions, a consulting firm that focuses on providing solutions to companies seeking to improve their development productivity. A regular columnist on StickyMinds.com, Ed can be contacted at edwardwellerIII@msn.com.

STAR WEST SPEAKER

EDITOR'S PICK

Heather Shanholtzer
Editor-in-Chief,
Better Software magazine
hshanholtzer@sqe.com

Driven to Test

My mom loves to tell stories about my misadventures while learning to drive. Her favorite tale is about the time I tried to squeeze my Nissan Pulsar between a truck and a brick wall through an opening that obviously was not wide enough even for my sub-compact car. After I hit the corner of the wall (duh, I knew better than to aim for the truck), my mother just stared at me in disbelief. Then she made me get out of the car.

Fortunately for my mother's nerves, driver's education was an elective at my high school, so she left it up to the pros to whip my driving skills into shape. Before the students were even allowed to sit in the school's cars we endured hours of instructional videos, computer simulated driving situations, and filmstrips showing the aftermath of car wrecks. When we were finally let loose on the driving course, the experience had an "inmates' exercise hour in the yard" feel to it. The driving school was surrounded by twelve-foot-high fences, and the instructors monitored us from what can best be described as a prison-like watch tower. All of the cars were equipped with radios through which the instructors barked orders and admonished instruction non-followers: "Headlights on; left turn signal; I said *left*, car four; where's your seatbelt, car three? Car six, put that car in park and take your hands off the wheel *now*!"

Driver's education was intimidating (and humiliating), but it definitely taught me the rules of the road. And when I finally took my driving exam, I passed the first time. Mom was so proud—and relieved. I think I turned out to be a pretty good driver—despite what the guy a couple of offices down might tell you after I bumped his car while parking—but sometimes I still think back to the lessons I learned in driver's ed in order to make a safe driving decision.

Michael Bolton had a flashback to his days as a driving student during a recent discussion on the best methods to train new testers. In his June 2008 Test Connection, "Know Where Your Wheels Are," Michael writes about his experiences while learning to drive and explains how he applies lessons he learned from written rules, experiential learning, and the advice of mentors to teaching testers some valuable skills.

Find out how to incorporate Michael's driving lessons into training new testers on your team. Read "Know Where Your Wheels Are" at www.StickyMinds.com/editorspick10-8.

Quotables

"Your recommendation of a Test Manager's Vade Mecum should help present and future test managers get organized in a superior manner.

Whether the notebook is physical or electronic, it has helped me take notes that do come in handy when I need to plan upcoming tasks or simply recollect the exact items discussed in meetings/one-on-one discussions/conferences, etc. There is one thing I worry about, though, and that is losing that notebook. It happened to me once and I was able to locate it after a hectic one hour of searching the most likely places I could have left it. After this incident, I started using an electronic notebook and backing up the data in it regularly."
Inder P. Singh commenting on Fiona Charles's column, "The Test Manager's Vade Mecum"
www.stickyminds.com/quotables10-8a

"Despite popular opinion, simply attending meetings and transferring pressure does not constitute test management. I'm sure all of you know what I'm talking about, because we've all undoubtedly seen those managers that have become experts at *looking* busy and authoritative. They are always going to and coming from meetings, and when they receive pressure from the project managers, they simply transfer that pressure to the subordinate test engineers. When they get yelled at by their bosses, they transfer that aggression by yelling at their subordinate test engineers. This approach to management is not helping anyone, because nothing is really changing."
Dion Johnson, "Test Managers—Start Managing!"
www.stickyminds.com/quotables10-8b

Building An Effective **Test Automation Framework** That Works For You

In today's extremely competitive global economy, the high degree of customer interaction and increased system complexity magnifies the risk of software application failure. Testing plays a critical role in application development and maintenance. Clients look at test automation as a universal solution to reduce cost and effort but automation in itself cannot solve all the problems.

Cognizant advocates obtaining a clear understanding of the automation landscape and defining the objectives before commencing automation. This is a crucial step to delivering high returns on your automation. An automation assessment will establish consistency around your automation approach across different teams, reducing the development and maintenance cost and ensuring reusability. The assessment will also help you standardize your automation in terms of process, frameworks, tools, metrics, operating model, environment and test data management.

In order to realize significant returns, the automation suite needs to be reusable and robust. And having a partner who can help you accelerate the process will ensure rapid implementation than doing it alone.

Test Automation Assessment Categories

Cognizant categories testing automation assessments into different ways depending on IT maturity and business needs. Assessments can be broadly classified as:

Starter Kit Assessments

When you decide that test automation is much needed for your organization but does not know where to start, this assessment reviews your current maturity of testing processes by analyzing your readiness for automation.

Intermediate Assessments

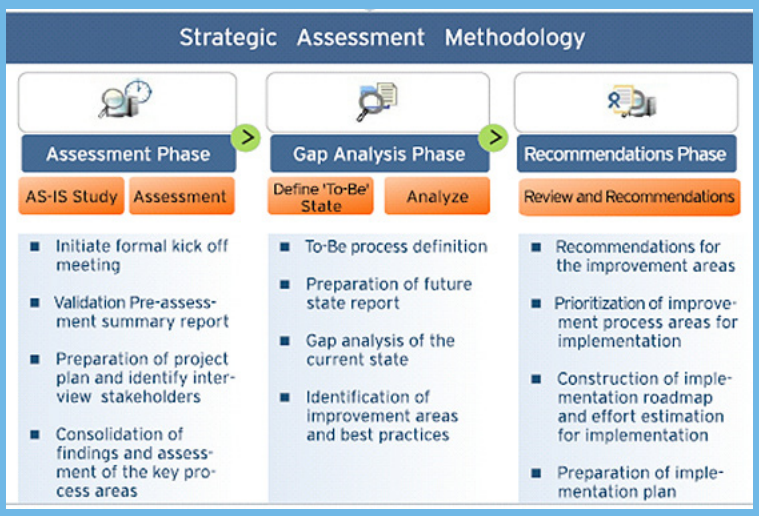
Cognizant assesses how your organization is creating, executing and maintaining the script base, and how your organization is using the tools at your disposal against its true capabilities: and also evaluates your capabilities against industry best practices.

Advanced Assessments

After your organization has relied on test automation for five-plus years, Cognizant evaluates the testing automation process to identify optimization opportunities in your end-to-end testing process. Cognizant not only evaluates the current automation framework, but also looks for opportunities to create value-added tools that bridge the gaps between what the tool vendor offers and established manual testing processes.

Strategic Assessment Methodology

An effective test automation process has to initially obtain the confidence of all your stakeholders involved, and be viewed as a strategic investment. Cognizant's Strategic Assessment Methodology is divided into the following phases:



In addition to an evaluation of the current automation practices, incorporating mechanisms to analyze existing manual processes for automation readiness will maximize automation benefits at an enterprise level. The various focus areas identified by Cognizant, based upon which an organization will be assessed across all related parameters, are depicted below:

Our Test Automation Expertise

Cognizant has developed an automation consulting framework to assess and provide a test automation implementation roadmap that benefits clients by increasing effort and cost savings, shorter cycle times and increased coverage. Cognizant enables clients to reach the defined future state by following a detailed test automation roadmap and tools strategy. Various focus areas covered in the automation consulting framework are organization, methodology, process, management, tools and infrastructure. Cognizant also provides proven recommendations covering scripting standards, test bed setup, test management tool integration, package applications automation and distributed test execution. Cognizant offers maximum transparency to clients by providing the ROI of the automation project using the enhanced in-house tools.

Cognizant is one of the largest testing service providers. We enable business agility by ensuring software quality. We help organizations achieve higher software performance, reliability and minimize the business risk through better stability of software applications while reducing overall cycle time and costs- ensuring a higher state of business readiness.

About Cognizant

Cognizant (NASDAQ: CTSH) is a leading provider of information technology, consulting and business process outsourcing services. With more than 40 global delivery centers and 58,000 employees our single minded passion is to work together with clients to make their businesses stronger. To learn more about Cognizant Testing Services, visit us at **http://www.cognizant.com/testing**

Cognizant | Testing Services
Passion for building stronger businesses

BOOK REVIEW

Software Testing and Continuous Quality Improvement, second edition

By: William E. Lewis

Reviewed By: Gene O'Flaherty

Software Testing and Continuous Quality Improvement contains an all-encompassing view of quality improvement. The author weaves his methods for continuous improvement throughout the book. This structure provides the reader with a solid understanding of how to determine the state of a test organization, how to identify problem areas, and how to work toward improvement. The book is organized in an excellent, logical fashion, beginning with helping the reader understand definitive quality and its varying measurement methods.

Author William E. Lewis reviews the traditional waterfall development methodology and provides a "phased" approach. He pays a significant amount of attention to all areas of test planning, from test group management to test project management and actual test activities. Lewis provides an extensive execution layout using the plan, do, check, and act (PDCA) approach as adopted from Deming. There are many metric points provided in all phases of test planning, execution, and reporting.

In the second half of the book, Lewis introduces a comprehensive analysis matrix including more than 170 various testing tools, how to evaluate tools for selection, planning purchases (e.g., proposals), and implementing the tools (training plans, use, and follow-up evaluation). Plenty of diagrams, charts, and checklists provide variety to this 500-plus-page volume. The tools used in the PDCA test methodology are also provided as templates on the bonus CD-ROM. Extensive appendices include templates for test activities (e.g., test plans, test cases, and summary reports), checklists for all phases of testing, and thorough explanations of various testing techniques.

Lewis does an excellent job presenting the material in an incremental manner. The content is extremely useful to test managers, testers, and anyone interested in learning more about quality and improving the testing process. The detailed checklists give testers and managers just one more method for improving the test process. I've been an SQA test manager and tester for more than ten years, and some of the checklists gave me an unexpected shot in the arm. This is one book on testing that provides the tools to conduct an in-depth analysis of the testing process.

Lewis hits the nail on the head by providing such a broad spectrum of areas where a quality-driven tester or manager needs to investigate and take action. I believe real-world examples would muddy the book's intent. Instead, the book provides the framework for you to think about what you can do to improve what is happening in your environment. This approach is refreshing and empowering. The style of the book allows the reader to quickly grasp the areas needed, but beware—you might find yourself sidetracked by all of the other useful topics!

I really enjoyed *Software Testing and Continuous Quality Improvement.* I liked the fresh content and look forward to applying the PDCA test methodology on my next project to experience it firsthand. You can bet that this book is on my bookcase with my name on it!

Visit www.StickyMinds.com/bythebook10-8 to post your comments on this book.

Books Guide

The StickyMinds.com Books Guide is one of the most popular areas on our Web site. With more than 880 books—including many that have been reviewed by thought leaders, industry experts, and your peers—the StickyMinds.com Books Guide is your first stop for finding a good read. Not sure what you're looking for? Browse books by topic, including:

- Project & Team Management
- Test & Evaluation
- Requirements
- Design & Architecture
- Development & Deployment
- Reviews
- Process Improvement
- Measurement & Reporting
- Security
- Defect Tracking
- Configuration Management

October Weekly Columns

Every Monday we bring you fresh ideas to jump-start your work week. This month we deliver thought-provoking articles from talented authors such as Johanna Rothman, Mary Gorman, Danny Faught, and Fiona Charles.

PowerPass Member Exclusive

Did you know that many of the expert speakers at the Software Quality Engineering conferences—STAR*EAST*, STAR*WEST*, Better Software, and Agile Development Practices—have also contributed content to *Better Software* magazine? Since this issue is being distributed to all STAR*WEST* 2008 conference attendees, we've compiled a list of feature articles written by some of the speakers at this year's conference. PowerPass members can access these articles and many more in the *Better Software* magazine archive. Not a PowerPass member? Go to www.stickyminds.com/getapowerpass to join.

James Bach
Inside the Mind of an Exploratory Tester
Renowned exploratory tester James Bach describes eight key skills that expert explorers possess and how you can develop them, too.
www.stickyminds.com/PPBach

Scott Barber
Hurry Up & Wait
There are no industry standards for Web response times. How long a user is willing to wait for a Web page to load depends on any number of variables and conditions. Find out how to determine and quantify performance criteria and use those criteria to create happy customers.
www.stickyminds.com/PPBarber

Rex Black
The Right Stuff
In the countdown to a product release, leaving load testing for last is a risky maneuver. In this cover story, Rex Black and Barton Layne present a case study of a project on which they adhered to a four-step strategy. By following these four steps, they were able to alleviate those nasty, end-of-project performance surprises.
www.stickyminds.com/PPBlack

Hans Buwalda
QA All-Stars
A testing team can mean success or failure for a project, but developing a team means more than putting a few people together and telling them to test something. Hans Buwalda shares his teambuilding experiences and gives some tips on how you can build the best team for the job.
www.stickyminds.com/PPBuwalda

Lisa Crispin
To Track or Not to Track
What's your take on the defect-tracking debate? Are defect logs helpful repositories of knowledge or cumbersome, inefficient inventories? Lisa Crispin took to the trenches to find out how different industry workers view defect-tracking systems. Find out what they had to say.
www.stickyminds.com/PPCrispin

John Fodeh
What's on Your Dashboard?
Just because a metric is easy to capture doesn't mean it is useful. The metrics that are really needed are the ones that can help you make good decisions. Find out how to establish a project dashboard with meaningful metrics that will guide your project safely to its destination without getting bogged down in an endless pursuit of unnecessary information.
www.stickyminds.com/PPFodeh

Fiona Charles
Bulking Up
Learn how strengthening people skills, such as teamwork and communication, can help shape you into a top-form tester.
www.stickyminds.com/PPCharles

Julian Harty
Know What's at Stake
Everyone knows the importance of well-defined functional requirements. We want our products to work, don't we? But how many of us are paying as much attention to defining our non-functional requirements? In this historically focused feature, we learn from past mistakes the potentially disastrous results of inadequately tested NFRs.
www.stickyminds.com/PPHarty

Jonathan Kohl
Man and Machine
Instead of viewing automation as an effort to replace manual tests, think of it as a means to extend the abilities of the tester. Combining the power of the human mind with automation tools helps fuel observation and discovery and provides a different perspective of the software under test.
www.stickyminds.com/PPKohl

WEB SEMINARS
by StickyMinds.com & Better Software magazine

StickyMinds.com & *Better Software* magazine Web Seminars: One Hour of Brain-Boosting Power

Performance testing, virtualization, quality assurance, and test-driven development are some of the topics experts in the software industry have covered in past StickyMinds.com and *Better Software* magazine Web seminars. If you missed a live presentation, check out the Web seminar archive on SQE.com today! www.stickyminds.com/Media/WebSeminar

A sampling of content from our eNewsletter archives

Better Software magazine and StickyMinds.com are pleased to offer readers free subscriptions to six eNewsletters delivered straight to your inbox. Whether you are looking for information about new content on StickyMinds.com, industry-related content from the newswires, handpicked content from StickyMinds.com, an interview with an expert on useful tools, first-person experiences from the agile development front, or what to expect in the next issue of *Better Software* magazine, our eNewsletters have something for everyone. Visit www.stickyminds.com/eLetter.asp to sign up today.

Sticky ToolLook: *June 13, 2006* www.stickyminds.com/eLetter10-8

A Word with the Wise: Choosing Performance Testing with Scott Barber

by Joseph McAllister

Every kid eventually puts some thought into the question "What do you want to be when you grow up?" For PerfTestPlus CTO Scott Barber, who specializes in context-driven performance testing and analysis for distributed multi-user systems, the answer was not "performance tester." He planned to follow in the footsteps of his father, an industrial arts teacher, and sought an ROTC- scholarship-funded degree in civil engineering. In his junior year of college, though, Scott learned that his first years with the Army Corps of Engineers would involve digging foxholes for infantry rather than building bridges with the Seabees.

"I decided that if I was going to be crossing the front lines, I'd much rather be carrying heavy weaponry than heavy shovels," he says.

He became a maintenance officer, with additional duties as an automation officer—essentially fixing computers. One day, while complaining about outdated software, he was serendipitously overheard by someone there to evaluate a replacement system and was recruited into a government contracting position. Later, while weighing his career options, a friend offered Scott a job as a performance engineer with a promise of "Don't worry, you'll like it."

"From that day, I've been doing performance testing," Scott says. "What I love about it is it spans every part of the software lifecycle. It dials into every discipline from development to business analysis."

Load-generation tools, which most people call performance-testing tools, are just one half of the equation, Scott says. The collected data often also requires automated analysis or "performance monitoring."

"As far as buying a tool and executing all your tests in house versus outsourcing, basically the first heuristic is this: If you've got one big performance test, the first one is going to be cheaper to outsource," Scott says. "Your break-even point, depending on what tool you choose, what your in-house resources are, and the price of a performance tester in your market, is somewhere between two and three performance-testing projects if you're in a software development shop. That's a huge ballpark."

Regular performance testing may call for an in-house process, while testing just once or once in a while may be better served through outsourcing, according to Scott.

If you choose in-house testing, Scott has a few recommendations for evaluating potential tools. Foremost is that you demo the tool on your application, particularly if your product is unusual or new technology.

"Virtually every tool out there does a good job against the type of application it was originally designed to test, but we develop with cutting-edge technology," he says. "Make sure that the tool that you're getting supports your application."

Also, don't pay for unnecessary features and support for communications protocols that you'll never use. Finally, be sure that your performance tester can, as Scott says, "make that tool sing"—let the tester guide the tool-evaluation process.

Of course, as with most software tools, one must also contemplate commercial or open source. The big differences are that each open source tool only works against a limited number of applications or communications protocols, and they are limited in terms of what Scott calls the "nice-to-haves"—extras or add-on features.

"When it comes to actually generating load, the difference in accuracy and in response time from enterprise pay tools and free open source—that's not the point to compare," he says. "Obviously some free and open source tools are going to be better than others. Some enterprise tools are better than others for your particular application. At that point, accuracy of load and response time is something that you need to evaluate against your particular application, as opposed to generally across whether you're paying for a tool or not."

Scott sees brighter days ahead for the field of performance testing. He believes the market no longer views performance testing as a nice-to-have option, but rather as a necessity. The industry, he says, "is finally starting to accept as a whole that performance testing isn't just functional testing on steroids or in 3-D. It is a unique set of skills."

LET'S TALK ABOUT IT

Members of StickyMinds.com use the Discussion Boards to seek out answers to troubling questions, hash out problems, and tap into the minds of other Discussion Board residents. The following are real questions posted to the StickyMinds.com Discussion Boards. Got an answer? Log on to StickyMinds.com and share your ideas today.

I am looking for a tool that can do two things: It needs to have some record functionality or something similar so that I can pass through two login checks. To be more specific: One is a normal site login with username and password required. The next does not require a login, only a username. From here, the system knows to grant access only to that user based on his/her privileges.

And, secondly, I need to be able to put a load on the system with multiple logins—as stated above—with multiple different users to the same site.

Anyone have any suggestions?

Posted by: Chuck

www.stickyminds.com/10-8DiscussA

Can anyone give me a name or two of good open source tools for test case design and execution?

Posted by: szohdi

www.stickyminds.com/10-8DiscussB

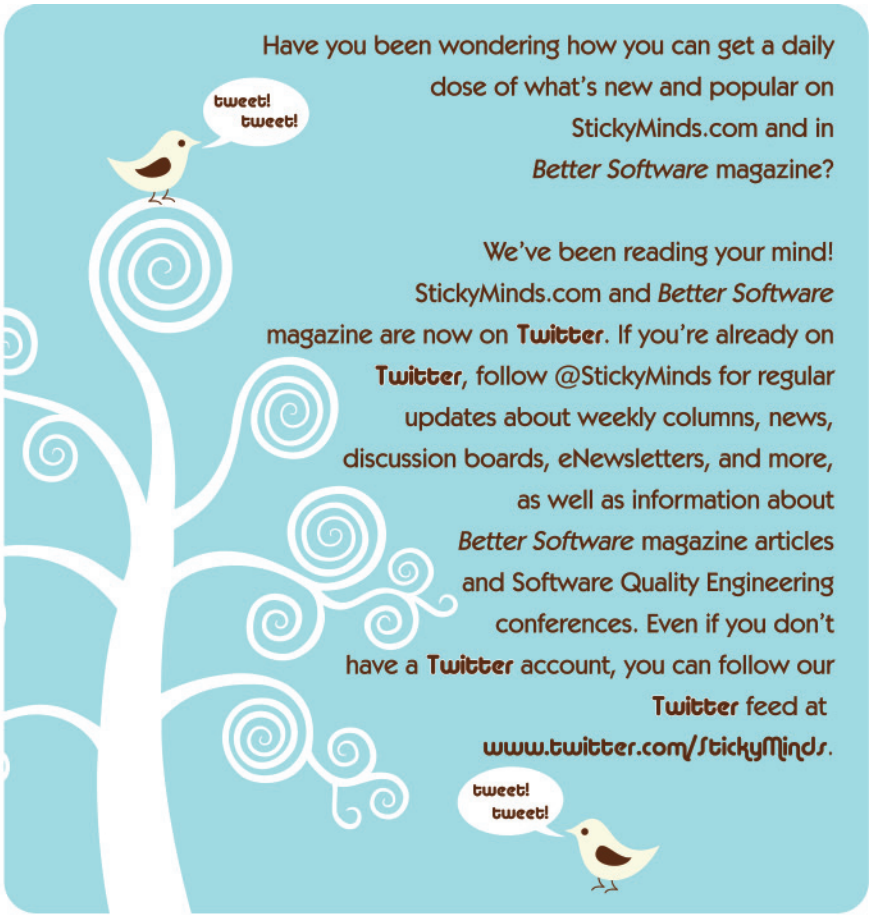
If you have insight or advice to share, log on to the StickyMinds.com Discussion Boards or email us at editors@bettersoftware.com. Select responses will be published in an upcoming issue of *Better Software* magazine.

We are using QTP for UI automation for our products. At high level, we were discussing whether to have in-house framework on top of QTP to meet the specific requirements of our product automation like we had implemented/used keyword-driven framework using winrunner. I would like to know:

1. Do we really need to have framework over QTP, as QTP itself provides a framework, sort of?
2. If you feel it makes sense to have framework to suit certain specific requirements or minimization of execution time of RTs, please share your thoughts and any already used frameworks (keyword driven) already in place in industry over QTP.

Posted by: rsubbu78

www.stickyminds.com/10-8DiscussC

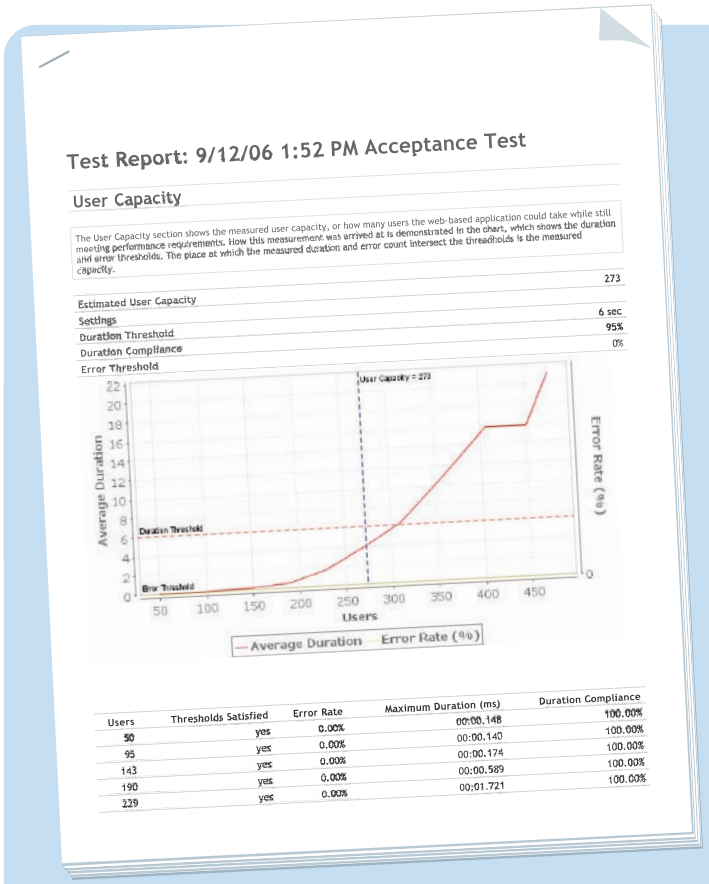


To load test your website, you could type this:

```
Definitions
  ! Standard Defines
  Include "RESPONSE_CODES.INC"  Include "GLOBAL_VARIABLES.INC"  CHARACTER*512 USER_AGENT  Integer
  USE_PAGE_TIMERS  CHARACTER*256 MESSAGE  Timer T_OBFUSCATED  CHARACTER*1024 cookie_2_0  CHARACTER*1024
  cookie_2_1  CHARACTER*1024 cookie_15_0  CONSTANT DEFAULT_HEADERS = "Host: " + domain + ".com^J" &
  "Accept-Encoding: gzip, deflate^J" & "User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0;
  .NET CLR 1.1.4322; .NET CLR 2.0.50727)"  CONSTANT S_cookie_1_0 = "B=1p1c30p38in6q&b=3&s=7m"  CONSTANT
  S_cookie_1_1 = "YLS=v=1&p=0&n=1"  CONSTANT S_cookie_1_2 = & "F=a=60rfgt4MvT9diVoFZzhX7WriqnlT2n2489Sx"
  CONSTANT S_cookie_1_3 = "PH=fn=fnP28mrdGyGdwWauu9qREw--&l=en-US"  CONSTANT S_cookie_1_4
  CONSTANT S_cookie_1_5 = "U=mt=dsuqeJ2MhYrp3Y7EjkT4qCUZIuWDW.w4_.E&ux=JHnhHB&un=24a238gchl7lv"
Code
  !Read in the default browser user agent field
  Entry[USER_AGENT,USE_PAGE_TIMERS]  Start Timer T_OBFUSCATED PRIMARY GET URI "http://" +
  ".com/ HTTP/1.1" ON 1 & HEADER DEFAULT_HEADERS & WITH {"Accept: image/gif, image/x-xbitmap,
  image/jpeg, image/pjpeg, " & "application/x-shockwave-flash, application/msword, */*", &
  "Accept-Language: en-us", & "Connection: Keep-Alive", & "Cookie: "+S_cookie_1_0+";
  "+S_cookie_1_1+"; "+S_cookie_1_2+"; "+S_cookie_1_3+";"  DISCONNECT FROM 1  WAIT 391
```

or this:

www.webperformanceinc.com



WEB PERFORMANCE LOAD TESTING

Why code every test case by hand, when our unique software detects and automatically configures the test cases for you – quickly and accurately, then gives you superior reports that are easy to understand? With Web Performance automatic load testing, the time and money you save could increase productivity as much as 500 percent.

For more information about how you can increase performance and productivity using Web Performance automated load testing, visit **www.webperformanceinc.com**

Keynotes by International Experts

Brian Marick

Seven Years Later: What the Agile Manifesto Left Out

Dan North

Beyond Best Practices: Keeping Agile Agile

Pollyanna Pixton

Collaborative Leadership: A Secret to Agile Success

David Anderson

Scaling Agile: Kanban and Beyond...

Conference Sponsor:

RALLY SOFTWARE

100% of 2007 attendees* recommend the Agile Development Practices conference to others in the industry

SOFTWARE QUALITY ENGINEERING CONFERENCES
www.sqe.com

*Based on the 2007 conference attendee evaluations

BETTER SOFTWARE

Agile Development Practices

November 10–14, 2008

Orlando, FL • Shingle Creek Resort

36 Pre-conference Tutorials — *Monday and Tuesday*

46 Keynotes & Classes and Networking EXPO — *Wednesday and Thursday*

Agile Leadership Summit — *Friday*

Agile Leadership Summit: An Executive View of Where Agile Is Heading
Co-located Event — November 14, 2008

Agile LEADERSHIP SUMMIT by APLN

Shingle Creek Resort

Register Early and Save $200!

www.sqe.com/agiledevpractices

A Gram of Prevention

by Chuck Allison

At the June 2008 Better Software Conference & EXPO in Las Vegas, I presented a tutorial on the principles and patterns of software design. Much of the discussion focused on how understanding and adhering to sound design principles lead to better code. Most design principles are based, it seems, on the notion of *separation*: separating things that change from things that don't in a given context, separating interface from implementation, separating object creation from object use, etc. While studying the virtues of separating command invocation from command internals, a tester in the audience observed that the Command pattern makes program functionality easier to test since functions can be tested independently of their calling contexts. Being somewhat of a focused (read *myopic*) developer, I never considered how design patterns could improve code testability.

Similarly, it seems that thinking before coding is also a Good Idea. Back when the turnaround time of program builds was much longer and more expensive than it is now, developers had to be very careful crafting code, or else they quickly ran out of time or budget or both. While modern IDEs and agile processes encourage a healthy project velocity, they seem to have begotten a generation of quick-draw programmers who are all too willing to let their tools do the programming. But following an *I-click-therefore-I-program* methodology does not lead to quality software. Agile methodologies, in part a reaction to the extremism of Big Up-Front Design, may not only have helped small-to-medium-sized projects break free from analysis paralysis but also may have encouraged a knee-jerk abandonment of almost *all* up-front design. Consequently, the sorry state of software quality throughout the industry has seen little measurable improvement [1,2].

Famed software visionary Grady Booch recently commented on how fast-and-loose code slinging compromises architectural integrity:

> In other disciplines, engineering in particular, there exist treatises on architecture. This is not the current case in software, which has evolved organically over only the past few decades. All software-intensive systems have an architecture, but most of the time it's accidental, not intentional. This has led to the condition of most software programming knowledge being tribal and existing more in the heads of its programmers than in some reference manual or publicly available resource … If I don't have a sense of the architecture, and I keep piling on code, it becomes a fetid mess [3].

> "Good code can and should evolve from clear, up-front descriptions of the solution to the problem at hand."

Some up-front design must be performed. Due diligence early on finesses much of the needless complexity that inevitably ensues with unarchitected code. According to programming pioneer Per Brinch Hansen, "Once you appreciate the value of description as an early warning signal of unnecessary complexity, it becomes self-evident that program structures should be described (without detail) before they are built and should be described by the designer (and nobody else). Programming is the art of writing essays in crystal clear prose and making them executable [4]."

If we can't manage our plans and descriptions, we won't be able to keep the code that follows under control.

This is all very old news, by the way, as you can see by the date of the previous quote. In his Turing Award Lecture a few years earlier, the legendary Edsger Dijkstra asserted that:

> Those who want really reliable software will discover that they must find means of avoiding the majority of bugs to start with, and as a result the programming process will become cheaper. If you want more effective programmers, you will discover that they should not waste their time debugging; they should not introduce the bugs to start with [5].

When I read this to an audience of testers at STAR*WEST* 2001, I had to wait for the laughter to subside before I continued, but Dijkstra knew what he was talking about. It pays to put in the effort to develop an architecture, to design for maximal cohesion and minimal coupling, to use or develop languages close to the problem domain, to create abstractions according to sound design principles—the list goes on. Testers will not lose their jobs if developers can deliver higher quality to begin with. But software just might be raised from its dismal state. **{end}**

References:

1] "The Economic Impacts of Inadequate Infrastructure for Software Testing." NIST Report, May 2002. www.nist.gov/director/prog-ofc/report02-3.pdf.

2] Frye,Colleen. "The State of Software Quality, Part 1." Software Quality News, February 16, 2007. searchsoftwarequality.techtarget.com/news/article/0,289142,sid92_gci1243311,00.html.

3] Booch, Grady. "Software's Dirty Little Secret." Scientific American, June 17, 2008. www.sciam.com/article.cfm?id=softwares-dirty-little-secret.

4] Brinch Hansen, Per. *The Architecture of Concurrent Programs*, Prentice-Hall, 1977.

5] Dijkstra, Edsger. "The Humble Programmer." ACM Turing Award Lecture, 1972.

Passing the Buck

by Allen I. Holub

ISTOCKPHOTO

Production code should be as easy to maintain as possible. It spends at least 80 percent of its lifecycle in maintenance—even more in the case of big enterprise applications. An extra ten minutes spent now can save months in the long term. The "long term," by the way, starts about ten minutes from now. A line of code is in maintenance the first time you have to look at it again while you're tracking down a bug.

One way that object-oriented systems address the maintenance problem is by using "implementation hiding" (aka "data abstraction.") Clients of an object shouldn't be dependent on its inner workings—they should only have to know how to talk to it. The acid test is to completely rewrite an existing class definition, throwing away all the method bodies or adding brand new behavior, without the client's noticing that a change has occurred. If you can't do that, a simple change to an object could ripple out into the rest of the system and wreak real havoc.

The basic rule for defining good interfaces is to have clients *ask for help, not information*. Or, to be more precise (and long winded): *Don't ask for the information that you need to do the work; ask the object that has the information to do the work for you*.

Consider the `Money` class in listing 1. There are a lot of problems with this class. The first is that it uses a floating-point number to represent a monetary value. If you think of a number line where each possible value that can be represented by a float is a dot on the line (there are a finite number of them), then the dots are very closely clustered around 0, and they spread out as you move away from 0. Like this:

```
                    0
.   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
```

If you don't believe me, run the following code, which prints "They're the same!"

```
Float big_value = Float.MAX_VALUE / 4.0F ;
if( big_value == big_value + 1 )
      System.out.println("They're the same!");
```

Some numbers just can't be represented by a float or a double, so don't use them if you really want to represent every possible value of a pile of money accurately.

```
class Money
{
      private float value;
      public float getValue() { return value; }
      public float setValue( double newValue ){ value = newValue; }

      public String toString()
      {
            // Convert to string (as if the DecimalFormat didn't know
            // about currency, for the sake of the example).

            DecimalFormat formatter = new DecimalFormat();
            formatter.setMaximumFractionDigits(2);
            formatter.setDecimalSeparatorAlwaysShown(true);
            return formatter.format( value );
       }
}
```

Listing 1

If you don't think that this problem has any real-world ramifications, consider writing code for a bank in Ecuador, where there are 29,944.00 Ecuadorean sucres to the US dollar. You can get into the realm of missing numbers pretty quickly with large transactions unless you scale, but if you do that, your code will either be full of special cases or won't be useful out of the box with standard currencies.

Let's fix the problem by storing the value as a long value, scaled by 100 to manage cents. Listing 2 shows a new version of the class.

But wait a minute! That change hasn't actually solved the problem; we're still converting our accurate representation into an inaccurate float when we call `getValue()` (and then returning that inaccurate value to someone who will, undoubtedly, perform inaccurate arithmetic on it). We're still in trouble.

```
class Money
{
      private long value;

      public float getValue() { return (float)(value/100.0); }

      public void      setValue( double newValue )
      {     value = (Math.round( newValue ) );
            newValue -= (float)value;
            value = (value * 100) +
                  (long)( Math.floor(newValue * 100.0) );
      }

      public String toString()
      {
            long integerPart = value / 100;
            return integerPart + "." + (value - (integerPart * 100));
      }
}
```

Listing 2

```
class Money
{
      private long     value;

      //...

      private Money(long v){ value = v; }     // for internal use only

      public Money plus (Money operand){
            return new Money(value + operand.value);
      }
      public Money minus(Money operand){
            return new Money(value - operand.value);
      }
      public Money assignFrom (Money operand){
            value = operand.value;
            return this;
      }
      public void increaseBy (Money operand){
            value += operand.value;
      }
      public void decreaseBy (Money operand){
            value -= operand.value;
      }
}
```

Listing 3

The only solution to this problem is to do what we should have done to begin with—hide all information about how the `Money` class works. The `getValue()` method has to go. It gives away too much information about how the value is stored internally. If the interface has to change so that `getValue()` returns something other than a long value, then all of the client code that calls `getValue()` will have to change, as well.

In any event, there's actually no reason to get the value. But what about all that code that used to use the value? The answer is the rule at the top of this article: "Ask for help, not information." The `Money` class itself should be doing all of the operations that used to be done externally.

Let's assume that `Money` operations are limited to addition, subtraction, and assignment. You can fix the class with the additions in listing 3.

Now, instead of saying:

```
Money x;
Money y;

x.setValue(
        x.getValue() +
        y.getValue() );
```

you can say:

```
x.increaseBy( y );
```

As an added benefit, the new code is both shorter and easier to read.

Note that all the operands to the new methods are *`Money` object*, not floats. The only place where a float still exists is in the original `setValue(float)` method, which may need to interface to legacy code or initialize a `Money` object from the result of database query or equivalent. Frankly, I'd probably eliminate that method, too, if I could get away with it.

Now comes the hard part. We have to search the code to find every place that `getValue()` used to be called and modify the surrounding context to use the new methods. This change is incredibly painful, and the scope of the change could be way larger than a single program if the `Money` class were in a library used by several programs. Of course, if we had designed the interface to `Money` the right way to begin with, without `getValue()`, the change of internal representation from a float to a scaled long would be completely transparent. All existing code would already be using the new methods, so no changes to the client code would be required.

Now that we've fixed the interface to `Money`, let's see if it's a reasonable interface by adding the notion of currency to the

class. If you think about the original getter/setter version of `Money`, adding support for multiple currencies would be incredibly difficult.

We could add a `Currency` class and `setCurrency()` and `getCurrency()` methods of course, but then what? The old (and not very pretty) code that added two values would have to change from this:

```
x.setValue( x.getValue() + y.getValue() );
```

to this:

```
x.setValue( x.getValue()
+ (y.getValue() * x.getCurrency().
                    convertTo( x.getCurrency() )) );
```

which is downright hideous. These aren't the sorts of changes that refactoring tools will do for you. In the new version of our class, however, the actual currency value can be a member of the class, and all the currency-related arithmetic can be hidden inside the arithmetic methods like `increaseBy()`. Consequently, existing code that added two units of `Money` remains unchanged, whether or not the `Money` objects were in the same currency. Voila!

The main clue that something was wrong with our earlier class was the existence of the accessor and mutator (`get/set`) methods. Though there's nothing inherently wrong with the words "get" and "set," it's seriously wrong—at least from an implementation-hiding perspective—for a `get` or `set` method to provide access to a field that might change if the implementation changes. Well-designed object-oriented systems simply don't do that. Since it's usually difficult to predict the future, I generally avoid accessors and mutators entirely, just so these problems will never occur.

The whole point of an object-oriented system is to improve maintenance, and the way to do that is to encapsulate your implementation so well that you can change the implementation without impacting any of the code that uses your classes. Consequently, you should avoid any code that gives away something about how a class works—and "getter/setter" functions often do just that. The best way to eliminate methods like these is to ask "What does this object do?" rather than "What does this object contain?" and then provide methods that do that work rather than provide information that someone else uses to do the work. **{end}**

Though getters and setters are responsible for significant maintenance headaches, they're not the only problem. Of the idioms that you've seen in code that you've worked on, which ones caused you the most grief?

Follow the link on the StickyMinds.com homepage to join the conversation.

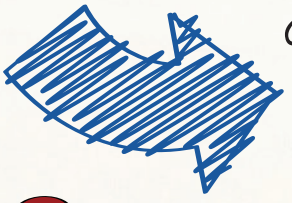
RALLY SOFTWARE

The Award-Winning-Est Agile Lifecycle Management Solution

Three-time Jolt Product Excellence award winner in 2006, 2007 and 2008 for project management tools

QSMA concludes—
"As compared to industry averages, Rally customers are 50% faster delivering their software to market and increased their teams' productivity by 25% while maintaining normal defect counts."

Forrester Research says—
"Rally designed the requirements management capabilities in Rally Enterprise, a software-as-a-service (SaaS) ALM solution, to suit teams using Agile processes. The product performs flawlessly in this regard..."



Get your FREE trial of Rally Enterprise at
www.rallydev.com/adp
No download, no installation, no commitment

Visit us at the Agile Development Practices Conference, booth #200
November 10 - 14, Orlando, FL

Got You Covered

by Michael Bolton

ISTOCKPHOTO

How much of the product have you tested? How completely have you tested it?

These are innocent little questions—the kinds of questions that a client might ask about *test coverage*. Their simplicity hides an enormous amount of complexity that clients and many testers might not understand. Answering these questions can be surprisingly difficult and can lead us and our clients into several traps.

Suppose that we were to define *complete testing* as "all possible tests having been perfectly performed." Most people would quickly recognize that this goal is impossible to achieve in anything less than an infinite amount of time, so when reasonable people ask, "Have you tested this product completely?" or "How much of it have you tested?" they *must* mean something else. They might mean "Have you identified all of the important risks that we've anticipated?" But they might also mean "Have you discovered important problems that we *didn't* anticipate?" They might mean "Have you tested on a wide variety of platforms?" or "Have you tested each element of the user interface?" Perhaps they simply mean to ask questions about the quantity or quality of our testing: "What has—and hasn't—been tested?"

In his book *Software Testing Techniques*, Boris Beizer defines coverage as "any metric of test completeness with respect to a test selection criterion." This makes some kind of sense when our test selection criterion is quantifiable: If there are 100,000 lines of code and we've only tested 80,000 of them, it follows that there is code that we haven't covered. In order to detect uncovered code, we could enlist the aid of code coverage tools. Such tools tell us about the statements or branches that have been touched (or not touched) during testing. For the code that we haven't touched, the tool tells us that we haven't touched it, and that might be important. Some people might believe that if we touch a line of code once, with a particular data value, it's been "covered," and in one sense it has. But coverage tools don't (and can't) tell us what we've looked for, how carefully we've looked, what we've observed, and what we've missed. We could perform tests that verify that a given function was performed, that a particular path was executed, that certain data values were entered, and that an expected result was returned, but while doing that, we could easily miss problems related to other data values, usability, robustness, security, performance, platform variations, timing, and so forth.

Yet there are other problems here, too. Code coverage tools can't read our minds and can't comprehend the intention of the product. Tools can't decide whether the code we're testing performs a function that we value, and tools can't recognize that a potentially valuable function is missing. Moreover, our source code doesn't give us the whole picture, because the program that we're producing is rarely part of a simple system. Our software tends to interact with other software and hardware in systems that are increasingly beyond our capacity to comprehend. In a presentation at STAR*WEST* 2007, Lee Copeland raised the problem that we have no clue of how to define coverage that includes behavior that emerges from complex systems given our limited understanding of them. This problem inevitably presents itself when our product interacts with the rest of a system—an operating system, third-party libraries, interconnected machines on a network. For such systems, we can't know about the completeness of the coverage we're getting because we can't comprehend what 100 percent is. We could do that for a closed system, but when software interacts with anything outside of itself, the system isn't closed any more.

There is a way out of the trap, though: We can model. A model is some idea, activity, or object that represents (literally, "re-presents") something—another idea, activity, or object—such that understanding something about the model may help us understand or manipulate the thing that it represents. Models hide or ignore certain kinds of information in order to help us focus on information that we care about. In his lengthy and vigorous (and quite amusing) attack on using lines of code as a metric, Beizer points out many valuable ways to model the code and its structure. For example, we could graph the control-flow, state-related, or logical structures in the program and then execute the program to cover the graphs—a much more robust metric of test completeness in terms of code coverage.

Yet the source code itself is a model of a computer program, because a computer program isn't just code. As Cem Kaner suggests, a program is not merely "a set of instructions for a computer." Instead, he says, a computer program is "a communication among several humans and computers who are separated over space and time that contains instructions that can be run by a com-

puter" and adds that "the point of the program is to provide value to stakeholders." That's why *code* coverage isn't the same thing as *test* coverage. Test execution involves configuring, operating, observing, and evaluating the product in some way, and operating the product always produces some code coverage. But our clients don't value the code as such; they value the things that the code does for them. That's what they're really asking when they ask about "how much of the product we've tested." The value of our work comes from describing what we've observed and evaluated, so if we want to understand, explain, discuss, or improve our coverage, we need a rich set of models.

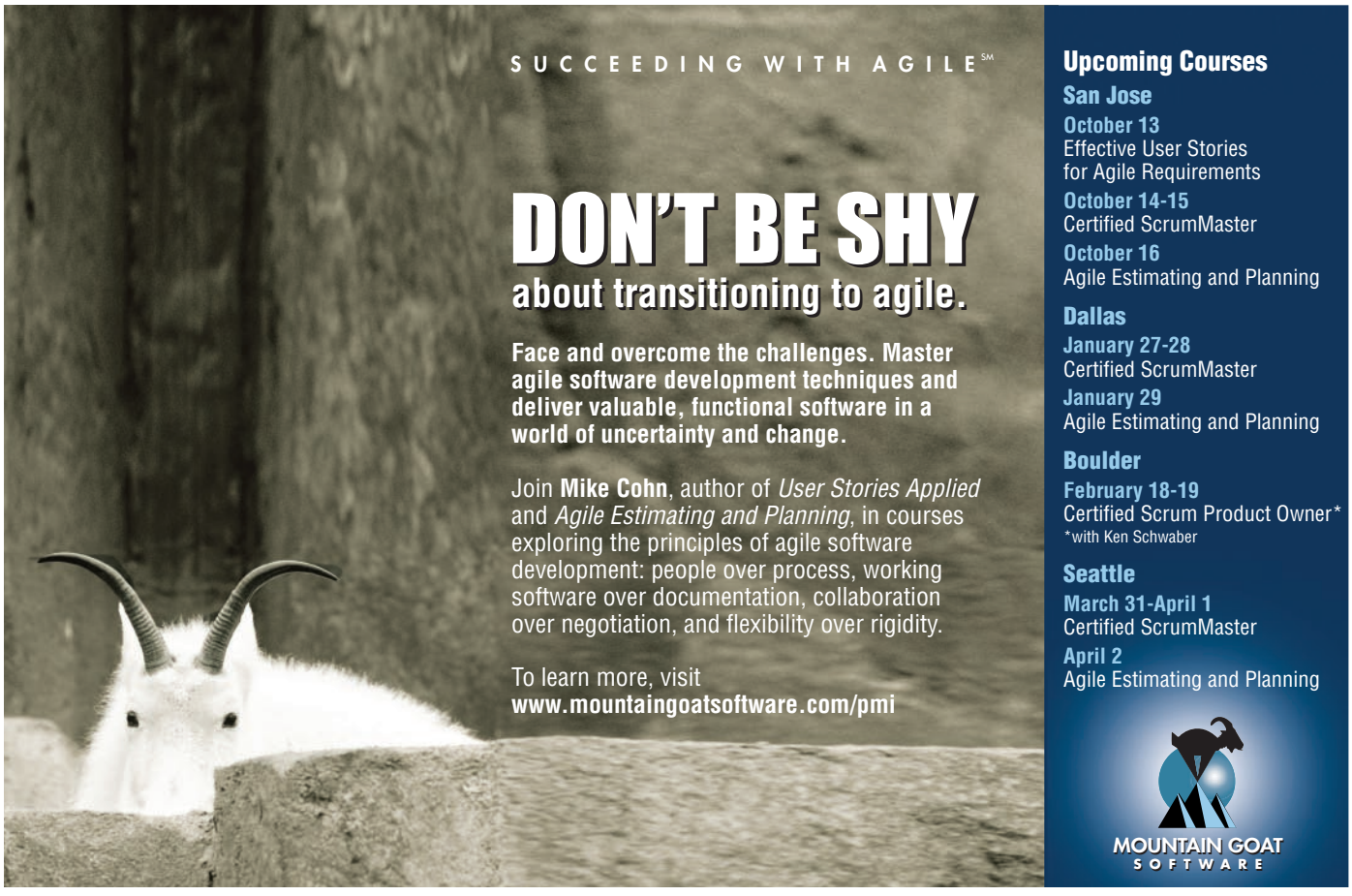
The first step when we're seeking to evaluate or enhance the quality of our test coverage is to ask who is interested and what they're interested in—that is, who and what we're determining coverage *for*. Excellent testing starts by first questioning the mission. The higher-level we go at first, the more context we have for identifying things that might be valuable to observe. So think "higher," or "hire"—*what have we been hired for?* Who is our client? Who are the people who will use the product, directly or indirectly? Who are the other stakeholders? What is our mission for this particular cycle of testing? What is the overall mission for the project and the product? What do people say about it? What is its history? Are there comparable products available—competitive products, or past versions of this product? These questions can help us set context and focus our attention on things that are known to matter, while also helping us recognize things that haven't yet been anticipated as being important. As Donald Rumsfeld said, "There are known unknowns, and there are unknown unknowns." One key objective of testing is to move things that we don't know in the direction of things that we do know. Known unknowns hide risk. But the biggest risks may be the ones that we haven't thought of—the unknown unknowns.

One of the most effective and efficient ways that I know to start addressing the unknowns is to learn about the knowns. Get close to your clients and start asking questions. Begin with, "May I ask questions?" If the answer is yes, then ask away. If the answer is no, then it might be useful and important to outline the things you don't know about and the risks associated with proceeding with insufficient information. Jot down some notes about that, and don't forget to include a date and time.

If the client is willing to answer questions but isn't clear on what to cover, then you can provide some suggestions as to what you might look for, and ask if they'd be relevant. If you're inclined to start by thinking about verifying functional correctness, that's OK, but you will soon want to broaden your models to think about other kinds of problems and risks, too—and how your tests will cover them. I'll talk about how to do that in the next column. {end}

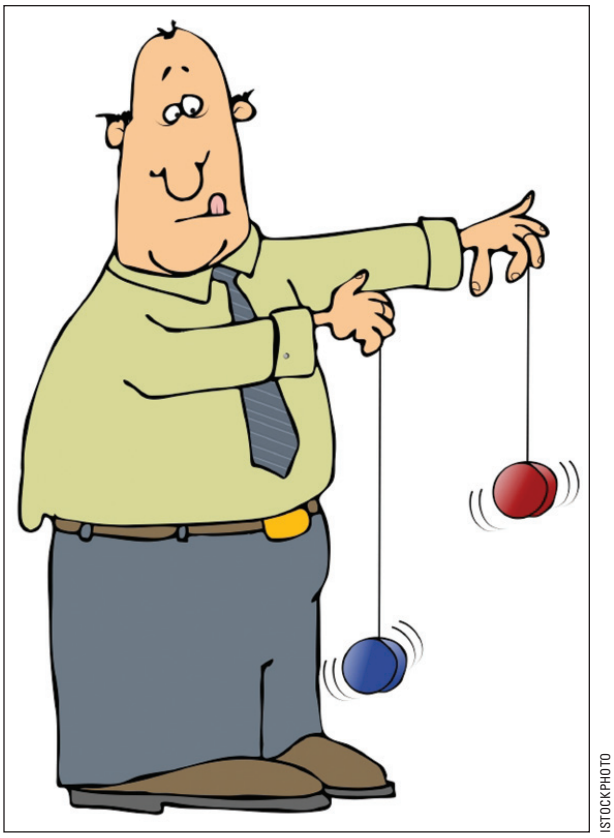
How do you start thinking about coverage? Where does that take you?

Follow the link on the StickyMinds.com homepage to join the conversation.



Tale of a Yo-Yo Manager

by Esther Derby



ISTOCKPHOTO

"I want you to be empowered," Bob told his team at the team meeting. But several weeks later, Bob expressed his disappointment to his colleague Sue. "My team members have a victim mentality," he said. "They're too passive. They wait for the team lead to tell them what to do. And the team lead checks everything with me. I want these people to show some initiative!"

Sue cocked her head. "What have you seen that makes you say that?"

"You remember the middleware bug we had last month?"

Sue nodded.

"Well, the team came up with a solution that was all wrong," Bob said. "It's a good thing I caught it before they forged ahead. If I hadn't wandered into the team room and seen the diagrams on the whiteboard, I never would have known about it."

Sue raised an eyebrow.

"I called the team into a huddle and explained why that idea wouldn't work. Then I had a meeting with the team lead. I really dressed him down. He should be leading them—not letting them make stupid mistakes."

Sue's other eyebrow went up.

"I want the team to dig in and solve problems," Bob said. "So last week at the team meeting, I asked everyone to offer at least one idea about how we can meet the deadline for our next release without dropping any features.

"One of the team members actually said it was impossible. Another one suggested we could implement a partial solution that included some manual work in the customer care department. When I told her that she hadn't examined all the issues with her approach, she backed down! How can we come up with creative solutions if we can't argue our ideas?

"Now they just sit there. It's like they are afraid to make any decisions at all."

Sue held up her hand to stop him.

"Bob," Sue said, as gently as she could, "do you think you might be sending mixed messages?"

Bob looked puzzled. "What do you mean?"

"I wonder how your team members felt when you overrode their technical design."

"I saved them from a big mistake," Bob said. "I'd think they'd be grateful."

Sue shrugged. "Maybe. And I bet they also felt like you told them the decision was theirs to make, and when they didn't do what you wanted, you took the decision back."

"But what they were planning was wrong!" Bob exclaimed.

"That may be true, but that doesn't change the way people feel. Do you think the team would have figured out the problems you saw before actually deploying the software?"

"Probably," Bob conceded. "But it wouldn't have been efficient."

Sue nodded. "No, it wouldn't have been very efficient. But it might have been more effective in helping team members learn how to think through the issues together. They are new to this and still need to learn through experience.

"Do you remember when you were learning how to drive?" Sue asked.

"Sure. My dad took me to a big, open parking lot and gave me instructions: Ease up on the clutch, press the gas pedal gently. After I mastered the clutch, he took me out on a quiet road."

"And after you received your license, did he let you drive anywhere, anytime, with no restrictions?" Sue asked.

"Of course not. At first I was allowed to drive to school alone, and later I was allowed to drive after dark. It was months before my dad let me drive with my friends in the car. At the time I hated it, but looking back, he was helping me build my driving skills before he turned me loose."

"Exactly," Sue said. "And that's what you didn't do with your team. In essence, you told your team members they could drive anywhere. Then, at the first small mistake, you pulled their license and showed you didn't trust them."

"Oh," Bob said. "I never thought about it that way. You mean I've actually brought about exactly what I didn't want? I'm making the team more passive instead of more empowered?"

"The key to empowerment is to tell team members what the boundaries are before they cross one by accident," Sue said.

"Otherwise, they'll feel jerked around," Bob said.

Sue nodded. "If you really want to empower your team members, you need to consider their skills *and* how much you trust them. Find the boundaries where you feel comfortable with the risks and give them free reign within that area. That's probably not the empty parking lot where your dad started your driving lessons, and it's probably not the freeway during rush hour, either."

STORY LINES

- **Clear decision boundaries help teams know when they can act on their own.**
- **Yo-Yo managers send mixed messages to the team and quash initiative.**
- **Analyze the skills of the team and the risk involved in decisions to create workable decision boundaries.**

"I have to work harder at this empowerment thing than I thought," Bob said.

"There's more to it than declaring, 'You're empowered,'" Sue agreed. "I think about the three Ws of empowerment: what, when, and why.

"*What* delineates the decisions and the standards that apply when making those decisions.

"*When* defines boundary conditions. For example, my test team can make decisions about test approaches where there isn't an existing test bed and when the tools team members need are open source or already available within the company. If they want to try a testing approach that requires a new tool or replaces a functioning test framework, then they come to me with a recommendation.

"*Why* sets the context. I find that when people see how their decisions fit into the value stream and affect customers and the bottom line, they make better decisions."

"I can't just go in and tell them what the boundaries are now," Bob said. "They won't believe me."

"You're right, Bob. First, you need to rebuild trust, and the best way I know to start that process is to admit you've made a mistake."

Bob gave Sue a pained look.

The very next day, Bob called a team meeting. "I've made a big mistake," he said. "I asked you to make decisions and offer ideas, and then I stepped in and took over. I was wrong to do that. I was worried about looking bad, and in the end, my actions made us all look bad."

Several team members looked down at the table. The team lead squirmed in his chair.

"Are you willing to try again?" Bob asked. "I have some ideas on how we can all be clear on which decisions are yours and which decisions are mine."

Tentatively, the team agreed. Together, Bob and his team created a grid that listed decisions and boundaries.

Bob realized that he wasn't ready to let go of every decision. He specified some areas where he would make the decision based on the team members' input and some cases where he wanted to review their decision before they implemented it. He was surprised to see that team members were grateful that they wouldn't have to carry the weight of all the decisions.

Bob asked the team to remind him if he started stepping over the boundaries he'd set.

It took some time for the team to trust him again, but eventually the team saw that Bob really had turned over a new leaf. Bob realized that the team was gaining skill and perspective in making decisions. They weren't always the decisions Bob would have made, but he had to admit the team's ideas and choices worked out reasonably well—most of the time. And when the decisions didn't work out as hoped, the entire team held a decision retrospective to learn from its mistakes.

Several months later, Bob went out to lunch with Sue. "The team is making better decisions and seems to have more ownership of its work," he said. "Once we agreed on the boundaries, the team stepped up to the plate. Empowerment really can work," Bob said. "But it works best when the *what, when*, and *why* are clear to everyone involved." {end}

How have you helped your team take the lead and make decisions?

Follow the link on the StickyMinds.com homepage to join the conversation.

ARE YOU IN SEARCH OF QUALITY?

SO ARE WE.

Quality Assurance Analysts

Quality Assurance Software Engineers

Apply online at www.blackbaud.com/QAcareers or contact Stephanie McDonald, senior technical recruiter, at 843.654.3547 or stephanie.mcdonald@blackbaud.com.

RELOCATION ASSISTANCE IS AVAILABLE

Quality of Work

Work at the high-tech industry leader in software and services for nonprofits.

Use your talents to make a difference for our customers, who make a difference in the world.

Be an integral part of our product development team and help bring the best possible products to our customers.

Quality of Life

Enjoy the history, culture, and charm of Charleston, South Carolina. Receive great benefits and a competitive salary.

Get started today!

Blackbaud®

ISTOCKPHOTO

SOFTWARE REGRESSION TESTING is an essential and challenging task for software test groups. By definition, regression testing is the process of verifying that none of the existing system features have been accidentally broken by any new features or recent bug fixes. The main challenges for regression testing are performing as many tests as possible in as short a time as possible and finding any serious regression defects as early as possible in the regression test cycle.

The classical regression test cycle is:

1. A final software version with all of the new features completed and with all of the major bugs fixed becomes available from the software development team.
2. Functional tests (for new features) and regression tests (for existing features) are executed.
3. Performance, integration, interoperability, and stress tests are executed.
4. If any serious defects are found, or significant feature changes are needed, steps 1 through 3 are repeated.
5. When all needed features are operational and no major defects remain, the software is released from test, ready for customer use.

PROBLEMS INHERENT IN REGRESSION TESTING

Although it is easy to follow this sequence, there are several problems with this classical approach to regression testing.

Lather-rinse-repeat

The goal of regression testing is to find problems in previously working software features. Although project management may choose to defer fixes or, in some cases, not to fix problems at all, there will be some defects found during regression testing that must be fixed before release. Then the fix-regression test cycle repeats itself, perhaps many more times than the schedule allows.

Any code change can break anything and everything

Project managers often believe that any defects introduced by a code change will be isolated to that particular area of the code and, thus, an abbreviated regression test focused on that particular feature area is sufficient. Eventually you learn that any code change—no matter how isolated—can have indirect, inadvertent, and catastrophic effects on seemingly unrelated areas of the code. Sometimes a bug lies latent in the code for years until some other change is made, which at last reveals the bug.

The best tests are often saved for last

Most regression test plans are arranged in a logical order that starts with some simple tests, such as "ping the server;" proceeds through the multiple functionalities of the product; and ends with complex performance tests, such as "measure the routing convergence time with one million routes in the table." Although logical from a tester's perspective, it is completely backwards from the regression test's goal of finding the most serious defects first. The tests that are the most complex, the most demanding, and the most critical for product and customer success should be executed first.

Getting meaningful results is slow

Both automated and manual approaches to regression testing tend to take a day or more before we know if the latest software version is acceptable

or if it has serious problems. More complex performance, interoperability, and stress tests can take weeks to perform. But managers may need to know within hours if the latest version is good enough for a customer demonstration of a new feature or if it fixes a critical issue being experienced at a customer site without introducing any new problems.

Regression testing is repetitious, dull, and tedious

Regression testing tends to be the least favorite activity for software testers due to its repetitive nature. After a few iterations, the same manual tests become mind numbing, especially if few regression bugs are found. Automated test scripts are usually developed to speed up regression testing and reduce the burden on human testers. But, multiply a simple automated regression test by the thousands of features that exist in most products and the result is tens of thousands of lines of regression test scripts to maintain—another dull task.

Regression testing is a major resource drain

Most test groups dedicate the majority of their resources to regression testing, either by writing and maintaining thousands of automated tests or by having a large number of testers spend weeks performing the regression testing manually. As problems are found in the software, the test team becomes trapped in an endless repetition of the regression test cycle, never able to escape in order to spend adequate time working on new feature tests or better tests.

Attempts to Deal With These Problems

Many test groups have tried to deal with the problems of classical regression testing by using one of these four approaches:

The complete regression test

Some test groups seek to avoid regression test problems by starting regression testing on one software version and continuing for days or weeks on that same version until the regression tests are completed, even though newer software versions are becoming available during the regression test cycle. These versions remain untested until the current regression test cycle is completed, but then regression testing starts again on the latest version.

The continuous regression test

Other test groups seek to avoid the problems by accepting new software versions on a daily or weekly basis during the regression test cycle. When a new version is available, it is loaded onto the test systems and testing continues without restarting. At the end of testing, the regression tests have been spread across multiple software versions. No version has been completely tested, but the regression test has been run only once.

The regression smoke test

Other test groups scale back the scope of regression testing so that a superficial, usually automated, "smoke test" can be completed within a few hours on each new software version. Full regression testing is done only on the "final" version.

The restartable regression test

Other groups restart regression testing as each new version of software becomes available, particularly if it is toward the end of the release schedule and there is a belief that this is the final (or nearly final) version that will need to be tested. Of course, any serious problems that are found require regression testing to restart (again). And rarely does the schedule clock restart to allow enough time to perform a full regression test. Sometimes the first test cases will get executed over and over for each software version, while test cases toward the end of the test plan may get executed only once.

Still, the Problems Remain

Each of these approaches to regression testing results in little meaningful testing being performed on the last software version. This lack of meaningful testing fails the purpose of regression testing, which is to give, in a timely manner, high confidence that all critical features are still working.

A common approach to mitigate the risk of performing very little regression testing is to perform a more comprehensive test on the "final" version. The problem with this is that it is impossible to know before regression testing starts that the "final" testing won't find any major problems and require yet another final version to be built and tested comprehensively, followed by another final version, and so on.

The One-Hour Regression Test Strategy

These problems can be minimized by the adoption of a "one-hour regression test" strategy. This strategy:

- Attempts to find the most serious bugs as quickly as possible, at the cost of finding the lesser issues later in the regression test cycle
- Adapts to the various styles of regression testing: the complete regression test, the continuous regression test, the regression smoke test, and the restartable regression test
- Does not require additional people, test equipment, or weekend work

Ask Yourself, "What If I Only Had an Hour?"

What if a customer asked you to demonstrate to him, within an hour, that your newest software is ready for use? What tests would you run? Now, think carefully—are these the same tests that you are now performing in your first hour of regression testing?

The importance of the first hour

"Wait a minute," you say. "I have six weeks to execute my regression test plan. Is the first hour really that important?"

In a word, yes. Finding the biggest problems as early as possible in the regression test cycle gives the development team the maximum time to develop a comprehensive fix—not just a quick fix.

Immediate feedback on the software quality keeps the entire project team focused on the goal at hand, which is to produce high-quality software as quickly as possible. In addition, immediate feedback prevents other projects from diverting (stealing) your development resources.

"Instead of spending several days performing low-value tests that will have to be repeated anyway, you can spend the bulk of your time performing, automating, and improving interesting, high-value regression tests."

Immediate feedback gives project management the maximum amount of time to react to problems. Sometimes it is necessary to reset (lower) customer expectations. Knowing this as soon as possible makes a difficult job easier.

Immediate feedback gives the test team greater awareness of where the problem areas are in the current software version. This guides testers to avoid further testing of that feature until it is fixed and then to perform additional testing of that feature after it is fixed.

Declaring the latest version "dead on arrival" can take pressure off the test team. Instead of spending several days performing low-value tests that will have to be repeated anyway, you can spend the bulk of your time performing, automating, and improving interesting, high-value regression tests.

How to pick the right tests for the first hour

The philosophy of one-hour regression testing is very different from the philosophy of classical regression testing. Instead of methodically validating that each and every feature meets the requirements of the product specification, the first-hour test approach tries to find the major problems with the version as quickly as possible.

The right regression tests to perform in the first hour will vary depending on the type of product, the maturity and stability of the software, and the test resources available. The following questions will help you select the right first-hour tests for your product.

Customer-focused tests

First, consider the typical use of the product from beginning to end and make sure that use is tested. Think of global, macroscopic tests of your product. What tests would be the most meaningful to your customers? How does the product fit into the customer's overall situation and expectation? What is the name of your product? The first tests should make sure that the product lives up to its name. Which features would have the greatest impact on the customer if they were broken? If stored data were lost? If security were compromised? If service were unavailable? These critical features must be tested in the first hour.

Complex and difficult tests

Which one usage scenario would tell you that the greatest number of important features is working? Make sure the scenario is a realistic, customer-based configuration. If there is no single scenario, then test your top-three customer usage scenarios that have a lot of "moving parts." Consider which top-level tests would reveal any problems with numerous underlying unit or feature tests. Which features can be tested together? Instead of regression testing features in isolation, use a real-world mix of compatible features, validating all simultaneously. Finally, what features are the most complex? Which uses are the most dynamic or the most challenging? What tests are the most difficult to pass or have the least margin to spare?

Big picture tests

Now step back and look at the big picture. Make sure that you test the overall system performance in large-scale use. What are the important top-level performance metrics? What performance tests can be used as benchmarks of major features? Which tests will quickly show any memory, CPU, or other resource-depletion issues? What tests can indicate overall system health with a busy system?

Expected failure tests

What features have historically been a problem for your product? Where have prior defects been found and fixed? These are areas that should be regression tested early in the cycle. Which regression tests are most likely to fail based on the changes that have been made? Use your testing experience and intuition to select the regression tests that have the highest likelihood of failing. What are the newest features in the regression test cycle? These often have problems that slipped past the new feature testing.

The overall goal is to test the software version as deeply and quickly as possible. These tests may take more than an hour, but try to prioritize and select the tests that will provide the most essential information about the quality of the software version within a matter of a few hours.

PUTTING IT ALL TOGETHER

Before starting any regression testing, it is critical to test the bug fixes, new features, or other substantial modifications to existing features that are in this latest version. These changes are the whole reason for building the version; if they aren't right, you know there will be another build.

Insert the one-hour regression test before your full regression testing but after testing the new features, as follows:

Test the reasons for the build.

- Test the bug fixes and test for any likely problems introduced by the fixes.
- Test any new features using a "one-hour new-feature test plan."

Perform your one-hour regression test.

- Select performance, load, and capacity tests.
- Select interoperability and compatibility tests.
- Select database backup and restore tests.

"After focusing on the first hour, ask yourself what you want to accomplish in the first four hours, the first eight hours, and the first week."

- Select redundancy tests, security tests, and data integrity tests.
- Select feature and functional tests.

Continue regression testing.

- Run any smoke or sanity tests.
- Perform load, capacity, interoperability, and performance testing.
- Perform stress, fault insertion, endurance, and longevity testing.
- Perform detailed tests of each feature, per classical regression testing.

Continuously optimize your first-hour regression testing as bugs are found, new features are added, and testing tools are improved.

Evaluate your current regression test strategy

Now that you have a select list of candidate first-hour tests, compare it with the tests that you are currently performing in the first hour of regression testing. Is your existing regression test plan written in a test-centric or customer-centric sequence? Your first tests should be focused on the complex, large-scale, customer-usage scenarios.

Is your existing regression test plan aging or up to date? Is it constantly updated to integrate the new feature tests into the regression test plan? Or are new feature tests simply tacked on to the end of the test plan, waiting until the end to be executed?

After focusing on the first hour, ask yourself what you want to accomplish in the first four hours, the first eight hours, and the first week. Make sure that you prioritize and rearrange your test cases so that if you are asked to release a version a few days before all of the regression test cases can be completed, all of the critical tests will already have been performed.

Smoke and sanity tests

This one-hour regression test has similarities with two other types of regression testing, smoke testing and sanity testing, but they are very different.

Smoke testing usually refers to superficial tests of the software, such as:

- Does the software install and run?
- Are all of the screens and menus accessible?
- Can all of the objects be accessed?

The one-hour regression test approach is more focused and does not replace the broader smoke test. If possible, the smoke test should be run in parallel with the one-hour regression test. If this is not possible, then run the smoke test immediately after the one-hour regression test.

Sanity testing usually refers to a small set of simple tests that verify basic functionality of the software, such as:

- Simple calculations provide correct results.
- Basic functions work correctly.
- Individual, isolated features work correctly.

There is typically no capacity, scale, or performance testing involved with sanity testing. Sanity testing provides the starting point for in-depth regression testing of each feature. Sanity testing should be performed only after the one-hour regression test is complete. Ideally, sanity tests should be re-examined to provide greater benefit to the regression testing, once the one-hour regression tests are in place.

The one-hour regression test differs from smoke and sanity testing in another important way. The goal of one-hour regression testing is to find the major problems as quickly as possible by testing complex features deeply.

New features

If there are any new features introduced during the regression test cycle, hit them hard in the first hour, shaking out any major design problems. After the new features survive this testing, methodically test the details of the new features using the full new-feature test plan.

SUMMARY

The one-hour regression testing approach performs a small number of the most important and most complex tests during the first hour after each new software version is received, with the goal of finding the big problems early in the regression test cycle.

The various approaches to software testing can all benefit from one-hour regression testing, whether your test group uses a traditional, agile, context-driven, or exploratory test strategy.

The one-hour regression testing approach requires a slight change of thinking for the first hour, but it can save days or weeks of repetitious regression testing, while increasing the overall software quality of your product. {end}

Sticky Notes

For more on the following topics go to www.StickyMinds.com/bettersoftware.

- Further reading
- Related links

SOFTWARE TESTING TRAINING

Accelerate Your Career & Empower Your Team

NEW SPRING 2009 SCHEDULE

BUILD-YOUR-OWN TRAINING WEEK

Maximize the impact of your training by combining courses in one location to create a customized training week. Pair two courses and save up to $300. For a complete list of courses available, visit www.sqetraining.com or call 888.268.8770 or 904.278.0524 for pairing discount options.

SQE TRAINING
www.sqetraining.com

Improve your skills and help your organization increase its performance through targeted high-value training. Delivered by top software consultants, training through SQE Training is one of the best investments you can make to meet your business objectives.

TRAINING WEEK LOCATIONS

March 23–27, 2009	*Boston, MA*
April 20–24, 2009	*San Diego, CA*
June 1–5, 2009	*Chicago, IL*

PMI Project Management Institute R.E.P.

The following courses are PMI Certified:

- Software Testing Certification
- Systematic Software Testing
- Mastering Test Design
- Test Management

Choose from 11 specialized training courses:

THREE-DAY COURSES (Monday - Wednesday)

- Systematic Software Testing
- Software Testing Certification
- Test Management
- Writing Testable Requirements
- Just-in-Time Software Testing

TWO-DAY COURSES (Thursday - Friday)

- Mastering Test Design
- Performance, Load, and Stress Testing
- Lean-Agile Testing Practices
- Requirements Based Testing
- Exploratory Testing in Practice

ONE-DAY COURSE (Thursday)

- Test Process Improvement

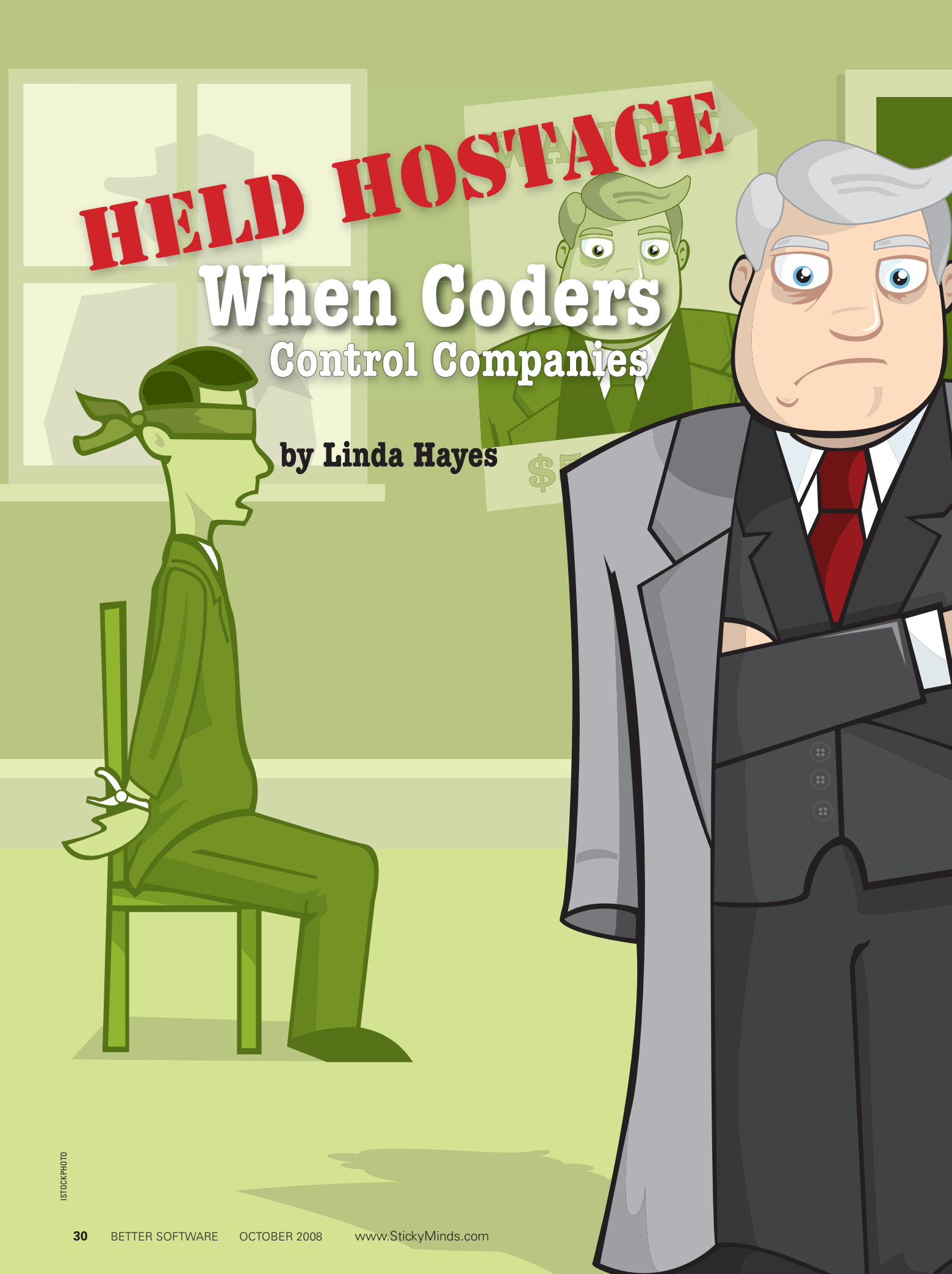
ON-SITE ADVANTAGE

Let SQE Training come to you. For more information about on-site training courses, contact SQE Training at 904.278.0524 x212 or 888.268.8770 or email onsitetraining@sqe.com.

HELD HOSTAGE

When Coders Control Companies

by Linda Hayes



ISTOCKPHOTO

Odds are, if you examined the organizational chart for any company that develops software, you would not see the developers at the top. Yet in some cases, developers are able to wield their knowledge and control of the code to hold management hostage to the developers' own agenda, playing on fear and ignorance to subvert corporate goals and customer needs. Understanding how this can happen—and what you can do about it—is key to avoiding being taken hostage and losing control of your company and its software.

My first inkling that such a power reversal could exist occurred many years ago as I was filling out security forms to enter the building of a major ATM network. Around me, the sea of professionals in business attire was parted by a biker-looking dude with long scraggly hair wearing ragged denim and chains. I was surprised that no one else seemed to take notice of him as he breezed past security without so much as waving a badge.

I mentioned this to the people with whom I was meeting, and they explained that the biker was Chuck and he wrote the highly complex, critical-switch software that controlled everything. As the original author, only Chuck understood it all. Management lived in fear that Chuck would quit or be killed in a motorcycle accident (he refused to wear a helmet). Management treated him like a rare commodity with special privileges, such as violating the dress code, coming and going as he pleased, and generally dictating the contents and schedule for software releases. Management acted as a supplicant when new functionality or interfaces were needed, allowing Chuck to dispense code at his whim.

Even way back then, before I had started three software companies, I sensed that there had to be more to the story. How could such a mighty corporation bow down to a single weird guy? Now, with the benefit of decades of expensive experience and the luxury of hindsight, I have come to recognize that there are certain patterns that create an opening for this reversal of power from a company to a coder, as well as proven and failed strategies for dealing with it.

I don't believe most programmers plot to wrest control from their employers. In fact, in some cases the coder is also a victim, forced by circumstances into what he feels is a no-win position. While every situation is unique, there are certain archetypes that are common. Let's look at them through the lens of actual—although somewhat disguised—case studies to see how they develop and what solutions will or won't work.

The Hijacker

The hijacker is a developer who takes control of strategic product decisions without permission or authority, holding the company and its customers hostage to his decisions through delays and diversion. Even though the hijacker's intentions might actually be good, the results are bad.

Take the case of a developer we'll call John. He was the unfortunate heir to a Web testing tool that was written in the early days when most Web applications used statically coded HTML pages. As the Web development paradigm shifted to dynamically generated applications with scripted controls, the basic architecture of the tool, which expected predictable page layouts, was no longer viable. As the tool failed to accommodate interactive content, bug counts started to climb and John was expected to fix these bugs. But from John's perspective, the bugs were symptoms of fundamental flaws and the entire product needed to be rewritten from scratch.

Even though John was technically correct, customers who were already using the product could not wait months or longer for a completely new version—they had issues that had to be fixed in real-time. When he failed to get management approval for the time to do the rewrite, John started elongating the turnaround time to deliver fixes, arguing that the code was so broken that it took forever to fix anything. Customers rebelled at the delays, and the company's sales and support team was feeling the heat.

By delaying the schedule, John was holding the company and customers hostage to his demands for a rewrite. Another programmer, who was onsite with a customer, discovered this when he was able to correct a problem with the

LOOK FOR THESE SIGNS AND CONSIDER THESE RESPONSES:

Archetype	Symptoms	Solutions
Hijacker	• Delayed schedules • Excessive defect backlog • Customer, support, and field frustration	• Independent evaluation of necessity for rewrite • Open strategy discussion regarding rewrite plans or reasons for no rewrite • If rewrite not planned, then implement strategy for ongoing support • If rewrite planned, then involve current developer but mentor maintenance developer
Prima Donna	• Separation of maintenance from initial development • Poor code design, instability, and scalability • Customer, support, engineering, and field frustration	• Establish minimum maintenance ownership for initial developer • Define and enforce development discipline and code management
Control Freak	• Single point of failure, inability to add resources • Slowing response to competition, customer needs • Increasing instability • Lack of central, current source control	• Define and enforce development discipline and code management • Independent evaluation of necessity for rewrite • Open strategy discussion regarding rewrite plans or reasons for no rewrite • Cross-training new or backup resources • If rewrite planned, involve current developer with rewrite but mentor maintenance developer

product in a matter of hours, instead of the weeks that John had estimated. The sales and support team initiated a campaign to replace John, which eventually succeeded.

In his exit interview, John explained that he was effectively rewriting the code piecemeal by solving the underlying design problem instead of just trying to remedy the specific reported bug. In his mind, he was doing the right thing for the company, because he knew that the bugs were symptoms of a much larger problem that would never be fully resolved without a complete redesign.

This situation had an unfortunate outcome for all involved. Customers suffered with lengthy response times for bugs, the support team bore the brunt of user dissatisfaction, the company lost a skilled resource with valuable knowledge about the existing product and how best to redesign it, and John lost his job and had his reputation tarnished as someone who could not or would not deliver.

There are many variations of this particular archetype, which boils down to this: One developer inherits someone else's problem code and wants to rewrite it. Such a rewrite may or may not be part of the company's strategic plan, thus setting up a conflict between the coder and the company. In cases like John's, the coder hijacks control of the product strategy by making the covert decision to move ahead with the rewrite without authorization.

What to do? First, recognize that not all code has to be rewritten just because the programmer says so. Who wouldn't rather start from scratch and put his own imprint on the code for which he is responsible? Development is always more fun than maintenance. On the other hand, sometimes code really has outlived its usefulness and has become more of a liability than an asset, consuming excessive resources to keep it on life support with no end in sight.

The best way to sort this out is to get a second opinion. Have another developer review the code and provide an independent analysis. If he agrees that a rewrite is truly necessary, then verify whether the time and expense is justified by the product's expected useful life, rate of change, or potential for revenue. Then have a full and open discussion with the developer about if and when the rewrite will happen. Most people's

attitudes and tolerance improve when they know they have been heard and the end of the tunnel is in sight—because of a planned product retirement, a slowdown in changes, or a rewrite.

If a rewrite is planned, though, don't leave the original developer to suffer with the bad code while allowing another to do the rewrite. As logical as this might seem, since you want bugs fixed fast, it has the double downside of penalizing the developer who knows the most while giving the rewrite to someone who likely knows much less. Instead, assign the original developer to the rewrite but require him to mentor the developer who takes over the ongoing support. This transfers the expertise on the existing product to another resource and lets you leverage product knowledge and experience for the new one.

The Prima Donna

Prima donnas are frequently the root cause of the rewrite problem. They are developers who enjoy writing new code but don't want to support and maintain it. They often see themselves as too creative to be bogged down in the daily grind of maintenance.

The risk is that a programmer writing code he won't be maintaining might be tempted to cut a few corners. Take the case of a developer I'll call Carl. Carl was the darling of sales and marketing because he seemed to have a superhuman ability to rapidly deliver sexy new products or features that customers wanted. Customers loved that he seemed so responsive—seemingly able to pull functionality out of a hat, without being hampered by formality and process.

His peers, on the other hand, hated Carl, because they were the ones who inherited his code. To no one's surprise, Carl's code was poorly designed, completely undocumented, and neither robust nor scalable. But by the time these issues surfaced, Carl was off to his next performance, and those left to pick up the pieces bore the brunt of customer frustration. The support and maintenance team was effectively held hostage by Carl's code and resented it bitterly.

Eventually this dissatisfaction reached the vice president of engineering, who satisfied himself that the complaints were real. He decreed that, henceforth, Carl would be responsible for supporting any code he developed. Carl rebelled and took his case to the field, warning the sales team and customers that he was being co-opted by the dark forces of formality and would no longer be available to blaze new trails. The sales and marketing team took his case to upper management, but the VP stood his ground.

Carl threatened to quit. The VP stood his ground. Customers complained to upper management. Still, the VP stood his ground. Eventually, Carl did quit—and the rest of engineering threw a party.

The end of the story took awhile to be told. The initial disappointment over slowed schedules was eventually erased as product stability and performance improved and schedules became more predictable. Carl's flash was forgotten as firefights subsided and customers became more satisfied.

The larger lesson in this story is that separating development and maintenance has hidden risks and costs. It creates orphan code, which sets up the rewrite problem, and allows one or more developers to hold the rest of the company hostage to their lack of ownership. The best way to address this is to hold developers accountable for the care and feeding of code they write so they will take a longer-term view of maintainability and stability.

This does not mean that a programmer should be shackled to code for life, as it is important for developers to constantly hone their skills and learn new technologies to stay engaged and marketable. A reasonable compromise might be a one-year-minimum commitment to maintenance and support, during which any inherent flaws should be revealed.

The Control Freak

Control freaks are coders who hoard the code—the flip side of Carl's refusal to own his code. Consider Sue, one of the first developers for a fledgling banking software company. She was young and smart—if somewhat inexperienced—when she was first hired, and since the company was a startup on a tight budget, she was a good fit.

"Most people's attitudes and tolerance improve when they know they have been heard and the end of the tunnel is in sight."

The first product she worked on was a telephone banking system that allowed customers to get balances and conduct transfers over the phone. Sue was comfortable with the technology and did a good job, but soon Web access eclipsed interactive voice response. While Sue was anxious to jump on the new bandwagon, she was inexperienced with the platform.

Due to her longevity with the company and experience with the phone banking product, which would form the backbone of the Web offering, Sue won the project to develop the online version. She literally learned on the job through trial and error, resulting in code that worked on the surface but underneath was a patchwork of fixes and workarounds. The increasing fragility and complexity of the code raised Sue's concern that anyone else looking at it would have a difficult time trying to understand it and might unjustly judge her abilities.

So Sue exercised increasing control over the code. She opposed any effort to train others on the product, arguing that it would take longer to teach someone else than just to do it herself. She did not keep the source control system updated

with the latest version, preferring to keep it on her own machine. Her dominion over the product was reinforced whenever a crisis erupted and she had to ride to the rescue at all hours. By rewarding her for resolving the crisis instead of holding her accountable for creating it, management perpetuated her position as a single point of failure.

The company grew and prospered, which increased the strain on the product. Scalability and security issues emerged, and competition became fiercer, fueling the demand for enhancements. Still, Sue resisted efforts to expand the available resources on the product, if anything tightening her grip.

Inevitably the demands on the product exceeded Sue's ability to deliver, but by then, Sue was so firmly in control that management felt powerless to do anything about it. Because the product was mission critical and could not be unavailable for any meaningful length of time, management feared antagonizing her lest she quit. Management was also frustrated in its efforts to transfer Sue's knowledge and did not believe she could be replaced without her cooperation. The situation simmered for what became years, during which time the company lost revenue and market share—hostage to Sue's insecurities and its own complicity in allowing the circumstances to reach this point.

Things came to a head when an upgrade to the platform crashed the software, which stayed down for more than a day with Sue scrambling frantically but failing to bring it back up. A recent recruit to the engineering team with extensive Web experience threw out a suggestion that turned out to be the right solution and, finally, the system was resurrected. This crystallized the issue for all involved. Something had to be done.

In this case, the company president didn't throw the baby out with the bathwater. Instead, he involved Sue in a complete rewrite of the product but was careful to institute sound project management and development processes so that no single person was the source of all knowledge or control. Technically there was a happy ending—but at a cost to the company of millions of dollars in lost revenue.

PROTECT YOURSELF FROM BEING TAKEN HOSTAGE BY TAKING THESE STEPS:

Do	Don't
Define and enforce source control procedures	Allow developers to keep source on their personal systems
Adopt documentation, design and coding conventions, code reviews, and walkthroughs	Permit developers to sidestep discipline in the name of expediency
Cross-train	Let one developer be a single point of failure
Openly communicate corporate strategies and plans	Expect developers to remain ignorant of what the future holds
Invest in developer career path planning	Expect developers to remain stagnant with the same code and technology forever
Require original authors to provide support and maintenance for a minimum time period	Allow developers to avoid accountability and ownership for the code they create
Reward stability, predictability, and team efforts	Reward drama and lone heroes
Aggressively address symptoms of existing or developing hostage situations	Delay resolving issues with developers out of denial or fear

If you have a control freak, don't delay. Force the issue of knowledge sharing and transfer as soon as you see the warning signs. Control freaks are less likely to quit than you might think. Until others have a chance to review the code, you won't know whether the developer was hiding bad code or just protecting his turf.

In addition, adopt and enforce coding conventions that require documentation in the form of architectural diagrams, code comments, and similar mechanisms to guide others through the code in the event the original author is unavailable for any reason. A huge red flag should go up any time a developer asserts that he is the only one who can understand his code; conduct code reviews and walkthroughs whenever possible. Code is a corporate asset—not a personal one—and should never be cloaked in such a complex or arcane way that it can't be deciphered.

And last but by no means least, institute and enforce stringent source control and backup procedures. The software source code is a valuable corporate asset that should no more be housed on a developer's personal machine than your bank teller should be making change out of her purse.

Management's Responsibility

The president of the banking software company where Sue worked summarized the common theme across all archetypes of the hostage phenomenon by pointing out that when a programmer is hired to write code, the result—the intellectual property—is owned by the company, not the creator. What's important to understand is that intellectual property isn't just the code itself; it's also the knowledge about how the code works. Developers cross the line when they believe that their knowledge of the code is a personal asset that can be used to hold the company hostage to their own agenda.

Management's role, then, is to limit the risk of being held hostage by en-

"By rewarding her for resolving the crisis instead of holding her accountable for creating it, management perpetuated her position as a single point of failure."

suring that there are processes and methods in place to make knowledge a usable and transferable asset while fostering a healthy environment for developers so that they don't have to resort to taking hostages. This takes the form of cross-training, development discipline, and a corporate culture that assures no single developer holds the keys to the software.

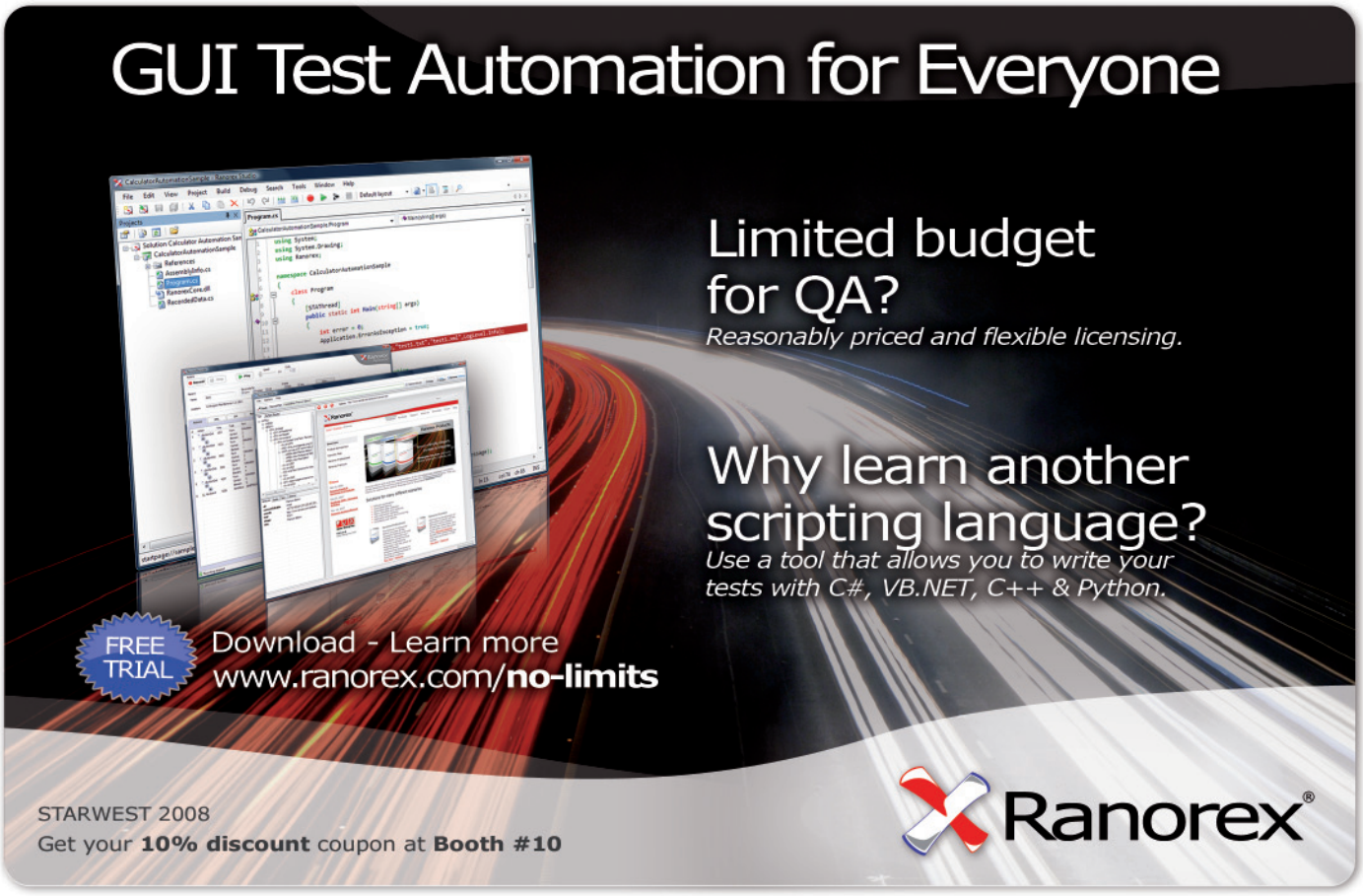
Cross-training is critical because anytime you have only one developer knowledgeable about a product, there is an inherent transfer of some level of control. If the company cannot afford more than one resource for a particular product, then compensate by having a backup, even if he is from a different product team.

Development discipline lends structure and uniformity to code that simplifies knowledge transfer and future maintainability. This includes naming and coding conventions, source control processes and procedures, code walkthroughs, inspections and reviews, and documentation standards.

An open and supportive corporate culture is also an essential component. Management has a responsibility to the developer to respect his knowledge by seeking his input and keeping him in the loop about plans for the product, as well as providing the developer with opportunities for professional advancement and achievement. Open communication in an atmosphere of mutual respect can do more to avert a hostage crisis than any exercise of power ever could. Create a culture where everyone is on the same team and hostages will not be needed.

The common underlying factor that creates the opportunity for a hostage situation or allows it to continue is fear: fear that losing or replacing a developer or otherwise calling his bluff will have catastrophic consequences. This leads to paranoia that the code will somehow fail or disappear—either by accident or even by deliberate sabotage—and only the now-absent and disgruntled developer holds the key to saving the day. If you feel this fear, treat it aggressively by implementing the steps outlined above. Don't wait for a crisis to act.

But most importantly, realize that being hijacked is almost always a symptom of more systemic problems that can and should be fixed, resulting in an overall improvement in product quality, customer satisfaction, and improved company morale. **{end}**



Google Web Toolkit

Your Shortcut to Ajax Web Applications

GETTY IMAGES

by Daniel Wellman

"Web 2.0 is the future! I need you to build us a new Web front end for our site. It should use all the slick animations, auto-suggest, and desktop-like features that are the standard these days. It will need to be fast, and it has to work with Internet Explorer, Firefox, and Safari so we can support all our customers. How soon can you be done?"

Rachel broke out into a sweat remembering her boss's request. She had a lot of experience programming in Java and was a whiz at refactoring using Java IDEs—but Ajax? She was pretty sure that required knowing JavaScript and understanding the subtle quirks of all the popular browsers. How would she get through this?

Then Rachel remembered hearing about the Google Web Toolkit, which promised that you could build Ajax applications using Java instead of JavaScript and that it would handle all the complicated cross-browser issues automatically. She downloaded the Google Web Toolkit distribution, fired up her Java IDE, and started writing ...

Building Ajax applications can be tricky—just as juggling chainsaws while taming angry lions can be tricky.

Ajax application code is written in JavaScript, which then runs in a Web browser. A public Web site must support multiple versions of multiple browsers (Internet Explorer 5, 6, and 7; Firefox; Opera; and Safari), each with its own differences and idiosyncrasies. A developer has the difficult task of writing JavaScript that safely manipulates the Document Object Model (DOM) in a way that behaves identically on all supported browsers. Of course, code on the server side usually uses a different language and set of libraries, so there's a lot of context switching to be done. And we haven't even considered the hardest part—the application's real business logic!

Ajax is short for *asynchronous JavaScript with XML*. Ajax applications are highly responsive and feel more like desktop clients than traditional, form-based Web pages. The fundamental difference between standard Web sites and Ajax applications is that standard Web sites deliver all their data in a single page load, whereas Ajax applications may make several requests for data while the application is running and a single page is displayed. Data is delivered *asynchronously*, which means the user may continue to use an application while waiting for data from the server. This means the server components need only send the raw data and let the Web browser handle all the formatting and presentation logic.

Google has created an innovative tool to aid Ajax developers: the Google Web Toolkit (GWT). GWT compiles Java source code into browser-friendly JavaScript. This means you can use your favorite Java IDEs, design patterns, and tools such as JUnit, FindBugs, and PMD to create Web 2.0 applications. GWT includes a rich component library of user interface widgets, remote server communication protocols, and an emulated Java Runtime library, which maps Java types to JavaScript types—all of which work seamlessly across the major browsers. Furthermore, you can debug GWT applications using a standard Java IDE. These features let you focus on writing your application instead of worrying about browser compatibility problems. Using GWT is like having a teammate who knows about all the different

```
// This is a Java method
public int add(int x, int y) {
  return x + y;
}

// This is a JavaScript method
public native void alert(String msg) /*-{
  $wnd.alert(msg);
}-*/;
```

Listing 1

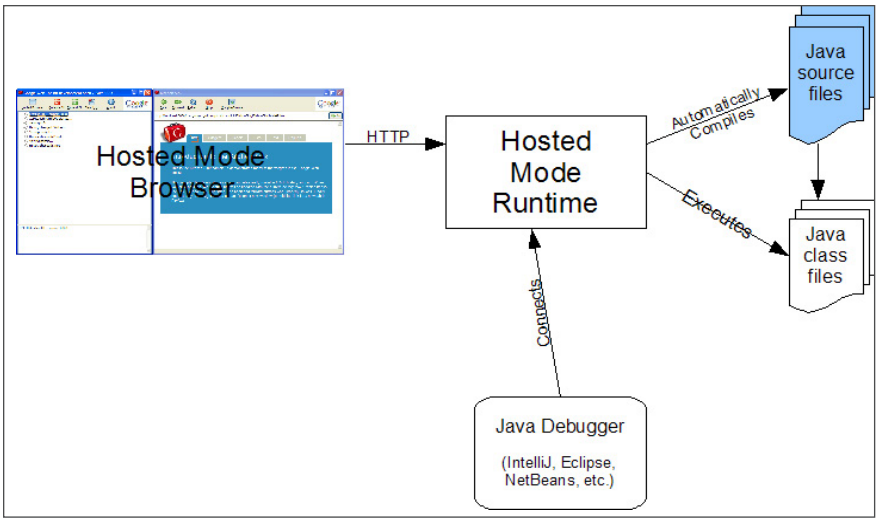


Figure 1: Hosted mode runs applications as Java bytecode and enables debugging using Java IDEs.

browser incompatibilities and handles them for you.

Java to JavaScript Compiler

What makes GWT unique is its Java-to-JavaScript compiler. You write plain old Java code and invoke GWT's compiler, which takes Java source files and outputs JavaScript. This is a true compiler—not a simple text transformation step like Ruby on Rails' RJS, which provides Ruby objects that use String manipulation to create JavaScript output. While Java is the preferred language for writing GWT applications, it doesn't mean that you have to give up using JavaScript. You may mix Java and JavaScript code in the same class with GWT's JavaScript Native Interface (JSNI) [1] by using the `native` keyword.

Listing 1 demonstrates a Java class with a native JavaScript instance method. You can pass objects back and forth across the Java/JavaScript boundary and generally treat the code as if it were all part of the same object. Using Java as much as possible to ensure browser portability is a best practice, as it leverages all the GWT safety features, such as preventing memory leaks. However, if you need it, you can always write JSNI methods.

Optimizations

Another benefit of using a compiler is that it allows for compiler optimizations both in terms of performance and generated JavaScript source file size. The GWT compiler can replace many levels of delegation with inline method calls, which can yield code that's as fast as code that has been hand-optimized. This means you can focus on writing well-factored, object-oriented code instead of obsessing about performance tweaks.

Ajax applications should be fast and responsive, which means minimizing initial page load time. One way to do this is to simply send less JavaScript code from the server to the client. The GWT compiler can create compact, obfuscated JavaScript optimized for short download time or human-readable code, which is useful when diagnosing problems. In addition, GWT analyzes your entire application and only includes the code that is actually used. In practice, this means that you can reuse existing Java libraries and other modules, and GWT will compile only the routines you actually invoke. This optimization comes at a price; you can't use reflection in your code since GWT can't tell ahead of time what methods are really invoked.

Debugging and testing

Some problems are easiest to diagnose with a debugger, and GWT makes debugging easy, as shown in figure 1. Simply start up your application in "hosted mode," which runs your application using Java bytecode instead of JavaScript. This means you can use your Java IDE's debugger just as you would for server-side code. To run applications in hosted mode, the GWT development kit includes an embedded browser instance of Internet Explorer for Windows, Firefox for Linux, and Safari for Mac. Hosted mode is really useful, and you'll probably end up using it for most of your development work with GWT. For example, you can run your application, change the source code, and hit the browser's refresh button to see your changes immediately. Hosted mode also provides a console window that you can use to log debug messages during development.

You can write unit tests with JUnit just as you would for server-side code. However, when you need to test applications that manipulate the DOM or use JavaScript, GWT can run your unit tests inside the same hosted-mode browser you use for developing applications. If you subclass GWTTestCase instead of the JUnit standard TestCase, your test will run in an invisible browser with access to the DOM. You can run these tests as compiled Java bytecode or as JavaScript to ensure your application behaves correctly after being compiled.

Component Libraries

GWT includes a rich set of class libraries and user interface widgets. A subset of the core Java SE libraries is emulated by GWT, which means you can use most of the core language data types and collections. GWT also provides a rich set of browser widgets as well as utilities for supporting multiple languages and a way to handle the browser's back and forward buttons.

Widgets and UI components

As figure 2 shows, Google provides many widgets with GWT—from simple, standard HTML widgets like buttons and text fields to rich widgets composed of simpler elements like a tree view. Simple widgets such as `Button`, `Checkbox`, `Hyperlink`, and `TextField` provide simple wrappers around basic HTML-form elements and are used for the majority of form-based data input. For more dynamic interfaces, GWT offers a rich set of widgets including: `TabPanel` for navigating through a tabbed set of pages; `SuggestBox` for a text entry field, which lists possible completions

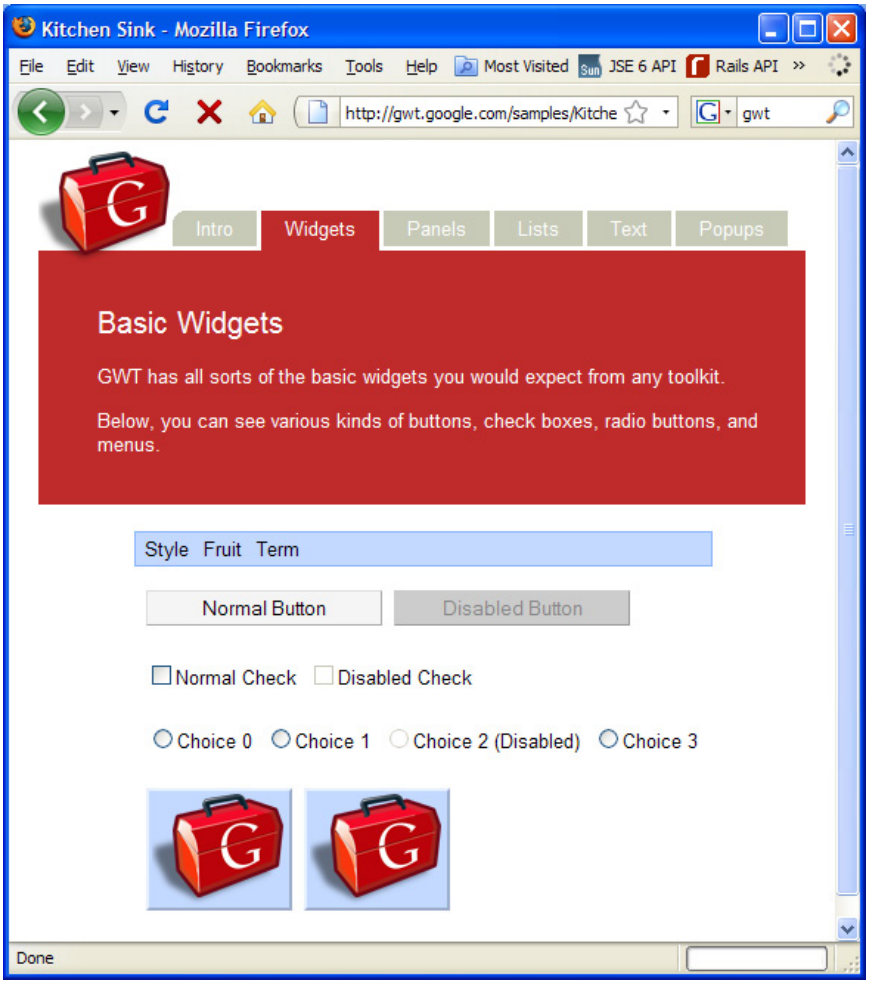


Figure 2: GWT provides many widgets from simple buttons to tabbed menus and dialog boxes.

as you type (think Google Suggest); and `DialogBox` for draggable popup windows.

GWT applications use an event-based programming model, where each button, text field, and other user interface element responds to a set of actions such as clicks, content changes, and focus changes. This programming model will be familiar to developers who have written Swing desktop applications. In fact, GWT code looks similar to Swing code. You add behavior to your widgets by adding event listeners that respond to user interactions. For example, listing 2 shows how you define a button that prints a message every time you click it.

By building your applications using GWT widgets and the event-based programming model, you let GWT handle the complex differences between browsers. For example, did you know that Internet Explorer versions 5 and 6 have a problem where select boxes appear through any covering popup windows? GWT prevents this display glitch by applying a well-used fix that creates IFRAMEs that sit directly between the select box and the popup window. Did you know that Internet Explorer propagates events through the document model differently from all other browsers? By using GWT's event classes, you never need to know about or deal with the differences.

Internationalization support

GWT's built-in internationalization features can help you build a multilanguage site. Your application's text and messages can be extracted into a separate property file for each supported language. GWT will then compile these property files and generate separate versions of your Web application for each language. Dates and currencies can be formatted for display in a given locale by using the `DateTimeFormat` and `NumberFormat` classes.

Back button and browser history

One of the most difficult issues in writing Ajax applications is handling the browser's back button. An Ajax application is usually a single page that requests new data from the server without navigating to another page. Since browsers track navigation history by full-page loads, handling the back button is tricky. For example, consider working with the Ajax-heavy Google Maps site: A user navigates to Google Maps from the Google start page then searches for the closest pizzeria. The user finds several results and clicks through a few choices to view their details. When the user hits the back button, he expects to go to the previously viewed pizzeria—not back to the Google start page.

GWT also provides a history mechanism for creating applications that respond predictably to the back and forward buttons. This mechanism relies on a common JavaScript trick that uses an IFRAME HTML element to generate page history. All you need to do for your application is add an IFRAME element to your page and let GWT handle all the JavaScript tricks for you. In your application, you tell GWT when to add an entry into the browser history, such as clicking a link or button. When the user presses the back button, your application will be notified and you can change your page as needed.

Remote Server Communication

Ajax applications wouldn't be very interesting if they didn't have access to data. GWT provides a number of ways to access data on remote servers, including a custom protocol that lets you easily use your server-side objects directly in the browser. To access data services from other applications, GWT also supports standard data interchange formats like JSON and XML.

Using server objects on the client with GWT RPC

If you're building both the server side and the client side of a GWT application,

```
Button button = new Button("Click Me");

button.addClickListener(new ClickListener() {
      public void onClick(Widget sender) {
            Window.alert("I was clicked!");
      }
});
```

Listing 2

then you'll get a big productivity boost by using GWT's Remote Procedure Call (RPC) feature. This mechanism allows you to use your server-side objects directly on the client side. Consider what would be required to use a plain-text format: You'd fetch the required data on the server side, write code to turn your objects into a text representation, send this text to the client, then write code on the client that can parse and interpret the message. With GWT RPC you can skip writing extra encoding and decoding layers and start actually *using* the data.

To use the GWT RPC feature, you can use GWT's servlet-based RPC framework for new applications or use the RPC libraries directly with your existing Web application frameworks, such as Spring and Struts.

```
{"widget": {
    "debug": "on",
    "window": {
        "title": "Sample Widget",
        "name": "main_window",
        "width": 500,
        "height": 250
    },
    "flags": [ 1, 8, 16 ]
}}
```

Listing 3

Communicating with other back ends with HTTP, JSON, and XML

Just because GWT client-side code is written in Java doesn't mean that the server must be Java as well. The `RequestBuilder` class can be used to communicate with any back end: Python, C#, or any language that can speak HTTP.

If you aren't using GWT RPC in Java or have a server back end in another language, GWT supports the standard XML and the JavaScript Object Notation (JSON) data interchange formats. JSON has been gaining popularity in Web services and is easily read by humans and browsers; in fact, it's a valid JavaScript object, as shown in listing 3.

JavaScript and GWT Third-Party Libraries

Successful development platforms have a rich community of third-party libraries, and GWT can use both JavaScript and other GWT libraries.

Since GWT applications are always deployed as JavaScript, you can easily use other JavaScript libraries. This means you can use any existing JavaScript code your company may have developed, as well as popular third-party tools, such as the animation framework Scriptaculous and other application frameworks like ExtJS. You'll need to write native JavaScript code, which acts as a bridge between your Java and JavaScript code. However, many folks have already done this work for you and published these bridges as GWT libraries.

In addition to JavaScript library wrappers, there are many useful open source components made specifically for GWT. GWT-Maven provides support for the popular Maven build tool to build and package GWT applications. Gwittir offers an alternative to reflection to provide simple JavaBean-style introspection that can be used for data binding and validation. Popular server-side frameworks are made GWT-friendly through projects like Hibernate4gwt, which allows you to use Hibernate objects easily in your client code, and GWT-SL, a server-side library that provides Spring framework support. Check out the GWT Google Group for discussions and announcements of other tools.

Google also publishes additional libraries for GWT. It wouldn't be a Google project if you couldn't access the Google online services, and the Google API Libraries for GWT project does just that: You get access to Maps, Search, offline functionality with Gears, and the multi-platform Google Gadgets library for desktop and Web widgets. In the eating-their-own-dog-food category, the gwt-api-interop library was used with GWT to help build the iPhone Google Reader application. It provides a way to bind JavaScript objects to Java objects, which means you can bind JSON to Java objects without writing a parser by hand.

The Real World

The story at the beginning of this article isn't pure fiction; a similar situation happened to me on a recent project. My team was hired to build the front end to a new site, and we had used GWT previously. We had to support the major browsers on Windows and Mac and deal with requirements that were evolving throughout the four-month project. GWT allowed us to use our Java experience with test-driven development and refactoring to keep our design clean and deliver on time. We started with two full-time developers and scaled up to four, then six—including two new hires. GWT's simple programming model made this an ideal first project, as the team spent more time focusing on the business rules than learning APIs.

The next article in this series will focus on testing and GWT, including how to build Ajax applications test-first to help create simpler, defect-free designs. **{end}**

REFERENCES:

1] "JavaScript Native Interface (JSNI)" in the Google Web Toolkit online documentation. code.google.com/webtoolkit/documentation/.

Sticky Notes

For more on the following topics go to www.StickyMinds.com/bettersoftware.

- More information
- Other Ajax framework options

StickyMinds.com

Map out your project success using StickyMinds.com as your resource guide

Benefits of joining StickyMinds.com:

- Access daily news articles geared to our industry
- Have any of our six eNewsletters delivered straight to your inbox
- Enjoy Podcasts and Videocasts featuring industry experts
- Find software solutions in the comprehensive Tools Guide
- Search and read book reviews
- Post a question to your colleagues on our Discussion Board
- Find the resources you need for building better software
- Submit articles or technical papers for others to download

StickyMinds.com is the Web's first and most popular interactive community exclusively engaged in improving software quality throughout software development.

Membership is free, so sign up today and start your journey to building better software!
www.StickyMinds.com/join

StickyMinds.com

GUI*dancer* Version 2.2

GERMANY—BREDEX GmbH has released GUI*dancer* version 2.2. New features include an option to retry failed test steps, compatibility with the Eclipse 3.4 Ganymede release, and the ability to start applications under test easily from execute table files. Alongside these features, there have been numerous improvements to the usability of the program. The addition of welcome pages, cheat sheets, and new example projects will help new GUI*dancer* users. The automatic inclusion of the Test Case library in all projects will benefit all GUI*dancer* testers, allowing reusable keywords to be created quickly using drag and drop.

GUI*dancer* version 2.2 retains all of the benefits of previous releases—tests are easy to create and maintain, no programming skills are required, and test creation is separate from the application under test. The new version has added features to make working with GUI*dancer* even easier. User interface improvements mean that tasks like object mapping and the maintenance or modification of large and complex tests have become simpler and smoother. The options available to GUI*dancer* testers now include the use of predefined test execution variables in tests.

Visit www.bredexsw.com for more information.

Ranorex Version 1.5

GRAZ, AUSTRIA—Ranorex GmbH has released Ranorex version 1.5—Ranorex Studio, a complete test development environment that brings all Ranorex features together, allowing the user to create and maintain professional test automation projects through a single, easy-to-use interface.

Ranorex is a cost-effective and comprehensive GUI-based software automation framework that simplifies both integration and acceptance testing for developers, SQA managers, and testing professionals, regardless of their programming or scripting experience. Ranorex supports standard programming techniques and common languages such as C#, C++, and Python, as well as VB.NET or any other .NET language. Its ease of use, increased testing accuracy, and low cost per seat makes Ranorex an ideal choice for software teams of virtually any size or level.

Visit www.ranorex.com for more information.

V1: Agile Enterprise 8.2

ATLANTA, GA—VersionOne has announced the release of *V1: Agile* Enterprise 8.2. Focused on strengthening collaboration between agile team members, the enhanced communication features promote useful interaction and provide quick access to the content in the application.

With the latest release, notifications now offer even more communication channels: user-generated messages between team members, system-generated alerts from subscriptions, and admin-level announcements to project teams. Users can send messages to raise awareness about a new issue or to provide immediate visibility into the status change of a story, task, or test. Members can set up subscriptions to alert them if there are changes to a specific asset or to let them know when someone responds to

their discussion post. Project or system administrators can notify all users about an upcoming system upgrade or send a reminder to complete an activity.

Key features include:

- Simplified report distribution and print-optimized page layouts available via direct printing or PDF generation throughout the VersionOne application. Reports and charts can be directly emailed to team members and outside stakeholders, or saved and archived from within the tool.
- Quick access to threaded discussions, links, and attachments available through new icons as well as hover popups that indicate if asset-related information has been added or changed.

Visit www.versionone.com for additional information.

Cruise

SAN FRANCISCO, CA—ThoughtWorks unveils Cruise: Continuous Integration and Release Management system. Cruise enables software teams to go live with new, tested features quickly and repeatably. By lowering the effort and risk associated with releasing complex applications, Cruise leaves your team free to focus on the stuff that really matters.

Cruise harnesses the concept of pipelines. A pipeline configuration is like a workflow; use it to define the stages through which a set of modifications must pass, such as:

- A "commit" stage, which compiles the software and runs unit tests
- Functional and performance test stages
- Stages for each of your testing environments
- Staging and production stages

Visit www.studios.thoughtworks.com for additional information.

Plastic SCM 2.5

SPAIN—Codice Software has announced its release of Plastic SCM 2.5. Plastic SCM 2.5 introduces a number of new features and enhancements such as the new trigger system, which makes smart branches go mainstream and implements attribute support in the GUI. Triggers can be used to enforce processes and development rules, add extended information to your metadata with attributes, and keep SCM status in synch with testing, production, etc.

The improvements on this new version go even further with more than one hundred usability enhancements resulting in a faster and more complete solution. New features include:

- "Task on changeset" mode supported on Plastic SCM extensions
- Extensions enhancement working on Visual Studio
- Trac Extension: SSPI and SSL connections supported
- Bugzilla Extension
- DevTrac integration improved
- CVS importer enhancement
- Installer enhancement
- Eclipse integration enhancement

Visit www.codicesoftware.com for more information.

PNSQC 2009

MOVING QUALITY FORWARD

October 26-28, 2009
Oregon Convention Center
Portland, Oregon

www.pnsqc.org

27TH ANNUAL
PACIFIC NW SOFTWARE QUALITY CONFERENCE

TestBench for iSeries 6.3

Original Software has announced the availability of TestBench for iSeries 6.3. Customer demand has driven the latest enhancements to TestBench for iSeries which is spearheaded by a new report compare facility, reducing the risk of undetected errors and allowing testers to accurately and quickly identify faults within reports.

Report compare enables testers to compare a new version of a report against a previously validated baseline version, ensuring that when a program change is made, testers can instantly tell if there has been any adverse impact to the report. It will identify key pieces of information, for example invoice numbers or product codes, then do an intelligent comparison of two report runs, identifying additional line items and correctly matching up those that exist in both reports.

Test tools have long struggled with anomalies in test reports. For example, if you had two reports, line one would have been compared with line one and so on. Any error or additional line item would have thrown it all off-kilter. If this happened right at the beginning, the whole report was useless. If your first error appeared on page 999, you may have lost interest after the first fifty pages and assumed it was working correctly. Now, in an industry first, report compare allows testers to identify changes straight away and trap errors they wouldn't have otherwise picked up—a huge potential risk if sending out invoices to customers with wrong invoice values.

Visit www.origsoft.com for more information.

HP, Intel, Yahoo Create Global Cloud Computing Research Test Bed

HP, Intel Corp., and Yahoo have announced the creation of a global, multi-data center, open source test bed for the advancement of cloud computing research and education.

According to the companies, the goal of the initiative is to promote open collaboration among industry, academia, and governments by removing the financial and logistical barriers to research in data-intensive, Internet-scale computing.

The HP, Intel, and Yahoo Cloud Computing Test Bed will provide a globally distributed, Internet-scale testing environment designed to encourage research on the software, data center management, and hardware issues associated with cloud computing at a larger scale than ever before. The initiative will also support research of cloud applications and services.

HP, Intel, and Yahoo noted they have partnered with the Infocomm Development Authority of Singapore (IDA), the University of Illinois at Urbana-Champaign, and the Karlsruhe Institute of Technology (KIT) in Germany to form the research initiative. The partnership with Illinois also includes the National Science Foundation.

Company officials noted the test bed will initially consist of six "centers of excellence" at IDA facilities, the University of Illinois at Urbana-Champaign, the Steinbuch Centre for Computing of the Karlsruhe Institute of Technology, HP Labs, Intel Research, and Yahoo. Each location will host a cloud computing infrastructure, largely based on HP hardware and Intel processors, and will have 1,000 to 4,000 processor cores able to support the data-intensive research associated with cloud computing. The test bed locations are expected to be fully operational and made accessible to researchers worldwide through a selection process later this year.

Supercharge Your Access to StickyMinds.com with a PowerPass!

Open the doors to more software quality information than you've ever seen when you buy a StickyMinds.com PowerPass.

PowerPass-only benefits include a fully searchable STQE and *Better Software* magazine archive, online reference books, conference materials, training discounts, and MUCH more!

Purchase yours today!

www.stickyminds.com/powerpass.asp

Want to receive complimentary copies of some of the latest books on Software Development?

Then you may be interested in the StickyMinds.com Book Review Program!

If you're an experienced software professional who likes to read and thrives on sharing opinions, join our unique book review program that caters exclusively to the software development community!

Tell us about your background, experiences, and which topics you'd like to review. If accepted into the program, you will receive a book selected for you to review—up to four a year. And the best part of the program? You keep the book—No Charge!

For an application or more information, contact Cheryl M. Burke, cburke@sqe.com.

10 Things You Might Not Know About Test Measurement

by Ed Weller

This list can be viewed as a test: Find all the locations where "not" should be added or deleted to make the statements true. If you get lost along the way, the statements following the quotations should set you straight!

1. The best way to measure testers is by the number of bug reports they write. This is true if you change "best" to "worst," as it encourages dysfunctional behavior; quantity over quality.

2. Test productivity can be measured by bug reports per hour. This is true only if "can" is changed to "cannot." This measure ignores severity, priority, and accuracy of the bug reports.

3. Test development productivity can be measured by the number of test cases written per hour. This is less useful than hogsheads per fortnight (unless you are a brewer) as it overlooks the complexity of test cases and that difficult-to-write test cases are often needed to verify complex functions.

4. Anyone can be a tester. The "butterfly net in the street" approach doesn't work for hiring testers. Good testers require system knowledge, the desire to find defects in products, and creative thinking. Measuring the cost of hiring and not the cost of poor testing is a losing strategy.

5. Testers do not need training. This statement assumes either testers know it all or that testing is so simple that anyone can do it (see point 4). Neither could be further from the truth. As with any profession, training helps shorten the time it takes to be productive or skilled.

6. The cost of test is the cost of all those testers. In most organizations, most of the cost of test is the rework cost. Who does the rework—those pesky developers?

7. Test is an overhead cost that does not contribute to the bottom line. Those who think this need only measure the post-ship maintenance cost, lost production cost of customers, and lost customers.

OK, the test is over, and I suspect all of you knew the correct answers without reading the answers I provided. Below are a few more things I have heard asked—or that should be asked—by or of testers and test managers:

8. We delivered 25 percent late because test finished late. And where was most of the time in test spent? Unless you have an exceptionally effective review or inspection process or designers and coders of exceptional skill, testing is a bug hunt, and the time spent is in fixing defects, not testing the product. If you have an open problem backlog, the delays in test are usually in the development camp. Are you measuring backlog? If not, do so!

9. When will you be finished with test? should be answered with "How many faults were inserted into the product?"

10. How good was unit test? Do you get test reports from development with analysis of the unit test activity? If you do not know how well the product was unit tested before you got it, how can you plan your activities and how will you know how good the product is at any point? When you reach the ship readiness review, you may be in a "I do not know" position.

The Tester Who Came In from the Cold

by Lisa Crispin

The stereotypical tester is a lonely, unloved soul waiting for code to drop "over the wall" for testing. She skulks about, searching vainly for someone who cares about actually fixing the bugs. QA teams often labor in their own little gulag, far from the programmers writing the code and even farther from the business people who thought up the requirements. Relationships between testing and coding teams often border on frosty.

Luckily for all us testers, the wall has started to come down, especially in organizations that have embraced agile principles, values, and practices. Countless developers have been "test-infected" and are eager to collaborate with professional testers to deliver the best possible product. The "whole team" approach—where everyone involved in producing software takes responsibility for quality and testing—has proven effective in creating high-quality, bug-free, testable code. Testers are finding the courage to break down the wall and contribute their valuable skills throughout the development lifecycle. Working closely with stakeholders and developers, testers find opportunities to learn and apply new tools and techniques.

As a testing and quality assurance professional, what can you do to come in where it's warm? As a manager, especially in a shop that's moving toward agile development, what can you do to help your QA team enjoy participating as valued members of a quality-focused development team? As a programmer, how can you get testers engaged in your development process?

Find a Common Language

Some of us may have a deep understanding of the business domain but a tenuous grip on object-oriented design concepts. Others may dream in Java but know only a narrow part of the application on which they're working. How can we collaborate if we can't communicate?

Examples are a good way. Ask business experts to illustrate desired system behavior with concrete examples. Have them draw on the whiteboard while you and your fellow developers, QA specialists, and DBAs ask questions. Explain a piece of functionality to a programmer using an executable test.

"Testers are finding the courage to break down the wall and contribute their valuable skills throughout the development lifecycle."

Make Quality the Goal

Five years ago, my team's manager asked us to deliver the best possible software—code we'd be proud to show off. He told us not to worry about deadlines or quantity but to take the time to figure out how we could do the best job. This was hard to believe, but with mutual trust and a lot of time, it happened. Since we took time to do things right, we were able to "work smarter" and save time over the long term. We also had more fun.

Testers can't come in from the cold without support from their managers. It takes time and training to learn how to work with people in other roles and to learn new technical skills. It takes a strong manager to let the team self-organize, solve its own problems, and continually find ways to do a better job.

Address Physical Logistics

For people to work together, they have to be together. If you're on a test team that sits on a different floor from the developers, get up from your desk and go find them—or lure them to your area with chocolate. Plenty of teams have ripped out walls or tiny cubes and made areas where they can collaborate.

This is tougher if you're in different time zones, but technology has come to our rescue. Experiment with the ever-improving tools that allow remote collaboration. Experiment with working different hours so you can at least talk or videoconference with remote team members.

Let Go of Your Fear

Some testers fear losing their identity if they suddenly become part of a team with many roles working together. Some people see developers writing unit tests and think, "If the programmers are doing the testing, what's left for testers?" And it's perfectly reasonable for a QA manager to look at the new "agile" organization and wonder, "Is there a place for me?"

No matter what your development methodology, you still need good people to produce good software—and you need a mix of skills. Testers still need managers to make sure they have the training and support they need. Programmer testing does help deliver defect-free code, but we still need to find out what the business needs and make sure the code provides it. It takes a village to deliver value to the business. Knowing you're not alone will give you courage.

Get Up and Try It

Testers, get up from your desk right now and find something you can do to help your team—virtual or otherwise. Ask a programmer if there's some testable chunk of code ready to look at. Ask a domain expert for some examples of how the new feature will be used. Get online or go to a local user group meeting and learn a new skill that helps you add value to your team.

Those of you who aren't testers, think about how your job might be made easier and how your product might be made better if testers participate from the start of each development cycle.

Break down that wall! {end}

Try our self-paced eLearning course delivered right to your desktop.

Travel from your desktop with Software Quality Engineering's eLearning course, eMastering Test Design. Experience classroom value with the convenience of self-paced instruction on the Web. Because people learn in different ways and everyone learns best with multiple learning options, we've combined audio, video, text, graphics, examples, questions, exercises, and additional learning resources for the best Web-based delivery possible.

eMastering Test Design

Classroom Value with the Convenience of Self-paced Instruction

- Instructed and designed by two of our most experienced instructors, Lee Copeland and Rex Black
- Same valuable information as the 2-day classroom course

The Perfect Solution for Test Practitioners with Travel and Time Constraints

- Complete an eLearning course from your own desktop
- Expert mentors provide answers to your specific questions

Course Content

- Superior lesson material developed and delivered by testing experts
- Tutorials that place content into real-world situations
- Exercises that immediately apply your new learning
- Assessment questions that help you evaluate your learning
- Questions linked to content to reinforce your learning
- Video and audio clips enhance your learning experience
- Web access to an extensive list of additional resources
- Hyperlinks to a glossary of terms used in the course

eLearning

SOFTWARE QUALITY ENGINEERING
www.sqe.com

TAKE A FREE DEMO TODAY! VISIT WWW.SQE.COM/EMTD FOR MORE INFORMATION.

Index to Advertisers

Aculis	www.aculis.com	18
Agile 2008 Conference	www.Agile2008.org	45
Agile Development Practices 2008	www.sqe.com/agiledevpractices	14
AutomatedQA	www.testcomplete.com/try	Inside Back Cover
Blackbaud, Inc.	www.blackbaud.com	23
Cognizant	www.cognizant.com	9
Hewlett-Packard	www.hp.com/go/software	Back Cover
Klocwork	www.klocwork.com	Inside Front Cover
Mountain Goat Software	www.mountaingoatsoftware.com/better	21
PNSQC	www.pnsqc.org	43
Rally Software	www.rallydev.com	19
Rally Software	www.rallydev.com/bsm	44
Ranorex	www.ranorex.com	35
Seapine	www.seapine.com	1
Seapine	www.seapine.com	1
SmartBear	www.smartbear.com?bse	Front Cover
SQE eLearning	www.sqetraining.com/eLearning	48
SQE Testing Training	www.sqetraining/Testing	29
STAR*EAST* 2009	www.sqe.com/ STAREAST	2
Stevens Institute of Technology	www.stevens.edu/sse	42
StickyMinds.com	www.stickyminds.com	41
TargetProcess	www.targetprocess.com	45
TechExcel	www.techexcel.com	5
Web Performance, Inc.	www.webperformance.com	13

Display Advertising
Shae Young syoung@sqe.com

All Other Inquiries
info@bettersoftware.com

Better Software (USPS: 019-578, ISSN: 1532-3579) is published ten times per year. Subscription rate is US $49 per year. A US $35 shipping charge is incurred for all non-US addresses. Payments to Software Quality Engineering must be made in US funds drawn from a US bank. For more information, contact info@bettersoftware.com or call (800) 450-7854. Back issues may be purchased for $15 per issue (plus shipping). Volume discounts available. Entire contents © 2008 by Software Quality Engineering (330 Corporate Way, Suite 300, Orange Park, FL 32073), unless otherwise noted on specific articles. The opinions expressed within the articles and contents herein do not necessarily express those of the publisher (Software Quality Engineering). All rights reserved. No material in this publication may be reproduced in any form without permission. Reprints of individual articles available. Call for details. Periodicals Postage paid in Orange Park, FL, and other mailing offices. POSTMASTER: Send address changes to Better Software, 330 Corporate Way, Suite 300, Orange Park, FL 32073, info@bettersoftware.com.

TestComplete™ Will Blow You AWAY!

Get superior automated testing software for a fraction of what the big guys want to charge you. TestComplete is an award-winning testing solution that gives you functional, regression, unit, load, client server testing and more in one low priced package. It tests any of your Windows applications: Desktop or Web, C++ or VB, .NET or Java, Flash or WPF. Don't throw your money away on overpriced software that does less than TestComplete.

- Award-Winning Features
- Easy Record & Playback
- Great Object Recognition
- Unlimited Extensibility

Download the free trial of TestComplete now!

www.testcomplete.com/try

AutomatedQA
test, debug, deliver!

(978) 236-7900

ALTERNATIVE THINKING ABOUT QUALITY MANAGEMENT SOFTWARE:

Make Foresight 20/20.

Alternative thinking is "Pre." Precaution. Preparation. Prevention. Predestined to send the competition home quivering.

It's proactively designing a way to ensure higher quality in your applications to help you reach your business goals.

It's understanding and locking down requirements ahead of time—because "Well, I guess we should've" just doesn't cut it.

It's quality management software designed to remove the uncertainties and perils of deployments and upgrades, leaving you free to come up with the next big thing.

Technology for better business outcomes. hp.com/go/quality

©2008 Hewlett-Packard Development Company, L.P.

Keynotes by International Experts

Brian Marick
Seven Years Later: What the Agile Manifesto Left Out

Dan North
Beyond Best Practices: Keeping Agile Agile

Pollyanna Pixton
Collaborative Leadership: A Secret to Agile Success

David Anderson
Scaling Agile: Kanban and Beyond...

Conference Sponsor:

RALLY SOFTWARE

100% of 2007 attendees* recommend the Agile Development Practices conference to others in the industry

SOFTWARE QUALITY ENGINEERING CONFERENCES
www.sqe.com

*Based on the 2007 conference attendee evaluations

BETTER SOFTWARE

Agile Development Practices

November 10–14, 2008

Orlando, FL • Shingle Creek Resort

36 Pre-conference Tutorials — *Monday and Tuesday*

46 Keynotes & Classes and Networking EXPO — *Wednesday and Thursday*

Agile Leadership Summit — *Friday*

Agile Leadership Summit: An Executive View of Where Agile Is Heading
Co-located Event — November 14, 2008

Agile LEADERSHIP SUMMIT by APLN

Shingle Creek Resort

Register Early and Save $200!

www.sqe.com/agiledevpractices

BETTER SOFTWARE

Agile Development Practices

November 10–14, 2008

Orlando, FL

Shingle Creek Resort

www.sqe.com/agiledevpractices

Conference Sponsor:

RALLY SOFTWARE

CONTENTS

4 Conference-at-a-Glance
Build Your Own Conference!

6 Special Events

6 The EXPO

7 Conference Speaker Index

8 36 In-Depth Pre-conference Tutorials

14 4 Keynote Presentations

16 42 Concurrent Classes

22 Agile Leadership Summit by APLN

25 Conference Venue and Things to Do in Orlando, FL

26 Conference Sponsors & Exhibitors

27 Registration Information

27 Event Location

27 Ways to Save

TOP TEN REASONS TO ATTEND AGILE DEVELOPMENT PRACTICES

1. More than 95 learning sessions: tutorials, keynotes, conference and summit sessions, and more
2. In-depth tutorials, half- and full-day options
3. Cutting-edge answers from top agile experts
4. Presentations from highly experienced agile professionals and thought leaders
5. Opportunities for networking with your peers in the industry
6. Special events—welcome reception, Open Space, book signings, Meet the Speakers, and more
7. Co-located with the Agile Leadership Summit, brought to you by the Agile Project Leadership Network (APLN)
8. Group discounts—bring your whole team
9. EXPO—find solutions to your agile development challenges
10. The perfect balance of learning and fun at the beautiful Shingle Creek Resort in Orlando, FL

WHO'S BEHIND THE CONFERENCE?

SOFTWARE QUALITY ENGINEERING
www.sqe.com

Software Quality Engineering assists professionals interested in improving software practices. Four conferences are hosted annually—the STAR conference series, the Better Software Conference & EXPO, and Agile Development Practices. Software Quality Engineering also delivers software training, publications, and research. **www.sqe.com**

BETTER SOFTWARE MAGAZINE

Better Software magazine brings you the hands-on facts you need to run smarter projects and to deliver better products that win in the marketplace. **www.BetterSoftware.com**

StickyMinds.com

StickyMinds.com is a complete online resource to help you produce better software. It offers original articles from industry experts, technical papers, industry news, a tools guide, forums, and much more. **www.StickyMinds.com**

WHO SHOULD ATTEND

Anyone investigating or implementing agile development practices, processes, technologies, and leadership principles.

Managers: Software Managers, CIOs, CTOs, Project Managers, Test & QA Managers, and Business Managers

Developers and Engineers: Technical Project Leaders, Senior Developers, Senior Testers, Business Analysts, Process Improvement Staff, and Auditors

BRING A BUDDY!

Bring a colleague and each of you saves an additional $200.

Any two people registering at the same time save an additional $200 off each registration.

Please call the Client Support Group at 888.268.8770 or 904.278.0524 to take advantage of this offer.

CONFERENCE SCHEDULE

Build your own conference—keynote presentations, pre-conference tutorials, concurrent sessions, summit sessions, and more—packed with information covering the latest technologies, trends, and practices in agile development.

Monday	Tuesday	Wednesday	Thursday	Friday
36 Pre-conference Tutorials in both full- and half-day formats		4 Keynote Presentations 42 Concurrent Classes Networking EXPO Special Events ...and More!		Agile Leadership Summit by APLN

Agile Leadership Summit: An Executive View of Where Agile Is Heading

November 14, 2008

By Agile Project Leadership Network (APLN)

New to 2008! Add a fifth day to your conference event and attend the co-located Agile Leadership Summit, in cooperation with the APLN. The Agile Leadership Summit gives you the opportunity to learn from industry leaders who have embraced agile principles and put them to work successfully in their organizations. See **page 22** for more information on the Agile Leadership Summit.

CONFERENCE-AT-A-GLANCE

MONDAY, NOVEMBER 10

8:30 Tutorial Sessions (8:30 a.m. - 12:00 p.m.)

FULL-DAY TUTORIALS

MA NEW **The Zen of Agile Management**
David Anderson, Valtech, Inc. and Modus Cooperandi, Inc.

MB **Principles and Practices of Lean-Agile Development**
Alan Shalloway, Net Objectives

MC **Test-Driven Development**
Rob Myers, Net Objectives

MD NEW **Using Metrics in Agile Environments**
Michael Mah, QSM Associates, Inc.

ME **Agile Requirements Interactive**
Ken Pugh, Net Objectives

MF NEW **Testing on Agile Projects**
Antony Marcano, testingReflections.com and Rachel Davies, Agile Experience

MORNING HALF-DAY TUTORIALS

MG NEW **Experiencing Agility from Requirements to Planning**
Mike Cohn, Mountain Goat Software

MH NEW **The Beginner's Mind: Keeping Your Agile Adoption Fresh**
David Hussman, DevJam and Jean Tabaka, Rally Software Development

MI NEW **Conducting Requirements-Driven Workshops for Agile Projects**
Ellen Gottesdiener, EBG Consulting

MJ NEW **User Story Mapping: Getting the Big Picture**
Jeff Patton, Independent Consultant

MK NEW **The Power of Retrospectives**
Linda Rising, Independent Consultant

ML NEW **Fostering Trust in Teams—A Leadership Practicum**
Pollyanna Pixton, Accelinnova

12:00 Lunch

1:00 Tutorial Sessions (1:00 p.m. - 4:30 p.m.)

FULL-DAY TUTORIALS

MA NEW **The Zen of Agile Management** *CONTINUED FROM MORNING*
David Anderson, Valtech, Inc. and Modus Cooperandi, Inc.

MB **Principles and Practices of Lean-Agile Development** *CONTINUED FROM MORNING*
Alan Shalloway, Net Objectives

MC **Test-Driven Development** *CONTINUED FROM MORNING*
Rob Myers, Net Objectives

MD NEW **Using Metrics in Agile Environments** *CONTINUED FROM MORNING*
Michael Mah, QSM Associates, Inc.

ME **Agile Requirements Interactive** *CONTINUED FROM MORNING*
Ken Pugh, Net Objectives

MF NEW **Testing on Agile Projects** *CONTINUED FROM MORNING*
Antony Marcano, testingReflections.com and Rachel Davies, Agile Experience

AFTERNOON HALF-DAY TUTORIALS

MM NEW **ADAPTing to Agile: A Guide to Transitioning**
Mike Cohn, Mountain Goat Software

MN NEW **Agile Product Planning: Building a Strong Backlog**
David Hussman, DevJam

MO NEW **A Day in the Life of a User Story**
Jean Tabaka, Rally Software Development

MP **Refactoring Your Wetware: Thinking about Thinking**
Andy Hunt, The Pragmatic Programmers

MQ NEW **Mixing Roles in Scrum: The Good, The Bad, and The Ugly**
Mitch Lacey, Mitch Lacey & Associates, Inc.

MR NEW **Designing Examples: Story Tests for Story Success**
J. B. Rainsberger, Independent Consultant

TUESDAY, NOVEMBER 11

8:30 Tutorial Sessions (8:30 a.m. - 12:00 p.m.)

FULL-DAY TUTORIALS

TA NEW **Lean Software Development: Mapping the Value Stream**
Mary Poppendieck, Poppendieck, LLC

TB NEW **Design Patterns Explained: From Analysis to Implementation**
Alan Shalloway, Net Objectives

TC NEW **Successful Agile Requirements: Collaborating to Define and Confirm Needs**
Ellen Gottesdiener, EBG Consulting

TD **Fearless Change: Introducing New Ideas**
Linda Rising, Independent Consultant

TE **Collaboration Explained: Facilitation Skills for Project Managers**
Jean Tabaka, Rally Software Development

TF NEW **Growing Agile Coaches**
David Hussman, DevJam

MORNING HALF-DAY TUTORIALS

TG NEW **Getting Agile with Scrum**
Mike Cohn, Mountain Goat Software

TH **Leading Successful Projects in Changing Environments**
Pollyanna Pixton, Accelinnova

TI NEW **Measuring and Using Your Team's Velocity**
Rob Myers, Net Objectives

TJ NEW **Hiring for an Agile Team: Candidates Who Fit**
Johanna Rothman, Rothman Consulting Group

TK NEW **Behavior-Driven Development: Writing Software that Matters**
Dan North, ThoughtWorks

TL NEW **xUnit Test Patterns and Smells: Improving Test Code and Testability Through Refactoring** — *Gerard Meszaros, Solution Frameworks, Inc.*

12:00 Lunch

1:00 Tutorial Sessions (1:00 p.m. - 4:30 p.m.)

FULL-DAY TUTORIALS

TA NEW **Lean Software Development: Mapping the Value Stream**
Mary Poppendieck, Poppendieck, LLC *CONTINUED FROM MORNING*

TB NEW **Design Patterns Explained: From Analysis to Implementation**
Alan Shalloway, Net Objectives *CONTINUED FROM MORNING*

TC NEW **Successful Agile Requirements: Collaborating to Define and Confirm Needs**
Ellen Gottesdiener, EBG Consulting *CONTINUED FROM MORNING*

TD **Fearless Change: Introducing New Ideas** *CONTINUED FROM MORNING*
Linda Rising, Independent Consultant

TE **Collaboration Explained: Facilitation Skills for Project Managers**
Jean Tabaka, Rally Software Development *CONTINUED FROM MORNING*

TF NEW **Growing Agile Coaches** *CONTINUED FROM MORNING*
David Hussman, DevJam

AFTERNOON HALF-DAY TUTORIALS

TM NEW **Avoid Integration Defects without Integration Testing**
J. B. Rainsberger, Independent Consultant

TN **From User Story to User Interface**
Jeff Patton, Independent Consultant

TO NEW **The Bridge to Agility for Traditional Project Managers**
Michele Sliger, Sliger Consulting, Inc.

TP NEW **Software Configuration Management for Agile Development**
Steve Berczuk, Cyrus Innovation

TQ NEW **Agile Estimating and Planning**
Mike Cohn, Mountain Goat Software

TR NEW **Practical Agile: Real World Practices**
Jared Richardson, 6th Sense Analytics

4:30 Welcome Reception 4:30 p.m. - 6:00 p.m.

WEDNESDAY, NOVEMBER 12

8:30 **Opening Remarks —** *Conference Chair - Lee Copeland, Software Quality Engineering*

8:45 **Seven Years Later: What the Agile Manifesto Left Out —** *Brian Marick, Exampler Consulting*

9:45 Morning Break

	Transitioning to Agile	Agile Management	Agile Projects	Agile Processes	The Agile Organization	Agile Design & Architecture	Requirements
10:00	W1 **Overcoming the Pitfalls of Transitioning to Agile** *Johanna Rothman, Rothman Consulting Group*	W2 **Are We There Yet? Defining "Done"** *Mitch Lacey, Mitch Lacey & Associates, Inc.*	W3 **From Concept to Product Backlog: What Happens Before Iteration Zero** *Gerard Meszaros, Solution Frameworks, Inc.*	W4 **Test-Driven Everything** *David Hussman, DevJam*	W5 **What Are They Doing Down There? A CIO's Perspective on Agile Software Development** *Niel Nickolaisen, Headwaters, Inc.*	W6 **Integrating Enterprise SOA Architecture with Scrum Development Methodology** *Steven Driver, Airlines Reporting Corporation*	W7 **Agile for Business Analysts** *Bob Hartman, Agile For All*
10:00	Open Space 10:00 a.m - 4:15 p.m. *(see page 6 for more information)*						
11:30	Lunch in the EXPO *(EXPO open 10:00 a.m. - 3:00 p.m.)*						
12:45	W8 **The Agile PMP: Teaching an Old Dog New Tricks** *Mike Cottmeyer, VersionOne*	W9 **Agile Contracting** *Rachel Weston, Rally Software Development and Chris Spagnuolo, Data Transfer Solutions*	W10 **Re-thinking Scheduling: Parkinson's Law Inverted** *Mary Poppendieck, Poppendieck, LLC*	W11 **Pragmatic Agility** *Andy Hunt, The Pragmatic Programmers*	W12 **It Takes a Village: Organizing to Fulfill the Product Owner Role** *Ronica Roth, Rally Software Development*	W13 **Refactoring: Where Do I Start?** *J. B. Rainsberger, Independent Consultant*	W14 **Driving User Stories from Business Value** *Guy Beaver, Net Objectives*
2:15	Networking Break in the EXPO						
2:45	W15 **Driving Agile Transformation from the Top Down** *Pete Morowski, Borland Software Corporation*	W16 **Calling All Agile Skeptics—the Curious and Die-Hard, Non-Agile** *Damon Poole, AccuRev*	W17 **Agile Project Metrics** *Dave Nicolette, Valtech Technologies*	W18 **Secrets of CMMI® for Agile Organizations** *Jeff Dalton, Broadsword Solutions*	W19 **Value Stream Mapping: Extending Our View to the Enterprise** *Alan Shalloway, Net Objectives*	W20 **Behavior-Driven Database Design** *Pramod Sadalage, ThoughtWorks*	W21 **Do the Right Thing: Adapting Requirements Practices for Agile Projects** *Ellen Gottesdiener, EBG Consulting*
4:15	Networking Break in the EXPO *(EXPO open 4:00 p.m. - 6:30 p.m.)*						
4:30	**Beyond Best Practices: Keeping Agile Agile —** *Dan North, ThoughtWorks*						
5:30	Reception in the EXPO, 5:30 p.m. – 6:30 p.m.						

THURSDAY, NOVEMBER 13

8:30 **Opening Remarks —** *Conference Chair - Lee Copeland, Software Quality Engineering*

8:45 **Collaborative Leadership: A Secret to Agile Success —** *Pollyanna Pixton, Accelinnova*

9:45 Networking Break in the EXPO *(EXPO open 9:30 a.m. - 2:45 p.m.)*

	People & Teams	Agile Management	Agile Projects	Agile Processes	Testing	Agile Design & Architecture	Special Topics
10:00	T1 **Who Do You Trust? Beware of Your Brain** *Linda Rising, Independent Consultant*	T2 **Selling Agile: Getting Buy-In from Your Team, Customers, and Managers** *Michele Sliger, Sliger Consulting, Inc.*	T3 **When to Step Up, When to Step Back: How to Lead Collaboration** *Pollyanna Pixton, Accelinnova*	T4 **Making People and Processes Congruent** *Ken Pugh, Net Objectives*	T5 **Agile Software Testing Strategies** *Jared Richardson, 6th Sense Analytics*	T6 **Test-Driven Development Takes on Embedded Software** *James Grenning, Renaissance Software Consulting*	T7 **Assessing Your Agility** *Mike Cohn, Mountain Goat Software, and Kenny Rubin, Innolution*
10:00	Open Space 10:00 a.m - 4:15 p.m. *(see page 6 for more information)*						
11:30	Lunch in the EXPO						
12:45	T8 **"With Great Power Comes Great Responsibility"— Empowering the Agile Team** *V. Lee Henson, VersionOne*	T9 **A Lean Approach to Managing the Project Portfolio** *Johanna Rothman, Rothman Consulting Group*	T10 **Maximizing Team Dynamics and Overcoming Dysfunction in Agile Environments** *Michael Mah, QSM Associates, Inc.*	T11 **The Business Value of Pair Programming** *Rob Myers, Net Objectives*	T12 **Agile Usability Testing** *John De Goes, N-BRAIN, Inc.*	T13 **Getting Agile with Legacy Code** *Steve Berczuk, Cyrus Innovation*	T14 **Agile Project Inception: Escaping the Waterfall** *Kenny Rubin, Innolution*
2:15	Networking Break in the EXPO *(EXPO open until 2:45 p.m.)*						
2:45	T15 **Mistakes Agile Teams Make** *J. B. Rainsberger, Independent Consultant*	T16 **Agile Growing Pains** *Michael Kirby, Xerox*	T17 **Scaling Agile Up and Out: A Tale from the Trenches** *Ade Miller, Microsoft Corporation*	T18 **Retrospectives in Action: Looking Back at the Conference** *Jean Tabaka, Rally Software Development*	T19 **Picking the Right Test Automation Strategy for Your Project** *Gerard Meszaros, Solution Frameworks, Inc.*	T20 **Agile Engineering for Architects** *Ryan Shriver, Dominion Digital*	T21 **Introduction to Multi-Stage Continuous Integration** *Damon Poole, AccuRev*
4:15	Networking Break						
4:30	**Scaling Agile: Kanban and Beyond... —** *David Anderson, Valtech, Inc. and Modus Cooperandi, Inc.*						

FRIDAY, NOVEMBER 14

Agile Leadership Summit: An Executive View of Where Agile is Heading

Add a fifth day to your conference event and attend the co-located Agile Leadership Summit, in cooperation with APLN. See page 22 for more information.

SPECIAL EVENTS

Welcome Reception

Tuesday, November 11, 4:30 p.m. - 6:00 p.m.

Kick off the Agile Development Practices conference with a Welcome Reception sponsored by Rally. Mingle with experts and colleagues, and enjoy complimentary food and beverages.

EXPO Reception

Wednesday, November 12, 5:30 p.m. - 6:30 p.m.

Socialize with others and enjoy complimentary food and beverages in the EXPO hall Wednesday evening. Look for answers to your pressing issues and meet other attendees with the same challenges. Also be sure to take part in the fun and games held at the EXPO for a chance to win fabulous prizes.

Bookstore and Speaker Book Signings

During EXPO hours, purchase popular industry books—many authored by Agile Development Practices speakers—from BreakPoint Books. Authors are available for questions and book signings during session breaks and EXPO hours.

OPEN SPACE

Wednesday and Thursday, November 12–13, During the EXPO

Want to discuss a topic in greater depth? Examine a topic that is not on the program? Meet with others who are also interested? Great! You're looking for Open Space. We supply an open space—a room, tables, chairs, and flipcharts...you supply the ideas and the leadership. Choose a topic you'd like to discuss, pick a time slot, promote your topic at the conference, enroll others, and have an Open Space discussion. That's what Open Space is all about—you are in charge of your own learning. Everyone who comes to an Open Space session should be passionate about the topic and willing to take some responsibility for creating learning out of that passion.

Don't be passive—create your own conference session on a topic of interest to you. If you've never tried Open Space, do it now. You'll teach and you'll learn.

An Open Space wiki will be available before the conference for delegates to sign up and propose topics. Check the Web site for updates to come. **www.sqe.com/agiledevpractices**

THE EXPO *November 12–13, 2008*

Visit Top Industry Providers Offering the Latest in Agile Development Solutions

Looking for answers? Take time to explore this one-of-a-kind EXPO, designed to bring you the latest solutions in technologies, software, and tools covering all aspects of agile software development. Throughout the EXPO, participate in technical presentations and demonstrations to help you find the tools and services you need to support and improve your software projects. Meet one-on-one with representatives from some of today's most progressive and innovative organizations.

EXPO Hours

Wednesday, November 12

10:00 a.m. – 3:00 p.m.

4:00 p.m. – 6:30 p.m.

Reception:

5:30 p.m. – 6:30 p.m.

All attendees are invited to the EXPO reception for complimentary food and beverages—and fun and games.

Thursday, November 13

9:30 a.m. – 2:45 p.m.

For Sponsor/Exhibitor news and updates, visit **www.sqe.com/agiledevpractices**

CONFERENCE BONUS! One-Year Digital Subscription to *Better Software* Magazine!

Agile Development Practices attendees receive a one-year digital subscription (ten issues) to *Better Software* magazine—the only magazine delivering relevant, timely information so you can tackle the challenges of building better quality software, regardless of your role in software development. www.BetterSoftware.com

If you are a current subscriber, your subscription will be extended an additional ten digital issues.

K Keynote Speaker T Tutorial Speaker C Class Speaker

C
Ade Miller
Microsoft Corporation

K T
David Anderson
Valtech, Inc. and Modus Cooperandi, Inc.

T
Jeff Patton
Independent Consultant

T C
Michele Sliger
Sliger Consulting, Inc.

T C
Rob Myers
Net Objectives

T C
Alan Shalloway
Net Objectives

T C
David Hussman
DevJam

T C
Johanna Rothman
Rothman Consulting Group

T C
Mike Cohn
Mountain Goat Software

C
Ronica Roth
Rally Software Development

T C
Andy Hunt
The Pragmatic Programmers

T C
Ellen Gottesdiener
EBG Consulting

C
John De Goes
N-BRAIN, Inc.

C
Mike Cottmeyer
VersionOne

C
Ryan Shriver
Dominion Digital

C
Antony Marcano
testingReflections .com

T C
Gerard Meszaros
Solution Frameworks, Inc.

T C
Ken Pugh
Net Objectives

T C
Mitch Lacey
Mitch Lacey & Associates, Inc.

T C
Steve Berczuk
Cyrus Innovation

C
Bob Hartman
Agile For All

C
Guy Beaver
Net Objectives

C
Kenny Rubin
Innolution

C
Niel Nickolaisen
Headwaters, Inc.

C
Steven Driver
Airlines Reporting Corporation

K
Brian Marick
Exampler Consulting

T C
J. B. Rainsberger
Independent Consultant

Conference Chair
Lee Copeland
Software Quality Engineering

C
Pete Morowski
Borland Software Corporation

C
V. Lee Henson
VersionOne

C
Chris Spagnuolo
Data Transfer Solutions

C
James Grenning
Renaissance Software Consulting

T C
Linda Rising
Independent Consultant

K T C
Pollyanna Pixton
Accelinnova

C
Damon Poole
AccuRev

T C
Jared Richardson
6th Sense Analytics

T C
Mary Poppendieck
Poppendieck, LLC

C
Pramod Sadalage
ThoughtWorks

K T
Dan North
ThoughtWorks

T C
Jean Tabaka
Rally Software Development

C
Michael Kirby
Xerox

T
Rachel Davies
Agile Experience

C
Dave Nicolette
Valtech Technologies

C
Jeff Dalton
Broadsword Solutions

T C
Michael Mah
QSM Associates, Inc.

C
Rachel Weston
Rally Software Development

MONDAY, NOVEMBER 10, 8:30-4:30 FULL-DAY TUTORIALS

MA The Zen of Agile Management NEW

David Anderson, Valtech, Inc. and Modus Cooperandi, Inc.

What is the essence of agile management? Empowerment, delegation, trust, continuous improvement—and a consistent focus on the leverage points to get the maximum advantage from agile—all with a light touch. Learn the Zen of agile management from David Anderson as he shares the techniques he's honed managing teams at Motorola and Sprint. While working as Senior Director of Software Engineering at Corbis, David pioneered the introduction of Theory of Constraints and Lean ideas such as kanban to consistently deliver value to customers. You'll explore ways to manage with queues using kanban boards, identify and eliminate bottlenecks, and reduce variability using cumulative flow diagrams. Learn about additional metrics, measures, and indicators you can deliver to line, middle, and upper management to keep them informed and onboard with the project. Take back the keys for building a kaizen culture of continuous improvement and ways to institutionalize enterprise-scale agile change in your development team and organization.

***David Anderson** is chief process scientist with Valtech and president of Modus Cooperandi, a management consulting firm based in Seattle, Washington. He has many years of management experience leading teams on agile development projects, most recently as senior director for Software Engineering at Corbis in Seattle. David was a founder of the agile movement through his involvement in the creation of feature-driven development in Singapore in the late 1990s. He can be contacted at david.anderson@valtech.com.*

MB Principles and Practices of Lean-Agile Development

Alan Shalloway, Net Objectives

As the popularity of agile development spreads, more and more companies are discovering that simply breaking down projects into small iterations is not sufficient. Agile methods require changes in management, analysis, architecture, design, testing, and quality assurance, as well as project management. Given the substantial adjustments required, where can a team or enterprise look for guidance in its transition? Learning the required skill sets individually is fraught with problems—analysis, design, code, and test are not independent; they must be integrated. Join Alan Shalloway as he describes the landscape of skills that a development team needs to become effective agile developers. He discusses a set of principles and practices that integrate the guidance provided by lean, agile methods, design patterns, and more. In particular, Alan details how agile analysis and design patterns support agile methods and how core "lean" principles support all agile methods, including design and test-driven development.

***Alan Shalloway** is the founder and CEO of Net Objectives. With more than thirty-five years of experience, Alan is an industry thought leader, trainer, and coach in the areas of lean software development, the lean-agile connection, Scrum, agile architecture and using design patterns in agile environments. He is a popular speaker at prestigious conferences worldwide as well as a trainer/coach. Alan is the primary author of* Design Patterns Explained: A New Perspective on Object-Oriented Design *and is currently writing a book on Lean Anti-Patterns.*

MC Test-Driven Development

Rob Myers, Net Objectives

Test-Driven Development (TDD) is a powerful technique for combining software design, testing, and coding to increase reliability and productivity. Rob Myers demonstrates the basic and essential TDD techniques, including unit testing with the common xUnit family of open source development frameworks, refactoring code, and using mock/fake objects in development. Use exercises to practice the techniques. With many years of product development experience using TDD, Rob will address the questions that arise during your own relaxed exploration of the techniques.

LAPTOP REQUIRED

LAPTOP REQUIRED. Attendees should have strong programming skills and be familiar with an object-oriented language and programming techniques. Each delegate should bring a laptop installed with your favorite programming language and IDE—and come prepared to write code. Rob can provide JUnit for Java and NUnit for any .NET language. For any other language choice (e.g., C++ or Ruby), you will need to install (and verify) your chosen xUnit framework prior to the tutorial.

***Rob Myers** has more than twenty years of professional experience in software development, including projects for industry leaders in medical, aerospace, and financial services. In the late 1990s, Rob became an eXtreme Programming coach and traveled throughout the U.S. and abroad assisting teams with agile practices and object-oriented design techniques. He teaches Test-Driven Development, Design Patterns Explained, Implementing Scrum for Your Team, Lean-Agile Testing, and other courses. Rob brings to each classroom his passion for Lean software development, team development, and sane work environments.*

MD Using Metrics in Agile Environments NEW

Michael Mah, QSM Associates, Inc.

How do you compare the productivity and quality you achieve with agile practices versus traditional, more waterfall projects? Join Michael Mah to learn about both agile and waterfall metrics, and how these metrics behave in the real projects. Learn how to use your own data to move from guesses on a project whiteboard to realistic agile project trends on productivity, time-to-market, and defect rates. Using recent, real-world case studies, Michael offers an inside look at agile measurements by showing you these metrics in action. In hands-on exercises, you will learn how to replicate these techniques to make your own comparisons for time, cost, and quality. Working in pairs, you will calculate productivity metrics using the templates Michael employs in his consulting practice. You can leverage these new metrics to make the case for changing to more agile practices at your company. Take back new ways for communicating to key decision makers the value of implementing agile development practices.

LAPTOP OPTIONAL

LAPTOP RECOMMENDED. To take full advantage of this session, participants should bring a laptop computer for metrics capture and productivity calculations.

***Michael Mah** is director of the Benchmarking Practice, an author with the Cutter Consortium, and managing partner of QSM Associates, Inc., specializing in software measurement and project estimation. Michael has written extensively and consulted with the world's leading software organizations while collecting data on thousands of projects worldwide. Michael's book-in-progress,* Optimal Friction, *examines the dynamics of teams under time pressure and its role in contributing to success and failure. He lives in the mountains of western Massachusetts with his two young children. Michael can be reached at www.qsma.com.*

ME Agile Requirements Interactive

Ken Pugh, Net Objectives

All projects, whether agile or traditional, need requirements. Ken Pugh explores the differences between agile and traditional requirements by interactively creating a set of agile-style requirements. These requirements are developed through progressive elaboration—rather than the big-bang, big-document approach. Ken first examines with you how stakeholders and requirements gatherers interact and communicate in an agile environment. Delegates will create a charter for a project that defines the overall scope and participate in a story-gathering workshop to create an initial set of stories. Learn when and how to revise stories by chunking and de-chunking to ensure that the requirements fulfill the characteristics of good stories. Explore user roles, personas, and narratives to determine additional stories. Practice prioritizing the requirements and estimating their business value to help in that prioritization. At the end of the session, delegates will begin constructing use cases and acceptance tests to add details to the requirements.

*A fellow consultant with Net Objectives, **Ken Pugh** has more than one-third of a century of experience in software development—from gathering requirements for stock market analysis to testing real-time radar systems. Ken consults, trains, testifies, and mentors from London to Sydney on agile processes and technology topics ranging from object-oriented design and test-driven development to Linux/Unix. He has written several programming books, including the Jolt Award winner,* Prefactoring, *and* Interface Oriented Design. *When not computing, Ken enjoys snowboarding, windsurfing, biking, and hiking the Appalachian Trail. Ken can be reached at ken.pugh@netobjectives.com.*

MF Testing on Agile Projects NEW

Antony Marcano, testingReflections.com and Rachel Davies, Agile Experience

Testing plays a major role for everyone on an agile development team-analysts, developers, testers, and product owners. Testing touches every facet of the process, helping the team understand requirements and scope with acceptance tests, guiding design with test-driven development, and providing feedback about the software with exploratory and automated testing. Unfortunately, agile team members often have limited access to training and documentation to help them take advantage of all that testing can bring to the team. Antony Marcano and Rachel Davies, who share a passion for agile development and testing, deliver a highly engaging and interactive session exploring the roles of testing on agile projects. Learn to use automated acceptance tests that drive development. Use feedback from testing to evolve the software and determine when the software is ready for release-and practice providing this feedback in the most constructive way. Develop your understanding of how exploratory testing fits in the picture. Leave with new skills to share with your team.

***Antony Marcano** has thirteen years of industry experience, specializing in software testing, the last eight of which as a practitioner on agile projects. Now, more often as a coach and trainer, Antony helps teams evolve their skills and abilities to attain the benefits offered by agile practices. Antony's specialities include User Stories, Acceptance Test-Driven Development and Test-Driven Development. Antony is also creator and curator of testingReflections.com, one of the most influential agile testing sites on the Internet for developers and testers, and is a Technical Editor for* Better Software *magazine. You can contact Antony at antony.marcano@testingReflections.com.*

*Based in the UK with eight years of agile experience, **Rachel Davies** provides expert coaching to teams in agile software development techniques, including test-driven development, heartbeat retrospectives, and planning with user stories. She is passionate about agile software development because it increases the chance of successful projects in the face of complex problems. A past chair of the Agile Alliance, Rachel is internationally recognized in the agile community as a frequent presenter at industry conferences worldwide. You can contact Rachel at rachel@agilexp.com.*

HALF-DAY TUTORIALS **MONDAY,** NOVEMBER 10, 8:30-12:00

MG Experiencing Agility from Requirements to Planning NEW

Mike Cohn, Mountain Goat Software

During this highly interactive session, you will experience an agile project from inception through its first iteration. Put theory into action working in small groups to create interesting case studies with product backlogs. Start by writing the high-level epic user stories that might be used to gain funding for a real-life project. Because no project proposal is likely to be accepted without a schedule, you'll learn how to estimate with story points and then forecast the team's velocity. You'll use these techniques on your product backlog to establish a baseline release plan for your project. Knowing it is important to work in priority order when a project begins, you'll learn the four essential factors to consider when prioritizing feature delivery. Practice using these factors to divide epic user stories into smaller user stories more suitable for the development team to produce during iterations.

*A Certified Scrum trainer, **Mike Cohn** is the founder of Mountain Goat Software, a process and project management consultancy and training firm. He is the author of* Agile Estimating and Planning *and* User Stories Applied for Agile Software Development, *as well as books on Java and C++ programming. With more than twenty years of experience, Mike has previously been a technology executive in companies of various sizes—from startup to Fortune 40. A frequent magazine contributor and conference speaker, Mike is a founding member of the Scrum Alliance and the Agile Alliance. He can be reached at mike@mountaingoatsoftware.com.*

MH The Beginner's Mind: Keeping Your Agile Adoption Fresh NEW

David Hussman, DevJam and Jean Tabaka, Rally Software Development

Join David Hussman and Jean Tabaka on a journey that guides participants in how the best agile teams truly engage and adapt. By investigating the notion of "Beginner's Mind" versus "Expert Mind" through interactive exercises, David and Jean invite you to embrace an agile adoption approach that keeps your mind in the present, open to new ideas, and always curious. They "open their agile kimono" by sharing experiences of teams that have successfully adopted Beginner's Mind versus teams that unfortunately embraced Expert Mind. David and Jean challenge you about your team's decision styles, agile practices, and notions of best practices—any of which can inadvertently invite the damaging blinders of Expert Mind. They invite discussion within small groups about how Expert Mind practices actually impede agile maturity and keep teams from gelling. Finally, they invite you to create and share a set of Beginner's Mind practices with other participants in a quiet reflection of how to stay fresh. Bring your experiences and curiosity—and expect to be surprised!

*For many years, **David Hussman** has led software projects in a variety of domains—digital audio, digital biometrics, medical, government, legal, security, industrial, financial, retail, and education to name a few. David now spends his time coaching and leading agile project communities worldwide. The author of* Cutting an Agile Groove *and contributor to several books, including* Managing Agile Projects *and* Agile in the Large, *David leads DevJam. As mentors and practitioners, DevJam focuses on using agile to help people and companies improve their software production skills. DevJam (www.devjam.com) provides seasoned leaders that strive to pragmatically match technology, people, and processes to create better and cooler products.*

*An agile coach with Rally Software Development, **Jean Tabaka** has more than twenty-five years of experience in IT. After studying DSDM in the late 1990s, Jean became an agile devotee, working with organizations worldwide to deliver more value faster through the adoption of agile principles and practices. Specializing in scaling agile practices, guiding leadership shifts, applying lean, and building continuous planning practices, Jean uses a collaborative approach in helping organizations adopt agile. A Certified ScrumMaster Trainer and a Certified Professional Facilitator, Jean is the author of* Collaboration Explained: Facilitation Skills for Software Project Leaders. *You can reach her at jean.tabaka@rallydev.com.*

MI Conducting Requirements-Driven Workshops for Agile Projects NEW

Ellen Gottesdiener, EBG Consulting

Learn the essential skills you need for planning and conducting a series of three agile workshops—*Product Roadmap, Release Plan, and Iteration Requirements*—for large or complex agile projects. Find out how to generate "just enough" requirements information at the right time and for the right stakeholders. Avoid the struggles some agile teams have grasping enough of the big picture to mitigate the myriad of problems that can arise—guessing which slice of the product to start with or build next, needing extensive rework due to undetected architectural dependencies, establishing a viable release strategy for business planning, and suffering from inadequate customer involvement. These three facilitated workshops help you combine "grokking" your product requirements with the benefits of a collaborative process that builds trust, mutual understanding, and accountability. Ellen Gottesdiener shares her toolkit of guidelines and practices—reinforced with practice sessions and group discussions—to improve the quality of your large agile product development efforts.

*Principal Consultant of EBG Consulting, **Ellen Gottesdiener** helps business and technical teams get product requirements right so their projects start smart and deliver the right product at the right time. An agile coach and trainer with a passion about agile requirements, she works with large, complex products and helps teams elicit just enough requirements to achieve iteration and product goals. Ellen is the author of* Requirements by Collaboration: Workshops for Defining Needs *and* The Software Requirements Memory Jogger. *Ellen writes articles, speaks, and advises at industry conferences, and provides training seminars to both traditional and agile clients. Contact her at www.ebgconsulting.com.*

MJ User Story Mapping: Getting the Big Picture NEW

Jeff Patton, Independent Consultant

Is your agile project buried under a mountain of user stories? As you add stories, does your vision of the product you are building grow hazier? As story count increases, do business stakeholders become more frustrated with priorities? Do you find it difficult to communicate the big picture about what the system does? User story mapping helps agile teams create a simple model that places user stories in the context of a complete system. With a story map in hand, you'll be able to see the big picture—the breadth of functionality the product implements, the users it serves, and the activities in which they engage. Jeff Patton shows how to visually prioritize user stories and create realistic, incremental release plans. Learn the essentials of story mapping, story splitting, story thinning, and incremental planning. Discover the characteristics of a good user story and how those characteristics differ when you are writing stories for planning versus development. With a living story map, all stakeholders—management, developers, and end users—will, for the first time, see a complete view of the entire system before you build it.

*For the past twelve years, **Jeff Patton** has designed and developed software on a wide variety of projects from online aircraft parts ordering to electronic medical records. A winner of the Agile Alliance's 2007 Gordon Pask Award for contributions to agile development, Jeff has focused on agile approaches since working on an early Extreme Programming team in 2000. Jeff has specialized in the application of user-centered design techniques to improve agile requirements, planning, and products. Some of his recent writing on the subject can be found at www.agileproductdesign.com and Alistair Cockburn's Crystal Clear. His forthcoming book gives tactical advice to those seeking to deliver useful, usable, and valuable software.*

MK The Power of Retrospectives NEW

Linda Rising, Independent Consultant

One of the Agile Manifesto principles states, "At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly." How can this best be done? Join Linda Rising to learn why project retrospectives are a critical part of your agile practices. By taking time to reflect, learn, and proactively decide what the team should do differently—in the next iteration, release, or project—teams discover what they're doing well so that successful practices can continue and identify what should be done differently to improve performance. Retrospectives are not finger pointing or blaming sessions. Rather, they are highly positive and interactive sessions in which teams reflect on the past to become better in the future. Linda shares her experiences leading several types of retrospectives for dozens of projects—both small and large. Her lessons learned will help your project and team become a true learning organization.

***Linda Rising** has a Ph.D. from Arizona State University in the field of object-based design metrics and a background that includes university teaching and industry work in telecommunications, avionics, and strategic weapons systems. An internationally known presenter on topics related to patterns, retrospectives, and the change process, Linda is the author of* Design Patterns in Communications; The Pattern Almanac 2000, A Patterns Handbook; *and co-author with Mary Lynn Manns of* Fearless Change: Patterns for Introducing New Ideas. *Find more information about Linda at www.lindarising.org.*

ML Fostering Trust in Teams—A Leadership Practicum NEW

Pollyanna Pixton, Accelinnova

In our business and personal lives, many of us know leaders who foster environments with incredible creativity, innovation, and ideas—while others try but fail. So, how do top leaders get it right? Going beyond the basics, Pollyanna Pixton explores ways that the best leaders create safety nets that allow people to take risks to discover and try new possibilities, fail early, and correct faster. Join Pollyanna to remove fear and engender trust to make your team and organization more creative and productive. They will spend less energy protecting themselves and the status quo and more energy creating and innovating. Pollyanna shares the tools you, as a leader, need to create open environments based on trust and to take the first step in collaboration across the enterprise. Learn how to do the right thing without breaking trust and find out when and how to acknowledge and reward trust within your team and organization.

*An international collaborative leadership expert, **Pollyanna Pixton** developed the models for collaboration and collaborative leadership through her thirty-five years of working in and consulting with corporations and organizations. She helps companies create workplaces where talent and innovation are unleashed—making them more productive, efficient, and profitable. Pollyanna is a founding partner of Accelinnova, president of Evolutionary Systems, director of the Institute for Collaborative Leadership, and co-author of the forthcoming book,* Stand Back and Deliver: Tools For Leading Agility *due out in early 2009. She co-founded the Agile Project Leadership Network (APLN) and chaired APLN Leadership Summits in London, Minneapolis, and Orlando. Contact her at ppixton@accelinnova.com.*

MONDAY, NOVEMBER 10, 1:00-4:30 HALF-DAY TUTORIALS

MM ADAPTing to Agile: A Guide to Transitioning NEW

Mike Cohn, Mountain Goat Software

Transitioning to an agile development process is unlike most transitions development organizations make. Many transitions begin when a strong, visionary leader plants a stake in the ground and says, "Let's take our organization there." Other transitions start with a lone team thinking, "Who cares what management thinks, let's do this." The problem in transitioning to agile is that neither of these approaches alone is likely to lead to the long-term, sustainable change you want. Mike Cohn describes how you can iterate toward more agility by combining a senior-level "guiding coalition" with multiple "action teams." Along the way, you will learn the acronym ADAPT to describe the five steps necessary for any successful agile transition: Awareness, Desire, Ability, Promote, and Transfer. Explore the true role of leaders and managers to guide self-organizing teams toward agility. Take back proven patterns for getting started—Start Small, Stealth Mode, Going All In, Public Displays of Agility, Impending Doom, and more. Leave knowing what you must—and must not—do to succeed with agile in your organization and team.

*A Certified Scrum trainer, **Mike Cohn** is the founder of Mountain Goat Software, a process and project management consultancy and training firm. He is the author of* Agile Estimating and Planning *and* User Stories Applied for Agile Software Development, *as well as books on Java and C++ programming. With more than twenty years of experience, Mike has previously been a technology executive in companies of various sizes—from startup to Fortune 40. A frequent magazine contributor and conference speaker, Mike is a founding member of the Scrum Alliance and the Agile Alliance. He can be reached at mike@mountaingoatsoftware.com.*

MN Agile Product Planning: Building a Strong Backlog

David Hussman, DevJam NEW

Although a strong product backlog is one key to sustained agility, this practice is less defined than other agile processes. Backlogs contain different work items—user stories, architectural spikes, investments in updating and maintaining development, and more. While it is clear that developers primarily code, it is often less clear who builds and grooms the backlog, and how to best find a groove with this practice. David Hussman explores how to create, prioritize, maintain, and groom your product backlog. He covers the core topics of user stories and personas as well as backlog items that do not neatly fit in the user story mold. Find out what other agile communities have done to establish a customer cadence that balances product planning with the work necessary to feed iterative development. Learn how to sustain a strong, continuing product backlog that feeds the innovation and evolution of your systems.

*For many years, **David Hussman** has led software projects in a variety of domains—digital audio, digital biometrics, medical, government, legal, security, industrial, financial, retail, and education to name a few. David now spends his time coaching and leading agile project communities worldwide. The author of* Cutting an Agile Groove *and contributor to several books, including* Managing Agile Projects *and* Agile in the Large, *David leads DevJam. As mentors and practitioners, DevJam focuses on using agile to help people and companies improve their software production skills. DevJam (www.devjam.com) provides seasoned leaders that strive to pragmatically match technology, people, and processes to create better and cooler products.*

MO A Day in the Life of a User Story NEW

Jean Tabaka, Rally Software Development

Jean Tabaka leads you through a series of simulations around the life of a user story. After setting the context within the Scrum methodology—explaining the roles and responsibilities in the care and feeding of user stories—the work and the fun begin! You will work in small teams applying what you have just learned about user stories. First, each team writes a set of user stories based on a Product Owner's view of how several features benefit the project. After prioritizing the new stories, the team sizes each one using "Planning Poker" and then reprioritizes the stories. Next, the team determines the tasks and estimates the effort to convert their user stories into acceptance criteria. As a new agile group, each team will debrief its work to the class. Come, join in, and be part of the active learning in this exercise-driven, on-your-feet session.

*An agile coach with Rally Software Development, **Jean Tabaka** has more than twenty-five years of experience in IT. After studying DSDM in the late 1990s, Jean became an agile devotee, working with organizations worldwide to deliver more value faster through the adoption of agile principles and practices. Specializing in scaling agile practices, guiding leadership shifts, applying lean, and building continuous planning practices, Jean uses a collaborative approach in helping organizations adopt agile. A Certified ScrumMaster Trainer and a Certified Professional Facilitator, Jean is the author of* Collaboration Explained: Facilitation Skills for Software Project Leaders. *You can reach her at jean.tabaka@rallydev.com.*

MP Refactoring Your Wetware: Thinking about Thinking

Andy Hunt, The Pragmatic Programmers

Software development happens in your head—not in an editor, IDE, or design tool. We're well educated on how to work with software and hardware, but what about wetware—our brains? Join Andy Hunt for a look at how the brain really works (hint: it's a dual processor, shared bus design) and how to use the best tool for the job by learning to think differently about thinking. Andy looks at the importance of context and the role of expert intuition in software development. Learn to take advantage of pole-bridging and integration thinking. Compare different laterally-specialized functions, including synthesis vs. analysis and sequential processing vs. pattern-matching. Discover the one simple habit that separates the genius from the "wanna-be." Andy helps you discover how to learn more deliberately by managing your knowledge portfolio. Explore practical learning techniques, including mind maps, reading techniques, and situational feedback that help you to cope with the torrent of new information that assaults each of us.

*In the industry since the early 1980s, **Andy Hunt** is one of the seventeen founders of the Agile Alliance, which launched the Agile Manifesto and the agile movement. Andy is a programmer, consultant, author, publisher, and co-founder of the Pragmatic Bookshelf. He co-authored the best-selling book,* The Pragmatic Programmer *and five others, including the recent award-winning* Practices of an Agile Developer. *At conferences and private corporations throughout the US and Europe, Andy is a frequent speaker on topics ranging from software development to management and cognition. When not working, Andy is an active musician composing, recording, and playing trumpet, flugelhorn, and piano.*

MQ Mixing Roles in Scrum: The Good, The Bad, and The Ugly NEW

Mitch Lacey, Mitch Lacey & Associates, Inc.

When is mixing roles on a Scrum team an acceptable solution on a project? On the surface, it seems innocent enough. Capable people should be able to switch roles among team members, ScrumMaster, and product owner. Why is it that this approach often ends in disaster for the team, the project, and the customer? What are the impacts of mixing roles on projects? What are some early warning signs that tell you that your role mixture is destined for failure? Mitch Lacey takes you through examples of three projects in which the teams mixed roles—one turned out good (well, OK), one was bad, and one was ugly. Through small group discussions, you will explore Scrum roles in great detail to identify the actions each team took that led to three dramatically different outcomes. Leave this session with a deep and clear understanding of Scrum roles and how they counterbalance each other. Ensure your team remains balanced—and productive.

*An agile practitioner and trainer, **Mitch Lacey** has managed plan-driven and agile projects for more than twelve years. Mitch honed his agile skills at Microsoft Corporation, where he released core enterprise services for Windows Live. While at Microsoft, he transitioned from program manager to agile coach, working hand-in-hand with groups throughout their transition to agile practices. As a Certified Scrum Trainer and a registered Project Management Professional, Mitch shares his experience in project and client management through Certified ScrumMaster courses, agile coaching engagements, conference presentations, blogs, and white papers. Mitch is currently writing a book,* Adopting Agile: 101 Tips for Surviving Your First Year, *scheduled for publication in 2009.*

MR Designing Examples: Story Tests for Story Success

J. B. Rainsberger, Independent Consultant NEW

Poorly written stories are a common source of friction for agile teams, and the outcomes are dire. Such stories lead to inaccurate estimates, resulting in broken commitments and, in the worst cases, to a death march. Well-designed story examples—also called acceptance tests or story tests—hold the key to effectively delivering features to customers. Teams struggling to design good examples end up dumping the responsibility for examples on the wrong people or worse, reverting to confusing and wasteful, formal specification documents. If you are having trouble designing story examples, J. B. Rainsberger can help. You'll learn how to combine familiar software design principles with specific domain expertise to produce excellent story examples. This session is for team members with any job description—programmers, designers, architects, business analysts, subject matter experts, domain experts, product owners, product managers—and for anyone who wants to deliver the right software the first time.

***J. B. Rainsberger** helps software organizations satisfy their customers and the businesses they support. Expert at delivering successful software, he writes, teaches, and speaks about why delivering better software is important—but not enough. He helps clients improve their bottom line by coaching teams as well as leading change programs. J. B. helps software organizations get off the treadmill of overcommitment and underdelivery, addressing all aspects of software delivery, including understanding the business, gelling the team, and writing great code. Learn more about how J. B. will inspire your software organization at jbrains.ca, in his* IEEE Software *magazine column "Not Just Coding", at conferences worldwide, or by writing him at get.started@jbrains.ca.*

FULL-DAY TUTORIALS TUESDAY, NOVEMBER 11, 8:30-4:30

TA Lean Software Development: Mapping the Value Stream NEW

Mary Poppendieck, Poppendieck, LLC

Of the many methods promoted to improve software development, Lean is emerging as one that is grounded in decades of practice in manufacturing. Mary Poppendieck delivers a practical, hands-on introduction for applying lean principles to software development. First, she focuses on workflow through a development organization—how to pull value from customers; how to level the workload and increase the flow of value; how to find bottlenecks that result in lumpy workflow, late delivery, and thrashing. Learn the advantages and limitations of Value Stream Mapping and how to use this lean tool to expose waste in your development process. Then, Mary describes the disciplines necessary for excellence: Stop-the-Line Quality in software development; Relentless Improvement using classic process improvement practices; and concise, useful documents to preserve and disseminate knowledge. You will work in small teams on a problem-solving exercise and develop an action plan for how to begin your journey toward lean development when you return to work.

***Mary Poppendieck** has been in the Information Technology industry for more than thirty years. She has managed software development, supply chain management, manufacturing operations, and new product development. Mary spearheaded the implementation of a Just-in-Time system in a 3M videotape manufacturing plant and led new product development teams, commercializing products ranging from digital controllers to 3M Light Fiber™. A popular writer and speaker, Mary is co-author of* Lean Software Development, *which was awarded the Software Development Productivity Award, and a sequel,* Implementing Lean Software Development.

TB Design Patterns Explained: From Analysis to Implementation NEW

Alan Shalloway, Net Objectives

Alan Shalloway takes you beyond thinking of design patterns as fixed solutions to a problem in a context. Design patterns are really about how to keep code from becoming complex and difficult to maintain as the system changes. In this clear, easy-to-understand presentation, Alan describes the classic use of patterns. He shows how design patterns illustrate good coding practices that should be followed, whether or not patterns are already present. He explains key design patterns including Strategy, Bridge, Adapter, Façade, and Abstract Factory. Within small group exercises, you will learn how to use patterns, in agile development to create robust architectures that can readily adapt as new requirements arise. Leave with a working understanding of what design patterns are and a better way to build models of your application domains.

***Alan Shalloway** is the founder and CEO of Net Objectives. With more than thirty-five years of experience, Alan is an industry thought leader, trainer, and coach in the areas of lean software development, the lean-agile connection, Scrum, agile architecture and using design patterns in agile environments. He is a popular speaker at prestigious conferences worldwide as well as a trainer/coach. Alan is the primary author of* Design Patterns Explained: A New Perspective on Object-Oriented Design *and is currently writing a book on Lean Anti-Patterns.*

TC Successful Agile Requirements: Collaborating to Define and Confirm Needs NEW

Ellen Gottesdiener, EBG Consulting

In this interactive workshop, requirements expert and agile coach Ellen Gottesdiener teaches the key practices of agile requirements so team members can successfully define and confirm customer needs. This session is for agile teams looking for better ways to build or prune their backlog, start a new iteration, or build a release plan. Learn why and how both the content and the timing of requirements analysis differ in agile and traditional projects. Practice agile approaches for defining user stories, as well as establishing requirements for quality attributes and external interfaces. Leverage EBG's Requirements Roadmap—a set of interrelated analysis models—and learn when, where, and how to draw on other analysis models. Find out how to calibrate the content, format, and timing of requirements analysis and know when the requirements are "done." Using collaborative techniques, you will practice conducting "just enough" requirements analysis to plan, estimate, and develop tasks for iteration planning. With these skills, you can start building a new product or improve requirements practices on your current project.

*Principal Consultant of EBG Consulting, **Ellen Gottesdiener** helps business and technical teams get product requirements right so their projects start smart and deliver the right product at the right time. An agile coach and trainer with a passion about agile requirements, she works with large, complex products and helps teams elicit just enough requirements to achieve iteration and product goals. Ellen is the author of* Requirements by Collaboration: Workshops for Defining Needs *and* The Software Requirements Memory Jogger. *Ellen writes articles, speaks, and advises at industry conferences, and provides training seminars to both traditional and agile clients. Contact her at www.ebgconsulting.com.*

TD Fearless Change: Introducing New Ideas

Linda Rising, Independent Consultant

Those who attend conferences or read books and articles discover new ideas they want to bring into their organizations—but they often struggle when trying to implement those changes. Unfortunately, those introducing change are not always welcomed with open arms. Linda Rising offers proven change management strategies to help you become a more successful agent of change in your organization. Learn how to plant effective seeds of change and identify what forces in your organization drive or block change. In addition to using these approaches to change your organization, you can use them to become a more effective person. Come and discuss your organizational and personal change challenges. Linda shows how the lessons from her book, *Fearless Change: Patterns for Introducing New Ideas*, can help you succeed. Learn how to overcome adversity to change and to celebrate your improvement successes along with your organization's newfound practices.

***Linda Rising** has a Ph.D. from Arizona State University in the field of object-based design metrics and a background that includes university teaching and industry work in telecommunications, avionics, and strategic weapons systems. An internationally known presenter on topics related to patterns, retrospectives, and the change process, Linda is the author of* Design Patterns in Communications; The Pattern Almanac 2000, A Patterns Handbook; *and co-author with Mary Lynn Manns of* Fearless Change: Patterns for Introducing New Ideas. *Find more information about Linda at www.lindarising.org.*

TE Collaboration Explained: Facilitation Skills for Project Managers

Jean Tabaka, Rally Software Development

Join Jean Tabaka for an exploration of collaboration and facilitation for project managers. In this exercise-packed session, you will learn an approach for decision-making. Jean leads you in evaluating and contrasting collaborative versus command-and-control leadership styles. See and experience collaborative decision-making and the vital facilitation techniques to plan for and run highly productive meetings and group interactions. Practice collaborative planning sessions, daily interactions, and review meetings. Working in small teams, each participant facilitates at least one of the class exercises. More than learning to plan collaborative meetings, you will practice helping teams gather insights and make decisions—without taking over the decisions yourself! Be prepared to think on your feet, be challenged, and grow your personal collaboration and facilitation skills.

*An agile coach with Rally Software Development, **Jean Tabaka** has more than twenty-five years of experience in IT. After studying DSDM in the late 1990s, Jean became an agile devotee, working with organizations worldwide to deliver more value faster through the adoption of agile principles and practices. Specializing in scaling agile practices, guiding leadership shifts, applying lean, and building continuous planning practices, Jean uses a collaborative approach in helping organizations adopt agile. A Certified ScrumMaster Trainer and a Certified Professional Facilitator, Jean is the author of* Collaboration Explained: Facilitation Skills for Software Project Leaders. *You can reach her at jean.tabaka@rallydev.com.*

TF Growing Agile Coaches NEW

David Hussman, DevJam

It's a fact—where agility lives and thrives, strong coaches are present. Finding and growing coaches is essential to sustainable agility and great teams. When strong coaches do emerge, it is often an organic, accidental process that organizations then struggle to recreate. Development organizations need tools and dedication to find potential coaches and grow them into great ones. Unfortunately, potentially great coaches often go undiscovered and untapped. David Hussman describes ways to help your development organization improve its coaching practices. David looks at ways to find candidate coaches in your company and provide them with tools and practices he has mined from many successful coaches. With a combination of presentation and participation, David challenges you to improve your coaching skills and take back a set of new coaching practices for your team and organization. Come prepared to share your experiences and skills, ask your toughest questions, and laugh out loud. Take back a new appreciation of the culture needed to foster the shared responsibility of coaching that is part of creating a sustainable agile groove.

*For many years, **David Hussman** has led software projects in a variety of domains—digital audio, digital biometrics, medical, government, legal, security, industrial, financial, retail, and education to name a few. David now spends his time coaching and leading agile project communities worldwide. The author of* Cutting an Agile Groove *and contributor to several books, including* Managing Agile Projects *and* Agile in the Large, *David leads DevJam. As mentors and practitioners, DevJam focuses on using agile to help people and companies improve their software production skills. DevJam (www.devjam.com) provides seasoned leaders that strive to pragmatically match technology, people, and processes to create better and cooler products.*

TUESDAY, NOVEMBER 11, 8:30-12:00 HALF-DAY TUTORIALS

TG Getting Agile with Scrum NEW

Mike Cohn, Mountain Goat Software

Since its origin on Japanese new product development projects in the 1980s, Scrum has become recognized as one of the best project management frameworks for handling rapidly changing and evolving development projects. With more than 30,000 Certified ScrumMasters, Scrum is one of the leading agile software development successes. It is especially valuable for product development projects with significant technology uncertainty or changing requirements. Through teaching, interactive discussions, and hands-on exercises, Mike Cohn covers the basics of what you need to know to get started with Scrum. Learn about all the key aspects of Scrum, including product and sprint backlog, the sprint planning meeting, the sprint review, conducting a sprint retrospective, activities that occur during sprints, measuring and monitoring progress, and scaling Scrum to work with large and distributed teams. Find out about the roles and responsibilities of the ScrumMaster, the product owner, and each member of the Scrum team. Equally suited for managers, programmers, testers, product managers and anyone else interested in improving product delivery, this session will help you and your team "get agile."

*A Certified Scrum trainer, **Mike Cohn** is the founder of Mountain Goat Software, a process and project management consultancy and training firm. He is the author of* Agile Estimating and Planning *and* User Stories Applied for Agile Software Development, *as well as books on Java and C++ programming. With more than twenty years of experience, Mike has previously been a technology executive in companies of various sizes—from startup to Fortune 40. A frequent magazine contributor and conference speaker, Mike is a founding member of the Scrum Alliance and the Agile Alliance. He can be reached at mike@mountaingoatsoftware.com.*

TH Leading Successful Projects in Changing Environments

Pollyanna Pixton, Accelinnova

There's no doubt about it—agile has gone mainstream. Short delivery iterations give organizations the means to incorporate change safely, reach go/no-go decisions early, and discover realistic team velocities. Managers can better determine if market windows can be reached, thus placing successful products in customers' hands. But what if the ground beneath the project team is changing rapidly even as it is trying to make progress? Pollyanna Pixton shares a collaboration model and iterative delivery process that will help you succeed, even in unstable conditions. She presents her ideas on creating an open environment, identifying the talent the team needs, managing risks, and creating team ownership to ensure great results. Among the skills you need are a collaborative, transparent leadership style; an approach to positively influence outcomes; collaborative communication; and then the knowledge of when to stand back and let things happen. Leave this session with some keys to successfully lead agile project teams—even in the midst of chaos.

*An international collaborative leadership expert, **Pollyanna Pixton** developed the models for collaboration and collaborative leadership through her thirty-five years of working in and consulting with corporations and organizations. She helps companies create workplaces where talent and innovation are unleashed—making them more productive, efficient, and profitable. Pollyanna is a founding partner of Accelinnova, president of Evolutionary Systems, director of the Institute for Collaborative Leadership, and co-author of the forthcoming book,* Stand Back and Deliver: Tools For Leading Agility *due out early 2009. She co-founded the Agile Project Leadership Network (APLN) and chaired APLN Leadership Summits in London, Minneapolis, and Orlando. Contact her at ppixton@accelinnova.com.*

TI Measuring and Using Your Team's Velocity NEW

Rob Myers, Net Objectives

"Velocity" is an oft-misunderstood agile term. Developers worry they're being evaluated based on this quantity. Managers want to know how to increase it. The team's definition of velocity—explicit or implicit—affects the way the team estimates stories, plans iterations, and tracks progress. The definition of velocity must be consistent and agreed upon by everyone or planning efforts will quickly unravel. Using money as a metaphor, Rob Myers illustrates how to successfully employ velocity in your iteration planning. In a hands-on simulation, you'll experience an agile iteration planning session that concretely demonstrates how velocity really works. Learn about estimation techniques, such as "planning poker," and try out a valuable, rapid-estimation technique based on story-complexity. See how Big Visible Charts reveal the team's progress through iterations and releases. Discuss what you need to do to plan for vacations, meetings, sick days, and surprises. Rob explores the "Four Variables" of software development and what to do when the answer to the question, "Are we on schedule?" is "No."

***Rob Myers** has more than twenty years of professional experience in software development, including projects for industry leaders in medical, aerospace, and financial services. In the late 1990s, Rob became an eXtreme Programming coach and traveled throughout the U.S. and abroad assisting teams with agile practices and object-oriented design techniques. He teaches Test-Driven Development, Design Patterns Explained, Implementing Scrum for Your Team, Lean-Agile Testing, and other courses. Rob brings to each classroom his passion for Lean software development, team development, and sane work environments.*

TJ Hiring for an Agile Team: Candidates Who Fit NEW

Johanna Rothman, Rothman Consulting Group

Even candidates who have experience on agile teams may not be experienced working the same way your team works. And, because not everyone is using the same—or any—agile approaches, some people who are not using the same approaches as your team may be perfect—or not. How do you decide? If you've tried to hire technical people, you know you can't rely on their résumé describing "agile experience" to be a predictor of success. Nor can a résumé that doesn't specifically cite "agile experience" be a predictor of failure. Will the candidate be just right for your open position—even if technical skill is not an issue? Hiring for your team doesn't have to be frustrating. Instead, you can reframe your interviewer role to be that of a detective. You need to define your team's culture in advance and determine during the interview process whether or not a candidate fits into that culture. In this hands-on session, you will practice defining the essential technical and non-technical skills for your teams, identify your cultural issues, and practice interviewing so that your next hire will be successful.

***Johanna Rothman** consults, speaks, and writes on managing high-technology product development. She has helped engineering organizations, IT organizations, and startups hire technical people, manage projects, and release successful products faster. Johanna is the author of the Jolt Productivity Award winner* Manage It! Your Guide to Modern, Pragmatic Project Management *and* Hiring the Best Knowledge Workers, Techies & Nerds: The Secrets and Science of Hiring Technical People, *and co-author with Esther Derby of the pragmatic* Behind Closed Doors: Secrets of Great Management. *Johanna is a host of the Amplifying Your Effectiveness (AYE) conference.*

TK Behavior-Driven Development: Writing Software that Matters NEW

Dan North, ThoughtWorks

Behavior-driven development (BDD) is a new approach in the evolution of agile software delivery. With its roots in test-driven development, domain-driven design, and automated acceptance testing, BDD focuses on the ways an application is expected to work—its behavior. By constantly reflecting on the varied points of view of different stakeholders, BDD helps ensure that product owners and the development team are in-sync on what is really needed and what to work on next. In this highly interactive session, Dan North introduces the principles behind BDD and describes how it works in practice. He provides an overview of the methodology of BDD: understanding your domain and who your stakeholders are, identifying and exploring requirements, automating acceptance criteria, and delivering working and tested software. Dan then looks at the nature of change and describes how to implement BDD in different contexts, including applying it to an in-flight project, managing distributed or large-scale development, and working with legacy systems. Dan encourages both experienced and novice agile managers and practitioners to join this session and bring with them their current challenges—and war stories.

***Dan North** has been working with software for twenty years, starting with playing Star Wars at a games company in the late 1980s. After graduating, he did real work, programming C on UNIX. Dan has worked in digital imaging, investment banking, ISPs, telcos, and car leasing. His sense of timing is demonstrated by his writing billing software for WorldCom in the late 1990s, consulting on sub-prime mortgage systems, and writing a trading platform for collateralized debt obligations (the product of choice for the credit meltdown). Since 2002, Dan has been a consultant with ThoughtWorks (thoughtworks.com) where he advocates simple, pragmatic common sense. He likes to talk about SOA, lean software development, NLP, and behavior-driven development (BDD). Dan occasionally blogs at http://dannorth.net.*

TL xUnit Test Patterns and Smells: Improving Test Code and Testability Through Refactoring NEW

Gerard Meszaros, Solution Frameworks, Inc.

Automated unit testing is the agile software development equivalent of "in-process inspection" in lean systems. xUnit, the generic name for the family of tools used to develop automated unit tests, is a way to automatically inspect software during development. Most experienced test-driven development (TDD) practitioners create as much or more test code as production code. "Yikes! Won't that double my cost of producing working software?" you ask. The answer, as always, is "It depends!" All test code is not created equal. Properly written, xUnit tests are easy to create and maintain; improperly written, they can significantly increase the total cost of ownership for your automated test suite. Learn the differences between Single Glance Readable test code versus Obscure Tests and Independent Tests versus Interacting Tests. Learn to avoid Test Run Wars and develop Robust Tests rather than Fragile Tests. Gerard Meszaros describes a vocabulary of patterns and smells to evaluate the quality of your test code and a set of reusable design patterns that you can use to eliminate those smells—either through avoidance or refactoring.

***Gerard Meszaros** is a software development consultant with twenty-five years of experience in software and nearly a decade of experience applying agile methods. A leader in test automation patterns, refactoring of software and tests, and design for testability, Gerard has applied automated unit and acceptance testing on projects ranging from full-on eXtreme Programming to traditional waterfall development. He is an expert in implementing and customizing agile methods, such as Scrum and eXtreme Programming, and is a proponent of including usability practices on agile projects. A frequent presenter at major conferences, Gerard authored* xUnit Test Patterns—Refactoring Test Code *which recently recently won a Jolt Productivity Award.*

HALF-DAY TUTORIALS TUESDAY, NOVEMBER 11, 1:00-4:30

TM Avoid Integration Defects without Integration Testing NEW

J. B. Rainsberger, Independent Consultant

Despite their good test-driven design (TDD) practices, end-to-end tests have sunk many TDD enthusiasts. Although end-to-end tests are not inherently bad, they can slow you down when you rely on them too heavily, causing you to make more mistakes. As a result, some developers give up on TDD entirely. Don't give up! You are ready to learn the next key TDD technique that will help you write better code that avoids integration defects. Learn the design principles and techniques that have enabled J. B. Rainsberger to not rely on end-to-end tests while writing correct code. With these new techniques, you will write fewer tests that cover more of the code base, provide far superior feedback, and deliver features sooner. Join J. B. Rainsberger to reduce your integration defects without adding costly end-to-end tests to the test base.

LAPTOP OPTIONAL

J. B. Rainsberger encourages you to bring a laptop with code that has your integration tests—tests that worry about the correctness of multiple objects. If you don't mind the critique, he'll show you how to write smaller, more focused tests.

J. B. Rainsberger *helps software organizations satisfy their customers and the businesses they support. Expert at delivering successful software, he writes, teaches, and speaks about why delivering better software is important—but not enough. He helps clients improve their bottom line by coaching teams as well as leading change programs. J. B. helps software organizations get off the treadmill of overcommitment and underdelivery, addressing all aspects of software delivery, including understanding the business, gelling the team, and writing great code. Learn more about how J. B. will inspire your software organization at jbrains.ca, in his* IEEE Software *magazine column "Not Just Coding", at conferences worldwide, or by writing him at get.started@jbrains.ca.*

TN From User Story to User Interface

Jeff Patton, Independent Consultant

You've chosen to take an agile approach to development. You've written down a set of user stories of what users want for their system. Now, the developers have questions on the look and feel of the user interface. How can you quickly, predictably, and with confidence move from user stories to a user interface? Jeff Patton introduces a practical approach for translating user goals and tasks into user interface designs that effectively support users' work. Discover how a user-centered design practitioner moves quickly from user tasks to user interface. Practice taking a set of user stories and transforming them into more tangible actions that users might take in the user interface; then, collaboratively build and test paper prototypes of your proposed user interface. In addition to paper prototyping skills and basic usability testing skills, learn the essential visual design skills that can help improve the appeal of your new user interface.

For the past twelve years, ***Jeff Patton*** *has designed and developed software on a wide variety of projects from online aircraft parts ordering to electronic medical records. A winner of the Agile Alliance's 2007 Gordon Pask Award for contributions to agile development, Jeff has focused on agile approaches since working on an early Extreme Programming team in 2000. Jeff has specialized in the application of user-centered design techniques to improve agile requirements, planning, and products. Some of his recent writing on the subject can be found at www.agileproductdesign.com and Alistair Cockburn's Crystal Clear. His forthcoming book gives tactical advice to those seeking to deliver useful, usable, and valuable software.*

TO The Bridge to Agility for Traditional Project Managers NEW

Michele Sliger, Sliger Consulting, Inc.

Traditional software project managers are feeling left behind by new agile software development practices. This is your opportunity to bridge agile development concepts by relating these new approaches to practices with which you are already familiar—the Project Management Institute's Body of Knowledge (PMBOK). Learn about agile frameworks and the meaning of value-driven development, Scrum, XP, lean methods, and more. Michele Sliger maps PMI's *PMBOK Guide*® knowledge areas to the corresponding agile development practices. The mapping includes answers to questions, such as how to manage risk, what happens to change control, what project plans look like, and whether or not scope creep has any meaning in agile projects. Michele describes how to redefine project managers' traditional jobs into a new—and more important—role in agile development.

The co-author of The Software Project Manager's Bridge to Agility, ***Michele Sliger*** *has extensive experience in agile software development, having transitioned to Scrum and XP practices in 2000 after starting her career following the traditional waterfall approach. A self-described "bridge builder," her passion lies in helping those in traditional software development environments cross the bridge to agility. Michele consults with businesses ranging from small start-ups to Fortune 500 companies, helping teams with their agile adoption and helping organizations prepare for the changes that agile brings. She is a certified Project Management Professional (PMP®) and a Certified Scrum Trainer (CST). Michele can be reached at michele@sligerconsulting.com.*

TP Software Configuration Management for Agile Development NEW

Steve Berczuk, Cyrus Innovation

Version management, build, and release practices are essential elements of any effective development environment. Yet many agile teams are puzzled by how to translate good software configuration management (SCM) practices into an agile environment. Some teams believe SCM interferes with the agile goals of rapid integration and change; others ignore SCM to the point that their code becomes more chaotic than agile. Steve Berczuk provides an overview of SCM concepts and explains the SCM patterns and practices that teams need to maintain an agile change rate. Robust SCM practices provide for the traceability and reproducibility you need for production environments. After this session, you will understand how both agile testing practices and continuous integration change how teams use SCM. Learn how to use SCM to manage both legacy (non-agile) code bases and new agile projects, and how to set up the essentials of an agile SCM environment. Leave this session with a better understanding of how SCM can help you be more agile—while still maintaining control of your code.

With twenty years of experience in the software industry, working with both startup and established companies, Certified Scrum Practitioner ***Steve Berczuk*** *leads and coaches teams in agile software development practices. In addition to developing software, he helps teams effectively use software configuration management (SCM). Steve is co-author of* Software Configuration Management Patterns: Effective Teamwork, Practical Integration*. He lives and works in Boston as a technical lead for Cyrus Innovation, an agile consultancy. Steve's Web site is www.berczuk.com.*

TQ Agile Estimating and Planning NEW

Mike Cohn, Mountain Goat Software

Planning is important for all projects, especially agile ones. Unfortunately, we've all seen many worthless plans and often want to throw out planning altogether. Unfortunately, too many teams today view planning as something to be avoided, and too many organizations view plans as something to hold against their development teams. Don't give up yet! With the right type of agile planning and estimating, you can create an accurate and useful project plan that looks six to nine months in the future. Join Mike Cohn to learn new skills that will help you create reliable plans that improve decision-making and break the cycle of poor estimates and failed expectations. Leave with a solid understanding of and experience in agile planning as you learn new approaches to estimating—unit-less points, ideal time, and more. Practice estimating with the popular Planning Poker technique and see how these techniques work on fixed-price and fixed-scope projects. With Mike and the other participants, you'll explore techniques that dramatically increase your project's chances of on-time completion.

A Certified Scrum trainer, ***Mike Cohn*** *is the founder of Mountain Goat Software, a process and project management consultancy and training firm. He is the author of* Agile Estimating and Planning *and* User Stories Applied for Agile Software Development*, as well as books on Java and C++ programming. With more than twenty years of experience, Mike has previously been a technology executive in companies of various sizes—from startup to Fortune 40. A frequent magazine contributor and conference speaker, Mike is a founding member of the Scrum Alliance and the Agile Alliance. He can be reached at mike@mountaingoatsoftware.com.*

TR Practical Agile: Real World Practices NEW

Jared Richardson, 6th Sense Analytics

Agile is a huge paradigm shift for most development organizations. It is difficult to decide which practices suit your organization and how to introduce them without alienating your team. Jared Richardson offers practical ideas that he has found are critical to agile adoption and its success. Join Jared to discuss and experiment with practices ranging from peer code reviews and daily stand-up meetings to learning about "The List." These practices will help you improve code quality, increase team cohesion, manage your feature backlog, and help keep your team on track. Jared describes how to get started with agile, pitfalls to avoid, and how to decide which practices are working for you. Experiment with and try out some of these practices to gain hands-on experience. Learn how a gradual, grassroots introduction of the right agile practices can help ease your team into change. Find out how to introduce the practices your team needs to ensure your next project stays on track. Everyone wants to succeed. Give your team the tools they need to do just that.

Jared Richardson, *co-author of* Ship It! A Practical Guide to Successful Software Projects*, is a regular conference speaker and an agile coach at 6th Sense Analytics. Jared has been in the industry for more than fifteen years as a consultant, developer, tester, and manager. Until recently, he was an independent consultant focused on helping teams build better software. He's now bringing that same focus to 6th Sense Analytics and its clients, using both the 6th Sense toolset and his unique perspective. Jared can be found online at www.AgileArtisans.com and www.6sa.com/blog.*

WEDNESDAY, NOVEMBER 12, 8:45 a.m.

Seven Years Later: What the Agile Manifesto Left Out

Brian Marick, Exampler Consulting

Although the Agile Manifesto has worked well to help many organizations change the way they build software, the agile movement is now suffering from some backsliding, lots of overselling, and a resulting backlash. Brian Marick believes that is partly because the Agile Manifesto is almost entirely focused outwardly—it talks to the business about how the development team will work with it. What it does not talk about is how the team must work within itself and with the code. Even though those omissions were appropriate then, now more is needed. Teams starting agile need to know that more discipline is required of them, and that discipline is fruitless without a strong emphasis on skills. Teams need to recognize that success is not just fulfilling requirements. It is also increasing productivity and decreasing the consequences of mistakes. Perhaps most of all, teams need to respect themselves and believe they deserve joyful work. Join Brian to find out whether you're really doing Agile or if you are agile in name only.

Brian Marick *(marick@exampler.com, www.exampler.com) was a programmer, tester, and team lead in the 1980s, a testing consultant in the 1990s, and is an agile consultant in this decade. He was one of the authors of the* Manifesto for Agile Software Development *and is a past chair of the board of the Agile Alliance. Brian is the author of two books—*The Craft of Software Testing *and* Everyday Scripting with Ruby*—and a number of articles. His consulting concentrates on blending formerly-independent test teams into agile projects, the use of executable examples (a.k.a, tests) to drive creation of products, and helping programmers learn their craft by pairing with them.*

WEDNESDAY, NOVEMBER 12, 4:30 p.m.

Beyond Best Practices: Keeping Agile Agile

Dan North, ThoughtWorks

Adopting "best practices" seems to be an intrinsic part of the transition to agile—with many organizations creating special process teams and hiring methodology consultants to implement and enforce best practices. These practices often are seen as a cornerstone of an agile change program and are even touted as a selling point—"Our projects will surely succeed if we follow best practices!" And of course, there are industries and ecosystems that have grown up around accreditation, auditing, and support of specific agile methods. Do they actually help you, or might they in fact be working against your organization? Dan North argues that best practices are useful only up to a point. Rigidly enforcing them is counter to the values of agile and will eventually drive away your best people. He looks at the motivations behind agile initiatives and introduces the Dreyfus model of skills acquisition—a framework for exploring the effectiveness of best practices in your team and organization. Dan shows that by understanding how people learn and develop, you can evolve your own agile practices and keep them relevant and applicable to the practitioners in your context.

Dan North *has been working with software for twenty years, starting with playing Star Wars at a games company in the late 1980s. After graduating, he did real work, programming C on UNIX. Dan has worked in digital imaging, investment banking, ISPs, telcos, and car leasing. His sense of timing is demonstrated by his writing billing software for WorldCom in the late 1990s, consulting on sub-prime mortgage systems, and writing a trading platform for collateralized debt obligations (the product of choice for the credit meltdown). Since 2002, Dan has been a consultant with ThoughtWorks (http://thoughtworks.com) where he advocates simple, pragmatic common sense. He likes to talk about SOA, lean software development, NLP, and behavior-driven development (BDD). Dan occasionally blogs at http://dannorth.net.*

THURSDAY, NOVEMBER 13, 8:45 a.m.

Collaborative Leadership: A Secret to Agile Success

Pollyanna Pixton, Accelinnova

When members of a development project are asked to become a self-directed agile team, some claim that leadership and leaders are obsolete. Or is a different type of leadership exactly what agile teams need to truly flourish? Pollyanna Pixton describes a new, collaborative leadership style that does not attempt to control or micro-manage. It's one that asks the right questions at the right time to generate new ideas and develop creative products that customers need and want. Pollyanna explains the four areas of collaborative leadership-creating an open environment where the best people can work, learning from stakeholders throughout the enterprise, prioritizing innovative solutions based on business value, and standing back to allow the team to succeed. She shares her battle-tested tips for leading collaboratively-fix processes, not people; take the fun out of being dysfunctional; eliminate constructive criticism, because it never is; and more. Whether you are a senior manager, product owner, customer, ScrumMaster, or an individual contributor, Pollyanna's collaborative principles will empower you and everyone on your team to become better leaders and deliver the business value that stakeholders deserve.

An international collaborative leadership expert, ***Pollyanna Pixton*** *developed the models for collaboration and collaborative leadership through her thirty-five years of working in and consulting with corporations and organizations. She helps companies create workplaces where talent and innovation are unleashed—making them more productive, efficient, and profitable. Pollyanna is a founding partner of Accelinnova, president of Evolutionary Systems, director of the Institute for Collaborative Leadership, and co-author of the forthcoming book,* Stand Back and Deliver: Tools For Leading Agility *due out in early 2009. She co-founded the Agile Project Leadership Network (APLN) and chaired APLN Leadership Summits in London, Minneapolis, and Orlando. Contact her at ppixton@accelinnova.com.*

THURSDAY, NOVEMBER 13, 4:30 p.m.

Scaling Agile: Kanban and Beyond...

David Anderson, Valtech, Inc. and Modus Cooperandi, Inc.

Agile software development has been around for almost ten years. Some believe lean is the next step in our evolution. How do agile and lean play together, and what does the lean influence mean for the future of agile? Kanban is a signaling system, devised by Toyota and used in their just-in-time manufacturing process. Often, it is implemented as cards on a board that shows the status of work. David Anderson describes how you can use the kanban approach to build a high maturity enterprise that can scale agile practices to support large, enterprise software development projects. He describes how kanban facilitates a quantitatively managed, predictable, and continuously improving organization. David also examines future trends in scaling agile, including the real option theory, CMMI's role in high maturity organizations, agile portfolio management, agile governance, and the emergence of lean software supply chains. Join David for a thoughtful look into the future of agile development—and your future, too.

David Anderson *is chief process scientist with Valtech and president of Modus Cooperandi, a management consulting firm based in Seattle, Washington. He has many years of management experience leading teams on agile development projects, most recently as senior director for Software Engineering at Corbis in Seattle. David was a founder of the agile movement through his involvement in the creation of feature-driven development in Singapore in the late 1990s. He can be contacted at david.anderson@valtech.com.*

WEDNESDAY, NOVEMBER 12, 10:00 a.m.

W1 TRANSITIONING TO AGILE

Overcoming the Pitfalls of Transitioning to Agile

Johanna Rothman, Rothman Consulting Group

If you've been trying to change your organization so that your projects are more agile, you may have encountered several problems—one is that it's difficult to have product management, senior management, and functional managers work together to lead in a way that makes sense for your agile project. You're also probably working with other parts of a large program that isn't agile; you have a geographically distributed team; your management wants to know at the beginning when the project will end; or you might have a project team that does not share a common vision of what "done" means. Johanna Rothman explores common organization, management, team, and individual team member issues. She offers suggestions for making the changes more acceptable and helping people work with you in a way that enables your projects to succeed.

W2 AGILE MANAGEMENT

Are We There Yet? Defining "Done"

Mitch Lacey, Mitch Lacey & Associates, Inc.

"Are you done yet?" The answer to this question may sink your career, your team, and your project. If you respond with a "yes," you may be forced to take on additional work you can't handle. If you say "no," you may be branded as someone who can't get things done. Mitch Lacey notes that this innocent question is asked countless times on almost every software project. Establishing an upfront, common understanding of "done" can save teams and businesses countless hours of rework, process-thrash, unclear communication, and hidden work. Mitch describes what a "done list" is, how it adds value, and the value it communicates to stakeholders. Mitch takes you through an exercise on how to establish a common understanding of done and provides an exercise that you can use with your project teams.

W3 AGILE PROJECTS

From Concept to Product Backlog: What Happens Before Iteration Zero

Gerard Meszaros, Solution Frameworks, Inc.

Many agile methodologies start with a product owner walking into a room with a pile of money and a stack of prioritized story cards, and then telling the development team to start building a system. These same methodologies often eschew any form of "big upfront" activities and leave us in such a rush to deliver business value that we don't have time to do architecture, user/task research, etc. While a pile of story cards may be the first thing the development team sees, this is rarely the first set of activities in a project. In reality, the customer usually comes with a problem and some vague idea of how to solve it with technology. Someone must help the customer crystallize his vision, design the product, get the necessary funding, and populate the initial product backlog. Gerard Meszaros provides an overview of what needs to go on "behind the scenes" from project conception to the start of development in earnest.

W4 AGILE PROCESSES

Test-Driven Everything

David Hussman, DevJam

When you hear people talk about test-first or test-driven, you probably think of testing the code. Test-driven practices help developers reduce defects and increase the value in the code and the designs they deliver. Sadly, "test-driven" is too often confined to the coding trenches, and project communities miss the value of test-driven as a way to produce more value and less waste in other areas. David Hussman challenges you to think about test-driven beyond the coding realm. In addition to test-driven development, it is possible to test drive projects, meetings, and more. David begins by describing test-driven development and why it is often devalued or even dropped. Then he explains about using project chartering and story test-driven development as concrete tools for infusing test-driven everything across your project community. As a result you will find defects, remove duplication, and discover dependencies sooner.

W5 THE AGILE ORGANIZATION

What Are They Doing Down There? A CIO's Perspective on Agile Software Development

Niel Nickolaisen, Headwaters, Inc.

What are the factors critical to the success of a CIO? How can a CIO consistently deliver business value? Do development teams, in general, and agile teams, in particular, understand how to contribute to this success? In this interactive presentation, Niel Nickolaisen presents the metrics and drivers that influence CIO behavior and longevity. These metrics and drivers also influence the organization's decision to embrace agile methods. Niel shares his experiences and the survey responses from his CIO peers on how development teams and CIOs can work hand-in-hand to make agile the preferred development method. Niel introduces and describes immediately-implementable, proven tools that dramatically improve IT and business value while reducing project risks.

W6 AGILE DESIGN & ARCHITECTURE

Integrating Enterprise SOA Architecture with Scrum Development Methodology

Steven Driver, Airlines Reporting Corporation

Many processes used to implement an enterprise architecture are in conflict with the agile development approach. An effective enterprise architecture framework represents the organization as it is today and as it is envisioned in the future. However, a key agile concept is that we design and build for today—and worry about the future only when it arrives. Steven Driver has found that a small change to the Scrum process flow allows easy integration of an enterprise architecture into the agile development of new systems. The translation of enterprise architecture into application architecture requires critical touch points within the Scrum process to emphasize service-based development required within the sprints. By combining an enterprise architecture approach using SOA (service-oriented architecture) and the Scrum development methodology, an organization can achieve effective system development—in both the short and long term.

W7 REQUIREMENTS

Agile for Business Analysts

Bob Hartman, Agile For All

A prevailing myth in the software industry is that business analysis requires a bloated requirements elicitation and documentation process. Although the Business Analysis Body of Knowledge (BABOK) is considered to be process agnostic, many business analysts create heavy requirements when they follow this document's guidelines. Bob Hartman busts this myth by explaining how to use generally accepted practices from the BABOK in an agile way. Drawing directly from the BABOK, Bob bridges the gap that many business analysts have regarding lightweight process, especially as it relates to larger projects and organizations. Gain the ability to use BABOK practices in an agile environment and develop an understanding of how to use them in more agile ways in traditional software development. Learn to eliminate waste in any bloated process and become comfortable regardless of the development methodology you use.

"I've attended some other types of conferences and this has been by far the best. The speakers were great, the food, the accommodations, the tutorials, and the set-up. Can't wait until next year to attend!"

— Chasity Johnson, Product Manager/Project Manager, MAP Software

W8 TRANSITIONING TO AGILE

The Agile PMP: Teaching an Old Dog New Tricks

Mike Cottmeyer, VersionOne

Agile methods put a great deal of emphasis on trust, empowerment, and collaboration. Agile moves us away from command and control project management toward an approach designed to harness the passion, creativity, and enthusiasm of the team. A successful transition to agile project management hinges largely on how well traditional project managers are able to adopt new ways of thinking about project structure and control. Building on the principles of PMI® and the Project Management Body of Knowledge (PMBOK), Mike will explore how a PMP can adapt their knowledge and experience to become an effective agile project leader. Mike will tackle the hidden assumptions behind the PMBOK and explore a more agile approach to managing time, cost, and scope. He will take an in-depth look at the PMI Processes and Knowledge areas and explore how to adapt them to agile projects. Project managers, business analysts, and other stakeholders will leave with a new way of thinking about project management best practices and new tools for delivering value in the face of uncertainty.

W9 AGILE MANAGEMENT

Agile Contracting

Rachel Weston, Rally Software Development and Chris Spagnuolo, Data Transfer Solutions

Many software development organizations work within the bounds of contractual agreements where the limitations imposed by the "Iron Triangle" of fixed timelines, budgets, and scope challenge their ability to embrace change and focus on value delivery. Agile practitioners often comment that agile contracting is a difficult problem, but proven solutions are rarely presented. Rachel Weston and Chris Spagnuolo offer some tools they have used in their own agile contracting work to help agile practitioners deal with different contracting scenarios while promoting agile practices, protecting the development organization, and still providing value and protection to the client's organization. Through a combined workshop and facilitated collaborative session, Rachel and Chris present new agile contracting tools that can be added to your toolbox. You will gain a deeper understanding of the problems associated with agile contracting as well as practical solutions for dealing with contracts in an agile manner.

W10 AGILE PROJECTS

Re-thinking Scheduling: Parkinson's Law Inverted

Mary Poppendieck, Poppendieck, LLC

The Empire State Building—the tallest building in the world for over forty years—took just 13½ months to build. Amazing as this may seem today, it was not remarkable at the time; most skyscrapers were built in about a year. How did they do that? In those days, cash flow was more important than cost, and schedule routinely trumped scope. The paradigm was the inverse of Parkinson's Law—work should **contract** to fit the time allotted. Today, Parkinson's Law is alive and well in current scheduling approaches that break work down into tasks, estimate the tasks, and sum up the result. This approach invites work on each task to expand to fit the estimated time. Mary Poppendieck will show why you should not ask, "How long will this take?", but ask instead, "What can be done by this date?" You will learn how to accomplish more with less by applying cash flow thinking and turning Parkinson's Law upside down.

W11 AGILE PROCESSES

Pragmatic Agility

Andy Hunt, The Pragmatic Programmers

What is agile software development all about? Why is it fundamentally different from other approaches and will it work for you and your organization? Join Andy Hunt, one of the seventeen original authors of the Agile Manifesto and a founder of the Agile Alliance, for his pragmatic answers to these and other questions. Examine the foundations of agile software development and learn what problems agility seeks to address. Don't be distracted by dogma—take some time to explore the core aspects of agile development. Andy presents the real foundations of agility and walks you through a typical day in the life of an agile developer. Find out what's really important about the agile approach and take back new ideas to help you transition to agile while avoiding common stumbling blocks. Join Andy to find out how to make agility work for you.

W12 THE AGILE ORGANIZATION

It Takes a Village: Organizing to Fulfill the Product Owner Role

Ronica Roth, Rally Software Development

At Yahoo!, the product owner role is defined as the "single wringable neck" who ensures that software products and projects deliver value. Many organizations struggle to fill this role that collaborates with stakeholders to define value and manage a backlog, provides tactical support to the delivery team, and directs the product and project vision and roadmap. For most organizations, the reality is that it takes a whole team of people to fill this role. Ronica Roth begins with a quick overview of the product owner responsibilities, particularly in the context of the five levels of agile planning. She then presents patterns and examples for organizing product and customer groups in product companies, consulting shops, and internal IT departments. Soliciting your ideas, Ronica leads a discussion of the successes and challenges of those patterns and of your experiences with them. Gain new ideas about how to organize your product and customer group to support value delivery.

W13 AGILE DESIGN & ARCHITECTURE

Refactoring: Where Do I Start?

J. B. Rainsberger, Independent Consultant

Since Martin Fowler completed his now-classic work *Refactoring: Improving the Design of Existing Code*, few programming practices have been more effective—and more controversial—than refactoring. Refactoring is effective when you study and practice it diligently. It remains controversial because many development managers think developers should be adding features, not reworking old code. J. B. Rainsberger takes you deep inside the process of refactoring, including how to start reaping the benefits of refactoring while minimizing the disruption of learning a new practice, how to safely refactor code you don't know well, and the four key elements of simple design that should guide your refactoring. He explains the hazards of refactoring, when not to refactor, and how to refactor in such a way as not to upset your boss. After this presentation, you will be able to refactor your own code more confidently and effectively. You might just impress some of your colleagues along the way.

W14 REQUIREMENTS

Driving User Stories from Business Value

Guy Beaver, Net Objectives

Implementations of agile and Scrum typically employ user stories as the primary method for discovering requirements. User stories provide the mechanism for the fast, flexible flow of ideas into completed increments of software. What's missing is a practical approach to discovering user stories from top-down, business valued, and prioritized capabilities. Guy Beaver shares proven approaches to allow a project-driven organization to transition to business features that can be predictably estimated and planned for release. The stories unfolded from business features have clear line-of-sight to business goals and allow for the timely discovery and management of technical considerations. Learn how to create release plans that maximize business value while minimizing waste, how to drive accurate enterprise release plan estimates with story point velocity, the Lean way to unfold user stories, and how to account for technical constraints in your release planning.

Shingle Creek Resort

WEDNESDAY, NOVEMBER 12, 2:45 p.m.

W15 TRANSITIONING TO AGILE

Driving Agile Transformation from the Top Down

Pete Morowski, Borland Software Corporation

While agile practices are starting to make their way into large enterprises, in most instances this has been a "bottom up" movement driven through grassroots efforts. But, as success stories draw attention to the benefits of agile practices, an increasing number of executives are considering making an organization-wide agile transition. It is an attractive idea, but what does an agile transition look like when it comes as a mandate from the top? How do you scale agile principles from a single team to an enterprise with multiple teams working on multiple projects? Pete Morowski shares practical answers to these questions, addressing issues, such as the role of management in creating an agile culture, bridging "two worlds" as traditional and agile co-exist in the enterprise, and rewriting the "rules" to fit the organization. Pete provides insight that can help you translate agile principles from theory into practice for your enterprise.

W16 AGILE MANAGEMENT

Calling All Agile Skeptics—the Curious and Die-Hard, Non-Agile

Damon Poole, AccuRev

Not convinced about agile? Curious about this new approach, but not sure it makes any sense? Does it feel like agile goes against everything your experience tells you is the right thing to do? Damon Poole examines your concerns, doubts, counter-examples, and horror stories. If you are interested in helping to answer the concerns of others, then bring your answers, positive examples, and experiences. In either case, bring an open mind, a sense of humor, and at least one anecdote. Delegates will share the floor and help to keep the atmosphere fun and relaxed. Come and learn how some of the practices that may be fueling your skepticism are either optional or only work when done in conjunction with other practices. For instance, frequent releases are not required and short iterations work best when coupled with automated regression testing.

W17 AGILE PROJECTS

Agile Project Metrics

Dave Nicolette, Valtech Technologies

Agile projects and traditional projects are tracked differently. The key difference is that agile projects track outcomes; traditional projects track activities. Project managers who are new to agile are often unsure which measures are relevant to which stakeholders and how to interpret them, and how agile metrics tie back to some of the more familiar forms of project reporting. Dave Nicolette explains how agile projects are tracked, which metrics are useful to which audiences, and how to monitor project health, delivery effectiveness, and the quality and value of the results. Dave describes the reasons to choose particular metrics, how to use metrics for informational, diagnostic, and motivational purposes, and the time-sensitivity of metrics. Dave also explains the meaning and use of measures peculiar to agile methods, such as "velocity," "running tested features," "earned business value," and "burn charts". Bring your metrics questions and apply the information to your real-world situations.

W18 AGILE PROCESSES

Secrets of CMMI® for Agile Organizations

Jeff Dalton, Broadsword Solutions

Are you convinced that agile development methods and process improvement methods such as CMMI® don't go together? Have you been the victim of a ton of process overhead dropped on your head? It doesn't have to be that way. CMMI® and agile methods can work together to supercharge software development performance, gaining the advantages of agility and the repeatability, reusability, and infrastructure that process maturity provides. Jeff Dalton presents an agile approach to CMMI®, both in content and in management of the process itself. Agile cultures need to approach and perform process improvement activities within a language and framework that makes sense to them—an agile framework. Jeff discusses iterations, releases, design slams, integrated teams, and JENTM concepts (just enough, not too much). If you are interested in agile methodologies and would like to learn how to apply CMMI® in your organization, this class is for you.

W19 THE AGILE ORGANIZATION

Value Stream Mapping: Extending Our View to the Enterprise

Alan Shalloway, Net Objectives

What if the process improvements you are trying to make are not where your real problems lie? Assuming where your problems are is often the biggest problem. Alan Shalloway presents value stream maps, a Lean tool that focuses on finding waste in your development process. Alan presents an example of a value stream map that resulted in a twenty percent productivity improvement to the development team without modifying how the team worked. After this introduction to value stream mapping, you will create your own maps to learn how to improve your own processes and to learn the basic lean principles of optimize the whole, deliver fast, and build quality in. Alan demonstrates how focusing on improving the flow of software development from a time perspective can lead to a higher quality, lower cost process.

W20 AGILE DESIGN & ARCHITECTURE

Behavior-Driven Database Design

Pramod Sadalage, ThoughtWorks

Agile methods focus on creating executable code quickly and with fewer defects. But what about the database? The database is "the" component of the application that is thought to be the least agile and often excluded from agile development. Pramod Sadalage explains how the concepts of Behavior-Driven Development (BDD) can be applied to database development to drive the design of the database using executable specifications. Pramod describes how performing BDDD (Behavior-Driven Database Design) allows us to specify the behavior of the database as it is expected by the code running against the database, how BDDD allows us to easily refactor the database, and how BDDD provides an easy way to document the database design and behavior. If we encode all the behavior we expect from the database, then we have a comprehensive set of tests to safely refactor our database in addition to an extensive behavior specification of the database itself.

W21 REQUIREMENTS

Do the Right Thing: Adapting Requirements Practices for Agile Projects

Ellen Gottesdiener, EBG Consulting

Some agile teams rely on user stories alone to articulate requirements, struggle with requirements rework on large agile projects, and spend too much time thrashing on requirements during iterations. Requirements expert and agile coach, Ellen Gottesdiener shares a wide spectrum of requirements practices ranging from traditional to agile to help you break out of the cookie-cutter mentality that some take toward requirements elicitation. Practitioners from a traditional environment learn how classic requirements practices are adapted on agile projects. Agile practitioners learn how they may lighten, tighten, or incorporate a subset of traditional requirements practices to mitigate risks associated with missing, erroneous, or conflicting requirements. Gain an appreciation of ways to adapt requirements practices to fit various project situations so you can do the right things for your project.

"I am excited to take so many great ideas back to work with me. The facility was first class. All of the speakers I had, tutorial and concurrent, were experienced in their fields and had valuable insight to share."

— Agile Development Practices 2007 Delegate

T1 PEOPLE & TEAMS

Who Do You Trust? Beware of Your Brain

Linda Rising, Independent Consultant

Cognitive scientists tell us that we are more productive and happier when our behavior matches our brain's hardwiring—when what we do and why we do it matches the way we have evolved to survive over tens of thousands of years. One problematic behavior humans have is that we are hardwired to instantly decide who we trust. And we generally aren't aware of these decisions—it just happens. Linda Rising explains that this hardwired "trust evaluation" can get in the way of working well with others. Pairing, the daily stand up, and close communication with the customer and others outside the team go a long way to overcome our instant evaluation of others. As Linda helps you gain a better understanding of this mechanism in your behavior and what agile processes can do to help, you are more likely to build better interpersonal relationships.

T2 AGILE MANAGEMENT

Selling Agile: Getting Buy-In from Your Team, Customers, and Managers

Michele Sliger, Sliger Consulting, Inc.

Are you excited by the potential of agile software development, but find that your colleagues are a bit reticent? Is your whole team ready to dive in, but your business partner is only interested in dipping in their big toe—if that? Or maybe you're wishing you could find a way to convince your clients that there's a better way to contract for a software development job—without having to do a full-blown detailed design upfront? Michele Sliger discusses all of these questions surrounding how to best "sell" agile in your organization. Michele focuses on the general idea of a "sales pitch," including what to say and what not to say. Then she discusses selling agile to the team, to management, to the customer, and to others in your organization. She wraps up with a pointed look at not selling, and instead focusing on finding other ways to promote and share agile.

T3 AGILE PROJECTS

When to Step Up, When to Step Back: How to Lead Collaboration

Pollyanna Pixton, Accelinnova

Leaders can stifle progress when they interfere with team processes. But as a leader, you don't want an on-track project to go over the cliff and deliver the wrong results either. There are times when leaders should stand back and let the team work and times when they should step up and lead. How do we know which is which? Pollyanna Pixton focuses on collaboration and teaches you how to step back by unleashing the talent in your organization and teams. Learn how to create an open environment that fosters innovation and creativity, and how to let your team members take ownership and hold themselves accountable. Equally important, develop the techniques to step up and lead without impeding the flow of ideas, yet keep the project on track. Master this balancing act and come away with tools to both motivate and guide effectively.

T4 AGILE PROCESSES

Making People and Processes Congruent

Ken Pugh, Net Objectives

Agile processes work better if developers and customers have specific aptitudes and attitudes, such as the ability and willingness to handle rapid change. Members of an agile product team cannot always be selected to ensure that they have these capabilities. Developers may not appreciate the need for unit testing while customers may not be able to interact easily to create just-in-time requirements. In this interactive class, you first outline people issues you have faced. Based on common issues, the class divides into groups to discuss their challenges in depth. Each group develops ways to approach these issues and improve their teams. At the end of the session, you will have the opportunity to share your key results with the entire group. You will learn to adapt your agile implementation so that team members can work effectively within their capabilities. Join Ken Pugh to find out about creating issue stories, looking for root causes, and making fix/avoid decisions.

T5 TESTING

Agile Software Testing Strategies

Jared Richardson, 6th Sense Analytics

Test automation is like exercise. We know both are great ideas, but most of us don't do enough of either. Although we know that creating a solid automated test suite is critical to any agile testing strategy, we are often just told to "do it" without much support—money or people. Jared Richardson examines the infrastructure and tools needed for your automated testing to succeed and prosper. He describes three strategies—test-driven development, defect-driven testing, and blitzkrieg testing—that you can use to ensure excellent test coverage on your projects. Gain an understanding of how to leverage your testing investments by employing continuous integration practices in your development projects. With real-life scenarios as a backdrop, Jared discusses appropriate testing strategies for your current project or the next one down the road. Jared will get you moving toward automated testing, whether you're starting fresh or trying to clean up an existing project.

T6 AGILE DESIGN & ARCHITECTURE

Test-Driven Development Takes on Embedded Software

James Grenning, Renaissance Software Consulting

Embedded software developers face the same challenges as other software developers—unpredictable schedules, poor quality, and the problems that follow. In addition, embedded software developers must overcome the realities of concurrent hardware/software development, scarce target hardware availability, long download times, high deployment costs, as well as the challenges of testing embedded C. Test-Driven Development (TDD), a key agile practice, helps software developers improve schedule predictability and product quality, but very few embedded software engineers apply TDD to their craft. James Grenning describes the problems addressed by TDD, as well as the additional challenges and benefits of applying TDD to embedded software. He provides valuable lessons for doing TDD in the hostile environment of C. After the class, get hands-on experience with James' independent study exercise, "TDD in C."

T7 SPECIAL TOPICS

Assessing Your Agility

Mike Cohn, Mountain Goat Software and Kenny Rubin, Innolution

Are you curious how "agile" your organization is? Do you wonder how you compare with other organizations that have been using agile for a similar amount of time? Do you want an authoritative source of information to help guide your successful transition to agile? Mike Cohn and Kenny Rubin present a framework for assessing organizational agility. Specifically, they examine the areas of teamwork, requirements, planning, technical practices, quality, culture, and knowledge creation. Mike and Kenny describe how to use a framework to assess agility at various times during an organization's adoption of agile and how to derive actionable information from each assessment. Mike, Kenny, and their colleagues have collected over 300 assessment surveys from participants working on agile projects around the world. They will present preliminary industry-specific findings derived from analyzing the results of these assessments.

"Excellent conference, especially for the first running. I plan to attend next year and encourage heavy participation from my organization."

— Deborah Moseley, Program Manager Specialist

THURSDAY, NOVEMBER 13, 12:45 p.m.

T8 PEOPLE & TEAMS

"With Great Power Comes Great Responsibility"—Empowering the Agile Team

V. Lee Henson, VersionOne

Managers at many levels are often afraid to let go of the reins for fear of losing control of the project (and their position of power). V. Lee Henson explains the benefits of letting go and outlines the expectations of a responsible, empowered agile team. Through presentation of multiple real-world scenarios and years of project management experience, Lee will show that often our own human nature is the greatest impediment to being a better manager. Lee focuses on the attributes of an effective agile manager/leader, the expectations and attributes of an empowered agile team, the pitfalls and warning signs of a "damaged" team, and the rewards an organization can expect from adhering to basic agile principles. You will leave with the tools to help any agile team become more empowered.

T9 AGILE MANAGEMENT

A Lean Approach to Managing the Project Portfolio

Johanna Rothman, Rothman Consulting Group

Whether you've been agile for a while or are still thinking about it, you have one thing in common with every other software team I've encountered. You have too much work to do. One way to organize your work is with a project portfolio. But if your portfolio is an "as desired" portfolio, you still haven't solved the problem of too much work. Fortunately, taking a lean approach to managing the portfolio helps make those problems of "desired" and "too much work" transparent. Johanna Rothman discusses the several choices you have in managing the portfolio. She presents a project portfolio plan at the highest level and the lowest level and describes how to apply rolling wave planning to your project portfolio. Johanna discusses ways you can evaluate the projects in your portfolio, whether it's a kanban board, a fixed queue, a fixed timebox, or other evaluation approach.

T10 AGILE PROJECTS

Maximizing Team Dynamics and Overcoming Dysfunction in Agile Environments

Michael Mah, QSM Associates, Inc.

Change can be painful, but staying stagnant can hurt even more. Deciding to "go agile" may be the right choice for many companies, but seeing Scrum or XP as the next silver bullet can be a mistake—or perhaps the right medicine at the wrong time. In the rush to be faster, better, cheaper, or super-innovative, it's possible to become trapped in organizational dysfunction, even to the extent whereby good medicine won't work. When companies seek to "become agile," what roadblocks might they hit that could increase risk of failure? Michael Mah presents examples of companies that have overcome problems, plus a few who didn't. Learn how systems theory plays a role in software development, why complex communication and expert thinking are the penultimate challenges facing knowledge workers today, and how accurate and reliable metrics are to revealing patterns that help managers find the right path through their software development jungle.

T11 AGILE PROCESSES

The Business Value of Pair Programming

Rob Myers, Net Objectives

After ten years in the public eye, pair programming (two people seated together, working on the same programming task) is still one of the most controversial of all the agile programming practices. Managers are concerned about the cost of having "two doing the work of one," and developers are concerned about what will happen to their privacy, their reputations, and their personal performance metrics. Rob Myers dispels these and other concerns by examining the Lean aspects of the technique and by describing subtle (yet high-dividend) benefits. Rob's "Top Ten" contains only benefits that provide clear and significant value to the organization, the team, and the individual. Rob also gives some clear advice on how to try this technique and evaluate the results. You will be better equipped to weigh the costs, benefits, and ROI of pair programming, and to decide whether or not it is valuable for your organization.

T12 TESTING

Agile Usability Testing

John De Goes, N-BRAIN, Inc.

Agile development has become mainstream in the past few years, and for thousands of companies around the world, it has succeeded in reducing risk and delivering more value for less money. Yet, with the emphasis on pleasing the customer and the philosophy of doing the simplest thing that could possibly work, there's one area where agile development has fallen short of more traditional methodologies—creating highly usable software. Practices such as test-driven development and continuous integration show little concern for the end-user experience. John De Goes explains the importance of creating humane software and how he has integrated user-interface design and usability testing into the tight feedback loop that is the hallmark of agile development processes. John illustrates this process with snippets from usability tests that he has conducted, feedback from end users, and data on how usability testing improves customers' ROI.

T13 AGILE DESIGN & ARCHITECTURE

Getting Agile with Legacy Code

Steve Berczuk, Cyrus Innovation

Applying agile methods to legacy code is challenging. You have to live with code that is not as testable or as modular as you'd like, and you have to manage support concerns that disrupt your iteration plan—all while trying to establish new build and testing practices. Even if your project is agile, you may have dependencies on legacy projects that are not delivering at iteration boundaries. Steve Berczuk explains how to be agile with a legacy code base using design, testing, build, and software configuration management practices so that you can be agile even when your code is not. You'll leave better understanding how to balance the requirements of working with an existing code base with the desire to have a more agile development environment.

T14 SPECIAL TOPICS

Agile Project Inception: Escaping the Waterfall

Kenny Rubin, Innolution

Whether you are working on a new development effort or the next release of an existing system, you are probably required to make a compelling business case for the proposed work to clear an approval committee's "go/no-go" process. As an approval prerequisite, many organizations require big up-front planning and estimating resulting in a "complete" project plan including dates, costs, and resources. However, a key aspect of agility is an incremental, just-in-time approach to planning and estimating. Kenny Rubin focuses on how to align management and agile teams to eliminate the message of "build in an agile way but still provide all of the same waterfall-like artifacts to get your project approved." Kenny describes different resource-allocation and organizational-constraint models and then focuses on agile techniques for incrementally answering reasonable versions of questions, such as: "When will the project be completed?" "How much functionality can be developed by a particular date?" and "How much will the project cost?"

"I captured new and helpful information from these sessions. Our company has been developing applications in agile for about 1½ years now. It feels good to know that we're applying practices that the speakers have indicated, and at the same time I've heard new things that will help us improve our process."

— Agile Development Practices 2007 Delegate

T15 PEOPLE & TEAMS

Mistakes Agile Teams Make

J. B. Rainsberger, Independent Consultant

The road to hell is paved with good intentions—with a special section reserved for those who have tried to "go agile." Agile adoption can fail because a number of common, large-scale, organizational issues. A lack of executive-level support can squash promising improvements among the day-to-day producers. Sometimes the organization is in such disarray that delivering perfect features perfectly wouldn't keep customers satisfied. While these are real and important, J. B. Rainsberger suggests you'll find it more productive to focus on issues over which you have real influence. J. B. describes a few relatively simple mistakes, the warning signs to look for, and how to solve the problems. Hear useful stories from an experienced agile coach that include, "If I'd only known then what I know now..." You'll laugh, you'll cry, and with luck, you'll catch a problem or two before it blows up on you.

T16 AGILE MANAGEMENT

Agile Growing Pains

Michael Kirby, Xerox

Often, examples of agile successes are presented in the context of small, software-only development teams. Michael Kirby describes what it took to deploy agile development techniques in a large, embedded software development organization. Michael describes the successes—and some of the failures—of deploying agile development in Xerox's Production Printer Development Team. Learn about the adaptation of Scrum (what happens when the project manager gets voted off the island), agile planning (what's a user story when the only observable behavior is to power up the device), and test-driven development and automated acceptance testing (until a paper jam occurs in the middle of the night). Michael describes the cultural barriers encountered at Xerox in trying to transition a large development team from "traditional" software development to a more agile development approach. Understand the challenges of deploying agile development in an embedded domain and how you can be successful in your organization.

T17 AGILE PROJECTS

Scaling Agile Up and Out: A Tale from the Trenches

Ade Miller, Microsoft Corporation

It seems like everyone wants to scale their agile teams. As projects grow in scope, the agile approach to software development needs to scale up to larger team sizes. Agile also needs to scale out to handle geographically distributed teams as businesses expand into new markets and seek the best talent available globally. These are challenging propositions for many teams. Ade Miller talks about his experiences at Microsoft®—scaling agile up on the Visual Studio® Tools for Office team and scaling out on the radically distributed teams within the patterns & practices group. Ade covers the approaches used—some that worked well some not so well—and shares that the important thing is what was learned and how this new knowledge can be applied successfully to other projects. Ade presents some successful practices when scaling agile projects as well as some key pitfalls to avoid on your projects.

T18 AGILE PROCESSES

Retrospectives in Action: Looking Back at the Conference

Jean Tabaka, Rally Software Development

In this last-day, last-hour session, Jean Tabaka invites you to apply a fundamental practice of agile teams—retrospection. Jean guides you in facilitating your own retrospectives about the Agile Development Practices conference you have just attended. You will mine your experiences by creating a timeline of your conference observations, your high points, your low points, and your conference activities. Each team will then use their timeline of observations, impressions, and actions to interpret how their overall conference experience might impact what they could do differently at the next conference, what they would recommend as changes for the Agile Development Practices conference organizers, and what they might recommend even outside of the conference context. Finally, each team will prioritize their recommendations to be collected and delivered to the Agile Development Practices conference organizers. Join Jean in a fun look back and a chance to impact future Agile Development Practices conferences!

T19 TESTING

Picking the Right Test Automation Strategy for Your Project

Gerard Meszaros, Solution Frameworks, Inc.

The choice of a test automation strategy is a key determining factor in whether your test automation initiative will repay your investment or become a sink hole devouring time and money. Gerard Meszaros helps you understand what kinds of tests you should be running, which ones should be automated, who should prepare the tests, what tools they should be using, and when the tests should be prepared and run. A well-executed test automation strategy is key to preventing defects rather than finding defects after they have occurred. Gerard gives you the information you need to make an intelligent decision about how to approach test automation to minimize cost and maximize quality.

T20 AGILE DESIGN & ARCHITECTURE

Agile Engineering for Architects

Ryan Shriver, Dominion Digital

Agile development requires architects and architecture, but the current user story-centric approach ignores the qualities of systems, instead overly focusing on features and functions. Sadly, developers who depend primarily on stories miss enormous value-creation opportunities for stakeholders. Developers often think system qualities (non-functional requirements) are someone else's responsibility to define. Even when defined, these qualities are typically not quantified, instead specified with vague terms such as "faster" and "better." Ryan Shriver believes that architects are needed to build successful systems. Architects think like engineers and actively participate in quantifying the key system qualities and then designing to implement those qualities. Ryan demonstrates techniques architects can use that are lightweight, but rigorous, and can be applied to agile projects of any size. Learn about practical tools and techniques you can immediately apply on your projects—proven techniques that are used in designing and building very complex and robust systems.

T21 SPECIAL TOPICS

Introduction to Multi-Stage Continuous Integration

Damon Poole, AccuRev

A full, continuous integration build and test is a key component of most agile processes. Unfortunately, as systems grow in size through consecutive iterations, these builds can easily take thirty minutes or more. Before you finish the build, other people's check-ins will invalidate your continuous integration (CI) results. Multi-stage CI solves this problem by limiting project-wide churn and allowing CI to scale to large projects. With Multi-Stage CI, each team does a team-based CI first and then cross-integrates the team's changes into the mainline code base. Damon Poole introduces the Multi-Stage CI process, discusses its benefits, presents examples of how to implement and automate it in both local and distributed environments. Join Damon as he shows how to avoid pitfalls, such as increased merging overhead and unnecessary integration delays, and describes how to include a variety of stages, such as multi-team integration, QA, code reviews, and more.

"I liked the ability to hear other views on agility—to [get a] better perspective of what others are doing."

— William Ploch, Project Leader, Energizer

Agile Leadership Summit: An Executive View of Where Agile is Heading

By Agile Project Leadership Network (APLN)

November 14, 2008

Whether you are skeptical of agile practices or a leader who is tasked with bringing about the changes needed for agile practices to reach their full potential, the Agile Leadership Summit is for you. Agile adoption offers opportunities for faster time-to-market and significantly higher product quality, with the bonus that applications remain malleable and easier to change over their lifetime. However, to gain these benefits, you need to align your organization's culture, priorities, and structure with agile principles. Join leaders from the industry—Steve Greene, Director of Tools & Processes at Salesforce.com; Sue McKinney, Vice President of Development Transformation and Integration at IBM; Niel Nickolaisen, CIO and Director of Strategic Planning at Headwaters, Inc.; Israel Gat, Vice President of BMC's Distributed System Management; and others—who have embraced agile principles and hear them share how they have put agile to work successfully in their organizations.

The Agile Leadership Summit is the perfect opportunity to meet and network with your peers and learn proven ways to improve software delivery by evolving people and practices to be the best they can be.

WHO SHOULD ATTEND?

Anyone interested in the exploration of agile leadership of projects and enterprises.

The Agile Leadership Summit offers you a unique opportunity to start or to continue your work toward being a great agile leader. The summit is for both new and seasoned agile leaders—at all corporate levels, including CIOs, CTOs, Directors, General Managers, Product Managers, and Project and Team Leaders.

Whether you are just thinking about bringing agile into your organization or you have been in the trenches before, this is your chance to spend time with some of the world's leading practitioners and experts in the area of agile leadership. Take this opportunity to network and share your experiences and concerns with them and other leaders who are in your same situation.

The program—for novice to expert agile leaders who are interested in learning more—educates through a tutorial and experience track with discovery in "Think Tank" breakout sessions that build on each other as the day progresses. The Agile Leadership Summit provides:

- Networking opportunities throughout the day
- A suite of intensive sessions and experience reports
- "Think Tank" discussion sessions on agile leadership with topics addressing advanced leadership tools, experiences, lessons learned, and issues yet to be decided by the group
- A panel discussion over lunch where practicing agile leaders discuss the value of agile leadership in the enterprise

BENEFITS OF ATTENDING

Build a network of leaders to connect with for support as you explore agile leadership. By attending the Agile Leadership Summit you'll have the opportunity to:

- **Learn how IBM is converting 25,000 developers to agile methods in distributed environments**
- **See what changes BMC made to complete projects in five months that would usually take one year to complete**
- **Discover how Salesforce.com converted their entire company to agile overnight and what preparation they did to make this conversion a success**
- **Gain advice from leaders who have "been there" and what they have learned in their transitions to agile**
- **Learn how IT and business leaders can rapidly align business and IT, develop pragmatic strategic and tactical plans, and improve decision-making in agile environments**
- **Explore the pitfalls other leaders experienced and how they successfully navigated them to deliver results**

Agile Leadership Summit Schedule

THURSDAY, NOVEMBER 13

5:30	**Leadership Summit Reception Sponsored by IBM** *(5:30 p.m. - 7:00 p.m.)*

FRIDAY, NOVEMBER 14

7:30	**Registration and Breakfast**
8:30	**Welcome — Think Tank Issues Identification: As a Leader, What Is Keeping You Up at Night?**
9:00	**Wherefore Art Thou Business Value?** *Niel Nickolaisen, CIO and Director of Strategic Planning, Headwaters, Inc.*
9:45	**Unleashing the Fossa: Scaling Agile in an Ambitious Culture** *Steve Greene, Director of Tools & Processes, Salesforce.com*
10:30	**Break — Prioritization of Think Tank Issues**
11:00	**Think Tank Session: Leadership Solution Brainstorm and Discussion**
12:00	**Networking Lunch Buffet**
12:30	**Panel Discussion: Advice for Leaders Wanting to Try Agile**
1:30	**Leading the Agile Disruption** *Israel Gat, Vice President of Distributed System Management, BMC*
2:15	**Networking Break**
2:45	**The Dancing Agile Elephant: IBM Software Group's Transition to Agile and Lean Development** *Sue McKinney, Vice President of Development Transformation and Integration, IBM*
3:30	**Breakout Session: Meet the Speakers**
4:30	**Wrap Up Session**
4:45	**Ongoing Informal Discussions with Speakers and Attendees**

Leadership Summit Reception *Sponsored by IBM* *(5:30 p.m. - 7:00 p.m.)*

Enjoy complimentary food and beverages during a private reception for Agile Leadership Summit participants.

From Adrenalin Junkies to Template Zombies...

Presented by Tim Lister

Join Tim Lister as he discusses how we all talk about "best practices" but few organizations actually practice them. Tim believes that patterns are stronger than best practices. They are the habits, the decision practices, the corporate culture, and the unstated rules, that dominate office life. Participate by looking at your own organization's patterns. If positive, how can you perpetuate them across all projects? If negative, how can you break the habit?

Agile Leadership Summit Sessions

November 14, 2008

9:00 a.m.

Wherefore Art Thou Business Value?

Niel Nickolaisen, CIO and Director of Strategic Planning, Headwaters, Inc.

Top leaders face a wide range of competing demands. We need to support legacy systems while also taking the lead in innovation. We must absorb changing technologies while sorting through which technologies can propel the business forward—and do all of this in a constantly changing business and competitive environment. How can we ensure that we make the right decisions at the right times? How can we ensure that our decisions improve business value? In this interactive session, Niel Nickolaisen presents the Purpose Alignment model and explains how IT and business leaders use this model to rapidly align business and IT, develop pragmatic strategic and tactical plans, and improve decision-making. Those attending this session will leave with the tools they need to immediately deliver improved business value.

***Niel Nickolaisen** has held technology executive (CIO) and operations executive (COO) positions in large- and medium-sized enterprises, typically in turnaround roles. He is expert in the rapid/adaptive selection, implementation, and deployment of enterprise business applications, analysis tools, and systems. He has developed a strategic and tactical alignment model that results in significantly improved returns on technology and business initiatives (by both improving the benefits and reducing the costs and risks). Named IT Executive of the Year at the 2004 Gartner Research Mid-Sized Enterprise Summit, writes for* CIO Leaders Magazine *and* Cutter Consortium.

9:45 a.m.

Unleashing the Fossa: Scaling Agile in an Ambitious Culture

Steve Greene, Director of Tools & Processes, Salesforce.com

How do you make an agile transformation? Should you transition your organization to agile all at once or proceed iteratively, team by team? Salesforce.com moved their entire R&D organization to an agile model. What was the key difference in their approach? They threw the switch on thirty teams all at once. Most agile experts thought they were crazy. However, in the end, the transition became one of the fastest and largest successful agile transitions. Steve Greene talks about how, in just three short months, they moved their entire team from a waterfall-based approach to a Scrum-based methodology call Adaptive Development Methodology (ADM). The technology team uses this methodology to regularly deliver three to four major releases a year to over 43,000 customers via more than 150 million transactions per day. Steve will present the business value they have achieved and the results of our team-wide survey sampled every quarter.

***Steve Greene** is the director of Tools & Process at Salesforce.com and is responsible for the implementation and evolution of agile methodologies and supporting tools for the Technology organization. He has held numerous senior management positions at On-demand startup and large enterprise software companies including DigitalThink, Hyperion, PeopleSoft/Oracle, SPC and AOL. He brings a wealth of expertise and experience in productivity, process and product delivery. He holds a BS in Computer Engineering from San Jose State University and is a board member of the BayAPLN.*

1:30 p.m.

Leading the Agile Disruption

Israel Gat, Vice President of Distributed System Management, BMC

One year delivery times shortened to four to five months? With productivity and quality levels far in excess of industry norms? Israel talks about how they reached this level of agile excellence in R&D at BMC, and what it meant from an end-to-end perspective—a major disruption to Marketing, Sales, Field Enablement, Software Consultants, Professional Services, and Customer Support. You will learn the set of processes developed and put into use in Israel's team that were used to mitigate methodical disruption and facilitate alignment across corporate functions. In particular, you will be exposed to the Agile-Based-Market-of-One concept that is showing great promise as the antidote to the innate corporate drive to "harness" disruptions. Israel also discusses candidly what he learned about leading agile and passes on to you what worked and what did not at BMC.

***Israel Gat** is the head of BMC's Distributed System Management—a business unit that is 100% "Scrummed." His executive career has spanned some of the world's top technology companies, including IBM, Digital, Microsoft, EMC and BMC. Digital's NetView, EMC's Cellera, Microsoft's MOM 2000 and the BMC Performance Manager are some of the products he brought to market during his tenure with these companies. In addition to his experience with industry giants, Israel is also well versed in growing smaller companies and has held advisory and venture capital positions for companies in new, high-growth markets. Israel holds a Ph.D. degree in Computer Sciences from the Israeli Institute of Technology and an MBA degree from Clark University.*

2:45 p.m.

The Dancing Agile Elephant: IBM Software Group's Transition to Agile and Lean Development

Sue McKinney, Vice President of Development Transformation and Integration, IBM

Twenty-five thousand developers transitioning to Agile? How does that happen? Faced with a results-driving organization, moving to agile had to deliver on the company's investment into a new methodology. In this session, Sue McKinney talks about the approach and challenges to transforming a multi-thousand person globally-distributed division, to adopt and deliver using agile methods. Learn how to inspire and motivate change in large organizations steeped in tradition, identify change agents, and discover the tools needed to enable the masses. Sue discusses how she navigates leadership expectations and what new leadership skills must be developed in teams to make a successful adoption of agile—and to maintain effective use of agile. What Sue and other IBM-ers did to overcome the barriers and issues will be included, especially around resistance to change in large organizations.

***Sue McKinney** is currently responsible for development transformational activities with IBM's software development group, her major emphasis is driving adoption of Agile and Lean principles into the mainstream of software development. Prior to this assignment, Sue was a Vice President of Development for the Lotus Division where she led worldwide development for Lotus Domino, IBM Sametime, and Websphere Portal. In addition to driving transformational activities within IBM, Sue works with large clients to share IBM's experience and help them scope opportunities for their own transformational activities.*

ROSEN SHINGLE CREEK RESORT

Orlando, Florida

Rosen Shingle Creek proudly presents 1,500 luxury guestrooms and suites that are both functional and classically styled with soothing earth tones and sumptuous bedding. Most rooms feature pool and golf views. The alluring suites, ranging from expansive hospitality parlors to the 1,744-square-foot presidential suites provide the perfect atmosphere for intimate gatherings.

Capitalizing on Rosen Shingle Creek's elevation, every single room offers a stunning view of the golf course and the area's incomparable setting—from the perfectly manicured fairways and greens to the picturesque creek, magnificent cypress trees, and lush natural vegetation. For vacationers or meeting guests, it's hard to imagine a better way to start the day than with the view from these rooms.

Book your reservation for your stay at the Shingle Creek Resort for a special discounted conference rate. To make your reservations, visit **www.sqe.com/go?ADP08Hotel** and reference the Software Quality Engineering Agile Development Practices conference.

THINGS TO DO AND SEE WHILE YOU'RE IN ORLANDO

- Visit Disney World® and discover the fun where imagination reigns
- Have a drink at the tiki bar in Tchoup Chop, Emeril Lagasse's restaurant
- Get up close with the wildlife at SeaWorld® Orlando
- Dine at the world's largest Hard Rock Cafe®
- Feel the g-forces on a roller coaster at Universal's Islands of Adventure®
- Swim with the dolphins at Discovery Cove®
- Play golf on the same courses as Tiger Woods
- Shoot hoops at NBA City®

Courtesy of Orlando/Orange County Convention & Visitor's Bureau, Inc.

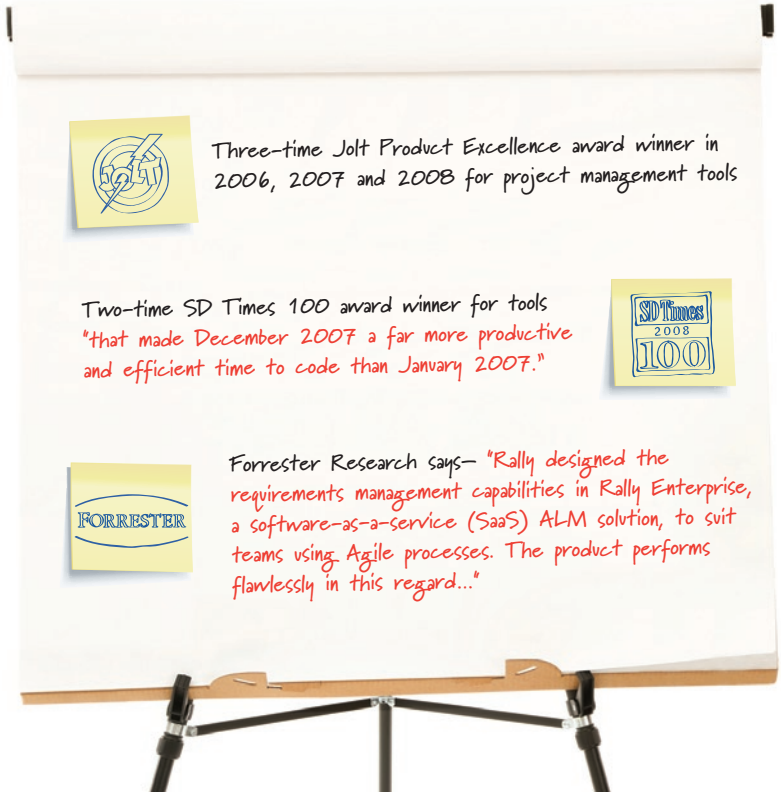
RALLY SOFTWARE

The Award-Winning-Est Agile Lifecycle Management Solution

Get your FREE trial of Rally Enterprise at

www.rallydev.com

No download, no installation
No commitment



CONFERENCE SPONSOR

Rally is the #1 partner for Agile success through its family of products, coaching and community support. With a dedicated focus on Agile, Rally helps software organizations of all sizes incrementally adopt Agile practices to shorten their development cycles and collaborate across distributed teams and silos. Rally's on-demand Agile lifecycle management products were honored with three-time Jolt Product Excellence Awards (the industry's Oscars) in 2006, 2007 and 2008 and support over 300 corporate customers and 12,000 subscribers across 30 countries. **www.rallydev.com**

INDUSTRY SPONSORS

Borland is the leading vendor of Open Application Lifecycle Management (ALM) solutions—open to customers' processes, tools and platforms—providing the flexibility to manage, measure and improve the software delivery process. Borland offers integrated Open ALM products, services and training to help customers transform software delivery into a managed business process. **www.borland.com**

Mountain Goat Software was founded by Mike Cohn, author of *Agile Estimating and Planning* and *User Stories Applied*, to offer coaching and training that helps teams succeed with agile. Onsite and public courses include Certified ScrumMaster, Certified Scrum Product Owner, Effective User Stories for Agile Requirements, and Agile Estimating and Planning. **www.mountaingoatsoftware.com**

Net Objectives provides targeted Lean-Agile Training Solutions with coordinated technical offerings in design patterns and test-driven development to help you foster sustainability, and deliver product and business value faster. We offer the most comprehensive Lean-Agile curriculum in the industry, Professional Scrum Certifications, Assessments, on-site Training, custom curricula, Coaching, and Enterprise Agility Consulting services for teams, management and IT. **www.netobjectives.com**

SolutionsIQ offers a full spectrum of services to develop software and fulfill technical talent needs, while improving your Agile knowledge and capabilities. A Microsoft Gold Certified Partner, clients include AT&T, Amazon, Corbis, Expedia, Federal Home Loan Bank, InfoSpace, Key Bank, Microsoft, Nordstrom, Safeco, and US Bank. Learn more at **www.SolutionsIQ.com**

VersionOne is recognized by Agile practitioners as the leader in Agile project management tools. Today more than 10,000 teams and 70,000 users in 50 countries use VersionOne's Agile project management tools to streamline and standardize their Agile development efforts. Since 2002, companies such as Adobe, Siemens, Sabre, Disney, IBM, Lockheed Martin, Sony, 3M and Business Objects have turned to VersionOne. **www.versionone.com**

IN COOPERATION WITH

MEDIA SPONSORS

PLUS SEE THESE EXHIBITORS AND SPONSORS AT THE EXPO (NOVEMBER 12–13)

Please note, these are the sponsors and exhibitors as of the brochure printing. Visit the Web site below for the most up-to-date information.

AccuRev

ACM Queue

Addison-Wesley Professional and Prentice Hall Professional

Agile Project Leadership Network

***Better Software* Magazine**

Borland

CM Crossroads

***CoDe* Magazine**

InfoQ

Methods & Tools

Mountain Goat Software

Net Objectives

Quantitative Software Management

Questcon Technologies

Rally Software Development

Scrum Alliance

Software Quality Engineering

SolutionsIQ

SQE Training

StickyMinds.com

TargetProcess

TechExcel

VersionOne

Agile Development Practices 2008 sponsors are listed in bold. For sponsor/exhibitor news and updates, visit:

www.sqe.com/agiledevpractices

AGILE DEVELOPMENT PRACTICES REGISTRATION

NOVEMBER 10–14, 2008 ORLANDO, FLORIDA, USA

Easy to Register

ONLINE:
www.sqe.com/adpreg

PHONE:
888.268.8770
904.278.0524

EMAIL:
sqeinfo@sqe.com

CONFERENCE PRICING

Registration Fees:*	On or Before Oct. 10	After Oct. 10	
❑ **VIP Package—All 5 Days** Includes: 2 days of Pre-conference Tutorials, 2 days of the Agile Development Practices Conference, 1 day of the Agile Leadership Summit	**$2,545**	**$2,745**	***Best Value!***
❑ Conference + Two Pre-conference Tutorial Days	$2,295	$2,495	
❑ Conference + One Pre-conference Tutorial Day	$1,995	$2,195	
❑ Conference Only (Wed. - Thurs.)	$1,695	$1,895	
❑ Two Pre-conference Tutorial Days	$1,390	$1,590	
❑ One Pre-conference Tutorial Day	$795	$895	
❑ Agile Leadership Summit	$795	$895	
❑ Agile Leadership Summit Add-on Add the Agile Leadership Summit to any package for additional savings!	$395	$395	

Special Early Bird Offer! Receive $200 off the regular conference registration fee if payment is received on or before October 10, 2008. See discounted pricing information above.

PAYMENT INFORMATION — The following forms of payment are accepted: Visa, MasterCard, American Express, check, or company purchase order. Payment must be received before the registration is confirmed. Make all checks payable to Software Quality Engineering. You will receive a confirmation package upon payment by check, credit card, or company purchase order. Payment must be received at Software Quality Engineering on or before October 10, 2008, to take advantage of the Early Bird conference rates listed above.

HOTEL RESERVATIONS — Take advantage of the special conference hotel rate. Call 866.996.6338 to make your reservation, or visit **www.sqe.com/go?ADP08Hotel**. Space is limited, and early booking is recommended.

CANCELLATION POLICY — Registrations cancelled after October 20, 2008, are subject to a 20% cancellation fee. No cancellations or refunds may be made after October 27, 2008. Substitutions may be made at any time before the first day of the program. Call our Client Support Group at 888.268.8770 or 904.278.0524 to obtain a cancellation code. All valid cancellations require a cancellation code.

SATISFACTION GUARANTEE — Software Quality Engineering is proud to offer a 100% satisfaction guarantee. If we are unable to satisfy you, we will gladly refund your registration fee in full.

MEDIA RELEASE — From time to time we use photographs, video, and audio of conference participants in our promotional materials. By virtue of your attendance at the Agile Development Practices conference, you acknowledge that Software Quality Engineering, Inc., reserves the right to use your likeness in such materials.

* Your registration fee includes $39 for a one-year digital subscription to *Better Software* magazine. If you are a current subscriber, your subscription will be extended an additional ten issues.

EVENT LOCATION

Rosen Shingle Creek is nestled on a 230-acre site along Shingle Creek just off Universal Boulevard, east of the Orange County Convention Center North/South expansion and just 10 minutes from the Orlando International Airport. This ideal location is just a short distance from a variety of Orlando's best attractions, restaurants, shopping, and entertainment venues.

SPECIAL HOTEL RATES FOR AGILE DEVELOPMENT PRACTICES ATTENDEES!

Book your reservation for your stay at the Shingle Creek Resort at a special discounted conference rate. To make your reservation, visit **www.sqe.com/go?ADP08Hotel** and reference the Software Quality Engineering Agile Development Practices conference. If you need special facilities or services, please notify the agent at the time of your reservation. Cancellations on a guaranteed reservation must occur no fewer than five days prior to the specified arrival time to ensure a refund. Make your reservation early.

ONLINE ACCESS AT THE CONFERENCE

All guestrooms have free high speed Internet. A WiFi lounge will be available to conference attendees during conference hours.

WAYS TO SAVE ON YOUR CONFERENCE REGISTRATION

Special Early Bird Offer!
Receive $200 off your registration fee if payment is received on or before October 10, 2008.

PowerPass Discount
PowerPass holders receive an additional $100 off their registration fee. Not a PowerPass member? Visit **www.StickyMinds.com/PowerPass** to learn more.

Bring a Buddy!
Bring a colleague and each of you saves an additional $200.
Any two people registering at the same time save an additional $200 off each registration. Please call the Client Support Group at 888.268.8770 or 904.278.0524 to register.

Alumni Discount
Both Agile Development Practices and Better Software Conference & EXPO alumni receive up to an additional $200 discount off their registration fee.

For **Group Discounts** or more details on our discount policy, please contact the Software Quality Engineering Client Support Group at sqeinfo@sqe.com or 888.268.8770 or 904.278.0524.

GROUP DISCOUNTS AVAILABLE

36 Pre-conference Tutorials — *Monday and Tuesday*

46 Keynotes & Classes and Networking EXPO — *Wednesday and Thursday*

Agile Leadership Summit — *Friday*

Agile Leadership Summit: An Executive View of Where Agile Is Heading
Co-located Event — November 14, 2008

THE EXPO
Nov. 12–13, 2008

Visit Top Industry Providers Offering the Latest in Agile Development Solutions

BETTER SOFTWARE
Agile Development Practices

November 10–14, 2008
Orlando, FL • Shingle Creek Resort

Conference Sponsor: RALLY SOFTWARE

www.sqe.com/agiledevpractices REGISTER EARLY AND SAVE $200!

Industry Sponsors:
Borland THE OPEN ALM COMPANY, Mountain Goat Software, Net Objectives, SolutionsIQ Knowledge. Software. Talent., VersionOne Simplifying Software Delivery

In Cooperation With:
ScrumAlliance your trusted source of Scrum knowledge

Media Sponsors:
acm queue, Addison Wesley, Prentice Hall, APLN, Better Software Magazine, cm//crossroads, InfoQ, Methods & Tools, CoDe www.code-magazine.com, StickyMinds.com

SOFTWARE QUALITY ENGINEERING CONFERENCES

330 Corporate Way, Suite 300
Orange Park, FL 32073

IF ADDRESSEE IS NO LONGER EMPLOYED:
Re-route to Director of Software Development