

September 2008

\$9.95 [www.StickyMinds.com](http://www.StickyMinds.com)

# BETTER SOFTWARE

**GPS OPTIONAL**  
Navigating  
acceptance test-  
driven development

**HIGH HOPES**  
Building trust on  
your team

The Print Companion to  **StickyMinds.com**

SO,  
**YOU'VE  
GOT A  
PROBLEM...**

CRAFTING REMARKS & ABSTRACTS FOR  
MORE EFFECTIVE DEFECT REPORTS



**FIND THE BUG INSIDE & WIN  
AN IPOD SHUFFLE™!**



# IBM®



IBM, the IBM logo, ibm.com, the Rational logo and Take Back Control are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol ("®" or "™"), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml). ©2008 IBM Corporation. All rights reserved.





**Rational**

\_INFRASTRUCTURE LOG

\_DAY 64: We're rushing our new business capabilities to the Web so fast that we might be taking unnecessary risks. Are we secure? Are we compliant? How prepared are we for the future? I wonder what's waiting for us around the corner.

\_Maybe I just have an overactive imagination.

\_DAY 67: The answer: IBM Rational AppScan. It gives us the tools we need to build security and compliance into our applications from the start and throughout their entire lifecycle. Now we can find the vulnerabilities and security issues in our apps and Web sites and fix them before they become a problem. I've never felt safer.

\_Maybe now I can turn the night-light off in my office.

Download a free trial of IBM Rational AppScan at:  
[IBM.COM/TAKEBACKCONTROL/SECURE](http://IBM.COM/TAKEBACKCONTROL/SECURE)





**Take quality to the next level**



## **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](http://www.techexcel.com) | 1-800-439-7782





Find the Bug  
and win an iPod Shuffle™!  
Search through the digital  
edition to find the flying  
bug. Click it to be entered for a  
chance to win an iPod Shuffle™.

## Cover Story

### SO, YOU'VE GOT A PROBLEM ... 30

Defect reports are among the most important deliverables to come out of test. It's worth the effort to learn how to write an effective defect report that conveys the proper message and simplifies the process for everyone. *by Kelly Whitmill*

## Features



### FROM HERE TO ACCEPTANCE TEST-DRIVEN DEVELOPMENT 24

Acceptance test-driven development (ATDD) means different things to different people based on their experiences. These nine landmarks will help you navigate ATDD no matter where you are coming from. *by Antony Marcano*

### A CULTURE OF TRUST 36

So, you've been asked to take over the leadership of a struggling, disconnected team. Now what? Discover how to create a culture where trust between team members is fostered, flourishes, and thrives. *by Pollyanna Pixton*

## Columns & Departments

### In Every Issue

Mark Your Calendar 4

Contributors 6

eLightenment 9

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](http://www.BetterSoftware.com)  
or call 800.450.7854.

### TECHNICALLY SPEAKING 15

Is "Agile" Distracting You? • *by Jonathan Kohl*

If a process, tool, or service claims to be agile it must be good, right? Not necessarily. Let's look beyond the label to what really matters—value.

### CODE CRAFT 16

Idioms and Idiosyncrasies • *by Pete Goodliffe*

Common language idioms help to show the elegance, beauty, and artistry of a piece of code. But sometimes the desire for beautiful, idiomatic code can trip us up.

### TEST CONNECTION 20

It's in the Way that You Use It • *by Michael Bolton*

Rapid testers think of test automation as any use of tools to support testing. With that definition in mind, it may not be the most obvious automation tool that is the most useful.

### MANAGEMENT CHRONICLES 22

Exit, Stage Left • *by Patrick Bailey*

Many technology workers are drawn to the industry from seemingly unrelated professions. Don't discount the importance of the experiences these workers can bring to your team.

### THE LAST WORD 47

Keys to Top-Notch Estimates • *by Howard Smallowitz and George Stark*

Inaccurate project estimates have become the norm in the software industry. Find out how you can turn your estimates into reasonable predictions of project performance.

**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](http://www.sqetraining.com/Public)

#### Testing

**October 20–24, 2008**

San Francisco, CA

**November 17–21, 2008**

Tampa, FL

#### Agile Software Development

**September 22–26, 2008**

Denver, CO

**October 6–10, 2008**

Chicago, IL

**October 20–24, 2008**

Seattle, WA

### SOFTWARE TESTING CERTIFICATION

[www.sqetraining.com/certification](http://www.sqetraining.com/certification)

**September 23–25, 2008**

Philadelphia, PA and Atlanta, GA

**September 28–30, 2008**

Anaheim, CA

**September 30–October 2, 2008**

Indianapolis, IN

**October 7–9, 2008**

Jacksonville, FL and Toronto, ON

### CONFERENCES

#### STARWEST 2008

#### Software Testing Analysis & Review

[www.sqe.com/starwest](http://www.sqe.com/starwest)

**September 29–October 3, 2008**

Disneyland Hotel

Anaheim, CA

#### Agile Development Practices 2008

[www.sqe.com/agiledevpractices](http://www.sqe.com/agiledevpractices)

**November 10–14, 2008**

Shingle Creek Resort

Orlando, FL

# BETTER SOFTWARE

Publisher

**Wayne Middleton**

Vice President of Publishing

**Holly N. Bourquin**

#### Editorial

Editor

**Heather Shanholtzer**

Managing Technical Editor

**Lee Copeland**

Technical Editors

**Chuck Allison**

**Jonathan Kohl**

**Antony Marciano**

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](mailto:editors@bettersoftware.com)

Subscriber Services: [info@bettersoftware.com](mailto: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



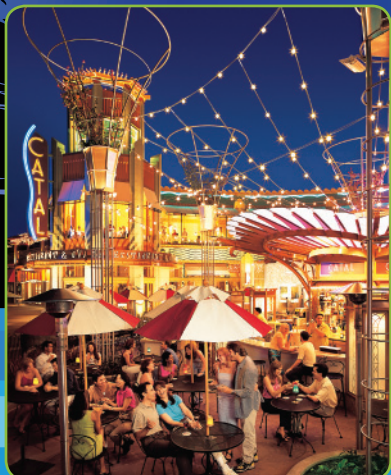
**STAR  
WEST**

# SOFTWARE TESTING

ANALYSIS & REVIEW

*The Greatest Software  
Testing Conference on Earth*

Anaheim  
**California**



Conference  
Sponsor:

**twist**<sup>TM</sup>  
collaborative test automation

[www.sqe.com/STARWEST](http://www.sqe.com/STARWEST)

**Register Early and Save \$200!**

**September 29–October 3, 2008**

**Disneyland® Hotel**

**Over 98% of 2007 Attendees  
Recommend STARWEST  
to Others in the Industry**



As to Disney photos, logos, properties: ©Disney

## KEYNOTES



**Testing Lessons from Springfield—Home of the Simpsons**  
Rob Sabourin, AmiBug.com



**Telling Your Exploratory Story**  
Jon Bach, Quardev, Inc.



**Six Thinking Hats for Software Testers**  
Julian Harty, Google



**Branch Out Using Classification Trees**  
Julie Gardiner, Grove Consultants



**Has the Time for the Adversarial Organization Passed?**  
Gerard Meszaros, Independent Consultant



**Testing Microsoft Office®: Experiences You Can Leverage to Drive Quality Upstream**  
Tara Roth, Microsoft

### Industry Sponsors:

**Borland**<sup>®</sup>  
THE OPEN ALM COMPANY



**McCabe**  
SOFTWARE



**ORACLE**<sup>®</sup>



### Media Sponsors:



**Methods & Tools**







**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](mailto:chuck@freshsources.com).



**PATRICK BAILEY** has more than twenty years of experience in software development. He now teaches full time at Calvin College in Grand Rapids, Michigan. Contact Patrick at [pmb4@calvin.edu](mailto:pmb4@calvin.edu).



**PACO HOPE** is a technical manager with Cigital, Inc. and has extensive experience securing Web applications, operating systems, and embedded devices (lottery systems, cell phones, casino gaming devices). He lectures on security testing and Web application security. Paco brings the techniques of security assessment into the mainstream activities of QA departments and testers.



**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](mailto:mb@developsense.com).



**PETE GOODLIFFE** is a software developer, columnist, speaker, and author who never stays in the same place in the software food chain. His projects range from OS implementation, through audio codecs, to multimedia applications. Pete's popular book *Code Craft: The Practice of Writing Excellent Code* is a practical and entertaining investigation of the entire programming pursuit—in about 600 pages. No mean feat! He has a passion for curry and doesn't wear shoes.



**JONATHAN KOHL** is a software testing consultant with Kohl Concepts Inc., based in Calgary, Alberta, Canada. Jonathan writes about and speaks on software testing and is a technical editor for *Better Software* magazine. Contact Jonathan at [jonathan@kohl.ca](mailto:jonathan@kohl.ca).

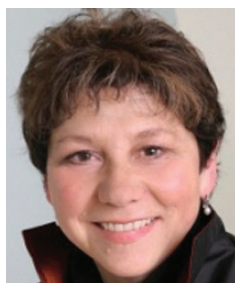


**BRIAN MIZELLE** brings twenty years of full lifecycle software development experience and leadership to Cigital. As a managing principal, he is responsible for working with our clients as a trusted advisor to help them achieve business and technical goals and objectives around software security, testing, and development.





**ANTONY MARCANO** has thirteen years of experience in software development and testing across numerous sectors. Since 2000, much of Antony's work has been on Extreme Programming projects. Now, as a practitioner, mentor, coach, and consultant, he helps teams realize the benefits associated with agile development. A regular speaker at peer workshops and conferences, Antony's views have been quoted in numerous publications including *Corporate Insurance & Risk* magazine and on VNUNet.com. Antony is creator and curator of testing Reflections.com, one of the most influential agile testing sites on the Internet, and is a technical editor for *Better Software* magazine.



**POLLYANNA PIXTON** developed collaboration tools for leaders through her thirty-eight years inside and consulting with corporations and organizations. She helps leaders create workplaces where talent and innovation are unleashed—making them more productive, efficient, and profitable while leading change in their marketplaces. She writes, teaches, and speaks internationally on collaborative leadership and business ethics. Currently, she is co-authoring a book titled *Stand Back and Deliver: A Leader's Guide to the Agile Enterprise*. Pollyanna is a founding partner of Accelinnova, president of Evolutionary Systems, and director of the Institute for Collaborative Leadership. You can contact Pollyanna at [ppixton@accelinnova.com](mailto:ppixton@accelinnova.com).



**GEORGE STARK** is a senior technical staff member in IBM's Technology and Integration Management Competency of the Integrated Technology Delivery organization. George has published more than forty papers in refereed journals. He is a member of the IBM Estimating Community core team and has worked for IBM for ten years. George can be reached at [gstark@us.ibm.com](mailto:gstark@us.ibm.com).



**HOWARD SMALLOWITZ, PMP**, is an ITIL- and PMI-certified project manager. He currently works as a systems engineer in IBM's Technology and Integration Management Competency where he manages the estimation review board. Howard has trained people around the globe on systems engineering tools and techniques, including estimating. He has worked at IBM for ten years and can be reached at [hsmall@us.ibm.com](mailto:hsmall@us.ibm.com).



**KELLY WHITMILL** has more than twenty years of experience in testing software. He is currently a senior software engineer at InfoPrint Solutions Company. Kelly's project experience includes client/server and Web-based applications, large mainframe applications, and a variety of other applications on multiple platforms. He has worked on teams of all sizes, which has provided an understanding of the varying team dynamics. Kelly has special interests in test automation, system testing, and practical test techniques and practices to improve testing. He has written many defect reports, both good and bad, and can provide practical tips on how to avoid the bad and embrace the good. Kelly can be reached at [kc\\_whitmill@yahoo.com](mailto:kc_whitmill@yahoo.com).

**EDITOR'S NOTE:** The bio of Rob Myers, author of "Encourage Pair Programming" (The Last Word July/August 2008), omitted that he is a senior consultant and instructor at Net Objectives. You can contact Rob at [rob@netobjectives.com](mailto:rob@netobjectives.com).



# 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](http://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.



# ET

## RANDOM TESTING OR TECHNIQUE?

In the face of tight timelines and large, complex projects, do you need exploratory testing (ET) or is it just a waste of time?

In an ideal world, the phases of software development go hand-in-hand with extensive documentation. Specifications and requirements for each part are set out neatly, all ready for testers to begin work.

But that doesn't happen very often, does it? You'd want the tester to get involved as early as possible in the software development life cycle, so that development and testing can happen in parallel. But that's when the software is just beginning to take shape, is likely to be unstable and buggy, and will probably be accompanied with little or no documentation. And, change requests, increments, and iterations will be the order of the day; so even if you have documentation, it's likely to be outdated and of little use. Can testers do anything in such a scenario?

The answer lies in exploratory testing (ET). Using this technique, skilled testers can gain understanding of your product—they learn more about it, explore various areas to assess how they work, and get a feel of the risky areas that are likely to throw up bugs. This will help them design test cases, execute them, and use the findings in further test design and execution.

That makes it sound like ad hoc testing. But nothing could be further from the truth. Ad hoc testing is usually short-term; it consists of tests that are run only once, unless a bug is discovered. ET is a wider and deeper concept—it covers not only ad hoc testing, but all forms of testing. All testing, even scripted or automated testing, is exploratory to a certain degree, since tests could be modified on the

basis of results obtained and defects discovered. For instance, if a tester has run a particular test case and found unexpected results, the tester is bound to 'explore' further, and run a few more tests—which may not be part of the test suite—to discover more about the bug. That's exploratory testing to a certain extent.

### A Waste of Time?

ET is often considered a waste of time—though the tester spends a lot of time trying to understand how the software works and in which scenarios it is likely to fail; what you get at the end is a list of bugs, which in any case, you would also get with automated testing. ET would be a far more productive exercise if the testers could share their understanding of the product and suggest improvements based on that. That is possible if processes to capture this information are introduced.

Moreover, several challenges are associated with ET. Since test design and execution occur simultaneously, the test design cannot be reviewed in advance. This could lead to errors in the code and in the test cases.

Challenges also arise in documenting the test cases that testers create on the fly, especially if they need to be rerun in future. Since there is very little documentation, it is difficult to find out which test cases were run at which point of time. Thus, if the tests are run a second time, it may not be in exactly the same way. This can cause difficulties if it is important to document what parts of the product were functional at each point in time.

### Make Exploratory Testing Work

Most of these challenges will cease to be important if you have a skilled testing team; or if you outsource the testing function to a skilled test services provider. ET relies heavily on the testers' knowledge, experience, skills, creativity, and intuition. So, to make ET work for your project, it is important to find the right skills. To optimize your use of ET, well-defined procedures and documentation are required.

That's why QA InfoTech is the right choice for you, if you have software that needs to be tested rapidly, but has little supporting documentation. We have processes and metrics in place for weekend projects—those with little or no documentation, which need to be tested over the weekend; the results would be ready for you by Monday morning.

Our highly skilled test team formulates an assessment strategy, by understanding what to test, why it should be tested, how to test it, and when to stop testing. We then manage the testing process to yield the best results. Our approach includes effective and timely communication with project managers, developers, and the marketing team. We use the Staticator to facilitate communication regarding test efforts, coverage and so on.

We also have a well-defined methodical process to develop the test metrics or the test deliverables. Other documentation would include summary of defects; and how severe or otherwise these are, so that defect fixing can be prioritized.

## How Exploratory Testing Helps

ET helps in testing the product more holistically and offers scope for improvisation in the test design.

**Faster defect finding:** Since ET requires no pre-documentation or preparation, the bigger and more important defects are found faster than with scripted testing.

**Better scope for improvisation:** Since testers are not bound to run the entire scripted test suite, they can modify their test design and execution on the fly, if the situation and the context require it. This helps to speed up the testing process.

**More defect finding:** ET is essential for finding bugs that are not detected in the course of normal testing. There may be scenarios that

## Managing ET

### INDEX

#### STATICATOR: Indicating Status Made Simple

S. No.	Data	Description
1	Project Dashboard	
2	TestCase Execution Spreadsheet	
3	First Round Defects Spreadsheet	
4	Second Round Defects Spreadsheet	
5	Defects Summary by State	
6	Defects Summary by Severity	
7	Charts and Graphs	
8	Defect Containment Effectiveness and Defect Resolution Effectiveness	

## Managing ET

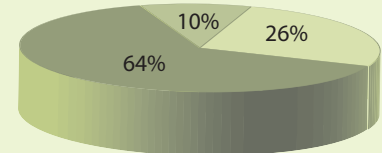
### Project QA Dashboard

Updated: [date] Last Build Tested: Project Build No. [xx], Name of Environment

Areas	Test Effort	Test Coverage (Approx.)	Quality Assessment	QAIT Comments
Login	Low	95%	Fine	
Functionality 1	Low	95%	Fine	
Functionality 2	Start	10%	In Process	
Functionality 3	Start	10%	In Process	
Functionality 4	High	5%	In Process	
Functionality 5	Start	5%	In Process	
Functionality 6	High	5%	In Process	
Functionality 7	Blocked	0%		
Functionality 8	Medium	15%	In Process	
Functionality 9	Low	95%	Fine	
Functionality 10	High	5%	In Process	
Functionality 11	Blocked	0%		
Functionality 12	Medium	40%	In Process	
Functionality 13	Medium	40%	In Process	

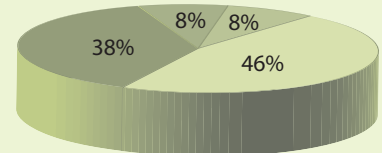
## Test Deliverables

### Defects by Priority



□ High    ■ Medium    □ Low

### Defects by Severity



■ Major    □ Minor    □ Cosmetic  
■ Critical

the scripted tests haven't covered, but which could be covered to detect bugs through ET.

**Improving testers' productivity and creativity:** Through ET, testers learn new testing strategies, gain further skills and experience, and use their intellect and imagination to understand the product and testing requirements. This leads to increases in their productivity and skills for future projects.

**Better understanding of requirements:** ET provides an opportunity for testers to understand the product's requirements and functionality more thoroughly than if they were using any other testing methodology. The level of engagement of the tester with the product under ET is unparalleled.

So, if your software has to be tested on tight timelines, or you have incomplete requirements and documentation, contact us for rigorous, methodical, and in-depth ET. You can also use ET to verify the results of automated testing. ET can help you find defects that are less obvious, but critical, and have escaped detection in other forms of testing. •

For further information, visit [www.qainfotech.com](http://www.qainfotech.com).  
For business queries, please contact Mukesh Sharma at [mukesh@qainfotech.com](mailto:mukesh@qainfotech.com), or call +91-981-077-1714



Top

Most  
Read  
on StickyMinds.com

Don't be the last one in the know.  
Read five of the most popular  
articles on StickyMinds.com from  
recent weeks.

**11 Synthesize Your Test Data**

by Danny R. Fought

[www.StickyMinds.com/10-7Top1](http://www.StickyMinds.com/10-7Top1)**21 May I Take Your  
Temperature?**

by Linda Hayes

[www.StickyMinds.com/10-7Top2](http://www.StickyMinds.com/10-7Top2)**31 Secrets to Automated  
Acceptance Test**

by Jeff Patton

[www.StickyMinds.com/10-7Top3](http://www.StickyMinds.com/10-7Top3)**41 Planning the Endgame**

by Fiona Charles

[www.StickyMinds.com/10-7Top4](http://www.StickyMinds.com/10-7Top4)**51 How Not to Create  
Customer Satisfaction**

by Naomi Karten

[www.StickyMinds.com/10-7Top5](http://www.StickyMinds.com/10-7Top5)**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.

"We have a suite of financial applications that has been developed through a number of years of 'bolting' onto an initial concept. It has been quite successful and has now grown into a much more complex application than when it was first conceived.

As a result, we have new developers on the team and a higher priority on testing. Historically, test documentation has been the poor relation in our development process.

I need to change our team's view on software testing, but don't really know where to start. Can anyone provide a solid framework to follow for documenting the testing process?"

Posted by: **blangley76**

"Anyone have any experiences, either positive or negative, to share regarding open source tools? I'm also exploring what's available for functional testing. I have previously used both WinRunner and SQA, but that was a long time ago!"

Posted by: **cls223**

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

**READ ALL ABOUT IT!**

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

**WEB SEMINARS**  
by StickyMinds.com & Better Software magazine

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.SQE.com/WebSeminars/Archive.aspx](http://www.SQE.com/WebSeminars/Archive.aspx)

## EDITOR'S PICK

### Caught Without a Master Plan

When the power window on the front passenger side of my car stopped operating, it was only three quarters of the way closed. I knew that this was destined to happen since my car is an older model, but I was hoping it wouldn't happen during the local rainy season. First thing I did was manually test both switches. Nothing happened.

**MISTAKE 1:** I based my following actions on the results of a single, simple test. I assumed that the electronic regulator motor was broken, so I bought a replacement and was content with the idea of installing it myself.

**MISTAKE 2:** I checked the service manual for additional information *after* buying the replacement motor. Lo and behold, the service manual listed instructions for a series of extensive tests to help troubleshoot the cause of a non-functioning power window switch and motor. I didn't have the proper tool (a multimeter), so...

**MISTAKE 3:** I skipped the tests. Not such a smart idea!

**MISTAKE 4:** I asked for help after I had wasted time, gas, and money believing the motor was damaged. My cousin and my father, both of whom have electrical engineering knowledge, offered to help troubleshoot the window motor.

They lent me a multimeter and taught me how to use it. We tested every single connection, the ground, and the relays. We even jumped the existing motor to see if it would work—it did! Then we backtracked through the wiring to the fuse box... where we found a burnt fuse. Once we installed a new one, the window automatically rolled up. Talk about an unexpected result! That led us to test other components that could have blown the fuse.

We finally determined that the master switch in the driver's door was broken. (For those of you who are curious to know the exact cause, the master switch for the front passenger window shorted while the signal relay to roll up the window was engaged.)

Besides my biggest mistake—jumping to a false conclusion, I am also guilty of not having planned for extensive testing in order to find the source of the problem. Looking back at this spate of errors, I realized I let the afternoon thunderstorms scare me into relying on a quick fix. In this state of panic, I rushed into action without having a master plan.

Rick Craig's article "Test Strategies and Plans" details how master test plans directly affect the success of all software testing. Yet because we're always strapped for time, we omit drafting a plan and opt to dive right into testing, usually late in the project when time is even more scarce.

Master test plans force us to analyze what should be tested and when. According to Rick, "Discussing issues of what and how to test early in the project lifecycle can save a lot of time, money, and disagreement later."

To read Rick Craig's "Test Strategies and Plans," visit [www.StickyMinds.com/editorspick10-7](http://www.StickyMinds.com/editorspick10-7).



Francesca Matteu  
Editor, StickyMinds.com  
[fmatteu@sqa.com](mailto:fmatteu@sqa.com)

## Quotables

"In 10+ years as a tester and test manager, I looked beyond the details of the product under test to the wider system issues and encouraged my team of testers to do the same. It led to a high degree of respect among the developers. We measured success in terms of customer use and tried to focus on objective measures that reflected customer experience. Along the way we rewarded both team performance (development and test) as well as individual performance. Delicate balance, but needed to reach top performance as an organization."

Ed Weller commenting on Linda Hayes's column, "May I Take Your Temperature?"

[www.stickyminds.com/quotables10-7a](http://www.stickyminds.com/quotables10-7a)

"Sometimes I like to describe security vulnerabilities as if they were people or characters instead of abstract concepts, with a human face on them so they're a little easier for people to understand.

For example, SQL injection is kind of like James Bond: He sneaks into places he's not supposed to have access to and then escapes with all of your secrets. A buffer overflow is like Mr. Creosote from Monty Python's *The Meaning of Life*: He tries to cram way too much data into a space that's not meant to hold it, and the results are disastrous. If Cross-Site Scripting (XSS) were a person, it would definitely be Rodney Dangerfield. XSS never gets any respect!

Bryan Sullivan,

"Show Some Respect to Cross-Site Scripting"

[www.stickyminds.com/quotables10-7b](http://www.stickyminds.com/quotables10-7b)

Nobody

wants to be a

PAGE  
16

drone—well, unless you have

an unhealthy fascination for bagpipes

or worker bees. As programmers, we are *not*

merely engineering drones, mechanically

pushing out reams of boring, ugly code. We

are also artisans. The act of programming

involves as much artistry as it

does technicality.



## Content Pointer

### Recognizing Agile Candidates

By JOHANNA ROTHMAN

Recognizing candidates who are capable of performing well on agile teams doesn't require keyword searches through a stack of résumés. It requires asking candidates questions that allow them to show you they understand the principles and can apply them in their daily work—even if their résumés don't list particular terms. In this StickyMinds.com weekly column, Johanna gives some excellent tips for the interviewer and the interviewee. [www.stickyminds.com/eLetterpick10-7a](http://www.stickyminds.com/eLetterpick10-7a)

## Book Review

### Agile Retrospectives: Making Good Teams Great

By ESTHER DERBY AND DIANA LARSEN

REVIEWED BY: JANET D. KENNEDY

Agile Retrospectives: Making Good Teams Great brings the skills that experienced facilitators use to the realm of the technical team leader. The forward by Ken Schwaber, well known for his work with Scrum methodology, lends credibility to the topic and should be sufficient to hold the attention of most skeptical readers. The book is well organized and informative. Following the instructions, even an inexperienced team leader can have a successful retrospective. Keep reading at [www.stickyminds.com/eLetterpick10-7b](http://www.stickyminds.com/eLetterpick10-7b).

## POWERPASS POINTER

### Magazine Archive: You Can Teach an Old PMO Agile Tricks

By MICHELLE SLIGER

Every manager has a story to tell. Find out how one management professional tackles a fictional dilemma. The story may be made up, but the solutions are tried and true. In this installment, Michele Sliger tells the tale of the movement of a Program Management Office away from waterfall toward agile. [www.StickyMinds.com/eLetterpick10-7c](http://www.StickyMinds.com/eLetterpick10-7c)

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](http://www.stickyminds.com/eLetter.asp) to sign up today.

**iterations:** June 11, 2008 [www.stickyminds.com/eLetter10-7](http://www.stickyminds.com/eLetter10-7)

### The Agile Experience: Three Common Misconceptions about Agile Teams

by Esther Derby

At a recent workshop, two managers had strong reactions to my description of self-organizing agile teams. "You can't just turn people loose and let the team make all the decisions. They'll mess things up," one manager declared.

"With all these ScrumMasters, agile coaches, and self-organizing teams, sounds like I'm out of a job," another said with resignation.

I frequently hear these points of view when I visit organizations that are adopting agile methods. In this article, I'll put these concerns to rest, along with two other misconceptions about agile teams.

#### MISCONCEPTION: SELF-ORGANIZING TEAMS DON'T NEED MANAGERS.

The fact that your company is using agile methods doesn't mean every manager is out of a job. Self-organizing teams need managers, too. But they need different things from their managers than traditional manager-lead teams do.

There's a reason we use the term "self-organizing" rather than "self-organized." That's because it's a process.

Most agile teams start from more traditional, manager-lead teams and begin the journey by organizing and managing their own work. Gradually, the team takes on some of the decisions that have traditionally been management decisions. That can include managing their own team membership, shaping product decisions, and allocating bonuses amongst team members.

But they don't get there without expert coaching from their managers, and they won't get there on day one.

Agile teams need their managers to communicate the organizational context, transfer knowledge and skills, coach, and remove impediments. And I'm pretty sure most agile teams would appreciate it if their managers dealt with the budget and HR matters.

#### MISCONCEPTION: TIMEBOXING FORCES ANY GROUP TO BECOME A TEAM. PUT A GROUP OF PEOPLE TOGETHER, HAND THEM A CHALLENGE, AND THEY'LL JELL.

I wouldn't bet on it. Teams do need a compelling task. They also need the technical skills required by the work and interpersonal skills to form a social system. They need resources such as tools and access to information. And they need coaching to form as a team.

The pressure cooker method of team formation is more likely to burn people out than result in the productivity of a real team. And calling a group a team doesn't make it so.

Timeboxing is one of the structures that can help teams succeed, and teams still need attention to the interpersonal skills that enable them to work collaboratively and build trust.

As a manager, you can provide resources and coaching to help people develop the skills they need.

#### MISCONCEPTION: TEAMS GROW LESS PRODUCTIVE OVER TIME, SO THEY SHOULD BE SHAKEN UP EVERY FEW MONTHS.

Team members need time to develop the social glue that enables high performance. They understand each other's strengths and weaknesses, develop shared knowledge, and discover how to learn together.

When new people are constantly arriving and leaving, a group may never develop the shared approaches and shared knowledge that permit teams to outperform a group of individuals.

Some teams—when they've had time to form and create a strong team culture—do become adept at adding new members. Even then, it's best to limit the number of new members added at any given time. Changing more than 30 percent of team membership causes a reboot of the team. Constant turnover prevents a team from truly forming.

As a manager, you can help by keeping teams together long enough to jell and by protecting teams from the revolving door syndrome. In the end, agile teams require the same coaching and attention as other highly productive teams. And as a manager, you are still needed.

## BOOK REVIEW

**The Art of Software Security Assessment****By:** Mark Dowd, John McDonald, and Justin Schuh**Reviewed By:** J.D. Kennedy

This book is written by three experts in the field who have shared the depth of their security knowledge and experience in a readable and instructive format. The basic premise of the book is that while it is desirable to avoid security vulnerabilities, this isn't always possible when you have software already in production. The experts start with the basics of software security assessment and make recommendations on how to "shortcut" the process when time or resources are limited. While the material is aimed at assessing the security of software, it also provides lessons in what to avoid when developing software.

The writing style is easy to read considering that some might not find the topic riveting. This book is suitable for novices developing a knowledge-based foundation on these issues and for experienced developers who need a reference to remind them of things to look for.

The book starts by covering the fundamentals of software vulnerabilities and then covers basic review methodologies. It goes into more depth by exploring specific vulnerabilities of both UNIX- and Windows-based systems. The final section deals with software vulnerability in practice. In the "How to Use This Book" section, the authors recommend that the book be read straight through "at least once" to get a feel for the material. Luckily they also recognize that this may not be feasible for readers who wish to put the principles into practice, so they organized the book so that specific topics and techniques are easy to find and use without having to read the entire book.

I liked the simple yet powerful examples used and how the experts build on them to create more complex issues. The concept of userids and passwords is fundamental to most computer systems; the discussion of how either may be easily compromised by inappropriate communication responses should hit home with any developer. Regardless of how obvious this subject matter is to even moderately experienced programmers, basic errors are repeated time and again by novices. I will be taking this book off the shelf and handing it to developers on my team whenever they are working on a new application.

Visit [www.StickyMinds.com/bythebook10-7](http://www.StickyMinds.com/bythebook10-7) 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—make the StickyMinds.com Books Guide 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

SEPTEMBER 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 Jeff Patton, Linda Hayes, Bryan Sullivan, Naomi Karten, and Payson Hall.



## 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:



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



\*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**



*Shingle Creek Resort*

**Register Early and Save \$200!**  
**[www.sqe.com/agiledevpractices](http://www.sqe.com/agiledevpractices)**

# 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=60rfgt4Mvt9diVoFZzhX7Wriqn1T2n2489Sx"
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:

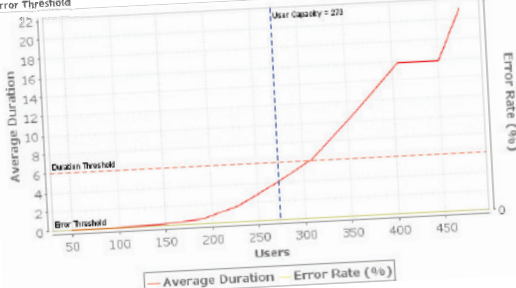
[www.webperformanceinc.com](http://www.webperformanceinc.com)

## Test Report: 9/12/06 1:52 PM Acceptance Test

### User Capacity

The User Capacity section shows the measured user capacity, or how many users the web-based application could take while still meeting performance requirements. How this measurement was arrived at is demonstrated in the chart, which shows the duration and error thresholds. The place at which the measured duration and error count intersect the thresholds is the measured capacity.

Estimated User Capacity 273  
Settings 6 sec  
Duration Threshold 95%  
Duration Compliance 0%  
Error Threshold



Users	Thresholds Satisfied	Error Rate	Maximum Duration (ms)	Duration Compliance
90	yes	0.00%	00:00.148	100.00%
95	yes	0.00%	00:00.140	100.00%
143	yes	0.00%	00:00.174	100.00%
190	yes	0.00%	00:00.589	100.00%
229	yes	0.00%	00:01.721	100.00%



## 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](http://www.webperformanceinc.com)



# Is “Agile” Distracting You?

by Jonathan Kohl

The term “agile” has become popular, and profitable. Early adopters of “agile” became in demand. Other groups jumped on the bandwagon, learning techniques such as Extreme Programming or Scrum, and adopting tools such as continuous integration, test-driven development, and refactoring. Still others wanted to mimic the success but didn’t want to change the product, process, tool, or service they were offering. Instead, they discovered that a little “agile” white-washing goes a long way. It’s easy to prefix whatever tool, process, or service you’re selling with the term “agile.” The consumer is distracted from the real offering; it contains the right buzzword, it must be good. Consumers buy both the buzzword *and* the product or service.

The term “agile” has become abused and, since we don’t have a standard dictionary definition, it is open to interpretation. (Agilists will tell you that being “Agile” is not the same as being “agile” in the dictionary sense.) That isn’t to say that the Agile Manifesto that brought us the term “agile software development” was a bad thing. It wasn’t. It was a wonderful reminder to return to a human-centered approach (we value people over process) and a product-centered approach (we believe in working software) that many had lost along the way. Just because something contains the word “agile” does not mean that the practitioners are not sincere, skilled, and working hard to create value for their customers and their teams. However, even when we have the best intentions, an ideal can distract us from our real goals.

“Agile” can distract from your message. I once started writing a book on agile testing and got stuck early on. My wife told me to look at Strunk and White’s *The Elements of Style*, specifically the chapter on “Omit needless words.” I realized my wife meant “agile” was a needless word. If I removed the agile prefix, I was left with content that was strong enough to stand on its own. It

was also much easier just to write about exploratory testing or test planning without having to fit the ideas into an “agile” slant. I was really talking about applying testing ideas on agile projects, or being an agile-fluent tester. Many of the test ideas we’ve developed over the years plug right in to agile environments, or any other iterative, incremental lifecycle with just a bit of adaptation. It was a hard realization, but my writing (and particularly my thinking) profited from it. Now I just focus on testing in any context, and I pride myself on being able to test in all sorts of process environments, and my work is more meaningful.

Agile can be a distraction from your mission to deliver software that your customers value, while supporting your team. I’ve seen far too many successful agile process implementations produce software that customers aren’t interested in. I’ve also witnessed agile bullies deliver working software but grind up teams through their zealotry, bigotry, and elitism.

Agile can distract from your skill development. We can talk about agile ideals, but someone needs to code, test, document, market, and sell the software. One of my friends mentioned that for the past few years he was so worried about following agile practices that his programming skills suffered. He said he learned more in several months spending his time reviewing and practicing the programming techniques in *Structure and Interpretation of Computer Programs* [1] than he had in several years of trying to follow agile practices.

Take a moment to ask yourself these questions: *To what extent are agile practices helping you create value in your product and within your team? To what*

“I’ve seen far too many successful agile process implementations produce software that customers aren’t interested in.”

*extent are they distracting you away from reaching your goals?*

I’m not against using the term “agile.” However, since it has come to mean whatever the speaker wants it to, it has lost significance. Rather, tell us your project stories. Let’s talk about what’s really going on—what practices you are using, how they work for you, what you have tried, and what you have learned—agile

or not. Tell us what your team learned when implementing the twelve practices of Extreme Programming, or how you successfully blended Scrum with existing practices on your project. I’d also like to hear about practices that don’t get a lot of air time in agile circles: What are Cleanroom or RAD teams doing? How are they delivering successful software? What lessons can we learn from successful “waterfall” projects? What about that process that has no name but is working wonders with your team?

The Agile Manifesto was a welcome development in an industry that seemed mired in paperwork and process. Unfortunately, “agile” can be carried too far. I propose our rallying cry be not Agile, but *Value*—both to our customers and our teams. If we are delivering what our customers need, and we are building up our teams and those we interact with, does it matter what it is called? Let’s stop worrying about whether what we do is “agile” or not, and go back to calling it software development. Let’s worry about how we can do *that* to the best of our ability. {end}

## REFERENCES:

1] Abelson, Harold and Sussman, Gerald Jay. *Structure and Interpretation of Computer Programs*. The MIT Press, 1996.

# Idioms and Idiosyncrasies

by Pete Goodliffe

Nobody wants to be a drone—well, unless you have an unhealthy fascination for bagpipes or worker bees. As programmers, we are *not* merely engineering drones, mechanically pushing out reams of boring, ugly code. We are also artisans. The act of programming involves as much artistry as it does technicality.

*Code is art. Kinda.* When we craft great software, we naturally consider, perhaps subconsciously, the beauty, the elegance, and the balance of our handiwork as much as the technical correctness. We aim to create order out of chaos; we derive satisfaction from solving complex problems with designs that look so simple they cannot possibly be wrong. And we ensure that our code expresses this clearly. Or at least, we should. In every good programmer there is an internal quality meter that automatically sounds an alarm when you see code like listing 1a and that quiets down when confronted by code like listing 1b.

Stop for a moment and consider the number of ways that listing 1a offends you. Count them all. How is listing 1b better? Is it any better? And what language is each one written in, anyway?

```
bool ouch()
{
    if (do_something() == FAILED)
        goto fail_1;
    if (do_something_else() != OK)
        goto fail_2;

    return true;
    fail_2:
        tidy_up_second_thing();
fail_1:
        tidy_up_first_thing();
    return false;
}
```

**Listing 1a: Ouch!**

```
bool better() {
    try
    {
        do_something();
        do_something_else();
    }
    catch (...) { return false; }
}
```

**Listing 1b**



ISTOCKPHOTO

When we're staring at code, we like to see clear structure and code patterns with which we're familiar. Certain code patterns are offensive—they naturally cause us to sit up, take note, and suggest extreme caution. The mere sight of a `goto` invokes the gag reflex. When we see messy and inconsistent layout, we feel bile rising, global variables cause an allergic reaction, and in the face of illogical and unmalleable structure, we're gripped with the urge to run away quickly.

## Beauty == Idiom?

Our sense of “beauty” is often shaped by familiarity—by the prevalent *idioms* of the implementation domain. What constitutes natural and beautiful code differs from language to

```
int list[] = { 1, 2, 3, 5, 8 };
for (int n = 2; n < 4; n++)
{
    do_something(list[n]);    // Process 3 and 5
}
```

**Listing 2a: Idiomatic C code**

```
list = [1, 2, 3, 5, 8]
for element in list[2:4]:    // Process 3 and 5
    do_something(element)
```

**Listing 2b: Equivalent idiomatic Python code**

```
list = [1, 2, 3, 5, 8]
n = 2
while i < 4:
    do_something(list[n])
    n += 1
```

**Listing 2c: Equivalent *non*-idiomatic Python code**



language. Idiomatic C code is quite a different beast from idiomatic Python. Listings 2a, 2b, and 2c illustrate this. The Python code in listing 2b is idiomatic, but it could have been written like listing 2c, which is a more direct translation of the original C code. But it's *not* idiomatic Python, and it doesn't look or feel "right" as a consequence.

Non-idiomatic code sets our internal alarm bells ringing, and rightly so. Idioms don't just look nice, they help us write safe, correct code, avoiding the subtle pitfalls in the language. As in all heuristics, there is a body of collected wisdom contained in our programming language idioms that is perilous to ignore. Learning the specific idioms of a language is a rite of passage and marks one's mastery of the language.

But important as they are, idioms are not sacred. Nor are idioms fixed and immutable. Fashion doesn't stand still; over time tastes change. Some classic programming idioms have become dated. Hungarian Notation used to be a conventional, safe, and well-regarded practice. These days it is not merely passé but socially unacceptable—the modern equivalent of software leprosy.

## Idiom Idiocy

Sometimes the desire for beautiful, idiomatic code can trip us up. Well-intentioned use of idioms can bite you. Here's a cautionary tale involving C++, which has particularly amusing idioms.

Recall the oft-cited Perl mantra, "There's more than one way to do it." Well, C++ is like that for idioms—there's always more than one idiom for anything in C++, and you can bet that each has zealots who fervently believe that theirs is the Only Right Way To Do It.

One of the most basic, contentious, C++ idioms is the naming member variables. Many of these idioms involve the subtle incursion of Hungarian Notation.

Many C++ programmers prefix member variables with an underscore. However, this is dubious practice as the language standard reserves many identifier names beginning with an underscore. Class member variables are not actually one of the reserved cases, but this convention sails dangerously close to the wind. Also common is the "m\_" prefix, where "m" stands for member. I've even seen the disgustingly cute "m\_member". Euch!

So what do the C++ experts do? Herb Sutter advocates a trailing underscore on member variable names. I have a personal dislike for this approach as the variable names read *very* strangely.

Scott Meyers is the sensible advocate of minimalism—he writes the variable name, the whole name, and nothing but the name. To my mind, this approach makes sense. If you need any extra indication of "member-ness," you probably have code that is too hard to read—your function has too many parameters or your class is too large. Fix that problem; don't mask it with silly variable names.

```
class Foo
{
private:
    int thing;

public:
    // How would you name the ctor parameter? (Choose one)
    Foo(int thing_in) : thing(thing_in) {} // (1)
    Foo(int t) : thing(t) {} // (2)
    Foo(int thing) : thing(thing) {} // (3)
};
```

Listing 3

Why does this simple naming issue matter? Well, it shouldn't, until we combine the Meyers Minimal Member Moniker Mechanism with another idiom. When constructing a C++ class, you can initialize members in an initialization list. There's another naming minefield here. Three of the options are enumerated in listing 3.

They are only subtly different, are functionally equivalent, and each seems perfectly adequate. They are all common in modern C++ code. My workplace tends to adhere to idiom 3; it's nicely symmetric and doesn't introduce another unnecessary name into the code. Great.

Well, not quite. Idiom 3 has a hidden sting in its tail. Sure enough, in listing 3 it works just as advertised. But consider what happens when you need some slightly more complex constructor logic, like listing 4.

```
Foo::Foo(int thing) : thing(thing)
{
    if (thing == 1)
        thing = 2;
}
```

Listing 4

What is the value of the `thing` member when a `Foo` is constructed with the value 1? It's 2, right? Well, no it isn't. It's 1. "But how can that be?" I hear you ask. The name `thing` inside the constructor is bound to the *constructor parameter*, not to the class' member variable. If you wanted to assign the member, you must write:

```
this->thing = 2;
```

Otherwise, you're just writing to a temporary variable that will shortly be thrown away. This is a subtle but nasty way to introduce obscure bugs into your codebase. So there you have a collision of idioms. Some idioms are bad for you!


How could you alleviate this problem? There are many ways. For example, you could make the `thing` parameter `const` in the constructor implementation. But for built-in types passed by value, this isn't idiomatic! How else could you avoid it?

## What Can We Learn?

So what do we learn from this sordid tale? It's important to consider the idioms of the language you're working in and to gauge the beauty and quality of code against the familiar idioms to which it should adhere.

Common language idioms help to show the elegance, beauty, and artistry of a piece of code. They help you write code that seems familiar and easy to work with, and they (usually) help you avoid simple bugs. You can gauge your mastery of a language by how well you know its idioms.

But don't blindly trust idioms. Idioms can be flawed. Always use your brain. Of course, if this seems like too much work, perhaps you should give up and produce boring ugly code. Or learn to play bagpipes, instead. **{end}**



**Do you always follow the idioms of your programming language? Have you ever stepped outside of the lines? Why?**

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

**SALARY SURVEY**  
Better Software magazine  
StickyMinds.com  
2008

Calling All Software Developers, Testers, and Managers! It's time to stand up and be counted.

The 2008 *Better Software* magazine/StickyMinds.com Salary Survey is in full swing.

Don't miss your chance to contribute to the collective knowledge of industry salaries and employment trends.

Follow the link that best describes your employment level, answer a few questions, and enter to win a \$50 Best Buy gift certificate!

**STAFF LEVEL:**  
[www.stickyminds.com/2008salarysurveystaff](http://www.stickyminds.com/2008salarysurveystaff)

**MANAGEMENT LEVEL:**  
[www.stickyminds.com/2008salarysurveymanagement](http://www.stickyminds.com/2008salarysurveymanagement)

**DIRECTOR LEVEL:**  
[www.stickyminds.com/2008salarysurveydirector](http://www.stickyminds.com/2008salarysurveydirector)



**STOP THE FREAK,  
KILL THE CREEP,  
BRING ORDER TO YOUR  
ALM TECHNIQUE**

**THE COMPLETE ALM  
SOLUTION ON TIME, ON  
BUDGET, ON THE MARK**

Without oversight, software projects can creep out of control and cause teams to freak. But with Software Planner, projects stay on course. Track project plans, requirements, test cases, and defects via the web. Share documents, hold discussions, and sync with MS Outlook®. Visit [SoftwarePlanner.com](http://SoftwarePlanner.com) for a 2-week trial.



[www.softwareplanner.com](http://www.softwareplanner.com)





# Hear that Sound?

**StickyMinds.com** has Podcasts!



As a StickyMinds.com member, you can tap into the newest opportunity your membership has to offer—Podcasts! StickyMinds.com podcasts offer expert interviews to help you build better software.

*Check out all three of the StickyMinds.com podcast series!*



**Gray Matters** - featuring interviews with software development experts and other audio to feed the stuff between your ears.



**StickyMinds SoundBytes** - the audio companion to *Better Software* magazine and StickyMinds.com previewing upcoming *Better Software* magazine issues and interviews with StickyMinds.com columnists!



**Conferences** - is a podcast series featuring content about and from Software Quality Engineering's four conferences: STAREAST, STARWEST, the Better Software Conference & EXPO, and the Agile Development Practices Conference.

Take a listen today at [www.StickyMinds.com/Podcasts](http://www.StickyMinds.com/Podcasts)

# It's in the Way that You Use It

by Michael Bolton

People sometimes ask me what my favorite test tool is, and my answer often confuses them. They seem to expect me to name a program that automates the operation of an application, makes some process more repeatable, or allows a machine to drive as quickly as possible through an application. But if you think driving is merely about getting from one point to another, ask a New York cabbie.

Rapid testers don't think of test automation merely as something that controls a program and checks for some expected result. Instead, we think of test automation as *any use of tools to support testing*. So my favorite test tool changes from moment to moment, but it's often (drum roll) my text editor. For years, I've used an editor called TextPad (see the Sticky Notes for a link). I'm not claiming that it's the best editor available, but it's *my* favorite. There are plenty of other text editors, and whichever one you use, if you're happy with it, I am, too. So what do I use a text editor for?

**Writing.** A lot of my work as a tester involves creating and editing text—taking notes, editing text, creating Fitness tables, building Web pages, or writing articles like this one. With my text editor, I can do these things efficiently without extra overhead like the tables, formatting and layout tools, styles, and thesauruses in sophisticated word processors. These extra features sometimes distract me or entice me to use them needlessly. As Marshall McLuhan said, “We shape our tools, and thereafter our tools shape us [1].” I sometimes use a text editor to emphasize to myself that I'm working on something temporary or preliminary. That's not to say that a text file is never important; a commercial software company that I worked for used to track bugs and feature suggestions by using a plain old ASCII text document. For us, the cost and overhead of an elaborate test management tool wasn't worth it



ISTOCKPHOTO

when ideas could be captured concisely, compactly, and sufficiently in text.

**Programming.** Good text editors support programming by providing syntax highlighting, which recognizes the programming language that I'm using and changes the color or font of keywords, symbols, and variables to make them more distinguishable. This improves readability and provides some cognitive hints to alert me when I'm making a mistake. More sophisticated programming environments automatically add closing parentheses, braces, or brackets when I type the opening symbol; provide context-sensitive help for the languages and libraries that I'm using; and offer very sophisticated ways to map and model larger programs. When I'm working on a large project, I find these features essential, but for a quick-and-dirty throw-away script, a good text editor is just fine.

**Cutting and pasting.** Unlike Notepad, TextPad can mark blocks of text in columns, which allows me to open a file that's organized as a table, grab specific blocks in columns, and manipulate them or copy them to another program. This is often handy in combination with command line redirection to supply input from a file or output from a program to a text file. In Windows, it's also possible to extend the virtual size of the com-

mand window, copy output from programs, and paste it into a text editor for processing.

**Data generation.** I often use a testing technique called an input constraint attack—hurling huge amounts of text at an input field to see whether the program protects itself from buffer overflows that could compromise security or stability. To generate data quickly, I type ten characters and then hit Ctrl-A (select all), Ctrl-C (copy to the clipboard), and then Ctrl-V ten times (paste ten copies). Now I have a file of one hundred characters. Then I hit Ctrl-A, Ctrl-C, and Ctrl V ten times again; now I have a thousand characters. Repeat that cycle a few times with a final Ctrl-A, Ctrl-C, and Ctrl-V, and I have a million or ten million characters on the clipboard, suitable for pasting into an input field, an application, or a Web page. Try it! The results are often fascinating—data corruption, strange behavior, or a crash. Danny Faught and James Bach's PerlClip tool [2] has more features for this task, but a text editor will do many useful things in a pinch.

**Searching and replacing.** Sometimes I'm looking for words, strings of words, or patterns of characters inside log files, Web pages, XML pages, comma-delimited files, or binary files. Sometimes I want to replace strings of text with something else, or delete markup



so that I can focus on the text. To assist with these tasks, better text editors include support for regular expressions—powerful forms of notation for seeking, matching, extracting, or replacing patterns in text. See Jeff Friedl's *Mastering Regular Expressions* [3].

**Learning.** The Perl, Python, and Ruby scripting languages provide powerful support for regular expressions, but it can be hard to appreciate their value until you've learned to use them well enough to perform your first miracle. More complex regular expressions can be hard to learn without visual feedback and recoverability. Using the text editor to step through pattern matching and substitution, seeing changes in the text in real time, and being able to press the Undo key helped to accelerate my learning.

**Data inspection.** When I need to learn something about a file—its content or format—I often start investigating by opening it in a text editor. I might see a header that gives me clues, data in clear text, or repeated patterns of strings or blanks. I often scroll through a log file very quickly, using a defocused perspective to identify patterns of output or dramatic differences from the norm. (We call this “blink testing” or using a “blink oracle.” See the StickyNotes for more information.) I can also use the editor to delete or change characters as a quick way to modify a clean data file to see how a program handles file corruption.

My colleagues and I sometimes use our editors for things that their publishers might not have expected. When I try to pour a lot of data into an input field, and the input field truncates the excess, I copy the field and paste it into the editor to get a word or character count right away. I often use text files as a backup mechanism when I'm entering text into a Web page. Jon Bach says that he uses text files as longer-term, safer storage for the clipboard. We both use text editors to peel the formatting off rich text when we're pasting large volumes of text to disjointed cells in an Excel file. James Bach once programmed his text editor to remind him to take testing notes every few minutes. When I forget the syntax for a tag or a special

character in HTML, or the character associated with a particular ASCII code, my text editor is the handiest and fastest reference.

There are several points to all this. First, the simpler the tool, the more flexible, adaptable, and useful it might be. Think of the difference between a \$50 Swiss Army knife and a \$15,000 computer-controlled table saw. Second, like “quality,” “purpose” is not something inherent in a product. Purpose is a relationship [4] between the product and the person who is putting it to some use. Third, some of those purposes are intended by the designers and adopted by the users, but novel and surprising purposes may emerge as we use a product—or as we test it.

As testers, we're usually asked to defend the value of a product by finding important problems with respect to its intended purpose. However, if we think of our products as tools that different people will use in different ways, we may *add* value by discovering purposes that haven't yet been recognized. For example, if we think of a test tool as “any tool that supports testing,” even the humble text editor might become a powerful test tool. {end}

#### REFERENCES:

- 1] McLuhan, Marshall. *Understanding Media*. The MIT Press, 1994.
- 2] [www.satisfice.com/tools/perlclip.zip](http://www.satisfice.com/tools/perlclip.zip)
- 3] Friedl, Jeffrey. *Mastering Regular Expressions*. O'Reilly Media, Inc., 2006.
- 4] Weinberg, Gerald M. *Introduction to General Systems Thinking*. Dorset House Publishing Company, Inc., 2001.



**What tools do you use that you or others might not have recognized as a test tool?**

Follow the link on the [StickyMinds.com](http://StickyMinds.com) homepage to join the conversation.

#### Sticky Notes

For more on the following topics go to [www.StickyMinds.com/bettersoftware](http://www.StickyMinds.com/bettersoftware).

- TextPad
- Blink testing



## ARE YOU IN SEARCH OF QUALITY?

## SO ARE WE.

Quality Assurance Analysts

Quality Assurance  
Software Engineers

Apply online at  
[www.blackbaud.com/QAcareers](http://www.blackbaud.com/QAcareers)  
or contact Stephanie McDonald,  
senior technical recruiter, at  
**843.654.3547** or  
[stephanie.mcdonald@blackbaud.com](mailto: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®**

## Exit, Stage Left

by Patrick M. Bailey

"I don't think we want him," Charlene said in her typical methodical manner.

The hung jury groaned. It was the eight-person development team's sixth meeting to pick a new business analyst from the final list. All agreed that Stan was the man, except Charlene, a fast-rising technical star at FastTech and already a team lead at twenty-four.

"Charlene," said Eric, the team architect, "we may have to go with a majority vote. We have to move forward."

Sarah, the development manager for the division that included this team of twenty-somethings, was also in the meeting. Although a believer in empowerment for her team leads, she also believed this stalemate was becoming counterproductive. "Charlene," Sarah said, "I think you should explain your reservations about Stan."

"OK. I know we're in a hurry to get a business analyst so we can focus more on technical issues, and talks with the customer about new marketing processes begin in two weeks. We want someone with so-called emotional intelligence, but selecting Stan is going too far."

"Why?" Eric asked.

"He has no technical experience outside of school," Charlene replied.

"We know," Eric countered. "All of us talked to him. We realize his background doesn't hint much at software development, but going back to school in his mid-forties to 'retool' himself shows motivation to the rest of us. Also, he's engaging and fun to talk to."

"Yes, he's charismatic, he's interesting, he's charming," Charlene said. "But he's ... he's ... he's ..."

"He's what?" Eric asked.

"He's an actor!"

"You've got something against actors?"

"Well, yeah!"

"Like what?"

"Do you actually think customers will take seriously someone from off, off, off Broadway?"



ISTOCKPHOTO

"Wait," Eric said. "You're ignoring some things. First, his interview with the business unit produced positive feedback. He has three degrees: theater, philosophy ..."

"Oh, great! Someone who wants to know the meaning of life during implementation."

"And, he just finished a degree in information systems."

"That's right. *Just* finished—no experience. We need someone to jump in quickly. And another thing—he's in his late forties. A little late to start a new field, don't you think? What about the other candidates? All of them have considerable coding experience."

"Charlene, we've been through this. No one doubts the candidates' technical prowess, but their visits with the business unit weren't exactly stunning. They certainly could talk geek with the best of us, but we're not the customers, either."

Sarah interrupted their exchange.

"Charlene, I wish we could convince you, but we need to move forward. We'll go with Stan."

While Sarah's and Charlene's eyes were fixed on each other, a unified sigh of relief filled the conference room. As the team departed, Charlene remained for a moment, thinking this was only the end of the first act.

Two weeks later, Stan waited in Charlene's cubicle. He'd just finished the standard orientation and had been wel-

comed by Sarah and the rest of the team. When Charlene entered, Stan stood up to greet her.

"Nice to see you again, Charlene."

Stan's warmth was greeted with Charlene's machine-like response: "Well, you're hired. So, let's get started. I emailed you the background information on the marketing project. Any questions?"

Stan respected this energetic efficiency. "Not yet, I'll look it over. Are you going to the first meeting with marketing today?"

"For a while," Charlene said. "But it's up to you to get things going to generate the first set of use cases, you've heard about use cases, right?"

"Yes, but I thought it was mostly introductions today."

"Well, Stan, if we're going to do iterative development—you do know what that is, don't you?—we need to start an iteration." Charlene ended the discussion and said they would meet later.

Within a short time, Stan established a relationship with the marketing department and soon created a regular dialogue between the developers and the business unit. This was an important step forward for FastTech. Stan's success was evident to Sarah, Eric, other developers, and the customer, but a never-ending tension existed with Charlene.

Charlene's aloofness kept her from noticing anything was wrong until Sarah called a short staff meeting.



## STORY LINES

- **Don't underestimate the value of a liberal arts education and other general life experiences in the technology field.**
- **Age discrimination can happen without realizing it.**
- **Consensus building does not mean all decisions must be unanimous.**
- **Highly talented technicians are very likely to become team leads early. These very talented people must have a seasoned manager to provide guidance as well as give them room to grow.**

With Charlene sitting next to her, Sarah said, "This is the three-month mark, and it's time for Stan's first performance review by the team."

"Should be a slam dunk," Eric said.

"Really?" Sarah said. "Why?"

"Well, for one, he writes incredible use cases. He captures the right amount of detail. Amazingly, what took several iterations before seems to happen the first time with him. How did he describe the process?"

"He calls it the magic 'IF,'" said Katy, another developer.

"What's the magic IF?" Sarah asked.

"It's an acting technique," Eric said. "What would I do IF I were the character in this given situation?" Stan uses that technique when working through use cases with the customers. He gets them to act out their roles. There's also the concept of a French scene—a significant change in the action. And the other cool thing is his mastery of business rules. He explained how things like the law of the excluded middle, the mood of a syllogism, and the types of fallacies helped him.

"Really?" Sarah said. "That's too bad, because Stan just gave me his notice this morning. I wanted to talk about transitioning his work. Given what you told me, I'll talk to him again."

As the stunned team left, Sarah placed her hand on Charlene's, who understood this meant to stay. Charlene was staring at the table, unable to look directly at Sarah.

"Charlene, we need to talk." {end}



**What non-technical skills or past experiences have benefitted you as a software developer?**

Follow the link on the [StickyMinds.com](http://StickyMinds.com) homepage to join the conversation.

### Sticky Notes

For more on the following topic go to [www.StickyMinds.com/bettersoftware](http://www.StickyMinds.com/bettersoftware).

■ Further reading

S U C C E E D I N G   W I T H   A G I L E <sup>SM</sup>

## DON'T BE SHY about transitioning to agile.

**Face and overcome the challenges. Master agile software development techniques and deliver valuable, functional software in a world of uncertainty and change.**

Join **Mike Cohn**, author of *User Stories Applied* and *Agile Estimating and Planning*, in courses exploring the principles of agile software development: people over process, working software over documentation, collaboration over negotiation, and flexibility over rigidity.

To learn more, visit  
[www.mountaingoatsoftware.com/better](http://www.mountaingoatsoftware.com/better)

### Upcoming Courses

#### San Jose

**October 13**

Effective User Stories  
for Agile Requirements

**October 14-15**

Certified ScrumMaster

**October 16**

Agile Estimating and Planning

#### Dallas

**January 27-28**

Certified ScrumMaster

**January 29**

Agile Estimating and Planning

#### Boulder

**February 18-19**

Certified Scrum Product Owner\*

\*with Ken Schwaber





# FROM HERE TO AG TEST-DRIVEN DEVELOPMENT

9 Landmarks  
to Help You Find Your  
BY ANTONY MARCANO

Make  
sure you turn right at  
the orange bank, not left  
at the blue one.

Go past  
the pink house.





**A** **acceptance** test-driven development (ATDD) means many things to many people—from “It’s all about testing” to “It has nothing to do with testing,” and from “TDD, ATDD—it’s all the same” to “TDD and ATDD are nothing alike.” Each of these statements describes ATDD from a single perspective, but neither encapsulates the entire meaning. This is often because different people notice different aspects of ATDD depending on where they are coming from.

It’s like asking a friend for directions to his favorite bar, but he can’t recall any of the road names so he gives you landmarks, instead (“Turn left at the clock-tower”). But ask another friend for directions to the same place, and you’ll get a totally different set of landmarks. It would be no surprise if you got lost along the way.

To explain ATDD and its benefits, I’ve retraced my own steps and also the steps of others to give you the most common landmarks, no matter where you’re coming from.

### Where Are We Headed?

If you aren’t familiar with ATDD, it picks up where user stories leave off. A user story—an artifact resulting from a related practice—is a brief summary of an end-to-end system capability. I say summary because that’s all it is; it is not a specification or a definition. I emphasize end-to-end because one of the most common mistakes is to decompose a problem into “horizontal” user stories (stories that are split by software component) when instead user stories should be written in terms of a “vertical” end-to-end capability.

ATDD bridges the gap between an idea summarized in a user story and the implementation of that story. It helps elicit the detail behind the idea to make it better understood by both the customer with the idea and the cross-functional development team that will implement it. This process begins when the first acceptance criteria for a user story is outlined. For new teams, acceptance criteria are discussed before iteration planning. For a more practiced team, acceptance criteria might be discussed during iteration planning. A highly experienced team on a well-established project may find it can still be successful when deferring such discussions until the iteration itself.

Whenever we begin to discuss acceptance criteria, it's safe to say the user story alone is not enough. To establish a shared understanding of a user story, the customer elaborates on the idea behind it by first outlining some acceptance criteria—often in terms of the new behaviors that the system will exhibit once the story is implemented. This is so that the team has a high-level view of the story's real meaning. Programmers, testers, and the customer exchange examples of specific inputs and outputs that further clarify the scope of the user story. This results in refinements to existing acceptance criteria or additional criteria.

During the iteration, programmers and testers work with the customer to draw out the detail required for them to expand outlined acceptance criteria into specific examples in the form of tests. From these tests, programmers infer the underlying rules that must be implemented. During this process, with the help of a variety of testing techniques, testers help to identify exception cases and reduce other example behaviors into equivalent cases. The result is a set of examples, expressed as automated acceptance tests. These tests are run and, since the feature isn't implemented yet, they fail—as expected.

For each of these automated acceptance tests, programmers implement enough of the software for the acceptance test to pass (ideally using TDD with unit tests). Once all the acceptance tests for a story pass, the story is done—almost. Often, exploratory testing is performed around this story to detect cases that the team didn't anticipate. This feedback can result in more tests for the story or inspire whole new stories.

So, that's a high-altitude image of where you'll end up. However, becoming proficient in ATDD isn't just a case of following a process and producing some artifacts. It requires a certain appreciation of the intent, value, and benefits of the practice. Many of us learned by doing, taking the journey to appreciating these factors with little or no guidance along the way. Fortunately, you can take a slightly easier path with the help of these nine key landmarks.

## Landmark 1: Understanding the Origin of ATDD

In the late 1990s, Extreme Program-

ming (XP) emerged and, along with other agile processes, has grown in popularity. Inspired by 1950s programming practices, Kent Beck proposed writing tests before writing the code—something in opposition to the common wisdom of the time. This idea became a core part of XP at two distinct levels—acceptance tests and unit tests.

Acceptance or customer tests are written by customers (with appropriate help) to explore and communicate a problem and the behaviors that would solve that problem. These tests are then automated by the cross-functional development team. In contrast, unit tests are written by programmers to explore and shape the software design and its implementation. As the system grows and evolves, thanks to both types of tests being automated, the team has confidence that the software continues to do all the things the team built it to do.

Perhaps because many programmers misunderstood TDD, Beck wrote the book *Test-Driven Development: By Example* [1], which leads you through a series of worked examples in how to drive your design with unit tests. From this point on, TDD became for many a specific reference to the unit-test level of the practice.

However, in the section “Mastering TDD,” Beck briefly discusses the issues with TDD as applied to unit tests and how application-level tests solve them. He calls this approach application test-driven development (now more commonly called acceptance test-driven development):

The problem with driving development with small-scale tests (I call them “unit tests,” but they don't match the accepted definition of unit tests very well) is that you run the risk of implementing what you think users want, but having it turn out not to be what they wanted at all. What if we wrote the tests at the level of the application? Then the users (with help) could write tests themselves for exactly what they wanted the system to do next.

Beck goes on to warn of the problems ATDD presents—the problem of writing application-level tests before the applica-

tion exists and the social and technical difficulties of getting users to write these tests. Today, I believe the practice has matured sufficiently that these problems have been solved—more or less.

## Landmark 2: Specification by Example

The word “test” is so prominent that many assume that ATDD (and TDD) are testing practices and ignore another key dimension of ATDD, “specification by example.”

To explain specification by example, I need you to forget the phrase acceptance “test-driven development” for a moment. Try to put the word “test” completely out of your mind. In fact, think about something else for a moment. Some random idea, say ... Internet radio.

Internet radio has come a long way since its inception. Now, services like Pandora.com and last.fm create for each user a personalized radio station that matches that user's own personal tastes. With last.fm you can provide one example of a song you like, or you can provide a wealth of examples by uploading your own music library to the site. Software analyzes the songs in your library, discovering common aspects of your songs' “DNA”—genre, tempo, and a number of other characteristics. From this, last.fm creates a user-specific Internet radio station that matches your tastes.

As you listen, you give last.fm feedback saying which songs you love and which ones you hate. Your future playlists are refined with each piece of feedback you provide. The feedback itself is, essentially, more examples of songs that do and don't match your taste. From this, last.fm evolves your playlist, introducing you to more and more new songs in the process, even evolving your own understanding of your musical taste.

This is an automated equivalent of specification by example, and this is what we are trying to do with ATDD. Instead of examples of songs, the customer provides examples of software behavior. These examples are represented as acceptance tests. The cross-functional team infers the underlying business rules from these examples, and since the tests are automated, the team can quickly determine when these rules have been implemented.



I first heard of “specification by example,” also known as “inductive inference of programs from input-output examples,” from Brian Marick, but it has been around for some time. Much of the original writings, however, were focused on automatically generating code from the examples. We’re not quite there yet with ATDD/TDD, but that isn’t to say we won’t make it a reality in the future.

### Landmark 3: It’s Not About Testing

That’s it, then; it’s all about specification by example—*right*? The whole point is to elicit and comprehend the customer’s requirements, deriving the logic and rules from specific examples of how the system is expected to function. Is it only about discovery of detail and creation of a specification that emphasizes the concrete over the abstract?

So, the fact that we express these examples as automated tests and that these tests help find regressions in the system is merely a useful side effect of the process. Or is it? Well, in short—*no*. When you see this landmark, I want you to turn around and double back until you see landmark 4.

### Landmark 4: It Is Also About Testing

Martin Fowler provides a nice analogy on this subject in a blog post about specification by example [2]:

These days it’s terribly unfashionable to talk about Test Driven Development (TDD) having anything to do with testing ... I do think that having a comprehensive automated test suite is more valuable than the term “side-effect” implies. Rather like if someone offered me a million dollars to hike up a mountain. I may say that the main purpose of the hike is the enjoyment of nature, but the side-effect to my wallet is hardly trivial.

So, ATDD is about requirements elicitation, and it is about testing. Just because the first benefit you encounter is requirements elicitation doesn’t necessarily mean that it is more about requirements than testing.

There is another rarely discussed aspect of ATDD in practice that also in-

volves testing. From the moment we start discussing the acceptance criteria for a user story and even while we are automating them, I find myself visualizing the application. Mentally I am attempting different scenarios, thinking, based on the examples provided so far, “How would I expect the system to respond if ... ?” or asking the customer “If I do this, what would you then expect?” I’m thinking creatively; I’m applying testing techniques to generate more potential examples or reduce overwhelming numbers of examples into a more comprehensible set, and none of this is scripted or automated. I’m learning as I discover more. I’m creating new test ideas or applying test-design techniques to identify another concrete example to ask about. Sound familiar? Indeed, this is very much like exploratory testing. In fact, discussions with exploratory testing expert James Bach led us both to conclude that this part of ATDD is a form of exploratory testing—just not as most people know it. It explores the *idea* rather than the *implementation*, avoiding the introduction of many of the bugs that would otherwise not be discovered until the code is deployed to testers.

### Landmark 5: Executable Specifications & Automated Validation

Barry Boehm’s book *Software Engineering Economics* [3] contains a clear distinction between validation and verification:

Validation: Am I building the *right* product?

Verification: Am I building the product *right*?

The *right product* is a product that meets the *real* requirements. The real requirements are the things that people will actually need to do with the system once they can use it. Traditional specifications (e.g., UML models, use cases, natural-language functional specifications, and the like) attempt to model our anticipation of these real-world requirements in a conceptualizing and generalizing way. I say generalizing because the rules within the specification contain variables rather than specific values—i.e., these rules apply *generically* for any relevant value used in place of the variables.

Of course, we assume that such generalizing specifications weren’t plucked out of thin air and were derived from concrete examples of anticipated usage in the real world. So on projects that derive tests (or examples of anticipated usage in the real world) from such a generalizing specification that was itself derived from examples of anticipated real-world usage is at best verification against the abstraction and at worst the software development equivalent of the “Telephone Game.” The very nature of such generalizing specifications makes it difficult, if not impossible, to use them in any effort to validate the software against its requirements.

So, testing our system against a conceptualization of anticipated requirements can’t tell us if we are building the *right* product. It can only tell us if the product matches the conceptualization. It can only tell us if we are building the product *right*.

Let’s also not forget that a significant reason for conceptualizing software formally on paper before writing the code is that software processes and technologies of old made software hard and expensive to change. If the technology is designed to make code easy to change and the process is designed to allow for the iterative discovery of requirements by making code safe and inexpensive to change, then formally modeling or conceptualizing on paper becomes of limited value. In fact, due to the additional maintenance overhead, it becomes a hindrance.

ATDD starts with concrete examples of anticipated real-world usage and automatically evaluates the software against those examples. Short of mind reading and time travel, this is about as close as we can get to *validation*—as close as we can get to knowing if we are building the *right* product.

### Landmark 6: It’s an Automated Test, Jim, But Not as We Know It

Because in ATDD our suite of automated acceptance tests is an executable requirements specification of sorts, this affects how we express the tests. Typical automated test tools just don’t cut it. I learned this the hard way on my first XP project in 2000.

Kevin Lawrence gave me (and a

bunch of other people in a workshop) a clever way of explaining how automated acceptance tests in ATDD differ from your typical automated system test. He referred to levels of abstraction used in interaction design. He explained that interactions can be described at three different levels, as shown in table 1.

Typical automated testing tools—both commercial and open source—work at the tasks level. For our executable specifications, made up of automated acceptance tests, we want to express the intent *and* automatically validate the system against examples of anticipated real-world usage. Furthermore, we don't want to duplicate the effort by writing the tests as a document and then writing them again in a programming or scripting language. This is wasteful and risky when the document and code become out of step with each other.

So that the tests are expressive of intent, we write them at the activities level. Because they express concrete examples, they also must be specific (rather than general). For example, a generalizing statement would be “valid username and invalid password” whereas a specific statement would be “username of ‘antony’ and a password of ‘thisaintright.’”

In reality, I find it useful to clarify intent in certain situations, so I might want my example to be more specific but hint at the general—for example, “a valid username of ‘antony’ and an invalid password of ‘thisaintright.’”

Thus, the acceptance tests, in the simultaneous roles of requirements specification and automated tests, must be more activities level than tasks level and more specific than general.

Let's consider this user story:

*As a registered user,  
I want to identify myself to the system  
So that I can use the personalization features.*

One of the automated tests for this story might look like figure 1.

Behind the test in figure 1 are an ATDD framework called FIT and some fixture code that is written by the team, acting as a gateway between the framework and the application you're developing, as shown in figure 2.

Level	Example
GOALS	Find blogs about software testing
ACTIVITIES	Search for “software testing blogs” Go to testingReflections.com
TASKS	Go to google.com  Enter “software testing blogs” into the search field  Click the search button  Wait for the page to load  Click the link for testingReflections.com  Wait for the page to load

Table 1: Levels of interactions

TEST: INVALID CREDENTIALS SHOULD PREVENT SIGN-IN				
Given that we have the following RegisteredUsers :				
username	password			
antony	pass			
raymond	morpheus			
damani	enzo			
When you attempt the Login action with valid username of antony, invalid password of pas				
Then you are Not Logged In				
and the message shown is You have entered invalid login details				
▼ Other Examples...				
Other examples include these LoginScenarios				
comment	username	password	message?	outcome?
Case sensitive username	Antony	pass	You have entered invalid login details	Not Logged In
Case sensitive password	antony	Pass	You have entered invalid login details	Not Logged In
Blank password	antony	BLANK	You have entered invalid login details	Not Logged In
User doesn't exist	nonExistentUser	pass	You have entered invalid login details	Not Logged In

Figure 1

The fixture code receives inputs given to it by the framework, performs the tasks associated with the activities, and returns the application's response to the framework. The test framework compares the returned result with the expectation in the test artifact and displays whether each condition passed or failed. This has many benefits, not the least of which is encouraging reuse of code that supports the automated tests.

This looks a lot like the idea of data-driven tests. Nothing new, of course, but what is different is how expressive these tests are and how people go about implementing them. The fixture code

tends to be written in the same language as the production code and is version controlled in the same way as the production code. It is also often tied into a continuous integration server so it is executed automatically several times per day as part of the software build process.

These characteristics result in tests being run more frequently, developers helping to maintain the test code as the application changes, and the test artifacts and production code being kept in sync through shared version control. These characteristics also make it simpler to run business-logic tests through

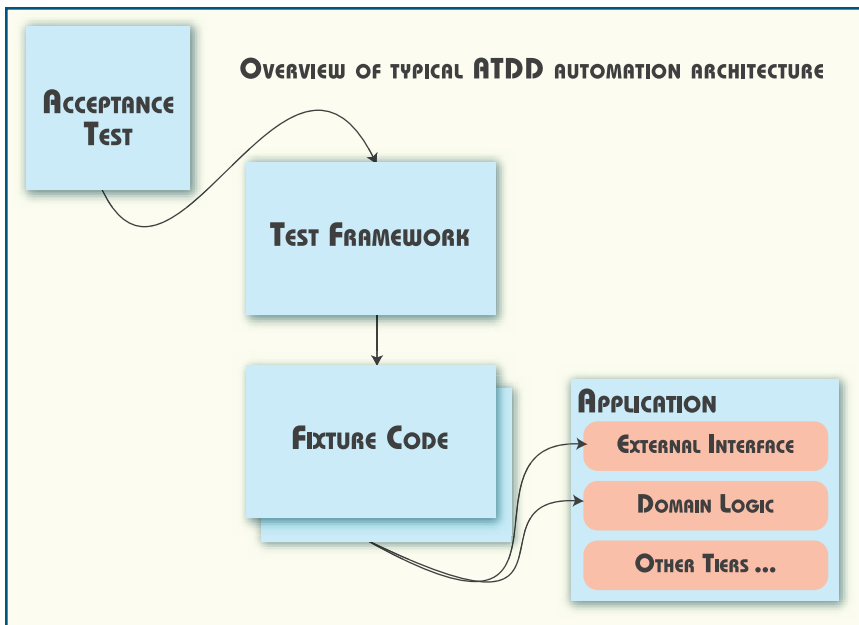


Figure 2

the business-logic tier rather than feeling restricted to an external interface.

### Landmark 7: Sometimes It Matters What You Don't Call It

Because the words “test” and “testing” are such overloaded terms, people hear them and immediately jump to all sorts of conclusions. I’ve had conversations with people about acceptance test-driven development and a few seconds later they refer back to it. “Yeah, so that acceptance testing stuff ... the users do that just before we go live, right?” Well, no, that sort of testing is to get feedback from the customer, which is part of ATDD.

Other times, people say, “Right, that stuff where the testers do all that automation?” Again, not quite, because members of the team contribute in different ways depending on their skills not their job titles.

The word test and the phrase acceptance test can, in the minds of the uninitiated, camouflage the other dimensions of the process—requirements elicitation and making sure we are building the right product. So, some prefer to call ATDD something else, such as story-driven development, example-driven development, executable-specification-driven development, behavior-driven development, and many more variations on the avoid-the-word-test theme.

### Landmark 8: Sometimes It Matters What You Do Call It

Having tried the avoid-the-word-test approach, I’ve noticed a detrimental side effect. People think it doesn’t involve testing at all, and later it becomes very hard to convince them otherwise. Meanwhile, testing specialists who would otherwise add value get sidelined. In the past, I’ve avoided the problem of ATDD’s being pigeon-holed as “a testing thing” by avoiding the word test, but I’m beginning to wonder if this is much like the “dark side” of the force—quicker, easier, and more seductive.

ATDD really is a cross-discipline activity, not dominated by one discipline or the other. Take one main skill area out of the equation and it suffers greatly. The value of representation from across the whole team—including the customer—is, in my experience, greater than the sum of its parts.

I’ve worked on teams that embraced the close involvement of testers with the development team for the first time, simply because of their interest in ATDD or because the developers were already “test infected” thanks to TDD. As the team tried to apply ATDD, team members realized that it was about eliciting requirements, driving the team to build the right product, *and* testing.

Whether you call it ATDD or something else depends entirely on which you feel is the priority for your organi-

zation—getting people other than testers interested or assimilating developers and testers into one team.

### Landmark 9: It’s More About the Dynamics Than the Artifacts

The Agile Manifesto has a pertinent statement: “We value people and interactions over processes and tools” (while there is value in the item on the right, the item on the left is valued more) [4]. ATDD in the context of XP is an example of how “people and interactions” are of great importance. ATDD within XP is first about people talking, sharing, and learning about the needs of the users in small and manageable iterations. The acceptance-test artifacts are a tool to help focus discussions on the detail that matters and capture the outcome in a way that adds enormous value in other ways, such as automated validation. Sending testers or customers an email and asking for the next batch of acceptance tests doesn’t work even half as well as having a conversation.

### You Will Then Arrive at Your Destination

If you’re just learning about ATDD, hopefully these landmarks have helped you see whether ATDD has potential value for you. If it has, then the next step is to learn by doing. Don’t expect to appreciate its subtleties overnight. Don’t try to do it alone; find someone who is experienced to guide you. The roads you’ll take between each of these landmarks, and any new ones you discover, will be different from one organization to the next. Indeed, the order in which you may pass each landmark may differ from one individual to the next.

Whichever route you take, remember to take in the scenery along the way by reflecting on your experience as you observe each landmark. Most of all, remember that it’s also meant to be fun, so make sure you enjoy the journey every step of the way. **(end)**

#### REFERENCES:

- 1] Beck, Kent. *Test-Driven Development: By Example*. Addison-Wesley, 2002.
- 2] [martinfowler.com/bliki/SpecificationByExample.html](http://martinfowler.com/bliki/SpecificationByExample.html)
- 3] Boehm, Barry. *Software Engineering Economics*. Prentice Hall, 1981.
- 4] [agilemanifesto.org](http://agilemanifesto.org)




So,

# YOU'VE GOT A PROBLEM...

CRAFTING REMARKS & ABSTRACTS  
MORE EFFECTIVE DEFECT REPORTS  
BY KELLY WHITMILL







**D**EFECT REPORTS are among the most important deliverables to come out of test. They are as important as the test plan and will have more impact on the quality of the product than most other deliverables from test. Thus, it is well worth the effort to learn how to write effective defect reports.

Effective defect reports can:

- Reduce the number of defects returned from development
- Improve the speed of getting defect fixes
- Improve the credibility of testers
- Enhance teamwork between test and development

The objective is not to write the perfect defect report but to write an effective defect report that conveys the proper message and simplifies the process for everyone. This article focuses on two aspects of defect reports: the remarks (or description) and the abstract. First, let's take a look at the essentials for writing effective remarks.

## DEFECT REMARK

### DON'T

Suffers from TMI  
(too much information),  
most of which is  
not helpful.

I was setting up a test whose real intent was to detect memory errors. In the process, I noticed a new GUI field that I was not familiar with. I decided to exercise the new field. I tried many boundary and error conditions that worked just fine. Finally, I cleared the field of any data and attempted to advance to the next screen, then the program aborted. Several retries revealed that any time there is not any data for the product description field you cannot advance to the next screen or even exit or cancel without aborting.

### DO

The exit, next, and cancel functions for the product information screen abort when the product description field is empty or blank.

Table 1: Condense example

## DEFECT REMARKS

### CHECKLIST

Here is a mnemonic to use as a mental checklist for self-inspecting your defect report. The first letter of each item on the checklist spells CAN PIG RIDE. These ten points ensure that your defect reports answer the questions that will be of most benefit to your organization:

- Condense
- Accurate
- Neutralize
- Precise
- Isolate
- Generalize
- Re-create
- Impact
- Debug
- Evidence

It is not just good technical writing skills that lead to effective defect reports. It is more important to make sure that you have covered the essential items that will be of most benefit to the intended audience of the defect report.

## ESSENTIALS FOR EFFECTIVE DEFECT REMARKS

### CONDENSE

Say it clearly but briefly. First, eliminate unnecessary wordiness, as shown in

table 1. Second, don't add in extraneous information. It is important that you include all relevant information, but make sure that the information *is* relevant. In situations where it is unclear how to reproduce the problem or the understanding of the problem is vague, you will probably need to capture more information. Keep in mind that irrelevant information can be just as problematic as too little relevant information.

### ACCURATE

Make sure that what you are reporting really is a bug. You can lose credibility very quickly if you get a reputation of reporting problems that turn out to be setup problems, user errors, or misunderstandings of the product. Before you write up the problem, make sure that you have done your homework. Consider the following:

- Is there something in the setup that could have caused this? For example, are the correct versions installed and all dependencies met? Did you use the correct login, security, command/task sequence, and so forth?
- Could an incomplete cleanup, incomplete results, or manual interventions from a previous test cause this?
- Could this be the result of a network or some other environmental problem?

- Do you really understand how this is supposed to work?

Make sure that you understand the numerous test-case influences and consider their roles in the perceived bug you are reporting. This is one area that quickly separates the experienced tester from the novice. If you are unsure about the validity of the problem, it may be wise to consult with an experienced tester or developer prior to writing up the problem.

Don't be afraid to write up problems, but do your best to ensure that they are valid problems. When you discover that you have opened an incorrectly reported problem, make sure that you learn from the experience.

### NEUTRALIZE

State the problem objectively. Don't try to use humor, and don't use emotionally charged zingers. What you think is funny when you write the defect may not be interpreted as funny by a developer who is working overtime and is stressed by deadlines. Using emotionally charged statements doesn't do anything for fixing the problem. Emotional statements just create barriers to communication and teamwork. Even if the developers doubted you and returned your previous defect and now you have proof that you are correct and they are wrong, just state the problem and the additional



DEFECT	REMARK
<b>Don't</b>	As could have been determined from the original defect with very little effort, function ABC does indeed abort with any negative value as input.
<b>Do</b>	Function ABC aborts with any negative value. Examples of some values tested include -1, -36, -32767.

**Table 2: Neutralize example**

information that will be helpful to the developer. In the long run, this added bit of professionalism will gain you respect and credibility. Read over your problem description before submitting it, and remove or restate those comments that could be interpreted as being negative toward a person.

Table 2 is a response to a developer's returning a defect for more information and requesting more details on what values caused the problem.

## PRECISE

The person reading the problem de-

scription should not have to be a detective to determine what the problem is. Right up front in the description, describe exactly what you perceive the problem to be. Some descriptions detail a series of actions and results. For example, "I hit the Enter key and action A happened. Then I hit the back arrow and action B happened. Then I entered the 'xyz' command and action C happened." The reader may not know if you think all three resulting actions were incorrect or which one, if any, is incorrect. In all cases, but especially if the description is long, you need to summarize the

problem at the beginning of the description. Don't depend on an abstract in a different field of the defect report to be available or used by everyone who reads the problem description. Don't assume that others will draw the same conclusions that you do. Your goal is not to write a description that it is possible to understand but to write a description that won't be misunderstood. The only way to make that happen is to explicitly and precisely describe the problem rather than just giving a description of what happened, as shown in table 3.

DEFECT	REMARK
<b>Don't</b>	Issuing a cancel print when job is in PRT state (job is already in the printer and AS/400 is waiting to receive print complete from printer) causes the twinax port to not time out. The printer never returns to a READY state and indefinitely displays "PRINTING IPDS FROM TRAY1" in the op-panel.
<b>Do</b>	Canceling a job while it is printing causes the printer to hang. Issuing a cancel print when job is in PRT state ...

**Table 3: Precise example**

## ISOLATE

Each organization has its own expectations of how much the tester is required to isolate the problem. Regardless of what is required, a tester should always invest some reasonable amount of effort into isolating the problem. Consider the following when isolating problems:

- Try to find the shortest, simplest set of steps required to reproduce the problem.
- Ask yourself if anything external to the specific code being tested contributed to the problem. For example, if you experience a hang or delay, could it have been due to a network problem? If you are doing end-to-end testing, can you tell which component along the way had the failure? Are there some things you could do to help narrow down which component had the failure?
- If your test has multiple input conditions, vary the inputs until you find which one with which values triggered the problem.

In the problem description, describe the exact inputs used. For example, if you found a problem while printing a PostScript document, even if you think the problem occurs with any PostScript document, specify the exact document that you used to find the problem.

Your ability to isolate in large part defines your value-add as a tester. Effective isolation saves everyone along the line a great deal of time. It also saves you a lot of time when you have to verify a fix.

## GENERALIZE

Often times, the developers will fix exactly what you report, without even realizing the problem is a more general problem that needs a more general fix. For example, I may report that my word processor “save file” function failed and the word processor aborted when I tried to save the file “myfile.” A little more investigation may have revealed that this same failure occurs any time I save a zero-length file. Perhaps on this release it aborts on every save to a remote disk, a read-only disk, and so forth. To already know this when you write the report will save the developer a lot of time and enhance the possibility of a better fix to handle the general case.

When you detect a problem, take reasonable steps to determine if it is more general than is immediately obvious, as shown in table 4.

## RE-CREATE

Some bugs are easy to re-create, and some are not. If you can re-create the bug, you should explain exactly what is required to do the re-create. You should list all the steps, include the exact syntax, file names, and sequences that you used to encounter or re-create the problem. If you believe that the problem will happen with any file, any sequence, etc., then mention that but still provide an explicit example that can be used to do the re-create. If in your effort to verify that the bug is re-creatable you find a shorter and more reliable means of re-creating, document the shortest, easiest means of re-creation.

If you cannot re-create the problem,

or if you suspect that you may not be able to re-create the problem, gather all the relevant information possible that may provide useful information to the person who has to try to fix the problem. This may be a time when you consider asking a developer if he wants to examine the system while it is still in the problem state or if there is any particular information that should be captured before cleaning up the problem state and restoring the system. Don’t assume that it can be re-created if you haven’t verified that it can be re-created. If you cannot or have not re-created the problem, it is important to note that in the defect remarks.

## IMPACT

What is the impact if the bug were to surface in the customer environment? The impact of some bugs is self-evident: “Entire system crashes when I hit the Enter key.” Some bugs are not so obvious. For example, you may discover a typo on a window. This may seem trivial unless you point out that every time someone uses your product this is the first thing he sees, and the typo results in an offensive word. In this case, even though it is just a typo, it may be something that absolutely must be fixed prior to shipping the product. Make your best judgment. If you think it is possible that this defect will not get sufficient priority, then state the potential impact and sell the importance of the defect. Don’t oversell, but make sure the readers of the defect report have an accurate understanding of the probable impact on the customer.

### DEFECT REMARK

DON'T

Error message for “file not found” error has garbage characters for the file name.

DO

Error message for “file not found” error has garbage characters for the file name. Every message I tried that expected data to be inserted in the message had the same problem.

Messages without inserts were OK.

Table 4: Generalize example



## WHAT YOU THINK IS FUNNY WHEN YOU WRITE THE DEFECT MAY NOT BE INTERPRETED AS FUNNY BY A DEVELOPER WHO IS WORKING OVERTIME AND IS STRESSED BY DEADLINES.



### DEBUG

What will the developer need to be able to debug this problem? Are there traces, dumps, logs, and so forth that should be captured and made available with this defect report? Document what has been captured and how it can be accessed.

### EVIDENCE

What evidence exists that will prove the existence of the error? Have you provided both the expected results and the actual results? Is there documentation that supports your expected results? Since you are writing a problem report, it is obvious that you believe there is a problem. Provide anything you can that will convince others that this is a valid problem. Evidence may take the form of documentation from user guides, specifications, requirements, and designs. It may be past comments from customers, de facto standards from competing products, or results from previous versions of the product. Don't assume everyone sees things the same way you do. Don't expect people to read between the lines and draw the same conclusions as you. Don't assume that three weeks from now you will remember why you thought this was a bug. Think about what it is that convinced you that this is a bug and include that in the report. You will have to provide even more evidence if you think there is a chance that this situation may not be readily accepted by all as a valid bug.

### DEFECT ABSTRACTS

The short, one-line abstract that gets associated with most defects can be a very powerful communication tool. Oftentimes, the abstract is the only portion of the defect that gets read by the decision makers. It is the abstract and not

the full description that gets included in reports. It is the abstract that the project managers, screeners, team leads, and other managers look at when trying to understand the defects associated with the product.

The abstract must be concise and descriptive and convey an accurate message. The abstract is usually very limited in length. Because of the space limitations, abbreviations are OK and short, accurate messages take priority over good grammar. A good use of keywords is essential since many searches are based on the abstract. Keywords such as abort, hang, typo, and so forth are both descriptive and useful as search words. Where space permits, it is helpful to mention the environment, the impact, and any of the who, what, when, where, and why questions that you can address in such a short space.

Be as specific as possible. For example, the following abstract is true but doesn't provide nearly as much information as it could:

*Abstract: Problems found when saving and restoring data member.*

Perhaps this would be a more descriptive abstract:

*Abstract: xyz's save/restore of data member on WinNT fails, data corrupted.*

You can never get everything you want in an abstract, so here is a list of items and tips that I try always to include:

#### Mandatory:

- Concisely, explicitly state what the problem is—not just that there is a problem.

#### Recommended (space permitting):

- Use meaningful keywords.
- State environment and impact.
- Answer who, what, when, where, why, and how.
- Using abbreviations is OK.

- Grammar is secondary over conveying the message.
- Don't use defaults.

### SUMMARY

Testers spend a significant amount of time seeking out and discovering software problems. Once detected, it greatly enhances productivity to report the defect in such a way as to increase the likelihood of getting the problem fixed with the least amount of effort. Making sure that the proper information is provided is more important than superior writing skills. The ten topics described in this paper will go a long way toward helping you provide the right information in every defect report.

The better you are at writing defect reports and abstracts, the more likely it is that problems will actually get fixed in a timely manner. Your credibility and value-add to the business will increase as developers, managers, and other testers are better able to do their jobs because your defect reports are well written and reliable. {end}

### Sticky Notes

For more on the following topic go to [www.StickyMinds.com/bettersoftware](http://www.StickyMinds.com/bettersoftware).

- Further reading



# A Culture of Trust Creating a Team Environment Where



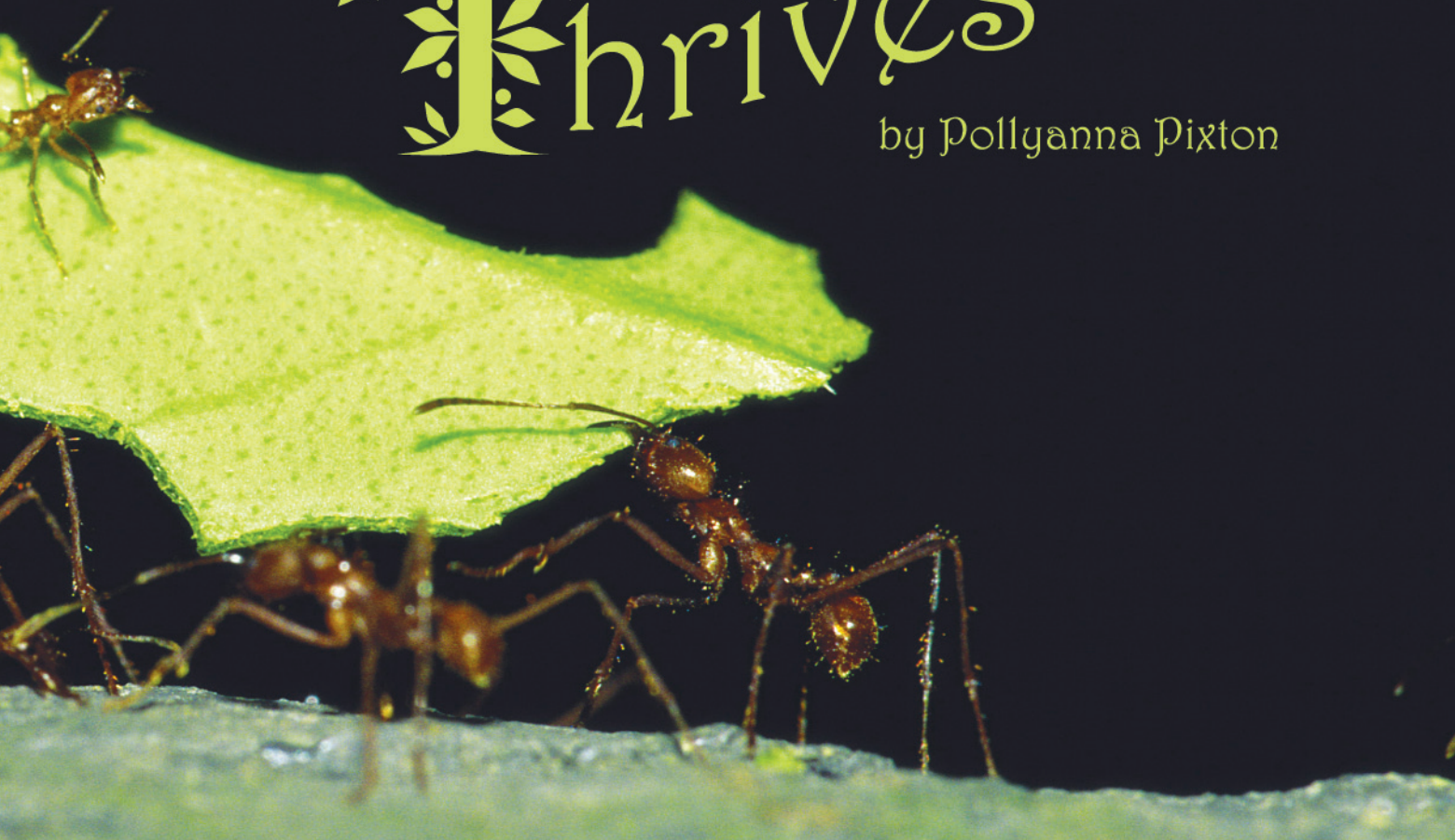
You've been asked to take over the leadership of a struggling team. The lack of trust within the team sticks out like a sore thumb.

Your current team exudes trust, and it has made quite a difference—morale and productivity are high, ideas and the information everyone needs to be successful flow freely, and team members help each other without taking over. In other words, it's a cohesive, collaborative team that delivers results.

So how do you help a struggling team become a trusting team? This article is not about how team members develop trust in their personal relationships; there are many articles written on that topic [1, 2]. Rather, I address how leaders can create a culture where the building of trust between team members is fostered, flourishes, and thrives—where people who have not begun to trust each other can discover the possibility.

# Environment Trust Thrives

by Pollyanna Pixton



## What Does a Team Without Trust Look Like?

How do you know that a team lacks trust? There are several signs of a non-trusting team:

- **Fear**—People are afraid to speak up, afraid to fail, and afraid of humiliation.
- **All members are out for themselves**—Team members pontificate and spout monologues while at the same time keeping valuable information to themselves.
- **Lack of engagement**—Team members do not support each other and don't participate in team discussions and decisions.
- **Defensiveness and negativity**—Team members exhibit closed

body language, never saying a good word about other team members or the tasks at hand.

- **Condescension**—Ideas are dismissed without consideration or are criticized unfairly.
- **Passive-aggressiveness and lack of integrity**—In meetings, people agree to one thing but outside the meetings say and do another.
- **Impatience**—There is tension in every working encounter, and team members lack initiative.
- **Rampant gossip and complaining**—Team members talk behind each other's backs and engage in mean-spirited humor instead of healthy, fun humor.

It is quite a challenge to take on

such a team and lead team members to trusting each other, especially when you are faced with the fact that, as a leader, you cannot change people. Ordering members to trust each other just doesn't work.

Still, you want to take on the leadership of this team. You respect many of the team members. They are a talented group and have produced great results on other teams. The project they are working on is important to the company, and you figure you can help. But there is one question you need to consider: Can I trust everyone on this team? You may not know all of them well, but you must make sure there is no one on the team you distrust.

As Ricardo Semler asks in his book, *The Seven Day Weekend*, if you don't



trust the people on your team, why are they on your team? For that matter, why are they in your organization?

If your answer to the question, “Can I trust everyone on this team?” is yes, then accept the leadership role for this team. Now what do you do?

## Creating a Culture of Trust

“Just pick people who are trustworthy.” I get this answer at times from colleagues.

“And, how often do leaders actually get to do that?” I reply.

You are assigned a project. You pick people you know are good, competent, and trustworthy; give them responsibility to deliver within the constraints and they become a trusting, high-performance team, all within fifteen minutes.

You may have this experience once in your lifetime—or not. Realistically, we are given our team members with all their foibles, shared history, and excess baggage.

## Can Team Members Trust Each Other?

I’ve been asked, “What do you do, Pollyanna, when there is one person on a team that no one trusts?”

“Are you sure they don’t trust him?” I answer. “Is it a lack of trust or has trust been broken?”

Broken trust is like a cut rope. There are many strands, wound together, that give the rope its strength. Once cut, repairing the rope requires matching each piece, strand for strand. Not only does it take time, but the rope will never be the same and will not have the strength of the original rope. Can a distrusted person repair his relationship with the rest of the team? Does he want to? And does he have the skills to do so? Possibly, but the time and effort to do so is very high, and the results might not be optimal or even acceptable.

However, there may be no broken trust in the team, just lack of trust, which is an easier problem to solve. Trust can be developed. Take the time to

observe and assess to see how deep the lack of trust might be and identify some possible causes.

Interview team members individually and in confidence. Ask them how they like their work on the team, what’s working, and what’s not. If they could fix what’s not working, what would they do and why? Ask what obstacles are getting in the way of their individual success and their team’s success. Do they feel like the team can deliver the expected results? If not, what can be done to improve their chances? Check to see if they feel the right people are on the team and that everyone has the knowledge, experience, and commitment to complete the

“As the leader, your role, style, and behavior will lay the groundwork for building a culture of trust.”

project. Most importantly, ask if they trust everyone on the team. Team members may be uncomfortable answering such direct questions from their new leader, so listen for the “ring of truth” in what people say and make note of what they leave out of the conversation.

Walk the floor. Watch and listen to how the team works. Is one person talking all the time? Are people ignoring one or more of their fellow team members? Are there constant put-downs or dismissals of one person’s ideas? Spend time in the break room. How do team members interact there? Do they avoid someone? Do they talk about ideas? Do they avoid eye contact with some of their team members? Listen to the interactions within the team and with people outside the team, and look at their results and progress.

Look for trends or threads in your conversations and observations. Did you sense any red flags or unauthentic answers? Did one name come up again and again as someone who did not deliver as he said he would? Did one person consistently withhold information? Was he constantly noted as hard to get along with, never listening, or saying one thing and doing another? If the issue of distrust seems to point to one person, you face a hard decision.

You have two choices: Keep this person involved with the team at some level or remove him. What is your first response? Your intuitive answer may be the right one, but, before you act, answer a few more questions. How valuable is he to your team and to your project? Can your team succeed without him? What are the negative impacts if he leaves? If he stays?


“Apply the ‘vacation test,’” advises my colleague Niel Nickolaisen. “See how the team does when the ‘problem’ team member goes on ‘vacation’ for a few days.” Take this person off the team and give him something else to do. Place him somewhere where he cannot interfere with the day-to-day functions of the team. What happens to team productivity? To motivation and morale?

If you come to the conclusion that the team benefits from his not being there, then remove him from the team as soon as possible. But what if you need to keep him involved with the team at some level?

Again, you have two choices: Ask the team to integrate this person in some way into the project, or create a “one-person island” inside the team. Both are difficult and will take time and effort to make happen. Without this person present, sit with team members and ask them how they can work with him. What team norms would have to be established to make it happen? What would they need to be successful with him on the team? What do they need from you to make it happen? What does the team want you to do when the disruptive person interferes too much with the other team members? Come to an understanding that, while the team must work with this disruptive person, he does not have to be viewed as a team member. The team can collaborate without him and make decisions with this person only when it involves his work.

Everyone must understand that sabotaging the disruptive person is not acceptable and would be seen as sabotaging the team’s efforts. Ensure that the team will be measured as a team, not as individuals. While team members can’t



A close-up photograph of several ants on a bright green, torn leaf fragment against a black background. The ants are brown and translucent, with their legs and antennae clearly visible. One ant is at the top, another at the bottom left, and a third at the bottom right, all appearing to be working on the leaf.

“Expect Success; Accept Mistakes.  
Stress the motto ‘Fail early, fail fast!’  
People learn from their mistakes.”

control the disruptive person in their midst, they can use his knowledge and experience to succeed as a team.

You have resolved the situation with the difficult person, and now your team issues are solved. You have a trusting, productive team, right? Not yet!

### Steps to Build a Culture of Trust

As the leader, your role, style, and behavior will lay the groundwork for building a culture of trust. There are a few things you need to pay special attention to about yourself. Authenticity is essential. Your team members will see right through you if you are missing your own “ring of truth,” and their lack of trust will continue. Be trustworthy, and own up to your own foibles, history, and mistakes. Share all information with the team, and, when you can’t, explain why. You first have to show you trust your team.

Give up command-and-control leadership, and stop micromanagement. Telling people what to do and how to do it shows a lack of trust. If you trust people, you know they will do what they say they will do—and that they know best how to do it. Micromanagement sends a message that you do not trust those you are leading.

Trust me, your leadership will be tested by your team. Team members will come back several times to see if you will rescue them or fix it for them, if you will tell them what to do and how to do it, if you will really accept mistakes, and whether you genuinely trust them to deliver. They will watch carefully and test your trustworthiness. Will you listen? Will you give them the information they

ask for? Will you admit your mistakes? Will you be honest?

Now focus on creating a culture where team members can build trust among themselves. Use the following steps:

- **Remove Debilitating Fear.** Without the team generating ideas and solutions, progress will be slow and perhaps impossible. Debilitating fear is what keeps team members from expressing their questions, solutions, analysis, and investigations. They are afraid of humiliation, ridicule, loss of respect, and—the deepest fears—loss of their position, pay, and perhaps their job. Fear results in paralysis and catastrophizing (making things seem worse than they are). There are ways to mitigate fear. As a leader, acknowledge openly with team members the fear that you feel is in the team. Imagine, reframe, and describe the team culture as it will be as a trusting team. Remind team members that they have choices in how they respond to the fear. Ask them to give the team a chance and to bring their ideas to team meetings or simply try their ideas out on other team members, one on one. Practice the steps to change: Celebrate resistance, figure out what to be—not what not to be—and take small steps.
- **Use Team-based Measurements.** Measuring individual performance is a deterrent to collaboration and team members’ working together. Often, if individuals are

measured on their own performance, they don’t care how well the rest of the team does; they’ll look out for themselves, first. To change this dynamic, measure the team, not the individuals. This motivates team members to work together and help each other deliver a team success.

- **Ask for Small Deliverables in Short Iterations.** Ask team members for rapid, incremental deliverables—small successes where they can see progress and successful results as a team. Let them make decisions on how they will do this, how they will do their own work, and how they will work together. They don’t need you to tell them. Step back and let the team decide.
- **Expect Success; Accept Mistakes.** Stress the motto “Fail early, fail fast!” People learn from their mistakes. Right now, your team may be worried about taking a risk and failing. Sure, removing disruptive fear might help. Most important is your protecting them and your organization. Create a way for the team to fail safely. What does that mean? First, you don’t want the team to be embarrassed in front of your customers, in front of organizational leaders, or in front of other teams. Add a step in the processes (or, better still, suggest team members evaluate adding a step) where they can walk through their results before they go outside of the team.
- **Take the “Fun” Out of Being Dysfunctional.** Ignore unprofes-



“In a non-trusting environment, people spend a lot of time protecting themselves. It’s no wonder teams without trust exhibit low productivity.”

sional behavior. What do you do about those who are “gaming the system,” where team members leverage the leader to discredit another team member? This rarely exists in a healthy team—one that is collaborative and in which team members understand and respect each other’s contributions, remain focused, and have ownership. But you don’t have that, yet. To get there, take the “fun” out of dysfunction. Remove the reward people are getting for playing games within the team. When someone causes distractions, such as asking rhetorical questions with no real purpose—trying to impress you with the “right answer” or embarrass you if you make a mistake—ignore this behavior. Stand quietly and do not say a word, or look to others in the room and change the subject. Remember, negative attention can be its own reward.

## Helping the Process Along

The team’s efforts to build trust will have their ups and downs. Here are a few things you can do to help that process move forward:

- **Focus on Purpose.** Team members don’t get to do whatever they want, whenever they want. But they do need time for exploration of their ideas and possible solutions. When chaos has gone on too long, step in and ask questions to help the team get back on track. Ask, don’t tell. Team members don’t need to be told what direction to go; they need

assistance in discovering their own direction as it relates to their current project.

- **Protect the Team Boundaries.** Be your team’s advocate. Get the team what it needs to succeed, and don’t let the distractions of corporate bureaucracy and politics creep into the work environment.
- **Stay Positive.** The team needs acknowledgement, feedback, and recognition. Affirm what is working. Don’t dwell on past failures or anything that could possibly be interpreted as judgmental. Negative feedback will be blown out of proportion, and its effects may take a long time to repair.

## The Lack of Trust “Price Tag”

Why should you expend the effort to create a culture of trust? Look at the transaction costs—the number of decisions and actions you make interacting with someone or something. How long does it take you to get a task done by someone you don’t trust? There is the preparation time, during which you figure out how to approach this person and how to clearly state your request. In the conversation, how many times did you say the same thing in different ways to ensure you were heard? How many times did you check to make sure that he correctly heard what you said? How often did you check to see if he will deliver what and when he said he would? The transaction costs of distrust are high. In a non-trusting environment, people spend a lot of time protecting themselves. It’s no wonder teams without trust exhibit low productivity.

## Summary

We know teams deliver great results when they take ownership. After working with a team on setting goals and objectives, leaders must step back and let the team work. You can’t do this without trust. It is essential in engaging teams, retaining talent, fostering innovation, creating great working environments, and delivering results. When the trust goes out of a team, what can a leader do?

Hard as it may be, you must decide what to do when one member of the team has broken trust with the others. It really does not matter how it happened. To keep such a person on your team is costly. You must decide if it is better—or not—to take the person off the team. If you keep that person, you and the rest of the team must decide how to work with him.

As the leader, make sure you trust or can build trust with everyone on your team. Be transparent and show you are open to new ideas and different ways of thinking. Practice collaborative leadership; give up micromanagement and command and control.

Leaders cannot make people trust each other, but leaders can create not only a culture that encourages trust but also one in which trust flourishes and thrives. {end}

## REFERENCES:

- 1] Hurley, Robert F. *The Decision to Trust*. Harvard Business Review, September 2006.
- 2] Benson-Armer, Richard and Stickel, Darryl. *Successful Team Leadership Is Built on Trust*. IVEY Business Journal, May/June 2000.



# SOFTWARE ENGINEERING TRAINING

*Accelerate Your Career  
& Empower Your Team*



## 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](http://www.sqetraining.com) or call 888.268.8770 or 904.278.0524 for pairing discount options.



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.

## REQUIREMENTS WEEK LOCATIONS

October 6-10, 2008      *San Diego, CA*

## CMMI® & SOFTWARE MEASUREMENT WEEK LOCATIONS

September 22-26, 2008      *Boston, MA*

November 10-14, 2008      *Las Vegas, NV*

### REQUIREMENTS

- Mastering the Requirements Process (Mon-Wed)
- Requirements Modeling (Thu-Fri)

### CMMI® & SOFTWARE MEASUREMENT

- Introduction to the Capability Maturity Model Integration® (CMMI®) (Mon-Wed)
- Business-Driven Software Measurement (Thu-Fri)



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](mailto:onsitetraining@sqe.com).



# Product Announcements

## RiTA

NEW YORK, NY—Redstone Software Inc. announces the migration to and general availability of the Redstone integrated Test Appliance (RiTA) on the Mac OS X 10.5 Leopard operating system. By supporting Leopard, Redstone is able to exploit the latest Apple Xserve hardware to create robust and powerful platforms from which Eggplant can be deployed to Windows desktops. The combination of the improved stability of Leopard and the high specifications of Apple's Xserve hardware also ensures that RiTA is well positioned to support the needs of customers for the foreseeable future.

Features include:

- Image Collections enable a single script to test an application or process across multiple platforms.
- Image Doctor simplifies update and maintenance of image-based test scripts in rapidly changing UI environments.
- Image Import accelerates test script development by leveraging

existing image resources.

- Text Image Generation simplifies interacting with text on the screen of the system under test, including dynamic content not known until test execution time.
- Readable SenseTalk scripts and a powerful interactive debugger simplify even complex automation development and script maintenance tasks.

Customers can choose between two models of Redstone's integrated Test Appliance. RiTA-1020, designed for mid-size testing departments, supports up to ten concurrent development sessions and twenty test execution sessions. RiTA-2050, designed for large testing departments, supports up to twenty concurrent development sessions and fifty test execution sessions.

Visit [www.redstonesoftware.com](http://www.redstonesoftware.com) for more information.

## ElectricCommander

SUNNYVALE, CA—Electric Cloud an-

nounces that ElectricCommander, an enterprise-class automation framework for the entire software build-test-deploy process, now features integration with HP Quality Center software. The technology integration speeds the overall development cycle and enhances quality through early, more frequent testing and improved collaboration between developers and QA teams.

With the integration of ElectricCommander with HP Quality Center, automated test sets defined and selected by QA are exposed within the ElectricCommander interface, so they can be run at any time by developers—on a scheduled basis, when triggered by a specific event, or on demand. Results of the tests are maintained in both systems, so everyone on the team has access to status and trend information. Software development organizations gain better code quality, faster time to market, and more productive QA teams.

ElectricCommander provides an automation framework for software production, unifying people, machines, processes, and tools. Software production is transformed from script-driven work by dedicated experts to unified, push-button processes driven by developers. With ElectricCommander, development organizations can benefit from:

- An automated system enabling self-service builds and tests by developers
- A shared QA and development dashboard to track status of testing activities
- Reusable test assets for both QA and development teams
- Push-button build and test cycles without manual scripting
- An end-to-end quality lifecycle encompassing development and QA

Visit [www.electric-cloud.com](http://www.electric-cloud.com) for more information.

## Lifecycle Management Solutions

BOULDER, CO—AccuRev, Inc. and Rally Software Development Corp. announce a technology partnership that will integrate AccuRev software change and configuration management (SCCM) with Rally's agile lifecycle management



### The Award-Winning-Est Agile Lifecycle Management Solution



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



Two-time SD Times 100 award winner for tools  
*"that made December 2007 a far more productive and efficient time to code than January 2007."*



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/bsm](http://www.rallydev.com/bsm)  
No download, no installation, no commitment

solutions. The combined solution will provide a platform to manage multiple agile processes and ongoing customer feedback, while improving visibility and requirements traceability between defects, issues and tasks, and the actual source code changes made to address them.

The results from agile software delivery teams are clear: faster delivery of quality software, significantly improved project visibility, and better alignment between business and IT organizations. But the ongoing challenge for many companies is taking early success—often at a team or division level—and scaling the success broadly across their companies. Impediments to success often include teams building the right skills and using the right tools to deliver results.

Visit [www.rallydev.com](http://www.rallydev.com) for more information.

#### LISA 4.5

DALLAS, TX—iTKO LISA announces the release of LISA 4.5 SOA Testing, Validation & Virtualization Solution. LISA 4.5 extends the functionality of

the award-winning LISA 4 suite, adding service virtualization for load and performance testing and customer-driven enhancements.

New features and enhancements for LISA 4.5 include:

- Performance-ready LISA Virtual Services Environment (VSE): LISA VSE provides a ready target for the test lab, capturing and simulating the rest of the architecture dependencies with a high level of functional accuracy in terms of business logic and data. VSE supports more than 2,500 plus transactions per second on a single typical CPU instance, at a fraction (average 3 to 5 percent) of the hardware and software provisioning and maintenance cost of attempting to replicate a test environment, when many of the components are distributed and not under the team's direct control.
- Native Integration Platform support increased: Most iTKO customers are seeking to better

manage technologies using leading integration vendor platforms. Therefore, much innovation has been applied to increasing support of business process validation for BPM tools and interaction with IT operations and monitoring frameworks such as TIBCO Hawk, CA/Wily, IBM Tivoli, and HP OpenView.

- Improved UI layer testing: Today's rich Internet applications (RIA) models are driving more complex functionality into the Web browser. LISA 4.5 further enhances testability of dynamic business interfaces such as those used in SaaS, eCommerce applications, and reporting and monitoring dashboards. In addition to supporting Web/HTTP, AJAX, Swing, AWT, DOM, and other presentation layer technologies, Adobe Flash, Flex, and Microsoft ActiveX are now testable with LISA.
- Usability and workflow enhancements: In addition to some look

An offering of the  
School of Systems and Enterprises at  
Stevens Institute of Technology

# Systems-Centric Software Engineering

**Graduate programs tailored to the real-world education needs of today's Software and Systems Professionals**

Integrating the principles of Systems and Software Engineering to provide students with the skills required to conceive, develop, assure, and maintain complex software-intensive systems.

Graduate Certificate and Master's Degree programs are offered in convenient, flexible delivery formats.

Courses held on-site at corporate and government facilities, online via Stevens award-winning WebCampus, and on-campus in Hoboken, NJ.

To learn more about Stevens Systems-Centric Software Engineering programs and degree options, visit:

[www.stevens.edu/sse](http://www.stevens.edu/sse)





and feel changes, new interface elements and workflows within LISA were streamlined to increase user productivity. New step creation, assertion, and filter wizards with context sensitive help are some notable examples. Users of Mercury TestManager/HP QualityCenter can also coordinate LISA tests within their tools in a few clicks.

For more information, visit [www.itko.com](http://www.itko.com).

### Accept Version 4.6

FREMONT, CA—Accept Software has released Accept version 4.6, the newest release of its award-winning SaaS innovation management suite. Major enhancements in this latest release include native support for the full range of agile development methodologies and processes. New agile capabilities build on prior Accept releases designed to help companies get the right products to market faster, while making it even easier for everyday users to manage complex product cycles and dependencies.

Features include:

- Agile methodologies, including Scrum, feature-driven development, agile unified process, Extreme Programming, and others
- Agile development processes, including iterations, stack-ranked backlogs, user stories, and others
- Agile project management, including velocity and burndown reports and a centralized repository of artifacts that facilitates reduced documentation
- Usability improvements including an updated user interface that makes it easier to use and manage filters and more intuitive to create implementation estimates
- Extended release hierarchies to support larger projects that consist of multiple sub-projects
- More robust Outlook integration, enabling users to create suggestions and other elements directly from emails

Visit [www.acceptsoftware.com](http://www.acceptsoftware.com) for more information.

### Parasoft Application Security Solution

MONROVIA, CA—Parasoft announces its Application Security Solution, which establishes a continuous process that ensures security verification and remediation tasks are not only deployed across every stage of the SDLC, but also ingrained into workflow.

Parasoft's Application Security Solution expands traditional data flow analysis from software quality to application security. This server-based technology statically simulates complex application execution paths to help teams find vulnerabilities including SQL injection, cross-site scripting, exposure of sensitive data, and other potential issues. The latest enhancements not only draw upon an extensive knowledge base of common attack patterns but also enable organizations to map the data flow logic to their own security policies.

Visit [www.parasoft.com/parasoft\\_security](http://www.parasoft.com/parasoft_security) for more information.



Have you been wondering how you can get a daily dose of what's new and popular on StickyMinds.com and in *Better Software* magazine?

We've been reading your mind!

StickyMinds.com and *Better Software* magazine are now on **Twitter**. If you're already on **Twitter**, follow @StickyMinds for regular updates about weekly columns, news, discussion boards, eNewsletters, and more, as well as information about *Better Software* magazine articles and Software Quality Engineering conferences. If you don't have a **Twitter** account, you can follow our **Twitter** feed at [www.twitter.com/StickyMinds](http://www.twitter.com/StickyMinds).



### 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](mailto:cburke@sqe.com).



# PNSQC 2008

## COLLABORATIVE QUALITY

### 26TH ANNUAL PACIFIC NW SOFTWARE QUALITY CONFERENCE

October 13–15, 2008

REGISTER EARLY AND SAVE		AFTER 9/8
3-Day Conference	\$950	\$1100
2-Day Technical Program	\$625	\$750
1-Day Workshop	\$475	\$550

Special discounts available.

#### ABOUT PNSQC

Now in its twenty-sixth year, PNSQC delivers business-critical education, training, and information to help you improve the quality of your company's software.

#### KEYNOTE SPEAKERS

**The Art of Building Consensus**  
Sam Kaner

**Quality Dynamics of Agile SW Development**  
Ron Jeffries & Chet Hendrickson

#### INVITED SPEAKERS

**Playing Nice in the Sandbox**  
Janet Gregory

**Selling Your Ideas to Management**  
Steve Smith

**It's Not Just an Update: Using Status Reporting to Expand Collaboration**  
Mike Kelly

**Expanding Trust and Collaboration with Earned Value Tracking**  
Tamara Sulaiman & Hubert Smits

#### WHO SHOULD ATTEND

Software managers, testers, and developers interested in building better software through best practices, quality design, or innovative management techniques in software development and testing processes.

#### FULL-DAY WORKSHOPS

**W1. Agile Planning - The Product Development Game**  
Tamara Sulaiman & Hubert Smits

**W2. SQL for Testers**  
Karen N. Johnson

**W3. Collaboration using the Agile Testing Taxonomy**  
Janet Gregory

**W4. Zeroing in on the Right Problem**  
Steve Smith

#### TECHNICAL PAPER TOPICS

A sample of the refereed technical papers topics scheduled for presentation at PNSQC 2008:

- Acceptance Testing: A Love Story in Two Acts
- Agile Retrospectives: Collaboration for Continuous Improvement
- Collaboration! — Guerilla Techniques
- “Ideastreaming”: Harnessing the Power of Collaborative Innovation Software
- Outnumbered: Ensuring Quality with a Low Test-To-Dev Ratio
- Software Testing As a Service (STaaS)
- Storytelling Techniques: Reporting Product Status in a Meaningful Way
- Tails in the Boardroom: Canine Lessons for Business Teams
- The TAO of Software Defect Test Collaboration and Estimation
- Trust: The Key to Project Team Collaboration

To register or for more information visit [www.pnsqc.org](http://www.pnsqc.org)



# 10 Things You Might Not Know About

## Getting the Most Value from Consultants

by Brian Mizelle & Paco Hope

- 1** **CONSULTANTS DO JUST WHAT YOU ASK.** Understand what you are and are not getting up front in terms of deliverables. Make sure statements of work (SOWs) are clear and well understood by both parties. Get sample deliverables to see if they will meet your needs, and don't be afraid to speak up if "cookie cutter" deliverables do not meet your objectives. *Value: Get what you need, not what you are sold.*
- 2** **LITTLE THINGS ERODE PRODUCTIVITY AND PUT BUDGETS AT RISK.** Work with the consultants and your own people well before the start date to take care of logistics. What do they need from you on day one? Badges? Work space? Documents? Network access? Appointments with resources? Be sure this "upfront" coordination time is unbillable (in most cases it should be). *Value: You want consultants working for you every minute the clock is ticking. Make the onboarding process as smooth as possible.*
- 3** **YOU SHOULD EXPECT TO SEE PROGRESS THROUGHOUT THE ENGAGEMENT.** Don't wait until the end of the project to see your deliverables. Work with your vendor to see "work in progress" to avoid misunderstandings. Establish a schedule for regular updates and make time to review, ask questions, and provide comments. *Value: Don't wait until the end to realize you are not getting what you expected.*
- 4** **YOU MIGHT BE PAYING FOR THINGS YOU DON'T SEE.** You don't want to be surprised by the invoice at the end of a project. Request updates along the way and see how you are doing against budget. There's going to be some management overhead, because the right amount adds value. Don't let your vendor bulk up the price with extra layers of management, though. If not kept in check, this can add an additional 10 percent or more to the cost of a project. *Value: Stay involved in monitoring your budget. Hold your vendor accountable.*
- 5** **YOU HAVE TO PUT TIME IN, TOO.** Be involved in the effort and provide timely feedback on progress reports and intermediate work products. It's easiest to keep a project on track by giving a lot of small course changes, instead of lurching from one big idea to the next. *Value: Be an active participant, not a passive observer.*
- 6** **FEATURE CREEP CAN HAPPEN WITH CONSULTANTS, TOO.** As in any endeavor, a little tinkering with scope can have a big impact on the final result. Get in the habit of asking, "Is this still within scope of our original agreement?" Document in writing any changes that differ from the original SOW. Make sure you're achieving your original goals or achieving goals that have evolved during the course of the project. *Value: Be diligent in your change documentation and understand budget impact.*
- 7** **THE LAWYERS HAVE TO BE HAPPY.** Most organizations have internal processes for establishing a master services agreement when you have never worked with a firm before. This process could delay your start date significantly depending on how quickly the legal folks can get done. Understand how the company will be handling your IP and be clear on your expectations (get a written policy on the handling of IP both during and at the end of a project). *Value: Anticipate delays that might be introduced elsewhere in your organization.*
- 8** **LONGER-TERM WORK IS USUALLY CHEAPER.** If you already have similar projects planned and have been getting favorable feedback on the work in progress, consider keeping the team on to do more, similar projects. If you keep the team, you keep its knowledge of your business and that makes the team more effective and able to do more in less time. *Value: Negotiate longer-term rate structures. Onboarding a (good) consultant is an investment that pays off over time.*
- 9** **TRUST IS IMPORTANT.** If you find good experts, stick with them and find ways to make your budget work. Although price is an important factor, consider the results you are getting. When you go to the trouble of getting an expert, get one that you'll be glad you hired. *Value: Reduce your risk.*
- 10** **YOU CAN VOTE WITH YOUR FEET.** Stay with people who do good work, but pay attention to those regular status reports. If you see staff changing a lot or you're less satisfied with the work, maybe it's time to find another vendor. At a minimum, be sure to make your vendors aware of declining quality and hold them to the standards originally set. *Value: Be prepared to shop if necessary.*

# Keys to Top-Notch Estimates

by Howard Smallowitz and George Stark

If the construction industry estimated projects as poorly as the IT industry does, we would still be living in mud huts. Yet inaccurate project estimates have become the norm in the IT industry. Consider the following facts:

- According to the Standish Group, only 29 percent of all projects are successful [1].
- In one study of 250 software development projects, 175 had major schedule delays and cost overruns, or were terminated without completion [2].
- Several white papers have identified inaccurate project estimates as one of the top causes of troubled projects [3, 4, 5].

Sometimes estimates amount to nothing more than gut feelings. Project managers try to focus on trying to calculate a project down to the last hour of effort or budget dollar. Rarely is either approach successful. With the benefit of painful experience behind us, we believe that the following ten keys can actually turn project estimates into reasonable predictions of project performance.

*Find something that you can count that is a meaningful measure of the project scope.* The literature is full of proposals: lines of code, function points, raised floor space, use case points, story points, etc. Just make sure that whatever you count is highly correlated to the amount of work to be done and can help to normalize the quality.

*Remember: Targets are not estimates.* This is a hard one for many managers to understand. Just because someone would like the scope delivered by the end of the quarter, doesn't mean it is possible.

*Use multiple techniques for computing the estimate and involve independent peer reviews.* There are a lot of tools and approaches to estimating—top-down, bottom-up, spreadsheets, tools, heuristics, Monte Carlo simulation, etc. They all give different answers, but if

you compare them and they are all “close,” you probably have a good estimate. Let other people review the approach and the outcome in a formal peer review to ensure that no major mistakes were made. We have found that a second set of eyes has uncovered potentially disastrous mistakes.

*Learn from the past; don't forget it.* Collect historical data on completed projects and then actually use it when making estimates for new projects. Without historical data, variation in processes, tools, and people cannot be visualized. Understand the impact of quality level, systems engineering, and management overhead on the estimate.

*Treat estimation as a process and take steps to make it mature.* Perform estimation maturity assessments, identify gaps, and take steps to improve. Measure estimation accuracy. Reduce common cause variation sources as much as possible.

*Iterate estimates.* The earlier you are in a project, the stupider you are. So don't expect your first off-the-cuff estimate to be spot on. Revise your estimates as you learn more.

*Always include boundaries that define a range of possible results.* The Project Management Institute recommends that initial estimates, sometimes called order-of-magnitude estimates, should have an accuracy range of -25 percent to +75 percent. Budget estimates, which often are used just to get initial funding and project approval, should be defined as having an accuracy range of -10 percent to +25 percent. Even final, definitive estimates, prepared from well-defined, detailed data should specify a range of -5 percent to +10 percent [6, 7].

*Avoid presenting estimates that promise levels of specificity you cannot deliver.* For example, early in a project it's impossible to predict that the work

“The earlier you are in a project, the stupider you are. So don't expect your first off-the-cuff estimate to be spot on.”

can be delivered in precisely fifty-eight calendar days for exactly \$123,956. You'll avoid trouble if you round up such estimates to two months and \$125,000.

*Make sure your estimate is complete.* We don't just mean that it should include all necessary work items and people. We mean that you must estimate ev-

everything that needs to be estimated. Too many people think a project is successful if it delivers on time and on budget. However, it is easy to meet a schedule or cost target by descoping work or delivering garbage. Estimates should be clear about the scope of the work to be delivered and the expected quality.

*Document, document, document.* Be sure to document the assumptions and constraints that went into your estimate and include them with the estimate. It's easier than polishing up your résumé.

{end}

## REFERENCES:

- 1] *PMI Today*. September 2005.
- 2] Jones, Capers. “Software Project Management Practices: Failure Versus Success.” *CrossTalk*. Software Technology Support Center, October 2004.
- 3] “Troubled Projects.” CBP Research News. [www.pmsolutions.com/uploads/pdfs/Troubled%20Projects%20Research%20Summary.pdf](http://www.pmsolutions.com/uploads/pdfs/Troubled%20Projects%20Research%20Summary.pdf)
- 4] “Major Causes of Software Project Failures.” *CrossTalk*. Software Technology Support Center, July 1998. [www.stsc.hill.af.mil/crosstalk/1998/07/causes.asp](http://www.stsc.hill.af.mil/crosstalk/1998/07/causes.asp)
- 5] “Troubled Project Telltales” (White Paper). [www.advalueservices.com/Advalue\\_White\\_Paper\\_Telltales\\_Troubled\\_Projects.pdf](http://www.advalueservices.com/Advalue_White_Paper_Telltales_Troubled_Projects.pdf)
- 6] *A Guide to the Project Management Body of Knowledge (PMBOK Guide)*, third edition. Project Management Institute, 2004.
- 7] *Project Management Professional Study Guide*. Osborne McGraw-Hill, 2003.



# Better Software Has Gone Digital!

**R**ead your favorite magazine straight from your desktop before it hits the street—check out the digital edition of this issue online today at [www.StickyMinds.com/September](http://www.StickyMinds.com/September)

Want to switch to the digital edition? Contact our subscription department at 1-800-450-7854 or [info@BetterSoftware.com](mailto:info@BetterSoftware.com) to find out how.



## Index to Advertisers

Agile Development Practices 2008	<a href="http://www.sqe.com/agiledevpractices">www.sqe.com/agiledevpractices</a>	13
Better Software magazine	<a href="http://www.BetterSoftware.com">www.BetterSoftware.com</a>	48
Blackbaud, Inc	<a href="http://www.blackbaud.com">www.blackbaud.com</a>	21
Hewlett-Packard	<a href="http://www.hp.com/go/software">www.hp.com/go/software</a>	Back Cover
IBM	<a href="http://www.ibm.com/rational">www.ibm.com/rational</a>	Inside Front Cover
Mountain Goat Software	<a href="http://www.mountaingoatsoftware.com/better">www.mountaingoatsoftware.com/better</a>	23
Pragmatic Software	<a href="http://www.softwareplanner.com">www.softwareplanner.com</a>	18
PNSQC	<a href="http://www.pnsqc.org">www.pnsqc.org</a>	45
QA Infotech	<a href="http://www.QAInfotech.com">www.QAInfotech.com</a>	9
Rally Software	<a href="http://www.rallydev.com/bsm">www.rallydev.com/bsm</a>	42
Rally Software	<a href="http://www.rallydev.com">www.rallydev.com</a>	Back Inside Cover
Seapine	<a href="http://www.seapine.com">www.seapine.com</a>	8
SQE Training	<a href="http://www.SQETraining.com/engineering">www.SQETraining.com/engineering</a>	41
STARWEST 2008	<a href="http://www.sqe.com/STARWEST">www.sqe.com/STARWEST</a>	5
Stevens Institute of Technology	<a href="http://www.stevens.edu/sse">www.stevens.edu/sse</a>	43
StickyMinds.com	<a href="http://www.StickyMinds.com/Podcasts">www.StickyMinds.com/Podcasts</a>	19
TechExcel	<a href="http://www.techexcel.com">www.techexcel.com</a>	2
Web Performance, Inc	<a href="http://www.webperformance.com">www.webperformance.com</a>	14

### Display Advertising Shae Young [syoung@sqe.com](mailto:syoung@sqe.com)

### All Other Inquiries [info@bettersoftware.com](mailto: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](mailto: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](mailto:info@bettersoftware.com).



# 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 Rally customer BMC Software  
"As compared to industry averages, BMC achieved nearly three times faster time to market, 20-50% improvements in individual team productivity, and one quarter the expected number of defects based on team sizes and schedules."



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..."



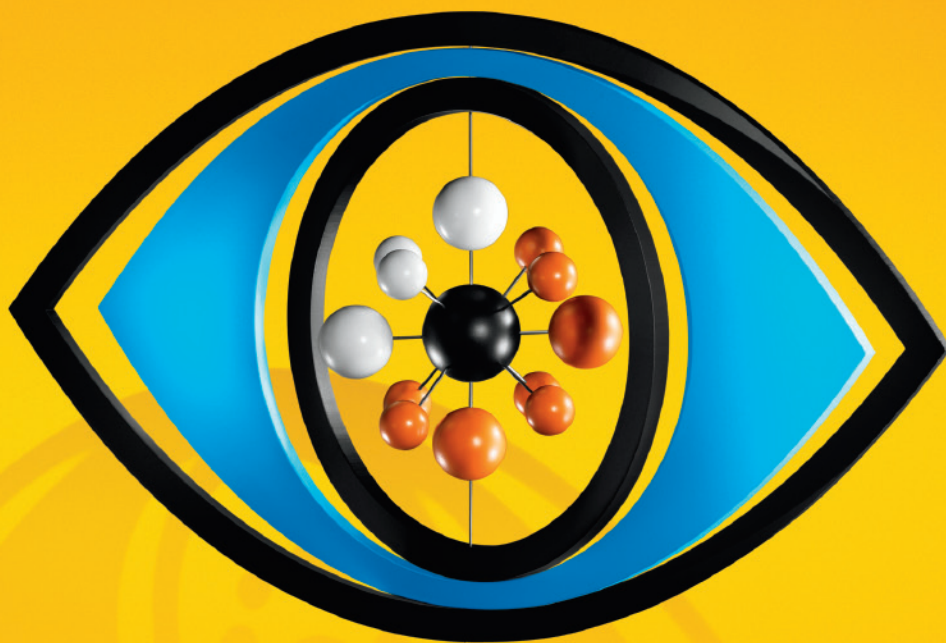
Get your FREE trial of Rally Enterprise at

[www.rallydev.com/adp](http://www.rallydev.com/adp)

No download, no installation, no commitment



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



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](http://hp.com/go/quality)





## 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:



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



\*Based on the 2007 conference attendee evaluations



**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**



*Shingle Creek Resort*

**Register Early and Save \$200!**  
**[www.sqe.com/agiledevpractices](http://www.sqe.com/agiledevpractices)**



**November 10-14, 2008**

**Orlando, FL**

**Shingle Creek Resort**

[www.sqe.com/agiledevpractices](http://www.sqe.com/agiledevpractices)

Conference Sponsor:



## 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 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](http://www.sqe.com)



*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](http://www.BetterSoftware.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](http://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, 6<sup>th</sup> Sense Analytics

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

## WEDNESDAY, NOVEMBER 12

8:30	<b>Opening Remarks</b> — <i>Conference Chair - Lee Copeland, Software Quality Engineering</i>							
8:45	<b>Seven Years Later: What the Agile Manifesto Left Out</b> — <i>Brian Marick, Exemplar Consulting</i>							
9:45	Morning Break							
10:00	<b>W 1</b> <b>Overcoming the Pitfalls of Transitioning to Agile</b> Johanna Rothman, Rothman Consulting Group	<b>W 2</b> <b>Are We There Yet? Defining "Done"</b> Mitch Lacey, Mitch Lacey & Associates, Inc.	<b>W 3</b> <b>From Concept to Product Backlog: What Happens Before Iteration Zero</b> Gerard Meszaros, Solution Frameworks, Inc.	<b>W 4</b> <b>Test-Driven Everything</b> David Hussman, DevJam	<b>W 5</b> <b>What Are They Doing Down There? A CIO's Perspective on Agile Software Development</b> Niel Nickolaissen, Headwaters, Inc.	<b>W 6</b> <b>Integrating Enterprise SOA Architecture with Scrum Development Methodology</b> Steven Driver, Airlines Reporting Corporation	<b>W 7</b> <b>Agile for Business Analysts</b> 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	<b>W 8</b> <b>The Agile PMP: Teaching an Old Dog New Tricks</b> Mike Cottmeyer, VersionOne	<b>W 9</b> <b>Agile Contracting</b> Rachel Weston, Rally Software Development and Chris Spagnuolo, Data Transfer Solutions	<b>W 10</b> <b>Re-thinking Scheduling: Parkinson's Law Inverted</b> Mary Poppendieck, Poppendieck, LLC	<b>W 11</b> <b>Pragmatic Agility</b> Andy Hunt, The Pragmatic Programmers	<b>W 12</b> <b>It Takes a Village: Organizing to Fulfill the Product Owner Role</b> Ronica Roth, Rally Software Development	<b>W 13</b> <b>Refactoring: Where Do I Start?</b> J. B. Rainsberger, Independent Consultant	<b>W 14</b> <b>Driving User Stories from Business Value</b> Guy Beaver, Net Objectives	
2:15	Networking Break in the EXPO							
2:45	<b>W 15</b> <b>Driving Agile Transformation from the Top Down</b> Pete Morowski, Borland Software Corporation	<b>W 16</b> <b>Calling All Agile Skeptics—the Curious and Die-Hard, Non-Agile</b> Damon Poole, AccuRev	<b>W 17</b> <b>Agile Project Metrics</b> Dave Nicolette, Valtech Technologies	<b>W 18</b> <b>Secrets of CMMI® for Agile Organizations</b> Jeff Dalton, Broadword Solutions	<b>W 19</b> <b>Value Stream Mapping: Extending Our View to the Enterprise</b> Alan Shalloway, Net Objectives	<b>W 20</b> <b>Behavior-Driven Database Design</b> Pramod Sadalage, ThoughtWorks	<b>W 21</b> <b>Do the Right Thing: Adapting Requirements Practices for Agile Projects</b> Ellen Gottesdiener, EBG Consulting	
4:15	Networking Break in the EXPO (EXPO open 4:00 p.m. - 6:30 p.m.)							
4:30	<b>Beyond Best Practices: Keeping Agile Agile</b> — <i>Dan North, ThoughtWorks</i>							
5:30	Reception in the EXPO, 5:30 p.m. - 6:30 p.m.							

## THURSDAY, NOVEMBER 13

8:30	<b>Opening Remarks</b> — <i>Conference Chair - Lee Copeland, Software Quality Engineering</i>							
8:45	<b>Collaborative Leadership: A Secret to Agile Success</b> — <i>Pollyanna Pixton, Accelinnova</i>							
9:45	Networking Break in the EXPO (EXPO open 9:30 a.m. - 2:45 p.m.)							
10:00	<b>T 1</b> <b>Who Do You Trust? Beware of Your Brain</b> Linda Rising, Independent Consultant	<b>T 2</b> <b>Selling Agile: Getting Buy-In from Your Team, Customers, and Managers</b> Michele Sliger, Sliger Consulting, Inc.	<b>T 3</b> <b>When to Step Up, When to Step Back: How to Lead Collaboration</b> Pollyanna Pixton, Accelinnova	<b>T 4</b> <b>Making People and Processes Congruent</b> Ken Pugh, Net Objectives	<b>T 5</b> <b>Agile Software Testing Strategies</b> Jared Richardson, 6 <sup>th</sup> Sense Analytics	<b>T 6</b> <b>Test-Driven Development Takes on Embedded Software</b> James Grenning, Renaissance Software Consulting	<b>T 7</b> <b>Assessing Your Agility</b> 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	<b>T 8</b> <b>"With Great Power Comes Great Responsibility"—Empowering the Agile Team</b> V. Lee Henson, VersionOne	<b>T 9</b> <b>A Lean Approach to Managing the Project Portfolio</b> Johanna Rothman, Rothman Consulting Group	<b>T 10</b> <b>Maximizing Team Dynamics and Overcoming Dysfunction in Agile Environments</b> Michael Mah, QSM Associates, Inc.	<b>T 11</b> <b>The Business Value of Pair Programming</b> Rob Myers, Net Objectives	<b>T 12</b> <b>Agile Usability Testing</b> John De Goes, N-BRAIN, Inc.	<b>T 13</b> <b>Getting Agile with Legacy Code</b> Steve Berczuk, Cyrus Innovation	<b>T 14</b> <b>Agile Project Inception: Escaping the Waterfall</b> Kenny Rubin, Innolution	
2:15	Networking Break in the EXPO (EXPO open until 2:45 p.m.)							
2:45	<b>T 15</b> <b>Mistakes Agile Teams Make</b> J. B. Rainsberger, Independent Consultant	<b>T 16</b> <b>Agile Growing Pains</b> Michael Kirby, Xerox	<b>T 17</b> <b>Scaling Agile Up and Out: A Tale from the Trenches</b> Ade Miller, Microsoft Corporation	<b>T 18</b> <b>Retrospectives in Action: Looking Back at the Conference</b> Jean Tabaka, Rally Software Development	<b>T 19</b> <b>Picking the Right Test Automation Strategy for Your Project</b> Gerard Meszaros, Solution Frameworks, Inc.	<b>T 20</b> <b>Agile Engineering for Architects</b> Ryan Shriver, Dominion Digital	<b>T 21</b> <b>Introduction to Multi-Stage Continuous Integration</b> Damon Poole, AccuRev	
4:15	Networking Break							
4:30	<b>Scaling Agile: Kanban and Beyond...</b> — <i>David Anderson, Valtech, Inc. and Modus Cooperandi, Inc.</i>							

## 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](http://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](http://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](http://www.BetterSoftware.com)

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





# CONFERENCE SPEAKERS

**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 Marciano**  
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**  
Innovation



**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**  
Broadword  
Solutions



**T C**  
**Michael Mah**  
QSM Associates,  
Inc.



**C**  
**Rachel Weston**  
Rally Software  
Development

### 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](mailto: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.** 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 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](mailto: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](mailto:ken.pugh@netobjectives.com).

### MF Testing on Agile Projects NEW

Antony Marciano, [testingReflections.com](http://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 Marciano 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 Marciano** 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 specialties 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.marciano@testingReflections.com](mailto:antony.marciano@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](mailto:rachel@agilexp.com).



## 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](mailto: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](http://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](mailto: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](http://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](http://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](http://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](mailto:ppixton@accelinnova.com).



### 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](mailto:mike@mountaingoatsoftware.com).

### MN Agile Product Planning: Building a Strong Backlog NEW

**David Hussman, DevJam**

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](http://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](mailto:jean.tabaka@rallydev.com).

### MP Refactoring Your Wetware: Thinking about Thinking NEW

**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 "wannabe." 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 NEW

**J. B. Rainsberger, Independent Consultant**

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](http://jbrains.ca), in his IEEE Software magazine column "Not Just Coding", at conferences worldwide, or by writing him at [get.started@jbrains.ca](mailto:get.started@jbrains.ca).

### 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](http://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](http://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](mailto: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](http://www.devjam.com)) provides seasoned leaders that strive to pragmatically match technology, people, and processes to create better and cooler products.



**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](mailto: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](mailto: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 xTreme 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](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>.

**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 xTreme Programming to traditional waterfall development. He is an expert in implementing and customizing agile methods, such as Scrum and xTreme 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.



**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.



**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](http://jbrains.ca), in his IEEE Software magazine column "Not Just Coding", at conferences worldwide, or by writing him at [get.started@jbrains.ca](mailto:get.started@jbrains.ca).

**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.**

**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](http://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](mailto: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](http://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](mailto: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](http://www.AgileArtisans.com) and [www.6sa.com/blog](http://www.6sa.com/blog).



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

### Seven Years Later: What the Agile Manifesto Left Out

**Brian Marick, Exemplar 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@exemplar.com](mailto:marick@exemplar.com), [www.exemplar.com](http://www.exemplar.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](mailto: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](mailto: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 Nickolaissen, 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 Nickolaissen 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.



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, 6<sup>th</sup> 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, *Innolusion*

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

THURSDAY, NOVEMBER 13, 2:45 p.m.

**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 Nickolaissen, 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 called 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 Celler, 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.

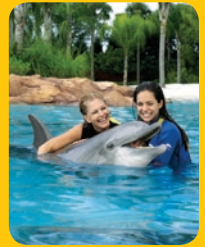
Capitalizing on Rosen Shingle Creek's elevation, every single room offers a stunning view of the golf course and the area's incomparable setting—from the perfectly manicured fairways and greens to the picturesque creek, magnificent cypress trees, and lush natural vegetation. For vacationers or meeting 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](http://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.



## The Award-Winning-Est Agile Lifecycle Management Solution

Get your FREE trial of Rally Enterprise at

[www.rallydev.com](http://www.rallydev.com)

No download, no installation  
No commitment



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

Two-time SD Times 100 award winner for tools  
"that made December 2007 a far more productive and efficient time to code than January 2007."



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..."

## 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](http://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](http://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](http://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](http://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](http://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](http://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](http://www.sqe.com/agiledevpractices)

# AGILE DEVELOPMENT PRACTICES REGISTRATION

NOVEMBER 10-14, 2008 ORLANDO, FLORIDA, USA

## Easy to Register



**ONLINE:**  
[www.sqe.com/adpreg](http://www.sqe.com/adpreg)



**PHONE:**  
888.268.8770  
904.278.0524



**EMAIL:**  
[sqeinfo@sqe.com](mailto:sqeinfo@sqe.com)

## CONFERENCE PRICING

### Registration Fees:\*

	On or Before Oct. 10	After Oct. 10	
<input type="checkbox"/> <b>VIP Package—All 5 Days</b> Includes: 2 days of Pre-conference Tutorials, 2 days of the Agile Development Practices Conference, 1 day of the Agile Leadership Summit	<b>\$2,545</b>	<b>\$2,745</b>	<b>Best Value!</b>
<input type="checkbox"/> Conference + Two Pre-conference Tutorial Days	\$2,295	\$2,495	
<input type="checkbox"/> Conference + One Pre-conference Tutorial Day	\$1,995	\$2,195	
<input type="checkbox"/> Conference Only (Wed. - Thurs.)	\$1,695	\$1,895	
<input type="checkbox"/> Two Pre-conference Tutorial Days	\$1,390	\$1,590	
<input type="checkbox"/> One Pre-conference Tutorial Day	\$795	\$895	
<input type="checkbox"/> Agile Leadership Summit	\$795	\$895	
<input type="checkbox"/> 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](http://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](http://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](http://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](mailto: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



**November 10-14, 2008**

Orlando, FL • Shingle Creek Resort

Conference Sponsor:



[www.sqe.com/agiledevpractices](http://www.sqe.com/agiledevpractices) REGISTER EARLY AND SAVE \$200!

### Industry Sponsors:



### In Cooperation With:



### Media Sponsors:



CONFERENCES

330 Corporate Way, Suite 300  
Orange Park, FL 32073

**IF ADDRESSEE IS NO LONGER EMPLOYED:**

Re-route to Director of Software Development