

BETTER™ SOFTWARE

A TECHWELL™ PUBLICATION

MIND MAPS
Organize your testing strategy

REVERSE MENTORING
Should younger workers be the ones mentoring others?



Configuration Management

THE ULTIMATE CONDUCTOR *In* THE PRODUCT LIFECYCLE



CONTENTS



14



18

in every issue

- Mark Your Calendar **4**
- Editor's Note **5**
- Contributors **6**
- Interview with an Expert **13**
- TechWell Spotlight **17**
- Product Announcements **29**
- FAQ **30**
- Ad Index **32**

Better Software magazine 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 at BetterSoftware.com or call 904.278.0524.

features

- 14 COVER STORY**
CONFIGURATION MANAGEMENT: THE ULTIMATE CONDUCTOR IN THE PRODUCT LIFECYCLE
When you think of configuration management, build automation and version control usually come to mind. Dave presents a perspective that shows the important role CM plays in the entire product and project lifecycle.
by Dave Lyon
- 18 BUILDING A SOLID SCRUM FOUNDATION REQUIRES CLEAR ROLES**
Without clearly defined roles and responsibilities, actions taken by key project stakeholders may result in project misfires. Kyle shows what you should do to avoid the situation when roles are misunderstood, overlapped, or completely missing.
by Kyle Roozen
- 22 REVERSE MENTORING: SHOULD YOUNGER WORKERS BE MENTORING YOUR EXECUTIVES?**
What happens when younger knowledge workers, the millennials, bring a new perspective to an organization? Reverse mentoring can dramatically improve employee retention, team collaboration, and the adoption of newer technology.
by Lew Sauder
- 26 FEELING LOST IN THE WOODS? MIND MAPS CAN HELP!**
Claire takes us on a nontraditional journey where designing and implementing testing approaches can be rapidly organized into a hierarchy of connected elements. Mind maps, used primarily for visual and conceptual thinking, may be just the answer for quality assurance professionals.
by Claire Moss

columns

- 7 TECHNICALLY SPEAKING**
UNDERSTANDING WHOLE TEAM TESTING
by Matt Heusser
Whole team testing makes product quality everyone's business. It can also make people uncomfortable. Matt explains how this new way to approach project quality helps with leading retrospectives, conducting defect analysis, and mitigating project risks.
- 31 CAREER DEVELOPMENT**
YOU CAN'T BE JUST A MANAGER ANYMORE
by Gunasekaran Veerapillai
It used to be that a project manager did one thing: manage the success of the project. As IT budgets shrink and job responsibilities expand, there is no such thing as a typical project manager role. You're expected to wear many hats, facilitate human resource issues, become a subject matter expert, and assist with key technical activities.

MARK YOUR CALENDAR

training weeks

<http://sqetraining.com/trainingweek>

Testing Training Week
March 24–28, 2014
Boston, MA

May 12–16, 2014
San Diego, CA

June 9–13, 2014
Chicago, IL

**Agile Software Development
Training**
June 1–3, 2014
Las Vegas, NV

software tester certification

<http://sqetraining.com/certification>

Foundation Level Certification
February 25–27, 2014
Baltimore, MD

March 4–6, 2014
San Francisco, CA
Philadelphia, PA

March 11–13, 2014
Cincinnati, OH
Denver, CO

Advanced Tester Certification
March 3–7, 2014
San Francisco, CA

April 28–May 2, 2014
Atlanta, GA

conferences

STARCANADA
<http://starcanada.techwell.com>
April 5–9, 2014
Toronto, ON
Hilton Toronto

STAREAST
<http://stareast.techwell.com>
May 4–9, 2014
Orlando, FL
Rosen Centre Hotel

Agile Development Conference West
<http://adcwest.techwell.com>
June 1–6, 2014
Las Vegas, NV
Caesars Palace

Better Software Conference West
<http://bscwest.techwell.com>
June 1–6, 2014
Las Vegas, NV
Caesars Palace

STARWEST
<http://starwest.techwell.com>
October 12–17, 2014
Anaheim, CA
Disneyland Hotel

Agile Development Conference East
<http://adceast.techwell.com>
November 9–14, 2014
Orlando, FL
Walt Disney World Dolphin

Better Software Conference East
<http://bsceast.techwell.com>
November 9–14, 2014
Orlando, FL
Walt Disney World Dolphin

Publisher
Software Quality Engineering Inc.

President/CEO
Wayne Middleton

Director of Publishing
Heather Shanholtzer

Editorial

Better Software Editor
Ken Whitaker

Online Editors
Cameron Philipp-Edmonds
Beth Romanik
Jonathan Vanian

Production Coordinator
Donna Handforth

Design

Creative Director
Catherine J. Clinger

Advertising

Sales Consultants
Daryll Paiva
Kim Trott

Sales Coordinator
Minda Crosby



CONTACT US
Editors: editors@bettersoftware.com

Subscriber Services:
info@bettersoftware.com

Phone: 904.278.0524, 888.268.8770

Fax: 904.278.4380

Address:

Better Software magazine
Software Quality Engineering, Inc.
340 Corporate Way, Suite 300
Orange Park, FL 32073



A NEW YEAR, A NEW FOCUS

Welcome to our first 2014 issue of *Better Software* magazine!

This promises to be an amazing year as the newly proclaimed Internet of Things blurs the lines between enterprise computing, mobile, and embedded devices. *Better Software* magazine is committed to broadening our topics this year, starting with this issue. If you weren't sure of the importance of configuration management in your project lifecycle, Dave Lyon's article should convert you. I know from experience that without a strong commitment to configuration management tools, it is almost impossible to keep up with the frequent plan-do-study-act iterations required for agile projects.

Claire Moss dispels the notion that mind maps are just for keeping thoughts and concepts organized (like a networked notepad). In fact, she illustrates how mind maps can be used for testing activities that take place during a project lifecycle.

I can't tell you how many times I've seen teams break down because roles aren't clearly identified at the beginning of a project. Kyle Roozen does a masterful job writing about the importance of identifying roles for every Scrum project.

Contrary to my experience, Lew Sauder suggests that we should consider younger staff members as mentors to seasoned, more experienced individuals on a team. Could this be a function of social media's influencing a wide range of collaboration? You'll need to read Lew's article to find out.

This is my third issue as editor of *Better Software* magazine, and I'm pleased that the awareness of *Better Software* is increasing. But timely, impactful information isn't just available in the magazine. *Better Software* magazine enjoys a strong relationship with our community sites: StickyMinds.com, AgileConnection.com, and CMCrossroads.com. These sites are continually being updated with articles and blogs about agile, project management, testing, configuration management, and other topics.

Our digital magazine is pushed bimonthly to subscribers worldwide. I thought I'd share with you how easy it is to view *Better Software* magazine on a tablet using Dropbox and an iPad:

1. On your PC or Mac, view the *Better Software* magazine issue in your browser and save it as a PDF in a Dropbox folder.
2. On your iPad, tap the Dropbox app and select the PDF located in the Dropbox folder.
3. Tap the share icon on the Dropbox toolbar and tap Open In to open the PDF in iBooks.

And, by the way, if you're interested in writing, don't hesitate to let us know by sending an email to editors@sqe.com.

A handwritten signature in black ink that reads "Ken Whitaker".

Ken Whitaker

kwhitaker@sqe.com, Twitter: @Software_Maniac

Contributors



MATT HEUSSER manages accounts for Excelon Development while doing technology consulting. A former member of the board of directors of the Association for Software Testing, Matt organized the workshop on technical debt, test coach camp, TestRetreat 2013, and, most recently, the Workshop on Self-Education in Software Testing. Probably known best for his writing, Matt was the lead editor for *How to Reduce the Cost of Software Testing* and recently became technical editor for *StickyMinds.com*. Learn more about Matt at <http://www.xndev.com>, to find him on Twitter at [@mheusser](https://twitter.com/mheusser), or email him at matt@xndev.com.



DAVE LYON has worked in industry (United Technologies, General Electric, Lockheed Martin, and General Dynamics) for thirty-nine years, primarily in the fields of project engineering and configuration management. Dave has spent the past several years writing books and teaching CM seminars in North America and Europe. Dave is the author of *Practical CM III: Best Configuration Management Practices for the 21st Century*, *Transparent CM: How to Get There*, and *Practical Project: Guidelines for Project Engineers and Program Management Personnel*. Reach Dave at lyon@configuration.org.



In 2003, **CLAIRE MOSS** became the first discrete mathematics business graduate from Georgia Tech and immediately jumped into software testing. She has been following this calling ever since, working with agile product teams as a testing teacher, unit and integration test advisor, exploratory tester, and test automator. Her hobbies in recent years have been writing, speaking, and nerding about testing. You can always count on her to make bad puns and to appreciate yours. Claire has always had a passion for writing and continues to use her evil powers for good on the job and on her blog at <http://aclairefication.com>.



KYLE ROOZEN is a director with ACME Business Consulting in Portland, OR. He has more than twelve years of global project, product line, and quality management experience, with positions at GE Energy, GE Security, and Climax Portable Machine Tools. You can reach Kyle at kroozen@acmeconsulting.com.



ROB SABOURIN has more than thirty-three years of management experience leading teams of software development professionals. A well-respected member of the software engineering community, Rob has managed, trained, mentored, and coached hundreds of top professionals in the field. He frequently speaks at conferences and writes on software engineering, SQA, testing, management, and internationalization. The author of *I Am a Bug!*, the popular software testing children's book, Rob is an adjunct professor of software engineering at McGill University. Rob can be contacted at rsabourin@amibug.com.



LEW SAUDER is the coauthor of *The Reluctant Mentor: How Baby Boomers and Millennials Can Mentor Each Other in the Modern Workplace* and author of *Consulting 101: 101 Tips for Success in Consulting*. Lew has been a consultant with top-tier and boutique consulting firms for twenty years. He is currently a senior project manager with Geneca, a custom software development firm based in Oakbrook Terrace, IL. Lew can be reached at lewis.sauder@geneca.com.



JONATHAN VANIAN is an online editor who edits, writes, interviews, and helps turn the many cranks at the websites of *StickyMinds.com*, *TechWell.com*, *AgileConnection.com*, and *CMCrossroads.com*. He has worked for newspapers, websites, and a magazine and is not as scared of the demise of the written word as others may appear to be. Software and high technology never cease to amaze him. Jonathan never sleeps, so send him an email day or night at jvanian@sqa.com.



GUNASEKARAN VEERAPILLAI heads the testing competency team at Wipro Technologies. He has more than thirty years of experience in retail banking, EDP, test management, and test automation. Guna is a certified software quality analyst (CSQA) from QAI and Project Management Professional (PMP). He contributes to web journals and has presented papers at international software quality conferences. Guna is the coauthor (with Bill Lewis) of *Software Testing and Continuous Quality Improvement*. He can be reached at gunasekaran.veerapillai@wipro.com.

Understanding Whole Team Testing

Testing used to be focused on just verifying that defects are found and fixed prior to product release. There's more to testing than meets the eye.

by **Matt Heusser** | matt@xndev.com

Every now and again there will be a discussion on the Agile-Testing Yahoo group that involves some amount of pain and frustration. [1] The interaction usually goes something like this:

Q: I am testing a credit-card processing system. What should I do?

A: Get the whole team together and discuss your strategy.

Q: Sure, we can do that, but I'm looking for specific techniques.

A: The whole team can figure that out.

Q: I'm the tester and they want me to test. What should I do?

A: WHOLE TEAM. SELF-ORGANIZING TEAM.

So the discussion goes until one group or another gives up. They may be doing something like Scrum and have cards flowing across a wall, but something is fundamentally different in the way the person answering thinks about her work. To an outsider, the answers look like nonanswers.

As it turns out, the person answering the question—whom I might call an agile enthusiast—is trying to protect the questioner from a certain kind of trap: the trap of expertise.

THE TRADITIONAL TESTING TRAP

Let's pretend the conversation went differently and the person asking the original question actually got a real answer. That would mean someone on the discussion list external to the original company engages the tester in a conversation about what to test and how to test it. Together, the two create a test strategy. On balance, this is a good thing; the strategy is better than what the person asking the question would have ever come up with alone. You'll notice what is missing here: any involvement at all from the end customers, the system users, the programmers, or anyone else with a stake in the project.

Bob, the questioner, decides to use this strategy, which consists of some judgments about risk made by Bob and people from the email list. Without perfect knowledge up front, the strategy will be a bit like a weather forecast—mostly right but never quite perfect. Because it isn't perfect, it won't cover all the defects resulting in a code released to customers who experience defects. As you might expect, Bob will be called into the executive suite and asked, "How did these bugs slip through?"

Thus begins an activity known around the world as time-wasting finger-pointing.

Instead of giving the reader an answer, the response directs the reader to the entire delivery team. In this case, the whole technical team creates its own strategy. When a bug slips through to production, it is not that the tester failed. Instead, the entire team was responsible for the mistake. Instead of a single point of failure, we have a group of overlapping circles, all of which failed.

Consider a bug in this case: The programmers weren't aware of the risk or didn't mention it to anyone, and the product managers forgot to mention it as something important to verify. Everyone made the same mistake about the risk; maybe it was a reasonable mistake to make. When it comes time to make sure the category of error does not recur, it will be the responsibility of the whole team, with an eye on prevention and not yet another item added to a test checklist.

Having the right people in the room has other advantages. If the programmers are part of defining the test strategy, they have a much better chance of delivering something that passes on the first try, and they'll be less likely to argue that a test isn't valid, not a bug, and so on. Likewise, because they are involved in the strategy, the programmers will tend to build test automation hooks as part of the process, thus removing another pain point. The tests the group comes up with become part of the shared understanding of requirements by example.

“Whole team testing creates a great opportunity for excellent testers to become coaches, guides, and mentors, both in the workplace and in the marketplace of ideas.”

The most common way to do this is probably something called a Three Amigos meeting, [2] a term used by George Dinwiddie in his groundbreaking paper. It also could be known as a test case creation meeting or a story kickoff.

WHOLE TEAM RETROSPECTIVES

The last piece of whole team testing is that the team has to figure out how to improve itself as a project team and not as a test team. That means we have to get everyone together periodically to review what is working, what isn't, and what to do next. The team needs to take ownership of implementing that change. That means no department, no executives, and no managers own the process. Instead, the technical team should meet periodically in retrospectives to build consensus on what to change.

In my experience, retrospective changes are the most likely to stick because the team members define the changes themselves instead of having the changes forced upon them. Retrospectives also position change as an experiment, something people tend to be more open to than directives.

BUT I THINK MY TEAM IS PRETTY AGILE, AND WE DON'T DO THAT!

I choose to look at only the idea of whole team testing because I find it causes the most friction in communication. People that adopt whole-team testing are often focused on entirely different things. They trade focus on traditional test skills for focus on full-time conversations about risk with the entire team. The assumption is that if you get the right people

in a room, the team will come up with a good enough test plan most of the time.

Whole team testing creates a great opportunity for excellent testers to become coaches, guides, and mentors, both in the workplace and in the marketplace of ideas.

TOMORROW

While this article is focused on building a whole team test strategy, the newest emphasis I see is whole team test activities. If the team has everyone pair on all activities, then anyone can shift focus to the most critical testing bottleneck at any time.

Whole team testing might not work for everyone. There are certainly other visions for test, such as the outsider who looks at product and market risk. There may be less of these, though, and they'll have to be better in order to add value in a world that is increasingly technical and general.

If you work on an agile team, is your whole team really involved in testing? If so, how has that changed from what you were doing before? Are you witnessing improvements in testing approaches and decision-making? If you aren't in that boat, what does your boat look like, and where is it headed?

Is your team resistant to this concept, or is it just you? **{end}**

**Sticky
Notes**

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

■ References

Jon Hagar

Years in Industry: **34**

Email: embedded@ecentral.com

Interviewed by: **Jonathan Vanian**

Email: jvanian@sqe.com

“In embedded, a lot of times the user of the software isn't really even aware that they're interacting with software compared with the PC, where it's pretty obvious you're using software.”

“I don't want to tell stories too much, but the system actually had some bugs in it. We had some interesting conversations with the flight crew. It was a very nice airplane, but as with any new high-tech thing, there's things to be worked out.”

“

Everybody is creating an app. There's tens of thousands if not millions of apps out there. When I started looking at a lot of the mobile app smartphone software, it was pretty obvious to me it wasn't well tested.

”

“I followed Dr James Whittaker's work on attack-based testing. The idea of attack-based testing is that you not only check the required functionality of the software, but that you also want to try to show that something in the software does not work with attack patterns.”

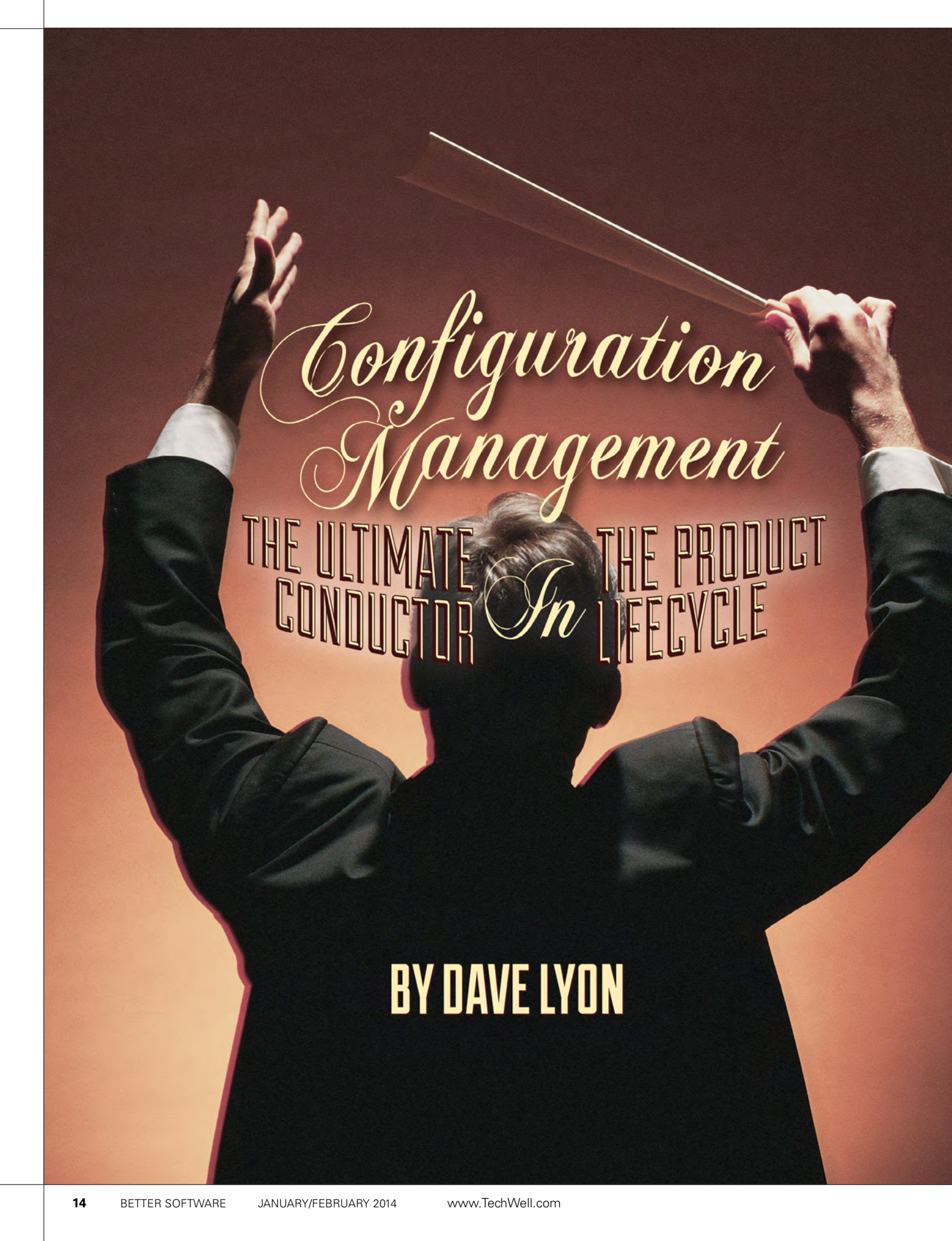
“In some of the mobile and embedded space, we see various horror stories—even lawsuits from systems not working correctly—so in my mind there's a fair amount of need to have just the right amount of testing.”

“As the users become more sophisticated and social media increases, I think a lot of people will start going, 'Gee, we need that right level of testing,' and, again, there's no one size fits all of that.”

“I'm not saying that every mobile smartphone app out there needs to be tested at the same level as, say, the space shuttle was tested, but you need enough testing early on so that when you get it out in the field—hopefully rapidly—that you don't get those bad reviews. Because, otherwise, you're going to probably not be around to have another go-round of changes.”

“We know from reports people have cracked into cars. We know people have cracked into pacemakers, another kind of embedded device. Again, a little bit of knowledge for me just makes me very scared. I've been studying this a fair amount, but I still have an awful lot to learn.”

For the full interview, visit
<https://well.tc/IWAE16-1>



Configuration Management

THE ULTIMATE CONDUCTOR *In* THE PRODUCT LIFECYCLE

BY DAVE LYON

Configuration management (CM) is like brushing your teeth—a nuisance, but a pretty good idea. Just ask any engineer what his first thoughts are when advised that a design change is needed: “Oh, my God! All that paperwork, all those meetings, and all the hassle! Why can’t I just fix it?”

This nuisance image of CM that so many of us have dealt with in the past has abated considerably as a result of the emergence of CM best practices, automated CM environments, and the need for reliable, dependable capture and control of product data.

When most folks hear the term *configuration management*, they usually think of change control. There is much more to CM than just change control.

A CM program is not the classic green-eyeshades occupation where rooms of squinty-eyed old men record data by hand in CM logbooks or enter data into out-of-date databases—when they are not carrying around forms to be signed and bothering engineers and other important personnel for technical clarification and approvals.

A sound CM program provides visibility throughout a project lifecycle that benefits functional managers, program and project managers, and your customer. An approved and documented CM plan addresses CM activities and includes a description and schedule of all inputs to and outputs from the CM system, including customer interaction with approval processes. CM also provides your company and customers with a way to monitor progress over your product’s entire lifecycle to ensure that contracted activities and events occur as expected.

Following CM best practices ensures that baselines are captured at design reviews, product design releases to manufacturing are made on time, and all necessary audits are conducted as planned. Ideally, an automated CM system should provide constant control and visibility into the progress of your design, development, integration, test, build, production, delivery, support, and maintenance activities to the team and management. A well-designed CM program will ensure that everyone is singing from the same sheet.

There are several project activities that can be tracked by CM throughout a product lifecycle.

Contracting: A sound CM program ensures compliance to a project’s contractual requirements. You will get both your business and your relationship with your customer off to a good start by tailoring CM activities to specific project needs coupled with your own best CM practices prior to negotiating this tailored CM program with your customer. This approach should eliminate unwelcome surprises as your program evolves. The use of a CM audit process will guarantee compliance to your negotiated requirements.

During the contracting process, CM staff should understand the customer’s requirements. In a misguided attempt to save money, proposals are often trimmed of CM-related activities even before the program or project begins. This practice is unacceptable in today’s business environment. CM should have a primary role in contract planning and execution.

CM personnel now attend proposal kickoff meetings and present the need for a CM best-practices program that identi-

fies inputs and outputs with the CM organization at specific events during the project’s lifecycle. In addition, a model for baseline capture and control and CM program details are introduced and negotiated.

Planning: Planning is closely linked to requirements tailoring. They are built on the same foundation, like the living room and kitchen are built on top of a home’s cellar.

If we think of the living room as the location where we invite the customer in to chat about how we are going to apply CM disciplines and processes to meet his requirements, then the kitchen is the place where we cook the meal by putting those processes to work after assembling the team of cooks.

Once the customer’s CM-related requirements are identified and understood, the next step is to plan and articulate what the CM organization sees as the most effective CM practices to apply to this specific program. It would be quite a coincidence if the results of this activity matched exactly the requirements articulated by the customer. Consequently, CM procedures should be tailored to meet both the customer needs and our own business requirements for program control and visibility.

CM personnel along with engineering personnel and program management personnel should then negotiate tailored best CM practices with the customer in order to achieve a resolution on future project issues. All of the project’s functional organizations should participate and agree to support the results of these negotiations. When final agreement is reached, the results of these negotiations should be documented in your CM plan and communicated throughout the organization.

Design and development: If it were up to engineers, they would work on their designs until they were thoroughly tested, updated, and validated before presenting them to the CM organization in order to establish baselines and subsequent configuration control activities. In fact, there’s a strong desire to postpone entering the project into CM control because each change usually points out the inadequacies of their design. Conversely, CM staff wants to put that design under CM control at the beginning of the project. This is when a tug-of-war usually ensues.

In today’s business environment, neither approach is acceptable or feasible. Shortened design cycles and early transition to production schedules require a more flexible yet controlled approach.

A consistent engineering design methodology, integrated with a CM mindset based on events that define the transition points from one level of control to another, will guarantee a good outcome. Figure 1 demonstrates an event-driven control system model where key project stakeholders know their roles and responsibilities and must understand what is expected of them at specific points in the evolution of the design.

The concept of planning engineering or CM milestones by calendar dates just doesn’t work well because schedules often slip. Event-driven milestones are much more effective. A CM-system methodology composed of multiple states can be tailored to fit any product, project, or program and support any design process or tool.

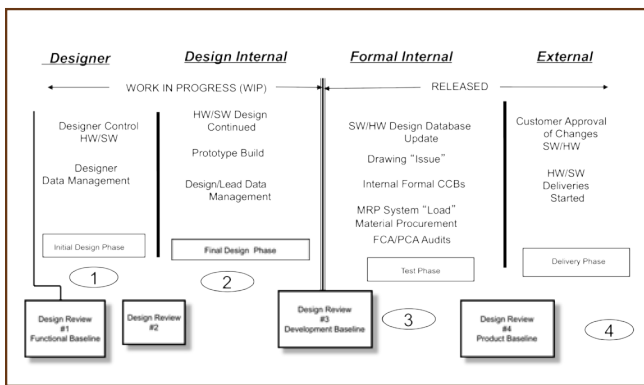


Figure 1: Key roles in a Scrum project

The design reviews shown in figure 1 not only represent baseline capture events and transition of control milestones, but they also provide an opportunity for CM personnel to present design data capture, control status, and checklists that identify required inputs and outputs to the CM system. They also present an excellent opportunity to remind folks of agreements made with the CM organization as documented in the CM plan.

CM personnel must put procedures and systems into place for controlling changes to baselines throughout the lifecycle of the product. They must establish best CM practices for creating and maintaining efficient change processes. They also must provide guidelines and procedures to ensure that changes to the design are incorporated into design documentation in a timely manner and into software programs and hardware products as planned. Obsolete or unacceptable software programs, hardware parts, and assemblies must be removed from service, whether to be upgraded or scrapped.

It's the job of CM personnel to verify the capture and deposit of all design data into the formal internal configuration control boards (CCB) repository at this point in the product lifecycle. The CCB is where the designs are transitioned from informal control to formal internal control (Design Review #3 in figure 1). CM personnel must verify the capture of all design metadata, files, drawings, and any related data at this point and confirm introduction of these items into the formal configuration control system.

Build and production: There are more similarities than differences between hardware and software configuration management. CM personnel capture software baselines, control changes to those baselines, report on the status of proposed and approved changes, and conduct functional configuration and physical configuration audits. The implementation of computer-based product lifecycle management/application lifecycle management systems leads to similar functionalities. The PLM/ALM system may record and display slightly different attributes or metadata, but the CM processes and interrelationships are basically the same.

In software CM, as in hardware CM, we need to think in terms of both the design, which will be captured and controlled, and the software code, which is identified and should be incorporated into the software executable source code. The

resultant software executable code can be treated similarly to a hardware physical product.

CM during software build and hardware production should represent an institutionalization of stable processes where problems are identified and resolved. One type of resolution is a design change. A design change is circulated for review and approval (or disapproval). Software and hardware designs are updated, and any changes are introduced into a software build or hardware production item. These changes may also have to be backfilled into previously built and delivered units.

Test: For military programs, a functional configuration audit (FCA) should be conducted on engineering prototype hardware and software during the formal internal state. The purpose of the FCA is to ensure that tests have been conducted to verify that each requirement identified in the system level specification has been met by the design. If tests cannot be performed to verify a particular requirement, then a theoretical error analysis must be performed to verify satisfactory compliance to the requirement. These tests are generally referred to as design evaluation and qualification tests. CM personnel either support or conduct these activities.

Following the successful completion of the FCA, a physical configuration audit (PCA) should be conducted for hardware products. During the PCA, the engineering drawings (or digital design file images) are compared against the first production unit. This unit should be built to manufacturing planning that was created from the engineering drawing package. Measurements are verified. All instructions, processes, and technical data specified on the engineering drawings are verified against the hardware. Drawing revision and serial number data are captured and compared to as-defined revision levels.

The best CM practices are those disciplines, processes, procedures, and mindsets that get the CM job done with the greatest degree of control and efficiency, minimal risk, and the least intrusion on other program activities and operations. The last thing we want to do is slow down any product lifecycle activity. Instead, we want the highest level of success for the program. This means we have to capture and control baselines, allow designer and draftsman access to engineering designs for updates and modifications, review and dispose of problems as they are identified, conduct audits without disruption to program operations, and provide visibility and access to program data to those who need it to do their jobs. By adhering to the methodology of a multiple-state level of control system model, you will be able to establish, fine-tune, and document best CM practices in your business.

CM is a necessary part of the product lifecycle, interwoven with and a vital part of each and every lifecycle activity. **{end}**

lyon@configuration.org

Featuring fresh news and insightful stories about topics that are important to you, TechWell.com is the place to go for what is happening in the software industry today. TechWell's passionate industry professionals curate new stories every weekday to keep you up to date on the latest in development, testing, business analysis, project management, agile, DevOps, and more. Here is a sample of some of the great content you'll find. Visit TechWell.com for the full stories and more!

Will Python Unseat R as the Programming Language for Data Scientists?

by Cameron Philipp-Edmonds

As this year comes to an end, a popular topic has been the debate that Python will displace R. A quick search on the Internet will bring up a myriad of recently published articles, blog posts, and forums with some pretty some strong debates on the subject.

But let's be clear: While Python may be great and is quickly becoming a popular choice for start-ups, small companies, and individuals looking to venture into new IT realms, R is still the main player when it comes to statistical sciences.

Continue reading at <https://well.tc/TSp>

Is Scaling Agile an Oxymoron?

by Louis Taborda

Agile methods were designed for and are most successful in small software development teams—that is their sweet spot. But the success of agile methods has meant that larger projects and programs also want a share of the benefits that can come from focused and cohesive teams delivering fast-paced incremental solutions.

As a result, there are ongoing attempts to scale agile methods, most notably Disciplined Agile Delivery (DAD), and more recently the Scaled Agile Framework (SAFe). Both of these processes attempt to “thread the needle” and find a delicate balance between agility and the more traditional plan- and architecture-driven approaches that larger organizations favor.

Continue reading at <https://well.tc/Sys>

Should Just-in-Time Training Be Just in Time?

by Eric Bloom

Conventional wisdom suggests that technical training should be provided just before its actual business use. For example, if you are teaching a .NET programmer to program in Java, the best time for him to attend a formal Java training class is immediately before beginning to program using it.

The theory is that if you provide the training too soon, the person will not remember much of what was taught. Additionally, he will be frustrated and resentful that he is not being given the opportunity to use the newly learned skills.

Continue reading at <https://well.tc/TSh>

Not-To-Do Lists Are Just As Valuable as To-Do Lists

by Naomi Karten

Some people can't get through the day without a to-do list. Benjamin Franklin was one such person. He used lists to encourage his own self-improvement. But he didn't just create a simple list as most of us list-makers do; he created detailed thirteen-week plans to guide him in practicing the thirteen virtues he held dear, such as temperance, frugality, sincerity, and moderation.

Some people are fond of to-do lists and even can be dependent on them. They get satisfaction not just from completing each task on the list, but also from crossing it off the list. (My secret: Make the first item on each day's to-do list “Cross off this item.” It gets the day off to a good start!)

Continue reading at <https://well.tc/Syh>

NSA Uses Special Google Cookies to Aid in Surveillance

by Jonathan Vanian

In another news item released from the seemingly never-ending treasure trove that is the leaked Edward Snowden documents, which detail the spying activities of the National Security Agency (NSA), the Washington Post reports that “when companies follow consumers on the Internet to better serve them advertising, the technique opens the door for similar tracking by the government.”

Yes, this means that the NSA is taking advantage of cookies and location data to help its surveillance operations.

Continue reading at <https://well.tc/TAq>

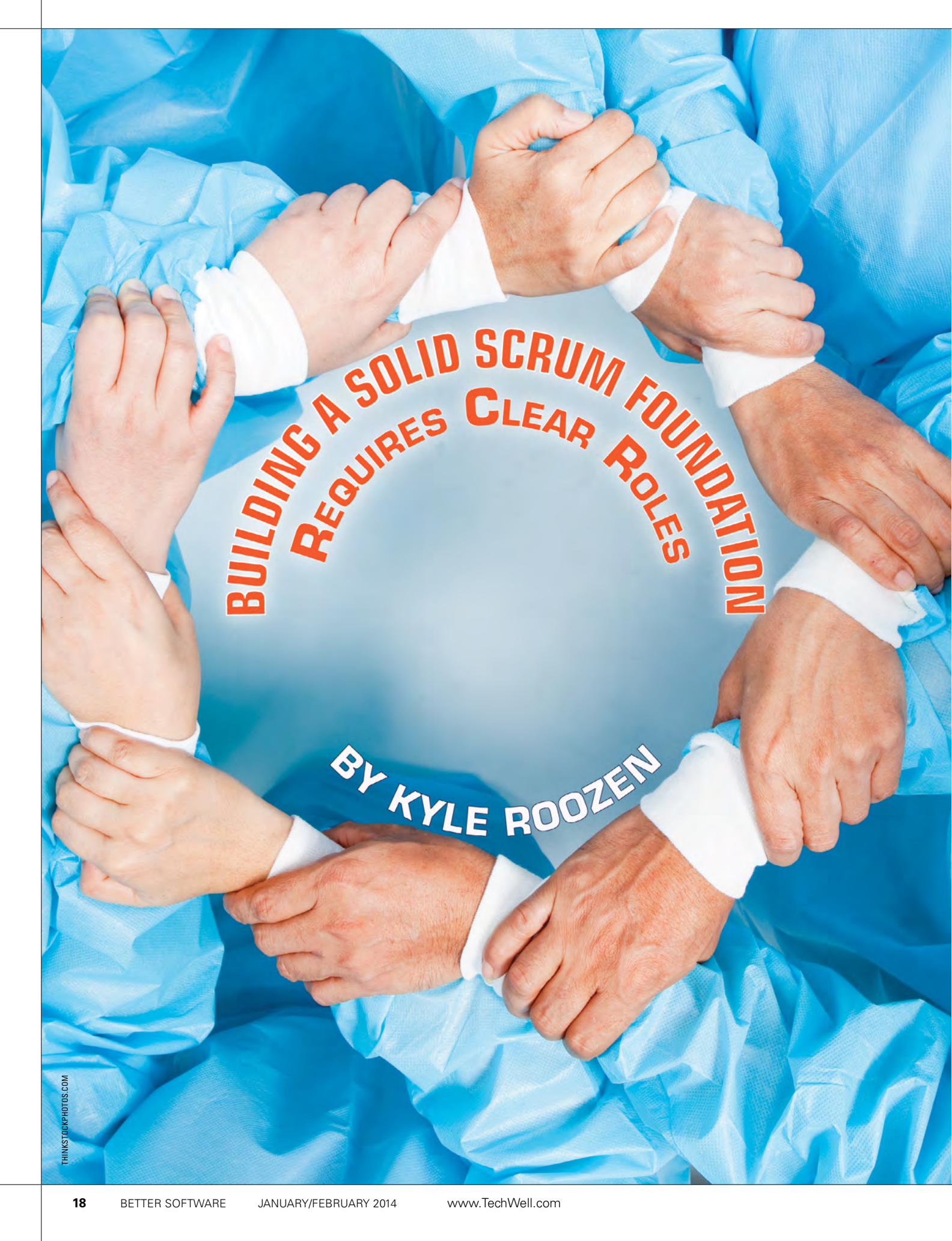
President Asks Every American to Learn to Code

by Beth Romanik

Hoping to spark students' interest in learning more about in-demand computer science and programming skills, nonprofit organization Code.org launched the Hour of Code as part of the annual Computer Science Education Week—December 9–15.

The idea is to get students to invest an hour to learn the basics of computer coding through interactive lessons, accessible on a range of devices. The activities are aimed at total programming beginners and include levels appropriate for kindergarteners all the way up to adults.

Continue reading at <https://well.tc/T3o>



**BUILDING A SOLID SCRUM FOUNDATION
REQUIRES CLEAR ROLES**

BY KYLE ROOZEN

Scrum can provide an ideal framework for project success. Scrum encourages perpetual plan, do, check, and adjust discussions among team members and project stakeholders. Established project-related artifacts provide a means for prioritizing and delivering business requirements. Clearly defined roles help build transparency around who is responsible for executing what. Conceptually, Scrum's foundation for project success is simple to build. In practice, this foundation is fragile and, if not handled with care, can erode quickly. This article explores the erosion of such a project, focusing special attention on three different scenarios where Scrum roles have not been adequately formulated.

Before examining the various scenarios, let's first establish a baseline for how the roles in Scrum are intended to be used and an understanding that the interdependent working relationship between these roles is a lifeline to the success of a Scrum project. Using figure 1 as a guide, consider the role of the product owner. The product owner translates the needs of the business through customer requirements, prioritizes those requirements, and ultimately approves what has been implemented by the development team. Second, consider the ScrumMaster. The ScrumMaster steers the project clear of obstacles and constantly ensures that the team can operate in an unobstructed, efficient manner. He does so by acting as a barrier between the development team and other outside forces, such as the business and product owner. And third, consider the role

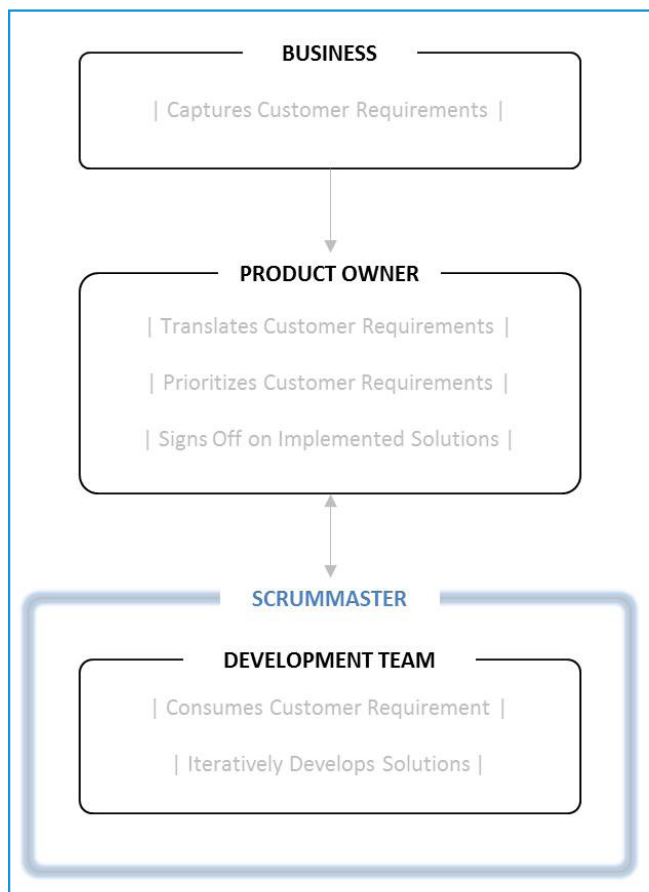


Figure 1: Key roles in a Scrum project

of the development team. The development team is composed of developers and quality assurance engineers. Ideally, they are laser-focused on iteratively implementing and testing the solutions that will, in the end, satisfy the customer requirements.

Scenario 1: Absent or Disengaged Product Owner

The product owner role is about much more than simply assigning a name to the role. Figure 1 shows that the product owner serves as an important broker between the business and development teams. That said, the absent or disengaged product owner scenario manifests itself more commonly than folks might initially think. And when such is the case, a Scrum project can quickly erode.

When put into forward motion, Scrum projects often appear to be set up for success. Names have been assigned to roles, customer requirements have been captured, and the development team is ready to start iterative development. The ScrumMaster reaches out to the individual who has been identified as the product owner, and that person is either absent, disengaged, or both (see figure 2).

So as to not hold up work, the ScrumMaster proactively encourages the development team to consume the customer requirements that have been captured by the business, and development begins. When the time comes to share these solutions with the product owner for sign-off, the only recourse is to go back to the business. This bypassing of the product owner can cause one or more of the following three challenges, leading to the erosion of the project:

1. Customer requirements aren't translated and, thus, are inadequately interpreted by the development team.
2. Solutions are developed for deprioritized or no longer applicable customer requirements.

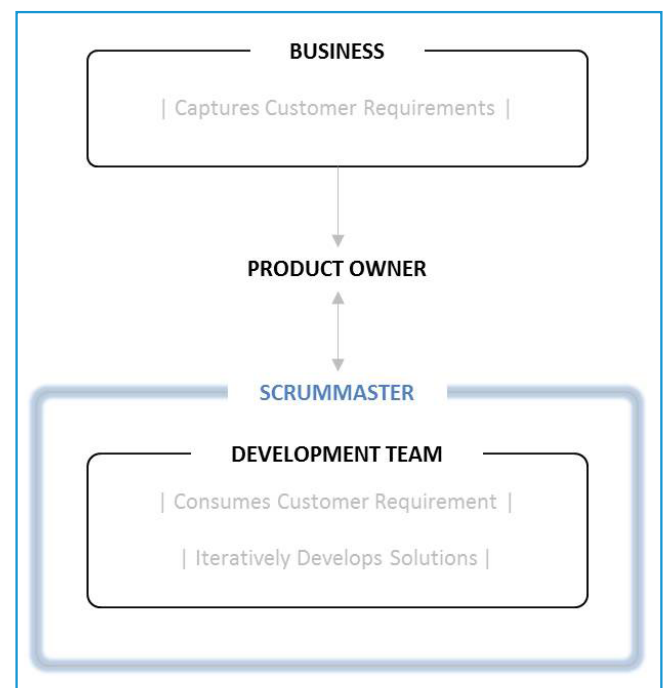


Figure 2: The product owner is not actively engaged in a Scrum project

- Business becomes frustrated with the development team, losing overall trust in the team's ability to deliver.

The key takeaway here is to not underestimate the importance of a present and engaged product owner. It is easy for a product owner to deflect his obligations in a Scrum project. This is why it becomes even more important to solidify the foundation and expectations for this role before moving forward with a Scrum project. Scrum projects can move very quickly, and in the absence of an engaged product owner, the project team may feel it is in its best interest to move forward while letting loose ends fall into place. Contrary to this belief, the product owner needs to be committed to taking full ownership and responsibility. In this situation the team members should consider it an obligation to escalate the concern and recommend that the project be temporarily paused until a dedicated product owner is identified.

Scenario 2: ScrumMaster Playing Dual Roles

In this scenario, you have at least two permutations that will play out in an organization. The first is when a project manager is brought in to help lead a Scrum project. The project manager has limited theoretical knowledge or practical

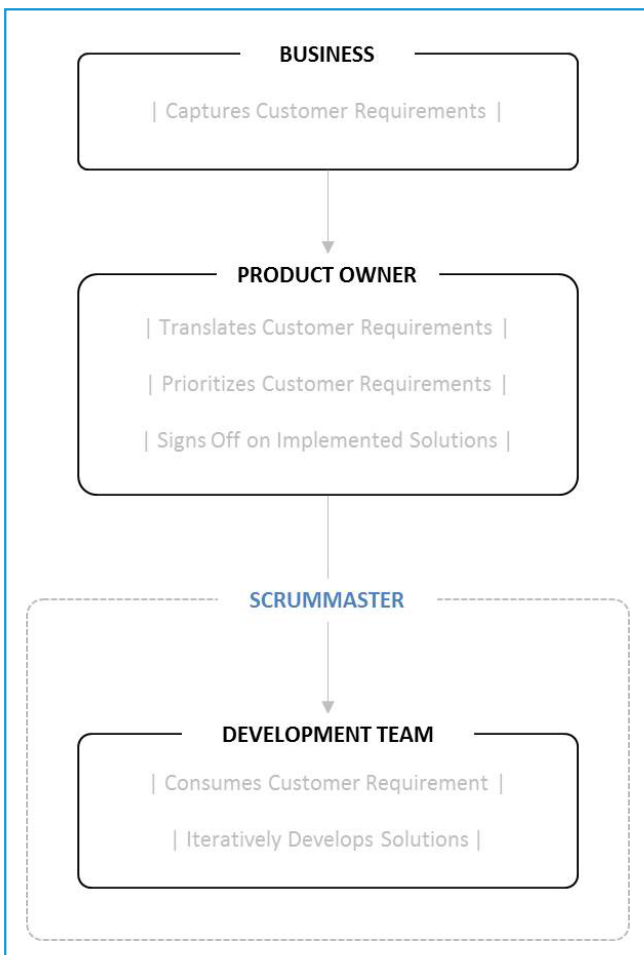


Figure 3: Requirements are passed through the ScrumMaster directly to the team

experience with Scrum yet has been asked to play the role of ScrumMaster. The second is when a ScrumMaster is brought in to lead a Scrum project, but he is also asked to cover project management responsibilities outside the periphery of the Scrum project itself. Neither scenario is ideal, and both can adversely impact the outcome of a Scrum project.

The inherent danger when the project manager, rather than a ScrumMaster, is brought in to lead a Scrum project is that a project manager, by definition, serves a different role than a ScrumMaster. So while knowledge and practice in Scrum will guide some project managers out of the potential pitfalls associated with this situation, others will be set up for failure.

Figure 3 portrays the situation in which customer requirements from the product owner are being passed, as if directly, through the ScrumMaster and on to the development team. A ScrumMaster would normally serve as an impermeable barrier to the development team, but in this case he is doing little to serve as a buffer. This can lead to a series of challenges resulting in an unfiltered environment where every customer requirement becomes the top priority for the development team.

In the latter condition, the ScrumMaster who was brought in to lead a project is asked to extend the scope of responsibility to other needs beyond the Scrum project itself. Such needs may involve supporting another project in the organization, developing a technical roadmap, or analyzing a portion of the organization's financials, for example. As a team player, it is a difficult decision to decline the opportunity to assist in these other areas. However, assisting in areas unrelated to the Scrum project puts the Scrum project on the path toward rapid erosion, due to the high likelihood that with additional responsibilities, the ScrumMaster will be stretched too thin to effectively play his role on the Scrum team.

The key takeaway is to stay focused on the importance of a dedicated and practical ScrumMaster. It is easy for an organization to ask anyone to play the role of ScrumMaster, and it is just as easy to ask a ScrumMaster to serve in a capacity beyond the Scrum project. If a Scrum team finds itself facing one or both of these predicaments, then a discussion should take place with someone who has the authority to offer assistance in order to rectify the situation. In the final analysis, a Scrum project can still be successful given this scenario, but the likelihood of the project eroding increases dramatically, so the risk should be carefully calculated.

Scenario 3: Failure to Adopt the Scrum Framework

Both novice and experienced Scrum participants understand the importance of fostering an environment where the development team is truly empowered to self-organize. When a dedicated ScrumMaster adequately buffers the development team from influences beyond the Scrum project, the team is able to do just this, and the resultant deliverables can be very impressive. Of course, with the freedoms of self-organization there comes responsibility. For Scrum projects, that responsibility belongs to each member of the development team.

The development team is composed of developers and quality assurance engineers, with each individual possessing a unique skill set for the organization. It is not unusual for developers and quality assurance engineers to become fixated on the unique skill sets they were hired for, and there is a tendency to operate in a capacity where only that skill set is exercised. In theory, this is an efficient use of an organization's investment in these individuals. But in reality, this is a disastrous mindset because in a Scrum project, it is everyone's job to help the project by helping each other. It is the responsibility of each member of the development team to eliminate the "me only" mindset before moving onto a Scrum project. Without doing so, the Scrum project runs the risk of quickly eroding.

Consider the following hypothetical example. A development team is halfway through its third of ten sprints. Within the sprint, a number of features remain open and still need to be coded and tested for compliance to the corresponding customer requirements. The quality assurance engineers are overwhelmed with functional tests that need to be executed against these features, yet the developers have plenty of bandwidth to continue developing. In an ideal Scrum environment, the developers will recognize this bottleneck and transition their efforts toward quality assurance rather than development. Just as important as the skills the developers were hired for is their ability to shift their mindset from individual contributors to team players when joining a Scrum project.

For a Scrum project to be successful, the development team needs to not only feel comfortable self-organizing but also feel comfortable operating outside the team members' individual areas of expertise for the overall success of the project. In a Scrum project, if the development team is operating with a "me first" mentality or using the self-organizing umbrella as an excuse to make up rules as they go, then changes to the team need to be made. And a dedicated ScrumMaster can help initiate these changes.

Final Word

Scrum can provide an ideal framework for project success, but the foundation can be fragile and needs to be handled with care. Defined Scrum roles help build clarity around who is responsible for executing what, but the definition itself is not enough. The product owner, ScrumMaster, and development team need to be able to fully invest themselves into the Scrum project. To do so requires a shift to a more unified, self-organizing mindset that will lead to project success. **{end}**

kroozen@acmeconsulting.com

Newsletters for Every Need!

Want the latest and greatest content delivered straight to your inbox every week? Have we got a newsletter (or four) for you!

AgileConnection To Go covers all things agile.

CMCrossroads To Go is a weekly look at featured configuration management content.

StickyMinds To Go sends you a weekly listing of all the new testing articles added

to StickyMinds.com. And, last but not least,

TechWell To Go features updates on the curated software development stories that appear each weekday at TechWell.com.

Visit StickyMinds.com, AgileConnection.com, CMCrossroads.com or TechWell.com to sign up for our weekly newsletters.

<http://forms.sqe.com/forms/SMeNewsletterSubscribe>



REVERSE MENTORING SHOULD YOUNGER WORKERS YO

BY LEV

I recently attended a presentation by a colleague of mine on using technology to market IT services. Her presentation included technologies such as social media, QR codes, and smartphone apps. It was very impressive. Afterward I asked her how she became so knowledgeable about these technologies.

“I had been working in traditional marketing for twenty-five years,” she responded. “A couple of years ago, we hired Michael to our marketing staff. He was just a couple of years out of school and only had a few years of marketing experience. He used all of this new technology that I didn’t know much about, so one day I asked him about it, and he demonstrated different applications and how they could be used in marketing.”

“You learned all of this in one sitting?”

“Oh, no,” she responded. “We would sit down for an hour or two every once in a while. It’s ironic because I was assigned by the marketing department to be his mentor.”

“And he ended up mentoring you?” I asked.

“We mentored each other. I taught him things like our competitive positioning and marketing strategy. He taught me all about these new technologies that I hadn’t kept up with.”

The conversation with my colleague got me thinking. Are organizations, particularly in IT, missing an opportunity here? The younger generation that is just entering the workforce typically knows technology—especially social media. My baby boomer generation saw the introduction of cellular phones and witnessed their evolution to the smartphones we enjoy today. We also saw the birth of the Internet during our careers and its rapid progression from the static pages of the mid-nineties to the interactive sites we currently use.

But these technologies have always been in existence for the millennial generation. They have never known a world that didn’t have mobile phones, the Internet, and so many other technologies still perceived as new.

The millennials are a much more collaborative generation. Perhaps because they have grown up with social media, texting, and the taking and sharing of pictures on the fly, they embrace new technologies. They are not afraid to try a new technology and share it with their friends. The baby boomer generation, on the other hand, tends to be later adopters, choosing to wait until new technology is more mainstream before trying it.

The result is a younger generation that has developed a



NG

WORKERS BE MENTORING OUR EXECUTIVES?

V SAUDER

more superior knowledge of advanced technology than its older coworkers. Meanwhile, baby boomers who have worked in IT all their lives have trouble facing the reality that they aren't keeping up with technology.

When new employees fresh out of college join the workforce, they are paired with more seasoned professionals to mentor them on such things as meeting etiquette, reading a balance sheet, and career management. This is still a valuable way to teach new employees about business concepts they may not have learned in school.

But what if we turned the tables a bit and made it more collaborative? While the more seasoned professionals coach the younger generation on how to handle various workplace scenarios, the younger generation can coach "upstream" and show the baby boomers how applications like Google+ and Twitter can be used in the business world, or more effective ways to network using LinkedIn.

Benefits to Reverse Mentoring

The concept of reverse mentoring—or, more appropriately, multi-generational mentoring could lead to a more productive

and knowledgeable workforce in many ways.

Better leadership development of young workers: Allowing members of the younger generation an opportunity to share their knowledge with more experienced coworkers provides the younger professionals with leadership opportunities they will continue to build on throughout their careers. Experienced workers can develop their own leadership skills by accepting that they are not always right and do not have all the answers, learning to become less authoritarian and more collaborative with diverse teams.

More respect between generations: Working with more experienced workers as peers in a collaborative environment develops stronger bonds at all levels, fostering greater tolerance of diversity and potentially more enhanced innovation.

Higher retention rates: Millennials want to make an impact immediately. Allowing them to mentor their more experienced peers on topics about which they are knowledgeable and experienced, coupled with the stronger bonds they will develop with their coworkers, provides higher job satisfaction and increases the likelihood of retention. Developing a friendship makes the interaction enjoyable and allows the younger worker to feel

more like a peer or colleague than a junior employee.

Higher organization-wide level of technical knowledge: The older people get, the more comfortable they become with their habits. More experienced workers may feel like technology that is only a few years old is quite advanced, whereas the younger professionals have moved to more recent technology (like smartphones, tablets, and social media) with ease. Having millennials share their knowledge of the newest technology with their coworkers allows the entire organization to access and benefit from the latest trends.

Earlier identification of high-potential employees: Having higher-level decision makers interacting with new employees provides upper management with earlier contact and exposure to rising stars, allowing them to take earlier actions to ensure retention and career development.

Keys to Effective Implementation

While the benefits are attractive, don't just jump into a reverse mentoring program without thinking things through. To be successful, expectations should be set appropriately for all involved.

Acceptance of vulnerability: Baby boomers are used to running the organization and calling the shots. They may need to be coached to question their own assumptions and consider alternative ways of thinking before allowing younger coworkers to teach them. This could involve persuading them to make decisions in a more collaborative approach rather than in a command-and-control decision-making process.

Tolerance and openness to sharing: Each generation enters the mentoring relationship with its own biases and assumptions. The initial instincts to each generational difference will be to resist. Each may see the other as weird and out of touch. Encouraging each generation to break down the walls of intolerance about the other's culture results in opening each generation to new ideas.

Persistence: When someone has a depth of knowledge on a specific technology or process, he may become impatient when the person of a different generation has trouble understanding. Explaining a new and complex technology to a baby boomer who has little experience with it will take persistence and understanding to help the person come up to speed. Similarly, explaining a concept such as cost accounting to a new employee without any exposure to business fundamentals will require comparable effort.

How to Get Started

Once expectations have been set, it's time to execute. Starting out on the right foot is critical.

Start with the right attitude: It is important that all participants approach multigenerational mentoring with an attitude of openness. Baby boomers should not summarily dismiss the millennials as too inexperienced and therefore unable to be taught anything. The younger generation also should not dismiss baby boomers as out of touch. Attitude shifts may be facilitated by an experienced coach.

Sell it to your employees rather than mandating it: If experienced workers believe in the benefits of multigenerational mentoring, they are more likely to commit themselves to such a program, resulting in greater success. This requires the organization to sell employees on the idea and obtain their agreement rather than mandating their involvement in the program. Convincing older employees that using technology is a more productive way to work in the twenty-first century, rather than insisting they adapt to the younger generation's way of doing things, may influence more buy-in with the program.

Don't necessarily call it mentoring: People often get caught up in terminology. If a majority of your staff insists that mentoring means more experienced people teaching the younger generation, change the lingo. Call it knowledge sharing, knowledge exchange, or something else that makes it feel more acceptable to all participants.

Start simple: Any mentoring program can be established informally. At Geneca, we have frequent lunch-and-learn sessions. Any employee is free to come up with a topic he thinks will interest other coworkers. Anyone interested can bring a lunch and sit in on the session. We've had people in the first couple years of their careers teach seasoned employees about new programming languages and techniques. This results in the older generation's learning something and realizing that the millennials may actually have some knowledge to share with them.

Pilot the system: If you anticipate difficulty getting buy-in or even participation, perhaps it would work to start out with a pilot of the system. Identify one boomer and one millennial who are open to the idea. After they mentor each other, allow them to be ambassadors of the program. Interview them on video and post it on your intranet site. They could host a lunch-and-learn on multigenerational mentoring. Once other people see it work, they may be more inclined to participate.

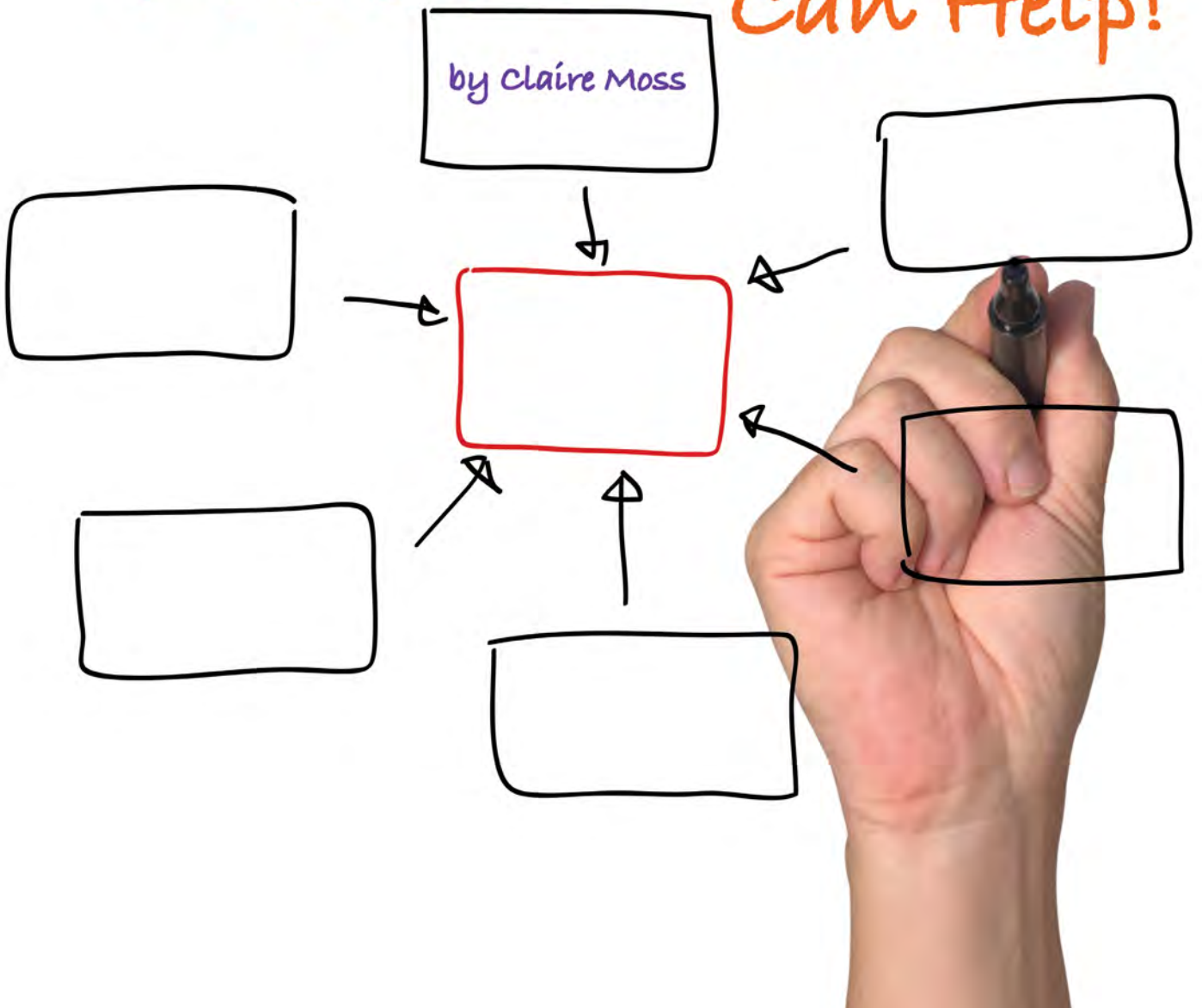
A lot is written today about millennials regarding how to integrate them into the workplace. Perhaps the best approach is to simply listen to them. It can be to everyone's benefit to take advantage of the knowledge and experience this generation brings to the business world.

One of the traits we hear so often about millennials is that they want to contribute right away. They also collaborate more than any other generation. As a baby boomer myself, I find that I have mixed feelings. I'll admit that there are times when I think the millennials are overstepping their boundaries. But it's great to see the new generation of workers collaborating and striving to contribute. It leads me to think that rather than fighting them, perhaps we should welcome their participation. We might just learn something. **{end}**

lewis.sauder@geneca.com

Feeling Lost In the Woods? Mind Maps Can Help!

by Claire Moss



What's the Big Idea?

Visual thinkers face a world where words make up the majority of work. It's easy to feel lost among the piles of paper and walls of text. Instead of caving to the pressure to use mere words, grow your strengths and branch out into creating mind maps. Other visual thinkers will thank you and follow your lead while your prose-oriented colleagues still enjoy a comfortable level of verbiage.

You can think of a mind map as an outline that's easier to read because it is displayed as a cluster instead of a never-ending list in a text document that you must scroll through. I like using them because it is easy to add things to the list without reformatting.

Why Mind Maps?

I learned about mind mapping when I was a kid. Someone thought it was a good idea to teach hand-drawing mind maps to elementary students, and I took to it immediately. As it turns out, I'm quite interested in representing information visually, but I didn't know what that meant at the time. It just fit in with some nonlinear thinking the public school was reinforcing. There are lots of connections between things in this world, and as an adult I see that connectedness—especially in technology—so mind maps are a natural way to represent those relationships.

Figure 1 shows a subset of the mind map used to create this article. Although it may not look very tidy and violates the hierarchical nature of mind maps, some branches can be related to other parts of the mind map, resulting in cycles in the graph. Recently, mind mapping has become fashionable among software testers. Due to their visual nature, mind maps are also more memorable and can be very effective for organizing thoughts and notes.

Drawing Pictures at Work? Really!

At Agile2013, Jeremy Kriegel explained using graphic facilitation to craft meetings that better involve attendees. [1] This sketching is a combination of note taking and wire framing, which is something user experience (UX) folks do routinely as

part of their work. He describes trading quality of the drawing for speed in order to keep the focus on communication, then enhancing the drawing later.

How Are You Using Mind Maps at Work?

There are many opportunities to use mind maps for testing activities.

Defect reporting: I had the good fortune to work on a cross-functional product team that included a product manager, developers, and UX designers. As we got to know one another, the other team members taught me about their past experience with software testers. Because we were all familiar with defects, I thought that would be a good place to start the discussion. UX designers are particularly responsive to visual information, so we worked on displaying known defects in the product we were building. Our bug board took many forms over time as we progressed toward providing essential information more effectively. Our favorite iteration was a hand-drawn site map of the whole product on a physical whiteboard with actual sticky notes arranged on the parts of the application where the symptoms surfaced. The hierarchical set of pages and transitions between pages that represented the application was essentially a mind map.

Test planning: Rather than only documenting an existing product, mind maps can record new ideas for future work as well. As new user stories passed through full-team grooming and arrived in sprint planning, we built a mind map of branches for each user story along with descriptions, acceptance criteria, and any additional notes. Once the team had a shared understanding, we expanded on the story acceptance criteria based on questions to create test ideas. This produced a set of intended tests that were now available to the whole team for review, perhaps eliciting additional test suggestions or, at times, simply pointing out a miscommunication before being removed.

Once the team agreed on the meaning of a story and began the next sprint, we grouped related test ideas together as testing charters to create story tasks for testing. Mind maps tracked my testing session results, including known issues,

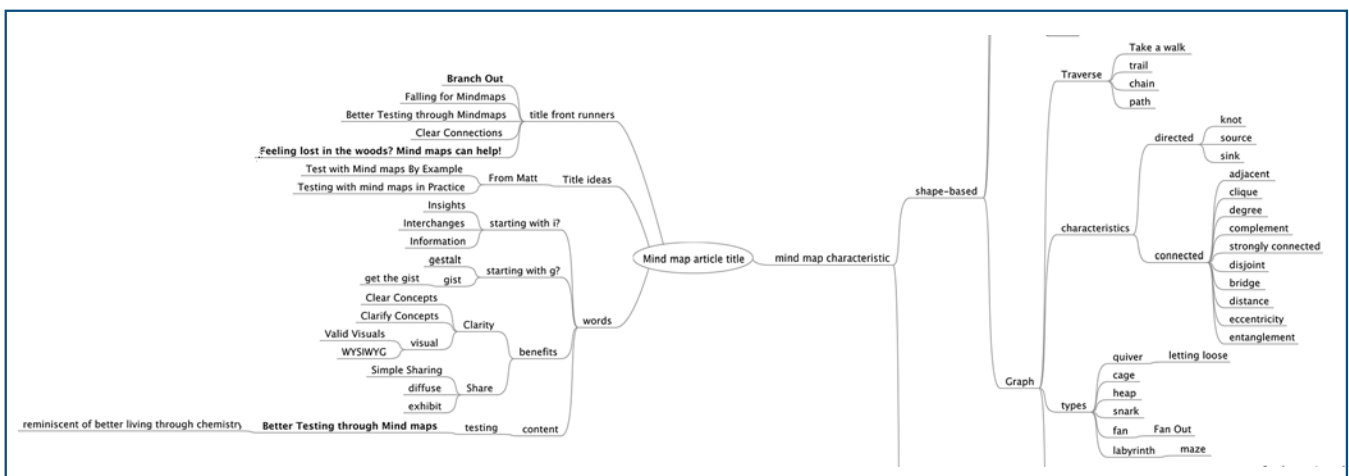


Figure 1: Mind mapping example of this article (partially shown)

blocking bugs, questions, and good results.

Test idea aids: Additional sources for test ideas include testing heuristics that are often themselves represented as mind maps of mnemonics for easier recall. These memory aids are a great reminder of different kinds of concerns and risks that might be relevant to the testing work at hand. Sometimes a shift of emphasis provides new insight, and these heuristics can reveal better test candidates. When a suggested area of focus brings out new ideas for tests, it augments the mind map of planned tests for the particular unit of work.

Mind maps provide a vehicle for conversation with business advocates to determine whether there is sufficient test coverage for a user story. User personas for the application or system can offer more context for making a first attempt at sequencing what needs to be tested. Someone with business value knowledge, usually the product owner or product manager, can review the list of test ideas to prioritize them.

When there are multiple stakeholders to consider, the mind map can serve as a meeting agenda to keep the discussions moving. This often results in better information about what matters to the business so that everyone leaves the meeting with a prioritized testing backlog for a particular story. This is a perfect opportunity to cut or add testing scope so that the business invests in the testing that is most valuable. When the business expects a testing estimate, providing a backlog of planned testing is easier to explain in the way the team expects.

Heuristic mind maps: Many testers have created mind maps of experience-based testing memory aids, or heuristics. When reviewing these existing mind maps for ideas that seem relevant to the testing at hand, highlight those branches. Although this leaves a lot of dead wood in the tree, the resulting distinction between what is in scope and what is out of scope is pretty obvious. Removing the irrelevant options results in a smaller mind map to track testing progress, results, and other test status considerations.

Project planning: If you are leading the testing of a project, you may wish to keep track of the discovery process. Specifically with agile projects, there will be considerable learning about a product during the project's lifecycle rather than having well-documented, intuitive information available from the beginning. Mind maps collect discoveries and key conversations in one place to benefit the whole project team, especially when that team makeup may change during the project.

Consider creating a mind map of reminders at the beginning of a new project. Usually, a project mind map's content is the subject and product of a series of conversations. Because it is collaborative brainstorming, record the observations and questions of project team members as part of this ongoing meeting documentation. The resulting mind map shows the development of shared understanding over time. Irrelevant past considerations can simply be folded or collapsed to reduce the complexity of the end result.

This complete history of conversations and decisions describes the chosen approach among many considered alternatives.

Exploratory testing documentation: If you are new to a project when you perform exploratory testing, you could build an outline of the structure of the application as you go. Alternatively, a mind map can capture the behavior of the application and focus discussion with other testers on the project. In this way, the mind map serves as both a site map and a record of the exploratory testing completed.

After further exploration and asking questions, update the mind map so it becomes a more complete form of documentation. When the observed behavior turns out to be a problem, adding a defect identifier from the bug tracking system to that branch of the mind map is easy.

Product coverage outline: I recently attended an end-to-end agile testing tutorial at the Conference of the Association for Software Testing. The class focused on reporting testing status to interested parties, including software testers, developers, test managers, development managers, ScrumMasters, product managers, clients, and other stakeholders. While the instructor used mind maps to organize his risk assessment and a heuristic approach to reporting on the work completed, I tried a different tack. I was reading through an email from the client, a developer whose software upgrade we were testing, and found that he had many specific points about what he valued.

His description of concerns was rather lengthy—almost a wall of text. Converting his email into a mind map allowed tracking progress of testing organized based on his concerns. Another mind map represented the structure of the product and how tests would be executed against it, as the teacher had recommended. The first mind map was vital to focus testing on usage scenarios and risks that were most urgent and important.

Using Mind Maps

Mind maps can be used in many other ways, such as software design, to-do lists, preparing presentations, and coaching. I have friends who use mind maps to collaboratively write books and to create and present material for tutorials. A mind map's key benefit is as a mechanism to visualize relationships between important pieces of information. This is particularly true in the testing world. **{end}**

claire@aclairefication.com

Sticky
Notes

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

■ References

Interneer Inc. Announces Intellect MobileApps

Interneer Inc., a provider of business process management software for process automation applications, announced Intellect MobileApps that enables business users and IT to easily build and manage native smart mobile apps on any iOS device for enterprise use. Smart mobile apps are integrated with back-end systems and enterprise applications and deliver enterprise-level security with bring-your-own-device capabilities.

Designed specifically for mid-sized companies and departments of large enterprises, smart mobile apps improve the productivity and efficiency of employees in the field, and customers, partners, suppliers and others who need access to vital enterprise data on the go. Intellect MobileApps, a free container app available on the App Store, is fully optimized for all iOS mobile devices, including iPhone 5S, iPhone 5C, iPod Touch, iPad Air, and iPad mini. Organizations can build an unlimited number of apps across the enterprise, all accessible within Intellect MobileApps as a library of native apps with access granted to each user based on individual permissions and security.

<http://www.interneer.com>

Skytap Inc. Introduces Skytap Cloud

Skytap Inc., a provider of self-service application development environments in the cloud, introduced Skytap Cloud built on Amazon Web Services (AWS) infrastructure. Customers can rapidly and repeatedly deploy development and test environments on AWS infrastructure using Skytap's complete SaaS offering to automate the build, deploy, and test lifecycle. The new offering allows users to build and test software on environments that mirror AWS production environments while utilizing the tools and solutions already used by development teams—such as Jenkins, Microsoft Team Foundation Server, and IBM Rational Team Concert. With this release, Skytap now supports both native AWS and VMware workloads.

AWS is an infrastructure provider for hundreds of thousands of the world's most innovative companies. Skytap's focus on development and testing means support for on-demand deployment of complex computing systems and networked environments to dev and test teams. Skytap Cloud on AWS helps ensure applications bound for AWS can be delivered faster and with higher quality.

<http://www.skytap.com/aws>

TestPlan Launches eggCloud

TestPlant, the maker of the eggPlant range of software quality tools, launched eggCloud, a private test cloud designed to help enterprises and test services businesses improve the way they manage software testing on mobile devices and desktop systems.

In conjunction with TestPlant's leading test automation tool, eggPlant Functional, eggCloud provides local and remote access to a centralized pool of test devices. This includes iOS and Android products, Windows machines running any browser, and any Linux servers. eggCloud is secure and compliant with

all industry guidelines and is designed for dedicated in-house or multi-project distributed QA teams, as well as external developer communities. Manual testers can have equal access via eggCloud to the centralized lab and its devices.

<http://www.testplant.com>

RightScale Inc. Rolls Out RightScale Cloud Analytics

RightScale Inc., an enterprise cloud management company, announced RightScale Cloud Analytics, the first multi-cloud cost management solution that integrates with a cloud management platform to enable users to take action on insights and implement budget controls.

Cloud Analytics is a new enterprise-grade cost management solution from RightScale that provides sophisticated cloud usage and cost analysis as well as forecasting and scenario planning across major public and private clouds. Cloud Analytics is integrated with the RightScale Cloud Management platform, providing a mechanism for enterprises to quickly execute optimizations to their cloud portfolio.

Together, RightScale Cloud Management and RightScale Cloud Analytics comprise the first in a new category of applications, cloud portfolio management (CPM), that provides an integrated solution to manage applications and optimize usage across a portfolio of public and private clouds. With RightScale Cloud Portfolio Management, enterprises can deploy applications in any cloud and seamlessly move applications between clouds in order to meet technical requirements or optimize costs.

<http://www.rightscale.com>

Kapow Software Announces Kapow Enterprise 9.3

Kapow Software, a Kofax company and leading big data integration software provider, announced the availability of Kapow Enterprise 9.3, which features a redesigned interface that simplifies the user experience and encompasses the entire information supply chain from data acquisition to enrichment, persistence, exploration, and distribution. Kapow Enterprise 9.3 is designed for organizations that need to turn data into actionable insights.

Using Kapow's Enterprise 9.3 Synthetic APIs and its patented data integration flows, organizations can easily extract and integrate big data from any source. Data integration flows can be deployed as Kapow Kapplets via the Kapow KappZone library. Kapplets enable users to run and manage thousands of automated data integration applications at any time and explore an integrated view of disparate data in an interactive pallet in order to act on their findings.

<http://www.kapowsoftware.com/products/whats-new-in-9.3/index.php>

FAQ

expert answers to frequently asked questions

by Rob Sabourin
rsabourin@amibug.com

Can Test Estimation Be à la Carte?

It comes up all the time. A key stakeholder corners me and inquires, “Tell me, Rob, how long will it take for your team to test project Fizbin?”

“What is project Fizbin? I have never heard of it,” I respond. Don’t stakeholders realize that preparing a reasonable test estimate needs at least some detailed information, such as product requirements, what quality factors matter most, how the product was designed, how the product will be used, and who will use it and why?

On the other hand, stakeholders may not know why this kind of information is needed. Why should they?

I’ve learned that a great way to answer this question is to actually give stakeholders a helpful tool.

Instead of engaging in a rhetorical flurry of profound argumentation and endless dialogue, why not present stakeholders with a menu—the type of à la carte menu you might see in fine restaurants? The menu contains nine items with three juicy tidbits of information about each choice.

The nine items represent recently completed projects, and each project has a different size and complexity. Project size can be small, medium, or large, and project complexity can be classified as simple, typical, or complicated.

Table 1 shows the actual test effort and time required to complete each project. In addition, reviews of each menu item are included, along with a count of post-release bugs that necessitated updates or patch releases.

	<i>Small</i>	<i>Medium</i>	<i>Large</i>
<i>Simple</i>	Project Kermit	Project Ernie	Project Bert
	Effort: 2 days	Effort: 20 days	Effort: 50 days
	Duration: 1 day	Duration: 8 days	Duration: 22 days
	Production bugs: 6	Production bugs: 33	Production bugs: 23
<i>Typical</i>	Project Grover	Project Elmo	Project Big Bird
	Effort: 100 days	Effort: 200 days	Effort: 400 days
	Duration: 30 days	Duration: 75 days	Duration: 150 days
	Production bugs: 12	Production bugs: 9	Production bugs: 12
<i>Complicated</i>	Project Count	Project Snuffleupagus	Project Oscar
	Effort: 200 days	Effort: 400 days	Effort: 600 days
	Duration: 65 days	Duration: 80 days	Duration: 244 days
	Production bugs: 1	Production bugs: 0	Production bugs: 15

Table 1: Recent testing project menu

By sharing the menu with an inquisitive stakeholder, stories can be analyzed for each of the menu items drawing on personal knowledge and experience. This is a golden opportunity to remind stakeholders of important context factors and relevant business, technological, and organizational components for each of the menu items. In addition, other project risks can be highlighted, such as the commercial pressures, political climate, and other priorities at the time.

By comparing and contrasting a new project to other recent projects, stakeholders can work with trusted advisors and confidants to determine which recent project closely matches Fizbin. The menu serves as an important estimation tool.

I encourage stakeholders to ask for estimates early and often. By using an à la carte menu system, stakeholders can identify important concerns before they commit to implement new projects in the business pipeline. **{end}**

You Can't Be Just a Manager Anymore

Just when you thought being a project manager was tough enough, you become responsible for a ton of other roles.

by **Gunasekaran Veerapillai** | gunasekaran.veerapillai@wipro.com

“I am managing a team of a thousand employees.” “I take care of five projects with more than six hundred employees.” “I am the program manager for a very big project rollout.” Do all these quotes from managers sound interesting or hollow to you? Statements like these may have been exciting a few years ago, but not these days.

I am playing the role of risk and mitigation manager in a SAP rollout project for a major retailer. I am the performance test architect in a core banking migration project for a bank. I am playing the role of test consultant for an insurance client who just recently acquired two other insurance companies. I am the cost optimization manager for a global securities client.

Should a project manager play the role of coordinator and people manager? With multiple technical and domain roles, it is easy to see how my original focus as project manager became diluted.

A *Guide to the Project Management Body of Knowledge*, or the *PMBOK® Guide*, defines project management as identifying requirements; addressing the various needs, concerns, and expectations of the stakeholders as the project is planned and carried out; and balancing competing project constraints, such as scope, quality, schedule, budget, resources, and risk.

Take a typical IT program and check out how many managers are performing an extraordinary number of disconnected tasks. What benefit are they providing in controlling scope, schedule, and budget? How effective are they in managing quality and resources? Is there any available time to effectively plan for project risk and risk mitigation? By dividing time among several concurrent projects, is the role effective in an agile environment? And because most of us in project management come from technical backgrounds, do we still have responsibilities to perform technical tasks as architects, consultants, domain experts, and quality auditors?

Due to the complexity of projects in terms of size, new technology integration, cloud, performance, and security, there can be multiple managers taking projects in various directions.

Especially in the agile environment, these responsibilities are shared with team members who are expected to scale up.

Assuming the traditional definition and responsibilities of a project manager, the role is independent of specific technologies employed in the project. It is very difficult, however, to manage a team effectively unless you are comfortable in the technology or domain of the project.

Can a project management office coordinate all the activities of multiple managers? Do we still need people managers or project managers who manage real-world issues that are simply added on to the already heavy workload of technical managers? Project management roles that do not require technical or domain expertise are rapidly disappearing. The percep-

tion is that consolidator or coordinator roles don't add value to the business.

What is your current role and responsibility as manager of projects? How technical or domain-experienced are you, and are you losing your technical edge the longer you're in management? Is more than fifty percent of your day spent in mundane coordinating of activities or people issues?

Think about your answers and decide whether it is time to reconsider how effective you are as a project manager. **(end)**

“It is very difficult to manage a team effectively unless you are comfortable in the technology or domain of the project.”

Interested in writing an article for *Better Software* magazine?

Contact Ken Whitaker at kwhitaker@sqe.com.

We're looking for article proposals for agile, testing, project and people management, configuration management, ALM, development, and any other topic you think is relevant to today's software professionals.

New Year, New StickyMinds.com

Software Quality Engineering has relaunched its popular software testing and development website, StickyMinds.com. With a cleaner look and feel and several new features, we remain dedicated to providing testers and other software professionals the resources needed to build better software.

The new StickyMinds.com retains popular features such as the weekly column, the Tools & Services guide, the Books Guide, blogs, and the jobs board. New additions include a Twitter feed, Q&A to connect community members, and improved navigation with access to the latest content right from the homepage.

A complimentary subscription to *Better Software* magazine is included with your StickyMinds.com membership. Members also receive free access to the *Better Software* magazine archive and Software Quality Engineering conference presentations. No PowerPass needed.

As with any new release, I'm sure you will find some bugs. Help us improve StickyMinds.com by submitting feedback, bugs, and suggestions via our feedback form: <https://well.tc/feedback>

index to advertisers

Cognizant	https://fastest.cognizant.com/webapps/home	Back Cover
Ranorex	http://www.ranorex.com/whyBSM	2
STARCANADA	http://starcanada.techwell.com	Inside Back Cover
STAREAST	http://stareast.techwell.com	9
SQE Training	http://sqetraining.com/trainingweek	Inside Front Cover
StickyMinds.com	http://www.stickyminds.com	8

Display Advertising advertisingsales@sqe.com

All Other Inquiries info@bettersoftware.com

Better Software (ISSN: 1553-1929) is published six times per year: January/February, March/April, May/June, July/August, September/October, and November/December. Back issues may be purchased for \$15 per issue plus shipping (subject to availability).

Entire contents © 2014 by Software Quality Engineering (340 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 904.278.0524 for details.

