# Communicating for Quality

Description:

Once we (back when “we” were QA) sat by a wall, looking up, wondering what code grenades would come flying over from the enemy (developers).

Thankfully, those days are (largely) gone.

The challenge and opportunity now is to embrace our role as equal partners by sharing our expertise. We have a responsibility to help the whole team contribute to quality.

How do we share our vision, for maximum impact? Through trust and lots of communication.

Jon will show the right ways to engage with developers to shift testing left, gain credibility with them, and (gasp) build trust. You will learn how early conversations with Business Analysts can find and fix bugs at the earliest possible point. Find out the value of connecting directly with the Product Owner. And learn to build community with fellow quality advocates across your organization. Become a Quality Multiplier!

Takeaways:

Attendees will learn:

- how to achieve technical credibility so your team will want to know your opinion

- specific communication techniques for navigating difficult conversations

- multiple ways to show how quality belongs to the whole team.

Speaker Bio:

Jon Allen is a Quality Architect at Red Hat, where he has brought a robust test automation and performance test practice to his team. He is learning a lot and having fun as the team moves its apps to a continuous cadence.

He also consults reqularly with other teams across Red Hat to advise on best practices as well as to drive improvements in large-scale processes. The Exploratory Test Parties he started have proven quite popular (and effective).

His previous roles over his 30-year career include game engine tester, risk compliance, bioinformatics, and a really terrible first job writing a UI in Pascal (really).

Speaker Experience:

Jon has spoken at Open Source 101, at Red Hat's internal quality conference, and at countless Toastmaster meetings, where he has achieved Competent Communicator status. He gave a guest lecture on performance testing to a graduate class in Computer Science at North Carolina State University.

References:

1. Paul Merrill, Principal, Beaufort Fairmont

[paul.merrill@beaufortfairmont.com](mailto:paul.merrill@beaufortfairmont.com)

984-244-2313

2. Shane McDowell

[smcdowell@redhat.com](mailto:smcdowell@redhat.com)

919-602-3690

====================================================================

**Communicating for quality**

**Intro**

Let’s start with a group testing session.

Everyone please stand if you are able.

We’re going to test the floor.

Actually we’re already performing a simple functional test, aren’t we? It’s holding us up.

Okay, let’s do a basic integration test - if you’re next to the wall, give it a little push, and we’ll make sure the floor, the walls, and the ceiling are all working together.

Great!

On to a load test - jump up and down!

Whew - glad it passed that one too.

Okay, can we call this floor ready to release?

Thank you, you may be seated.

How much did any of us think about the floor before our test session?

Not much, right? It’s just there, doing its job; we don’t have to think about it.

**[SLIDE]**

What if software quality was like that? What if, when we built our projects, quality was just a given in everyone’s mind? Like when you build a building, and you know there has to be a floor.

I believe we can achieve that goal. Software as a discipline is still young, and we’ve already come a long way since the bad old days of waterfall. Quality IS everyone’s responsibility. But I also think that we, as quality professionals, have a great opening to lead the way to “Quality by Default”. That can take us from being testers to being Quality Multipliers.

Today I’m going to focus on one critical aspect of that opportunity - great communication that can help make quality a given.

I’ll cover specific communication methods to help us achieve that goal.

**[SLIDE]**

I’ll break it down three ways:

First, how to communicate effectively with the different groups involved in your projects.

Second, how to do it with your whole project team.

Third, special cases: informal, bad, and difficult communication.

Then, before the general Q&A at the end, I’ll invite you to play Stump the Speaker, giving you a chance to bring me your thorniest communication issues to see if I can help. The biggest challenge wins a prize.

**PLEASE NOTE** that this is based on my experiences and those of my team, and the way we make software. And though I believe most of the ideas are pretty universal, you have no doubt had some different experiences. I would love to hear **your** stories and insights - please bring them to the Q&A, or flag me down after this talk.

Now - specific techniques you can use to influence for quality:

* First, communicating with different groups that you work with:
  + The first group is developers and operations folks:
    - **[SLIDE X - Cherry pie]** By the way, food is the visual theme for today, mostly pie, because I love pie. Each group is represented by a different slice.
    - This relationship can be the most challenging. A lot of the practices here relate to building trust and credibility, showing that you are working together as partners toward a shared goal.
    - First, be respectful: **[SLIDE]**
      * If you have a question for a developer, don’t interrupt flow unless it’s urgent. If you see her at her desk, headphones on, nothing but terminal windows and IDEs on her monitors, fingers banging away on the keyboard, it’s probably not a great time to interrupt. Maybe it should be an email, or let it wait til you run into her in the hallway.
      * And we’re going to talk more later about using the different channels of communication well.
    - Ask a lot of questions, but do your homework too - don’t make them be Google for you. Read the docs, and ask them questions that only they know the answers to.
    - Similarly, when you find a potential issue, try to do some debugging, some narrowing down before reporting it. If a dev sees that you have used your skills to do at least first-level debugging when you find something questionable, they can see that you’re not just “throwing it over the wall to them”.
      * This concept extends to your relationship with your ops peers as well - being able to be your own first-line ops support is greatly appreciated. **ANY** time you ask for help, it makes others feel a lot better about helping if they can see you’ve tried at least some basic investigation and problem-solving.
      * All of these are ways to establish your credibility as someone who works outside their box, who wants to pull more than their weight, for the sake of the team. This also helps set a high standard for the team.
    - Insert yourself into design reviews. This one is a little more subtle, and I kind of stumbled into it: I already had a feeling that some design decisions were being made without QE. Then after a grooming meeting one day, I overheard a hallway conversation between a BA and dev on a story we’d just groomed. I walked up and joined them, and they realized immediately that I should’ve been there all along. Watch for conversations, meetings, etc that look like you should be in, especially early in a project or story, and politely join. You have questions and ideas to contribute to these conversations, and the team needs you to be there. This can be a way to start an informal practice of 3 Amigos meetings, which I will come back to later.
    - Read code reviews - Understanding how the app code works makes you a better tester.
    - Related to that, work in the same code language as them, if possible.
      * tell the story of our golang adventure
      * benefits:
        + devs can help review test code
        + both can share practices like branching strategies, code libraries, document management, CI/CD integration
    - keep learning, always
      * this could be a good life motto
      * also helps w/ credibility
    - share the things you're learning
      * do demos or blog about best test practices or things you're reading. The teams I work with set a high standard for regular demos, tech talks, and blog posts, and this practice makes everyone better.
    - help them advocate to mgmt for things that will make their lives easier - often these things also improve quality. I’m talking about anything from software licenses to additional hardware, from static analysis tools to IDEs to standalone performance test environments.
      * help them advocate for monitoring of production and other environments; ie, “shifting testing to the right”. We have adopted a log analysis tool and an app monitoring tool that have been VERY helpful in our performance analysis, alerting for problems, and in understanding our customers’ behavior and needs better.
      * You may have to back up your requests with data. Make an estimate of how much time (and therefore money) is saved when a bug is found in development, versus in production.
  + with BAs, Scrum Masters, and Agile Coaches
    - **[SLIDE - Apple Pie]**
    - Please note - we have Business Analysts in our organization. This function may be served by a Project Manager, Product Owner, or someone else in yours. The principles still apply.
    - Remember - BAs are a crucial channel to your customer! You should probably be talking to them almost as much as to devs. And the same goes for your Agilists - they help keep the team on the rails!
    - **[SLIDE] WARNING -** “Agile” ahead! I’m about to talk about how we practice Agile. And because it’s Agile, it’s probably different from the way you do it!
    - **[SLIDE]** Four essential types of BA communication:
      * Discovery or “pre-grooming”
        + The BA talks with the Product Owner or Stakeholders to get an initial understanding of what is needed.
        + In my experience, this is mostly helpful from the standpoint of gaining context/business domain knowledge. It’s especially informative when you have an entirely new product or functional area. It can be asynchronous.
        + Let your BA know you’re interested.
      * Grooming - see Team Rituals
      * BDD - Behavior Driven Development
        + This consists of specifications written by the BA in plain language, using simple conventions, that can be executed as tests, in combination with the underlying framework code which is written by QEs or Devs.
        + Through these written examples, BDD drives shared understanding.
        + Can be practiced on its own or as part of:
      * “Three Amigos”
        + It is a group, and it is a specific meeting
        + The three essential perspectives: BA, Developer, and QA.
        + The meeting usually occurs between grooming and start of development on the given story. It focuses on how to test that story, and in a way similar to BDD, the conversation creates shared understanding and drives out bugs - EARLY!
        + Rebecca Mayes is doing an interactive session on this right after this talk, in this very room!
      * Work with your scrum master/agile coach to establish a clear Definition of Done and exit criteria. This can take a good bit of work, but having these standards helps stories flow smoother and more quickly. For instance, on our Kanban board, unit tests have to be complete to an 80% line coverage standard and passing before the story can leave the development column. To move from the QA to the Staging environment, any cross-browser testing has to be completed. There are many others.
      * Extending this, talk strategy with them on a regular basis to re-examine how things are working and what could be improved.
      * Get them to share their agile metrics with you like burndown charts and average time to completion per story. They can help you drill into these to find where you may be spending too much time.
  + with Product Owners
    - **[SLIDE - Pumpkin Pie]**
    - Maybe you don't have a direct connection to them
    - Create one - reach out, go to lunch
    - ask about the customer, about specific bugs
    - share with them what you're doing to build in quality.
    - All of this builds trust, which you will need someday - <Tell the story about Shea letting us take a sprint to catch up on unit (rspec) tests>
  + with other testers
    - **[SLIDE - Chocolate Pie]**
    - With testers on your team:
      * We do a weekly sync on:
        + priorities
        + upcoming projects and the implications of those
        + infrastructure needs
        + “pre-grooming” stories to indicate the testing we know will be needed, plus questions to get answered
      * We have our own email list, plus we use IRC for chat.
    - With testers outside of your team
      * Ask about their work
      * Share your work, **humbly** and with a genuine interest in getting feedback and learning from others.
      * build community
        + setup internal confs

**[SLIDE X - QECamp]**

Red Hat has a series of internal quality conferences called QECamp, where we can share what we’re doing and learning. We have a team of volunteers from various groups that work together to put it on. It’s great networking and cross-pollination of ideas.

By the way, this work has led to a new test conference, sponsored by Red Hat, and open to the public. <https://opentestcon.org/>

It’s in Beijing on March 30-31, 2020, and the CFP is open now - please check it out!

The theme is related to what we are talking about right now - “Quality is Everyone’s Responsibility”

* + - * + setup informal viewings of outside confs - book a room

OnlineTestConf - organized by Joel Montvelisky of PractiTest - good content, and FREE!

STPCon shares really good webinars and podcasts, and they’re FREE!

Ministry of Testing - FREE!

StarWest and StarEast have virtual confs - FREE!

* + - * + setup a mailing list or chat channel for all QEs in your organization, to share articles, books, local meetups and other events
        + All this encourages a growth mindset among your peers.
        + be aware that not all will respond
  + with the whole team
    - **[SLIDE - Whole Pie]**
    - **[Slide X - team meeting]**
    - Influence the team rituals, agile or otherwise. **[SLIDE]** If you have story grooming, for instance, is there time allotted for discussing test-related acceptance criteria?
      * What kinds of questions should we ask in grooming a story? Examples:
        + For an input field, what are the boundary values and allowable data types? Is the checking done by a centralized, reusable routine?
        + How important is performance for this story; should we measure it?
        + How should the front-end behave when the back-end goes down?
        + These are just examples; obviously there are many types.
        + But ALWAYS ask with curiosity and humility, to LEARN, not as a “gotcha question” or to show how much you know. That’s just annoying.
      * Prepare well ahead of time. Having a well-formed opinion on how and whether to test something is essential to your **credibility**.
        + Being able to provide a good risk-based explanation of why NOT to test something is almost as important as the opposite.
      * At the same time:
        + **[SLIDE]** be able to say, “I don’t know”
        + **[SLIDE]** it’s okay to be wrong
        + **[SLIDE]** be able to change your mind.
        + Thank those who help you even when it hurts. A common trait of the VERY BEST people I have worked with is the **humility** to do these things.
        + LISTEN
      * Asking good questions and listening well is catching - you will notice others will pick up the habit over time.
    - What is the team structure? Is it even a team?
      * Communication is easiest when all the essential roles are organized **[SLIDE]** under shared management, in the same location **[SLIDE]** - this is the ideal.
      * At the other end of the spectrum, the roles are siloed, organizationally and perhaps also physically.
        + How to compensate for these challenges?

team-building

Lightning talks - We started this to help us get to know each other better and to give people an easy first experience with speaking to a group.

IRC karma

Use monetary rewards for high achievement. Our rewards system lets us give coworkers money and recognition for doing great work. And one of the best features is that it works across geographies. So we can “virtually” buy our coworkers in Pune a beer, so to speak.

Share the timezone challenge equally - take turns getting up early or staying up late.

**[SLIDE X - screenshot of group members in daily standup]** VIDEO - always use video for your meetings if at all possible! Make it a team rule, part of the culture! We’ll talk more in a bit about the Hierarchy of Communication Methods

* + - **[SLIDE X - team culture]** Speaking of the team culture, what is yours?
      * Does it include trust?
      * Does it allow mistaeks?
      * Does it support team members, while challenging them to be their best?
        + Communication flows more freely and openly in teams that answer yes to all of the above. And I would maintain that those teams make better software.
      * You may think it's above your pay grade to influence the team culture. But it's not. You should be able to use team meetings to ask questions and start discussions on topics like this. [**Think about inserting Buscemi story**] And any time you model good culture, it can influence your team - and grow your **credibility**. This is grassroots leadership **(LEADERSHIP!)**. For instance, if you find a bug **[SLIDE]**, remember that the whole team is responsible for that bug, the whole team failed to prevent it - communicate it that way. Don’t let a culture of blame sneak in! **[SLIDE X - Blame]**
      * And sometimes creating a strong team culture doesn’t happen and your best efforts are thwarted. Sometimes you have to either live with it or move on.
* Now, let’s look at communicating in special situations **[SLIDE]**
  + Informal - hallway
    - **[Slide]** These spontaneous conversations are often very fruitful, with your team members as well as others. While refilling your water bottle in the breakroom, you run into your DBA. Ask if she got a chance to see that email you sent 2 days ago. Break the awkward elevator silence and ask people what they are working on. They may be starting on a new framework you’ve been curious about. **[SLIDE]** These little connections really add up over time!
    - **[SLIDE X - cappucino]** Set up a lunch or coffee outing - get to know people! Some of my favorite meetings are doing high-level brainstorming or just getting to know someone over a Cortado at Sir Walter Coffee. It can be mentally stimulating to get a change of scenery. We actually have a spreadsheet detailing how far of a walk it is from Red Hat Tower in downtown Raleigh to the good coffee shops.
  + Bad communication!
    - IE, communication smells
    - When you and your teammates are getting 10 exchanges deep into an IRC/Slack/Google Chat conversation, it’s probably time to change the medium! We think we can work through a complex technical conversation while also writing code on the side. Guess what - we can’t! During that extended chat, no other real work is getting done because you’re working in micro-slices of time, context-switching every 10 seconds. Or worse, bad code is probably being written. **[SLIDE]** Roll your chair over to their desk or take it to video chat.
    - Save your energy and time! Like extended chats, also:
      * **[SLIDE]** Avoid checking your email every 5 minutes. If it’s urgent, they should be calling or messaging you! Check your email once an hour, tops.
      * Avoid constant messaging about trivial things. Messaging interrupts productive flow time.
        + One simple response: “Is it urgent?”
        + If not, say, “Hey, I’m in the middle of something, can you give me 30 min.”
        + Or even, “Sure, drop that in an email to me and I’ll get to it as soon as I finish this.” **[SLIDE]** This helps your team form better habits.
    - HCM - Hierarchy of Communication Modes
      * **[SLIDE X - hierarchy of comms modes]**
      * face to face in person **[SLIDE X - Kyle and Brittany]**
      * face to face over video **[SLIDE X -**
      * phone call
      * IRC/google chat/slack/etc
      * Twitter - aka the Anti-Communication Channel
    - And by the way - For audio, **PLEASE PLEASE PLEASE** get a decent headset, probably with a boom mic. People might put up with those crummy earbuds you use that make you sound like you’re inside a bass drum and then they make a staticky, rustling sound when the mic drags across your shirt every time you reach for your coffee cup, but TRUST ME - it has an effect on communication and therefore QUALITY.
  + Difficult communication
    - When it gets crucial
      * Emotions, fatigue, hunger all affect how we communicate.
      * Here’s how to keep it professional when you receive an imperfect message from an imperfect human:
        + First, breathe - seriously
        + Assume positive intent
        + Even if the message is unprofessional - maybe it is blaming, or it targets you (or someone else) instead of the action/behavior/mistake, isolate the truth in the message and focus on that.
      * Another difficult situation is when we disagree with developers as to how something should be tested, or the extent to which something should be tested.
        + A recent story: A library we'd written mostly as an experiment, without much unit or other test coverage, started finding its way into production usage, delivering data to customers. We had a story on the backlog to catch up on the coverage, but it moldered. When the library suddenly became part of a major new project, I cried foul, pointing out the risk. Dev pushed back because we truly were slammed with other work at the time. My first reaction was to freak out and escalate the situation. My reptile brain was fully engaged! **[SLIDE X - reptile brain]** Fortunately, my hominid brain regained control before I finished the angry email, and I reworded it to include some proposals for how we might solve the coverage problem. The dev lead responded with another proposal, I thanked her for the idea, and we continued negotiating until we had a good 80% solution. And she awarded me karma in our IRC channel for championing quality! We found a third way, in large part due to our willingness to take the high road.
        + The point is, in situations like this, to take a deep breath, and keep an open mind **and** a determination to find an approach that works for all involved. This will enable you to reach the best outcome for the whole team, and most importantly, the customer.
      * Communicating as an introvert
        + Are you an introvert? **[Show of hands]** So am I. Every time I speak, I get a great adrenaline charge, and when it’s over, my energy is completely drained. But I love it, if I think I’ve provided anything of value to another.
        + **[SLIDE X - cereal]** This is a favorite introvert food for me, because...
        + I’m no psychologist and I won’t try to tell you how to communicate more/better as an introvert, but I’ll share my experience.

Toastmasters and lightning talks are highly recommended. As I said earlier, lightning talks are the easiest possible way to get more comfortable speaking to a group.

And even though you will probably never become Super Gregarious Person (thank goodness - we have plenty of blabby extroverts as it is), It gets better with practice, I promise.

* + - * + And honestly, modern software team processes don’t have much room for the completely noncommunicative approach any more. You simply can’t deliver quality software at speed without LOTS of good communication.
* **Closing**
  + I hope you heard something today that you want to go back and try. Please get back to me with your stories of what helped you succeed at making quality part of the fabric of your team.
* **Time to play Stump the Speaker! (5 min)**
* **Q&A**
* **THANK YOU**

===

To Do:

* Add github info to contact info
* PRACTICE PRACTICE PRACTICE
* Get swag for the winner of Stump the Speaker
* Get a slide clicker to take w/ you - ORDERED due by Sep 4
* Submit final talk by Aug 31
* Have copies of the talk in multiple places
* Get a backup battery