Hard Lessons About Soft Skills

Marlena Compton & Gordon Shippey
The Writers

Marlena Compton
- Tester at Mozilla, formerly Atlassian & Equifax
- @marlenac
- marlenacompton.com

Gordon Shippey
- Licensed Counselor
- Formerly a tech writer at Equifax
- vastandinfinite.com
The Abstract

- The psyche of a software tester
- The tester’s paradox
- Crucial conversations for testers
Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.
What I learned about myself when the walls came down...

I AM smiling
Where did this tester come from?

- Teammates told me it was me vs. them
- Everything I know about negotiating I learned from my dad, a successful attorney...who will cut you.
- Popular testing culture insists the tester persona is a force of destruction and the test cycle is judgment day
Some asshole behaviors...

- Threats and intimidation
- Flame-y emails
- Public shaming
- Interruptions
- Nasty Looks
- Personal Insults
- Sarcastic jokes and teasing as an insult delivery system
Where does this happen?

- Stand up meetings
- Reviewing specifications
- Software demos
- Bugs, especially bug comments
- Email, Irc, twitter
How do you know if you're a testing asshole?
Take the A.R.S.E test and replace the plural nouns with “developers”
Wait...are you saying testers are assholes?
# The Tester’s Paradox

- Many testers are solo testers or a small band because money is not allocated for testing.
- Often, testing occurs when a team is late and over budget.
- Testers are supposed to expose problems/critique an application.
- Easy to feel ostracized and ignored.
- When testers finally get to work, the team’s nerves are frayed and tempers are short.
- Participation on a software team is limited to dispensing negative information. Easy for critique to be misconstrued as blame.
It takes 5 positive interactions to counter one bad one.
Testers are set up to fail at communication.
um...so what?
Developing software happens at a creative edge

- If people don’t feel safe talking to each other, problem solving will occur in personal silos
- More time is spent finger-pointing than problem solving
- Developers are more likely to hide problems from testers than to be upfront about them.
- An unsafe work environment can result in heightened anxiety, depression and trouble concentrating
Testers care.
Let's acknowledge our challenges and move on

- Understand our own stress reactions
- Control what we can control
- Set the communication example
- Work on improving our “crucial conversations”
The conclusion from 25 years of research

- The best companies rely on skillful face to face discussion over procedures and forms to solve problems
- Individuals and interactions over processes and tools
Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.
What is a crucial conversation?

Any conversation between two or more people where:

- Stakes are high
- Emotions are strong
- Opinions vary
Crucial Conversations in Software

- Any discussion involving whether or not a bug is fixed
- Ship/No Ship
- Why a bug happened at all
- Suggesting changes to a feature in development
- Asking for more time to test
- Asking if something is ready to test
- Talking with a developer about seeing the same bug *again*
The guts of a crucial conversation

- Recognize that the conversation has “turned crucial”
- Start with heart
- Look beyond the sucker’s choice
- Create space from emotions with observations
Recognizing Crucial Conversations

- Remember the three elements:
  - high stakes
  - emotions are strong
  - Opinions vary
- They don’t always happen when you plan.
- They don’t always happen face to face
Start with heart

- What do I really want for myself?
- What do I really want for others?
- What do I want for the relationship?
When to hold off...

- If you feel the need to “win”
- If you’re out for revenge
- If you are concerned about remaining safe
Notice Stress Reactions

- We have them because we’re human
- Fight, Flight or Freeze
- Blood is diverted away from our brains
Where do you feel your stress reaction?

- Is it your:
  - Shoulders?
  - Neck?
  - Chest?
  - Hands?
  - Face?
3 steps to avoiding the “sucker’s choice”

1. Clarify what I *want*

2. Clarify what I *don’t want*

3. Combine the two into an “*and*” question to present the brain with a more complex problem

   Ex. I really want to find out why this bug isn’t fixed and I don’t want the developer to think I’m blaming him
Create space from emotions with observations

- Notice how you are feeling
- Notice assumptions you are making
- Find a way to give the other person the benefit of the doubt
Having a fair argument

- Don’t argue until everyone understands what the argument is about
- Don’t argue while generating ideas or solutions
- Once the argument is resolved, cease the criticism
- If the argument gets nasty, take a timeout to turn down the venom
- After the fight is over, recover & repair
Recovery & Repair

- Take the blame fully
- Take control over what you can
- Explain what you've learned
- Communicate what you will do differently in the future.
- *Stick to it.*