How do I Know I am Context-Driven?

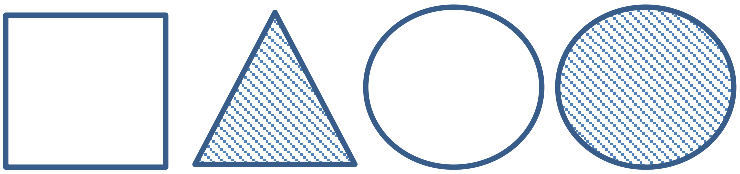
James Bach
@jamesmarcusbach
james@satisfice.com
Consulting Software Testing, Satisfice, Inc.

Thank you to my context-driven colleagues:
Pradeep Soundarajan, Parimala Hariprasad, and Michael Bolton
for reviewing and improving this work!

Do you agree to these principles?

1. The value of any practice depends on its context.
2. There are good practices in context, but there are no best practices.
3. People, working together, are the most important part of any project's context.
4. Projects unfold over time in ways that are often not predictable.
5. The product is a solution. If the problem isn't solved, the product doesn't work.
6. Good software testing is a challenging intellectual process.
7. Only through judgment and skill, exercised cooperatively throughout the entire project, are we able to do the right things at the right times to effectively test our products.

Wait, let’s try something simple…



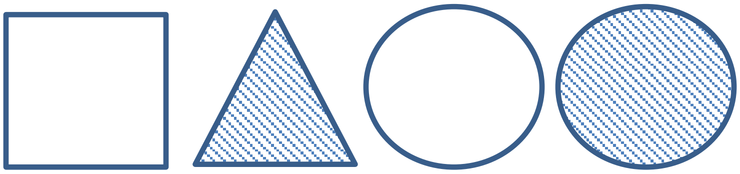
Can we agree? Can we share common ground?

“There are four geometric figures in **blue** on this slide.”

“There is one **square** among those figures.”

“The square is shaded in **blue**.”

Wait, let’s try something simple…



Beware of Shallow Agreement!

Three Different Things are Called "Context-Driven"

- A paradigm (the model by which I understand, experience, value, explain and categorize good testing)
- A community (people I influence and am influenced by who espouse the principles of the Context-Driven School)
- An approach (a heuristic I apply in my test projects)

All these things relate to the seven principles.

What is behind these principles?

- Problem-solving
- Problem-solving and scientific discipline
- Humanism, community and scientific discipline
- Non-linearity and scientific discipline
- Humanism and community discipline
- Problem-solving and humanism
- Problem-solving, humanism, community, non-linearity

1. The value of any practice depends on its context.
2. There are good practices in context, but there are no best practices.
3. People, working together, are the most important part of any project's context.
4. Projects unfold over time in ways that are often not predictable.
5. The product is a solution. If the problem isn't solved, the product doesn't work.
6. Good software testing is a challenging intellectual process.
7. Only through judgment and skill, exercised cooperatively throughout the entire project, are we able to do the right things at the right times to effectively test our products.

(The Approach)

"People evaluating a product
by learning about it through
experimentation…

…in a manner
organized and motivated by
a systematic consideration of all the factors that significantly
influence the problems and
solutions that lie within the
scope of their mission."

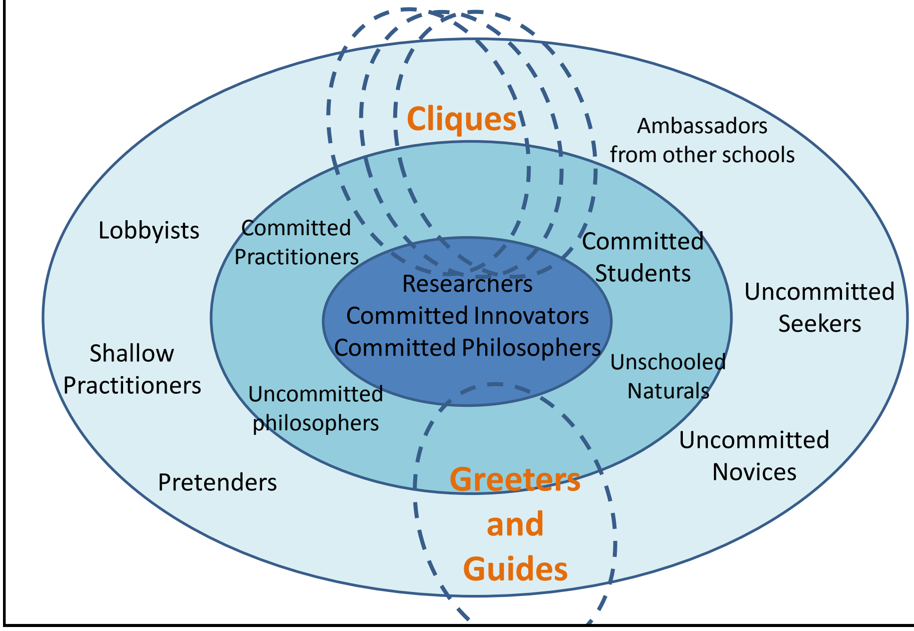
What is

by

Context-Driven

Testing

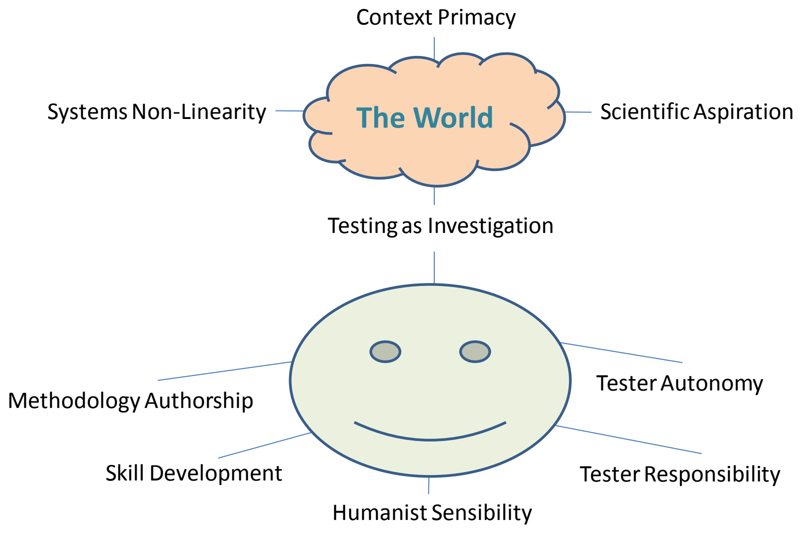
(The Community)



Implicit principles of the Context-Driven School of Testing

- **Context Primacy:** Context is not inert scenery, it embodies vital information, resources, constraints, and other agents that must inform all competent work.
- **Scientific Aspiration:** Folklore is not a basis for a respectable craft. Our work is informed by evidence, cleaned and tempered by skepticism and vigorous debate. Community status is accumulated through demonstrated and demonstrable merit. We avoid groundless and exaggerated claims.
- **Systems Non-Linearity:** Our systems are not practically predictable or reducible in terms of linear or statistical equations. We must use non-linear, cybernetic control methods, and learn to live with uncertainty.
- **Testing as Investigation:** Testing is not just fact checking and it is *not* quality improvement. It is an open-ended investigation and learning process focused on discovering problems.
- **Humanist Sensibility:** Technical workers are not interchangeable resources. All technical work is done by unique, unreliable people, and to be good at technical work we must develop as people.
- **Tester Autonomy:** We are not robots or slaves: we have *agency*. We manage the value of our time and bear responsibility for doing ethical work. We must cultivate the courage to do that.
- **Tester Responsibility**: We are not alone. We work within a social network in which value is constructed and responsibility is shared. This happens on project, corporate, professional, and societal levels.
- **Methodology Authorship:** Ignorantly mimicking behavior is not competent work. Competent testers must design (or adapt) and test their own practices and heuristics.
- **Skill Development:** Technical work is not brute labor. Methodology skill , in both tacit and explicit form, is absolutely required to fulfill our mission, and development of such skill is an ongoing obligation.

Implicit principles of the Context-Driven School of Testing



What About Bad Work?

Bad work is any work that YOU think should be done differently in order to reasonably fulfill YOUR obligations to your client, project, company, or society in terms of the Context-Driven Testing principles as YOU interpret them.

You may experience pressure to perform bad work or to allow your work to suffer due to the impact of other people's bad work.

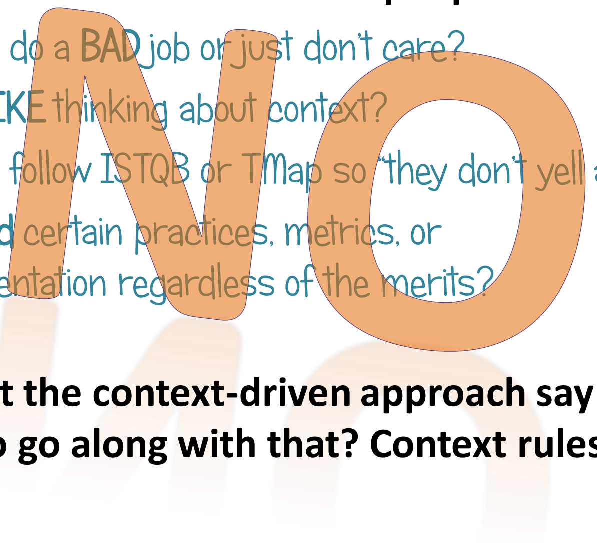
What if the context includes people who…

…want to do a BAD job or just don't care?

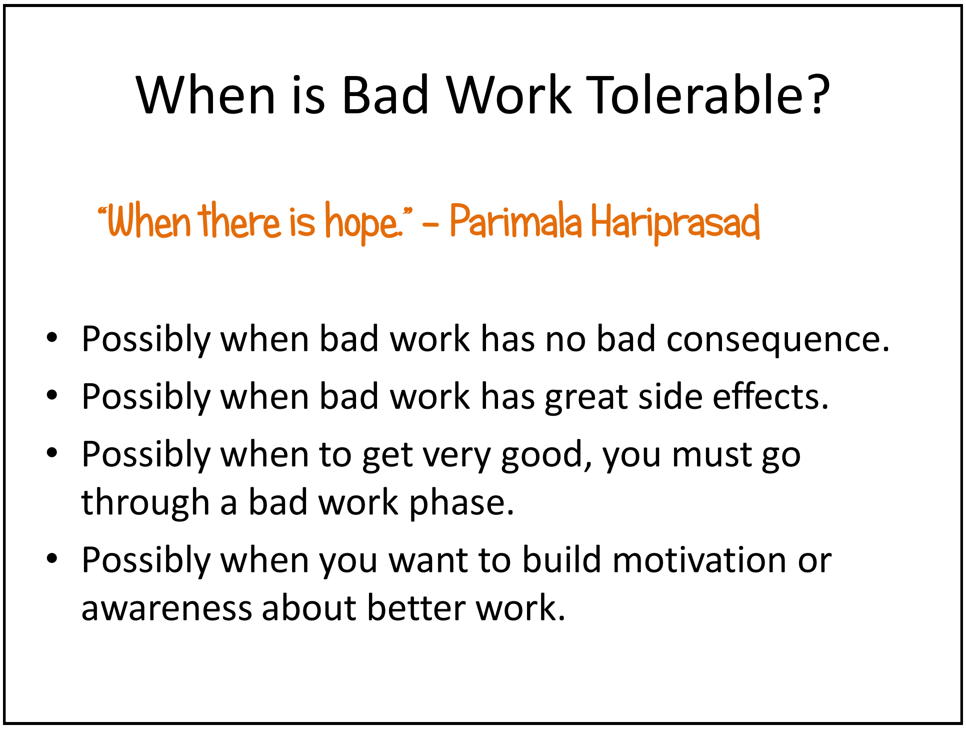
…don't LIKE thinking about context?

…want to follow ISTQB or TMap so "they don't yell at me?"

…**demand** certain practices, metrics, or documentation regardless of the merits?



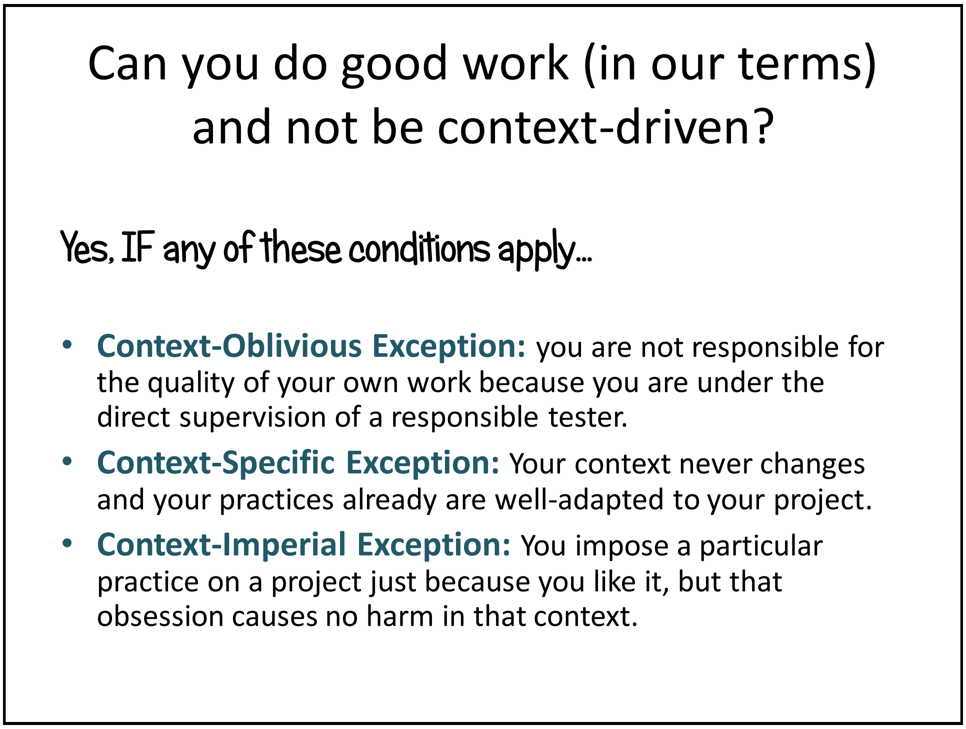
Doesn't the context-driven approach say that we have to go along with that? Context rules, right?



When is Bad Work Tolerable?

"When there is hope." – Parimala Hariprasad

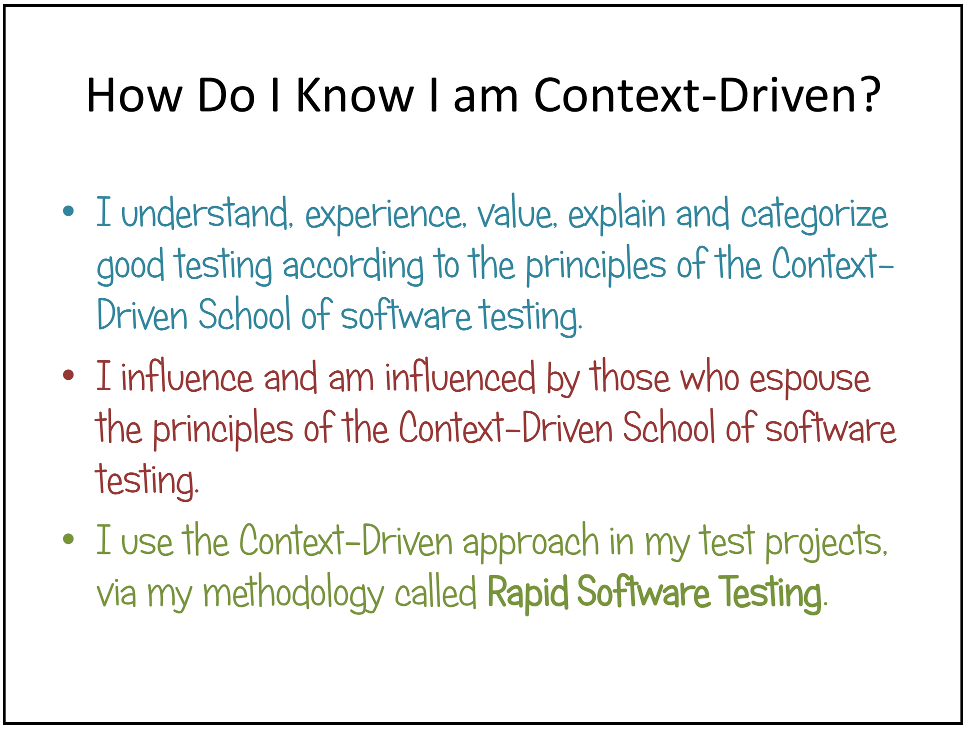
- Possibly when bad work has no bad consequence.
- Possibly when bad work has great side effects.
- Possibly when to get very good, you must go through a bad work phase.
- Possibly when you want to build motivation or awareness about better work.



Can you do good work (in our terms) and not be context-driven?

Yes, IF any of these conditions apply…

- **Context-Oblivious Exception:** you are not responsible for the quality of your own work because you are under the direct supervision of a responsible tester.
- **Context-Specific Exception:** Your context never changes and your practices already are well-adapted to your project.
- **Context-Imperial Exception:** You impose a particular practice on a project just because you like it, but that obsession causes no harm in that context.



How Do I Know I am Context-Driven?

- I understand, experience, value, explain and categorize good testing according to the principles of the Context-Driven School of software testing.
- I influence and am influenced by those who espouse the principles of the Context-Driven School of software testing.
- I use the Context-Driven approach in my test projects, via my methodology called **Rapid Software Testing**.