
expoQA®25
MADRID
May 20th,
21st & 22nd
2025
sogeti
expoQA
expoqa.eu

Bright Future or Dark Age: Why Testers Matter

James Bach, Satisfice, Inc.
james@satisfice.com
www.satisfice.com
+1 (360) 440-1435

Michael Bolton, DevelopSense
mb@developsense.com
www.developsense.com
+1 (416) 656-5160

This talk, TL;DR version

How does the business get what it wants?

Testing Responsibly

Discover something worth building

Envisioning Success

Study what we built

This is how the business finds out whether it *really* got what it *really* wanted.

This is how the business plans to keep customers happy and make money.

Low cost of developmentand support helps keep the business sustainable.

This is how the business gets something it can sell.

Build some of it

Anticipating Failure

Build it cleanly and simply—with change in mind

Building Diligently

How does the business get what it wants?

Testing lights the way.

Testers strive to discover the **true status** of the product.
Testing makes informed decisions about quality possible,
by looking critically at software.

Testers focus on the possibility of **TROUBLE**.

Or do we?

We had better, because
there's a world of trouble coming.

This talk, longer version…

Maybe You've Heard of Stockton Rush

A REPORTER AT LARGE

THE TITAN SUBMERSIBLE WAS "AN ACCIDENT WAITING TO HAPPEN"

Interviews and e-mails with expedition leaders and employees reveal how OceanGate ignored desperate warnings from inside and outside the company. "It's a lemon," one wrote.

By Ben Taub

July 1, 2023

Maybe You've Heard of Stockton Rush

Toward the end of that year, Lochridge became increasingly concerned. OceanGate would soon begin manned sea trials for Cyclops II in the Bahamas, and he believed that there was a chance that they would result in catastrophe. The consequences for Lochridge could extend beyond OceanGate's business and the trauma of losing colleagues; as director of marine operations, Lochridge had a contract specifying that he was ultimately responsible for "ensuring the safety of all crew and clients."

On the workshop floor, he raised questions about potential flaws in the design and build processes. But his concerns were dismissed. OceanGate's position was that such matters were outside the scope of his responsibilities; he was "not hired to provide engineering services, or to design or develop Cyclops II," the company later said, in a court filing. Nevertheless, before the handover of the submersible to the operations team, Rush directed Lochridge to carry out an inspection, because his job description also required him to sign off on the submersible's readiness for deployment.

https://www.newyorker.com/news/a-reporter-at-large/the-titan-submersible-was-an-accident-waiting-to-happen

Maybe You've Heard of Stockton Rush

On January 18, 2018, Lochridge studied each major component, and found several critical aspects to be defective or unproven. He drafted a detailed report, which has not previously been made public, and attached photographs of the elements of greatest concern. Glue was coming away from the seams of ballast bags, and mounting bolts threatened to rupture them; both sealing faces had errant plunge holes and O-ring grooves that deviated from standard design parameters. The exostructure and electrical pods used different metals, which could result in galvanic corrosion when exposed to seawater. The thruster cables posed "snagging hazards"; the iridium satellite beacon, to transmit the submersible's position after surfacing, was attached with zip ties. The flooring was highly flammable; the interior vinyl wrapping emitted "highly toxic gasses upon ignition."

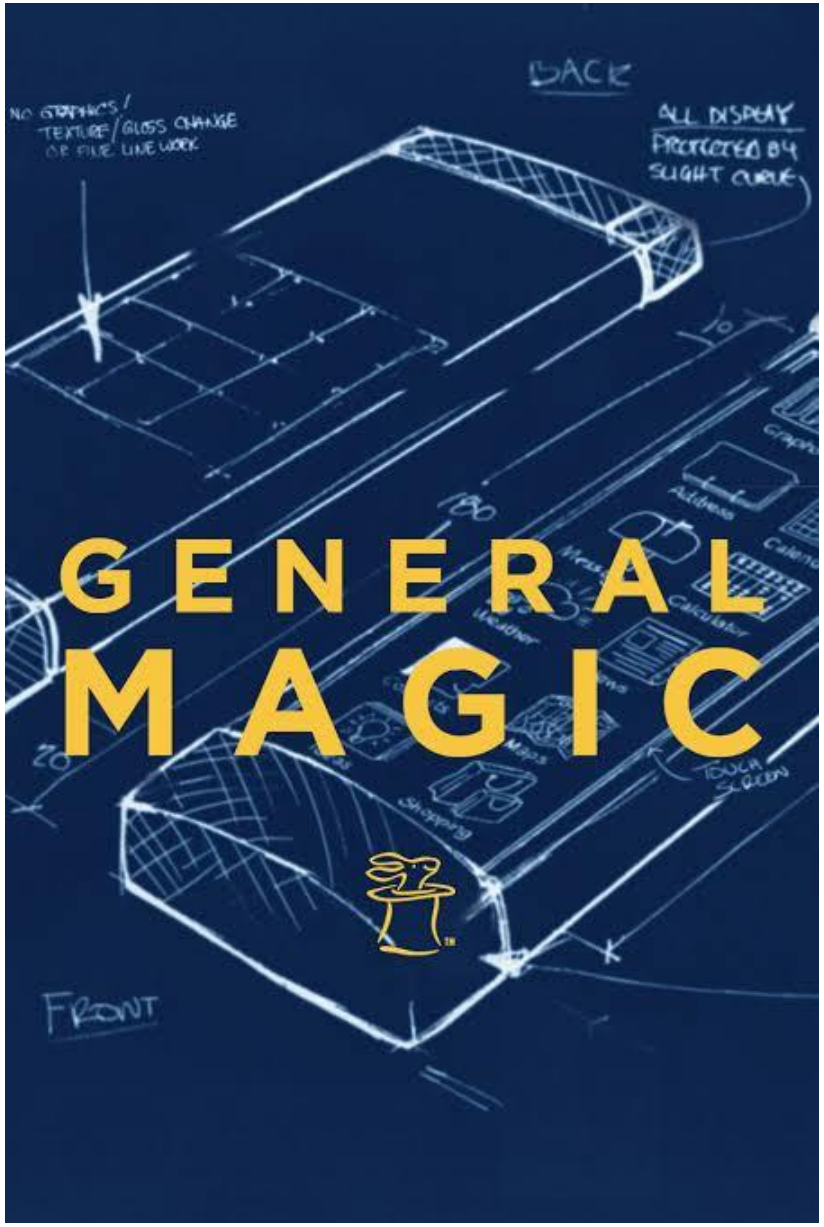
https://www.newyorker.com/news/a-reporter-at-large/the-titan-submersible-was-an-accident-waiting-to-happen

Maybe You've Heard of Stockton Rush

Rush was furious; he called a meeting that afternoon, and recorded it on his phone. For the next two hours, the OceanGate leadership insisted that no hull testing was necessary—an acoustic monitoring system, to detect fraying fibres, would serve in its place. According to the company, the system would alert the pilot to the possibility of catastrophic failure "with enough time to arrest the descent and safely return to surface." But, in a court filing, Lochridge's lawyer wrote, "this type of acoustic analysis would only show when a component is about to fail—often milliseconds before an implosion—and would not detect any existing flaws prior to putting pressure onto the hull." A former senior employee who was present at the meeting told me, "We didn't even have a baseline. We didn't know what it would sound like if something went wrong."

https://www.newyorker.com/news/a-reporter-at-large/the-titan-submersible-was-an-accident-waiting-to-happen

Maybe You've Heard of General Magic



https://www.computerhistory.org/revolution/mobile-computing/18/323/1780

Maybe You've Heard of General Magic

While the company did have a target customer of "Joe Sixpack", no one took it seriously. It was a bit of a company joke. Engineers simply designed features they liked and assumed that Joe would like it too. Much later, Fadell described Joe Sixpack in his 2022 book *Build*:

> General Magic's target customer was "Joe Sixpack." Seriously. It's a derogatory caricature of an average American slumped on his couch, drinking beer, watching football, not thinking about much of anything. It's a terrible way to imagine your customer. And even though we repeated it over and over, claimed we were doing it all for him—it was meaningless. **Because even if Joe Sixpack existed, he was never going to buy the Magic Link** (*emphasis added*)**.** This was before the internet was remotely accessible, at a time when most people didn't have a desktop computer, didn't have email, couldn't imagine mobile games or movies.

https://commoncog.com/c/cases/general-magic/

It was a great idea! But… Developers be like…

Tony Fadell: "We can be engineers without a manager and we know what's best. Managers are just going to get in the way. We don't need program managers, we don't need any of that stuff. We're just going to make it happen."

Marc Porat: "There was a fearlessness and a sense of correctness; no questioning of 'Could I be wrong?'. None. Because that's what you need to break out of Earth's gravity. You need an enormous amount of momentum. And that momentum comes from suppressing introspection about the possibility of failure."

Or maybe you read the newspaper today…

Or maybe you read the newspaper today…

Why

weakest argument

- Empirical (
- Skille
-

ve)

ndant)

f ambivalent)

ad of preoccupied)

Responsible companies hire testers because good testers are...

Empirical (instead of wishful)

- finding bugs with respect to the intentions of the designer/developer
- predicting if the users will accept it, and anticipating how they might want it to be improved
- understanding the risk of bugs yet unknown
- discovering surprising behavior and making sense of it
- discovering how the product behaves vs. time and data and environment
- discovering the practical limits of the product
- discovering how the product fails and how it recovers from failure
- following-up on specific user experiences or complaints
- finding new uses for the product
- understanding how recent changes may have affected the product
- understanding the product in comparison with competitors
- understanding how the product interoperates or integrates with the ecosystem
- discovering how the product may violate relevant laws or industry standards

Skilled (instead of merely intuitive)

- accumulating and curating professional and institutional knowledge about testing
- extracting test ideas from a variety of sources and materials
- gathering and curating repositories of test data
- acquiring, adapting, commissioning, or developing tools as needed
- developing tacit knowledge about the world of the users, rather than relying solely on explicit specifications
- making educated judgements about what to test and how and when
- able to select the least burdensome approach that gets the job done
- tracking what has been tested and how well it has been tested
- looking ahead to anticipate and be ready to test upcoming products
- evaluating their process and themselves as they go
- advocating for testability
- explaining and justifying their work
- reporting test results in a manner that allows their clients to make timely and informed decisions
- resisting bias due to team's unhappiness to hear unpleasant news?

Different (instead of redundant)

- focusing on trouble rather than success; testers have faith that trouble exists
- focusing on users and usage rather than technology
- considering the "big picture" while working with the details
- exploring obscure possibilities and possible diversities
- having "critical distance"; unbiased by detailed knowledge of how the code should work
- functioning as an independent source of information about the status of the product

Motivated (instead of ambivalent)

- enjoying the process of testing, rather than seeing it as a chore or distraction
- being supported in doing it
- unbiased by the responsibility to fix things
- unburdened by the pride and hope of being a maker

Available (instead of preoccupied)

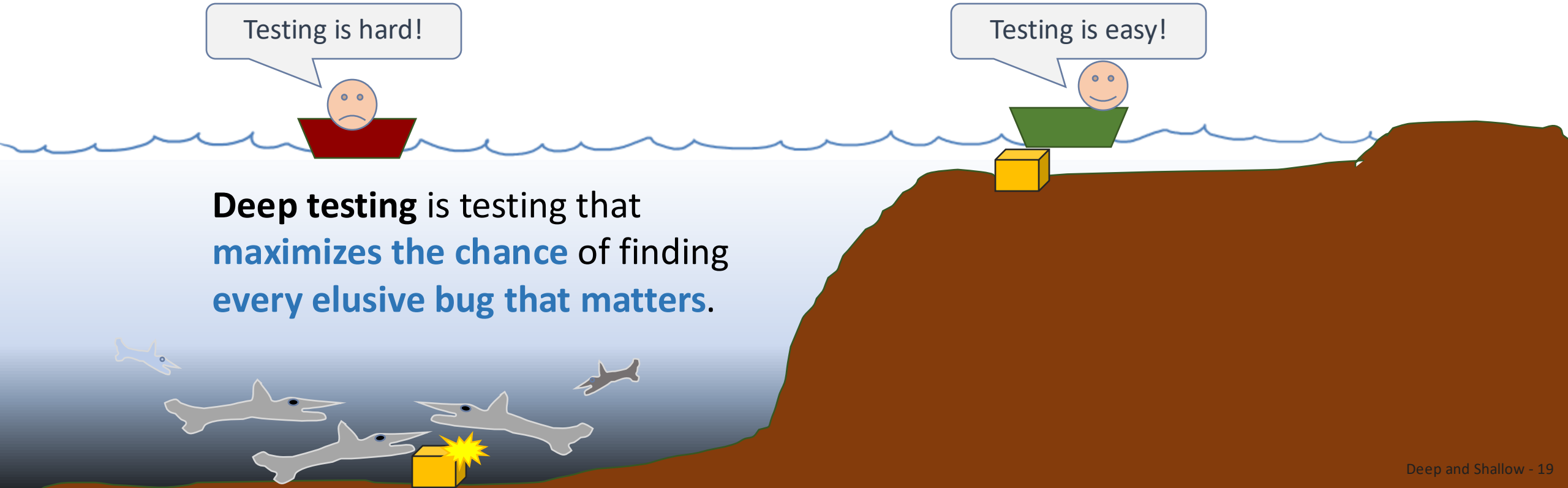
- working full-time in parallel with development, so everyone stays in flow
- having time to follow up on subtle clues and cues that might lead to trouble
- being flexible and responsive to the needs of their clients
- not doing development!

no conflicts of interest

https://www.satisfice.com/download/why-testers

Deep or Shallow?

Shallow testing is testing that **has a chance** of finding **every easy bug**.



Deep testing is testing that **maximizes the chance** of finding **every elusive bug that matters**.

Most developers agree that testing is needed, but they won’t test deeply.

But this is not because they’re lazy, or incompetent testers, or Just Bad People.

“Show me the incentives
and I’ll show you the behaviour.”

—Keith Klain

Four Frames for Testing

Testing Thoroughly

Envisioning Success

High Value of Product

Studying what we built

When people are focused on developing intentions, they often sweep the idea of failure aside from their minds. It's the testing role to notice things that could go wrong, and to ask for testability.

Intention

…ective testing—design-focused testing and review can help us to discover what the customer and the business really want, and how we might go about building it.

Developers like to program. (Lots of testers do too.) Meanwhile, many programmers think this is all there is to testing. But there's a big difference between *checking the build* and *testing the product.*

Discipline

…ew, pairing, shallow testing, and lower-level output checking can help us to confirm that what we have just done is reasonably close to what we intended to do.

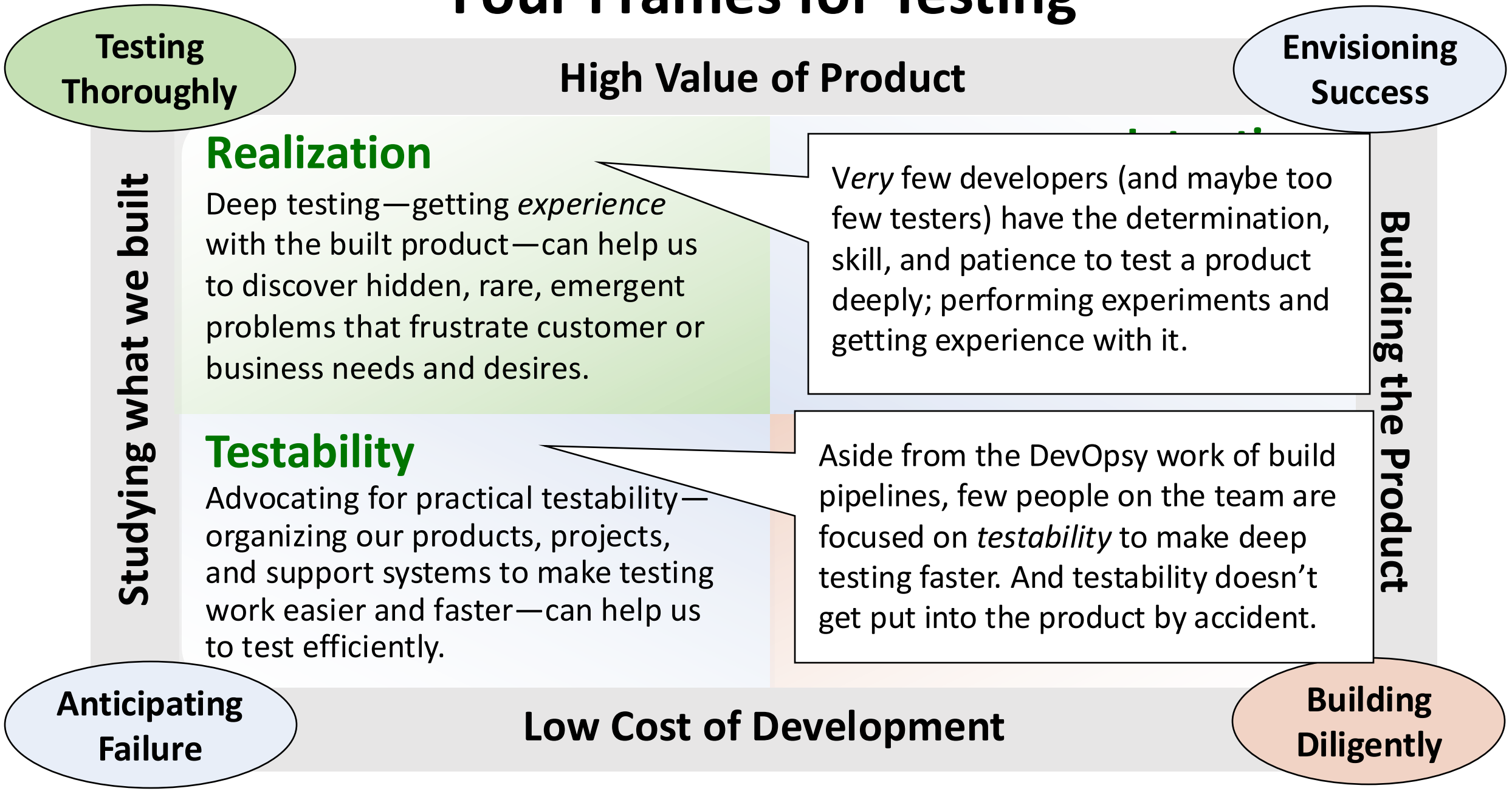
Building the Product

Anticipating Failure

Low Cost of Development

Building Diligently

Four Frames for Testing



The four frames can apply to any and all levels of our work to all the precursors of the final product, as well as to the final product.

Intention

Prospective testing—design-focused testing and review can help us to discover what the customer and the business really want, and how we might go about building it.

Discipline

Review, pairing, shallow testing, and lower-level output checking can help us to confirm that what we have just done is reasonably close to what we intended to do.

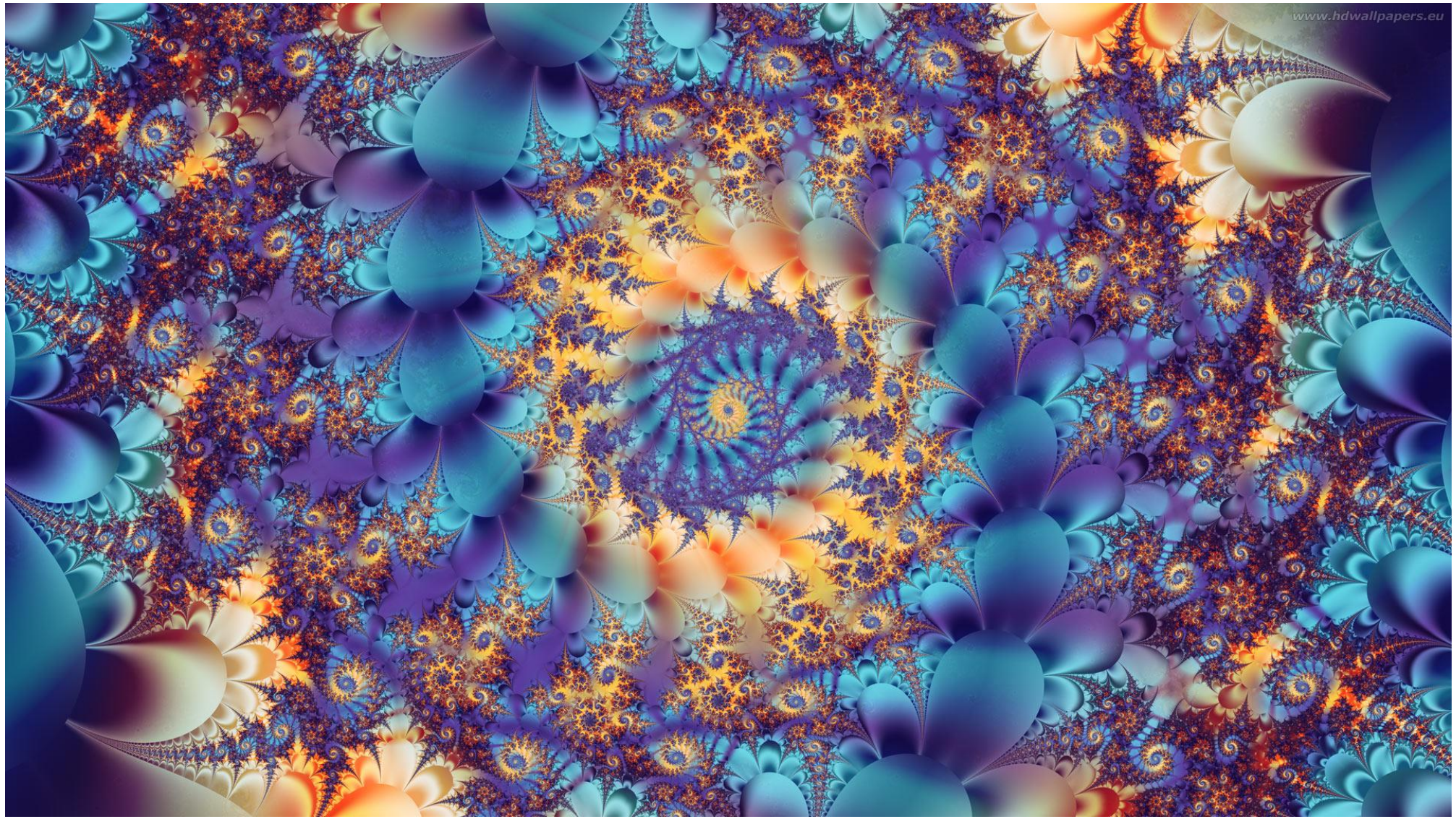
Testability

Advocating for practical testability— organizing our products, projects, and systems to make testing work easier and faster—can help us to test efficiently.

Realization

Deep testing—getting *experience* with the built product—can help us to discover hidden, rare, emergent problems that frustrate customer or business needs and desires.

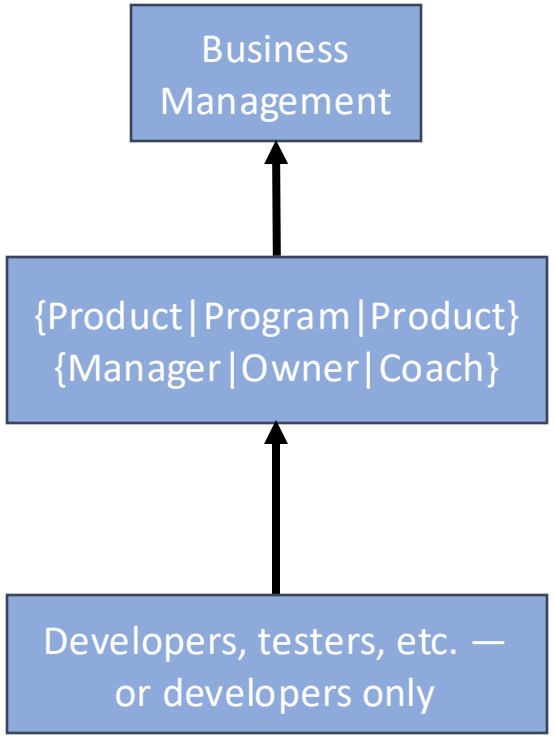
…but not just in lines; not even just loops.



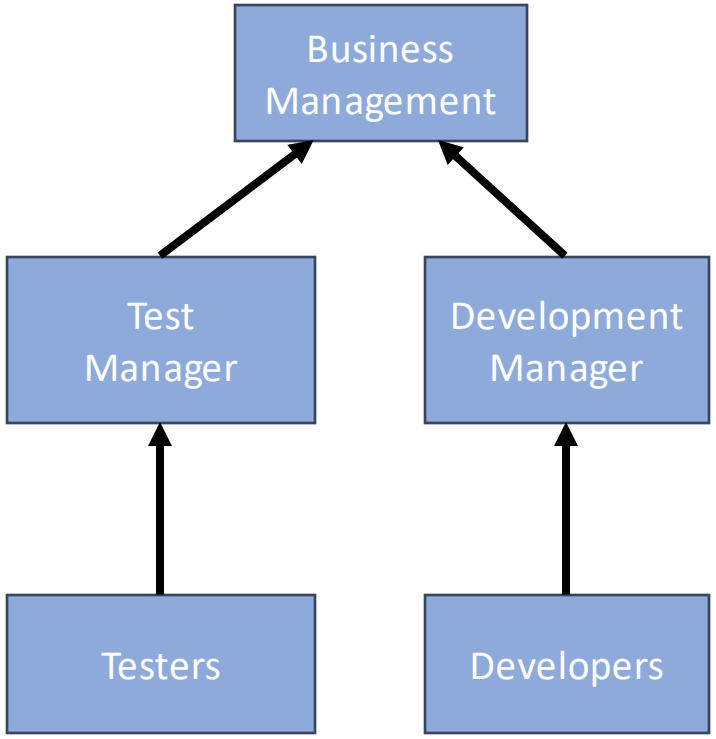
Development is a fractal!

Why Tester Independence?

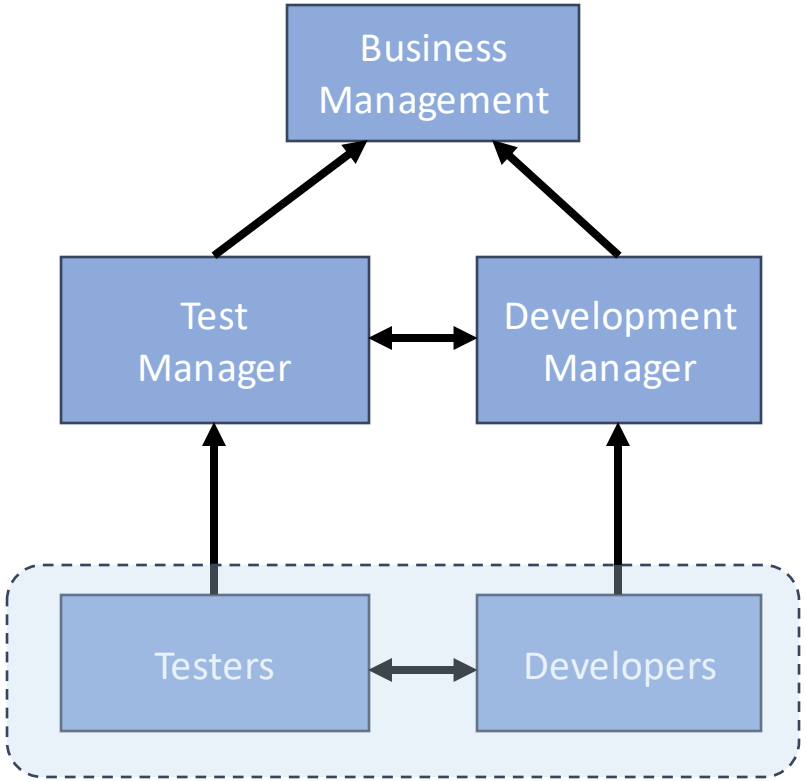
Assume that the team
fully Investigates and reports
the true state of its own problems



Divide the team based on
People Who Built It and
People Who Didn't Build It



Colllaborative access to
product status information
throughout the project community



The Burning Question:

But are we independent?

Not isolation, but independence of thought, and of *stance*

Critical distance is the ability to think differently, oppostionally from another person, and to investigate a matter without pre-judging it.

Testers must maintain critical distance as outsiders to the maker mindset, not emotionally attached to a bug-free product. Testers must love looking for trouble, and finding it.

Critical distance falls apart when we lose our differentness; when we fear displeasing powerful people; when we become attached to certain outcomes; when we focus on confirmation.

Talking about product trouble brings *social* trouble.

Talking about product trouble brings *social* trouble.

Talking about product trouble brings *social* trouble.

Demonstration

"See? It does *this*!"

To *show* something we know

Rehearsal focuses on outcome

Repetition is often ceremonial

Inconsistency in results is undesired and troubling

Variation of factors is risky and may undermine the demonstration

Suppression of risk

Deeper truths are beside the point

Experiment

"See *whatever* it does."

To *challenge* what we *think* we know

Rehearsal focuses on integrity

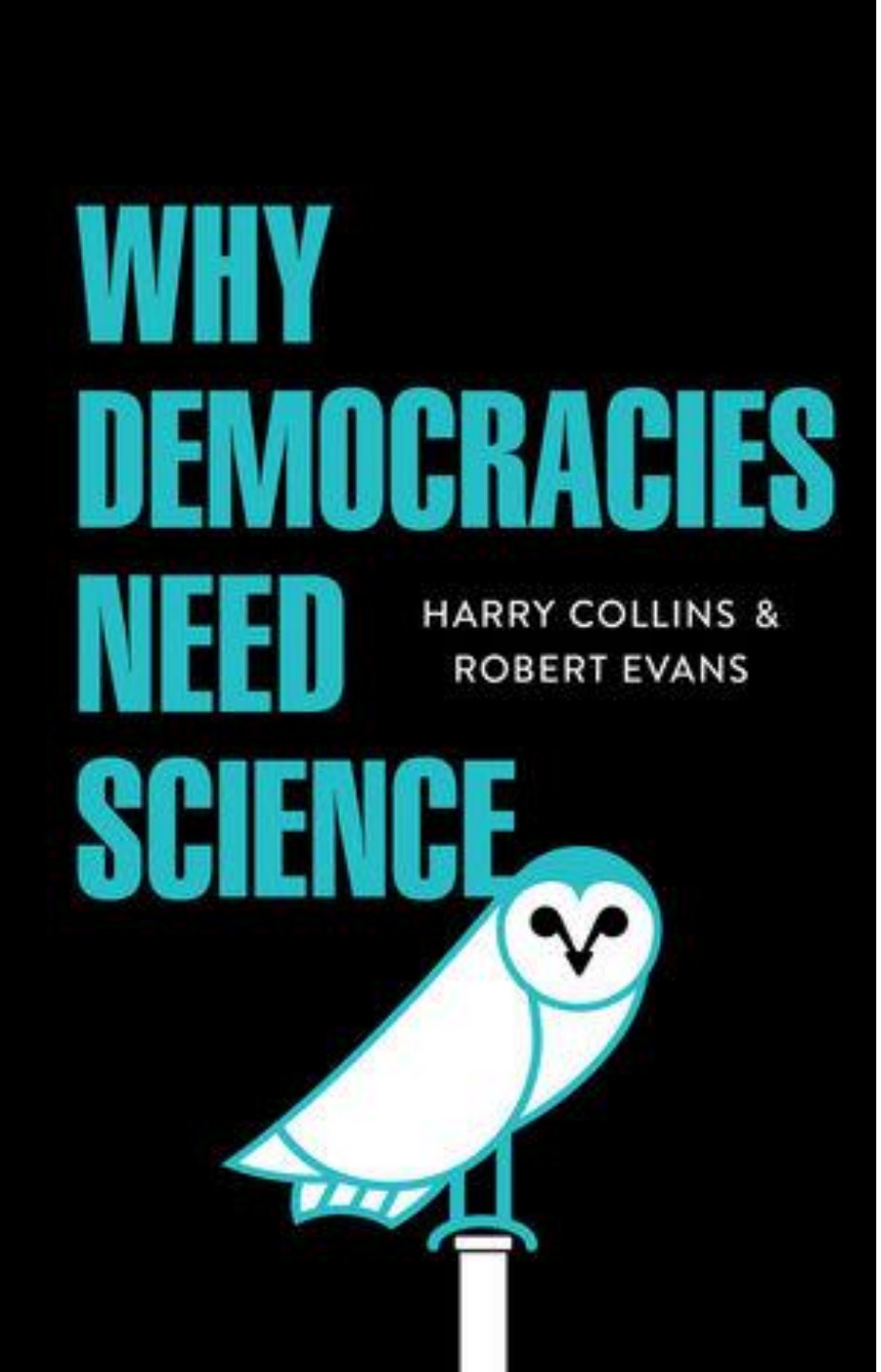
Repetition is contingent

Inconsistency in results is welcomed and intriguing

Variation of factors is desirable and may improve the experiment

Identification of risk

Deeper truths *are* the point

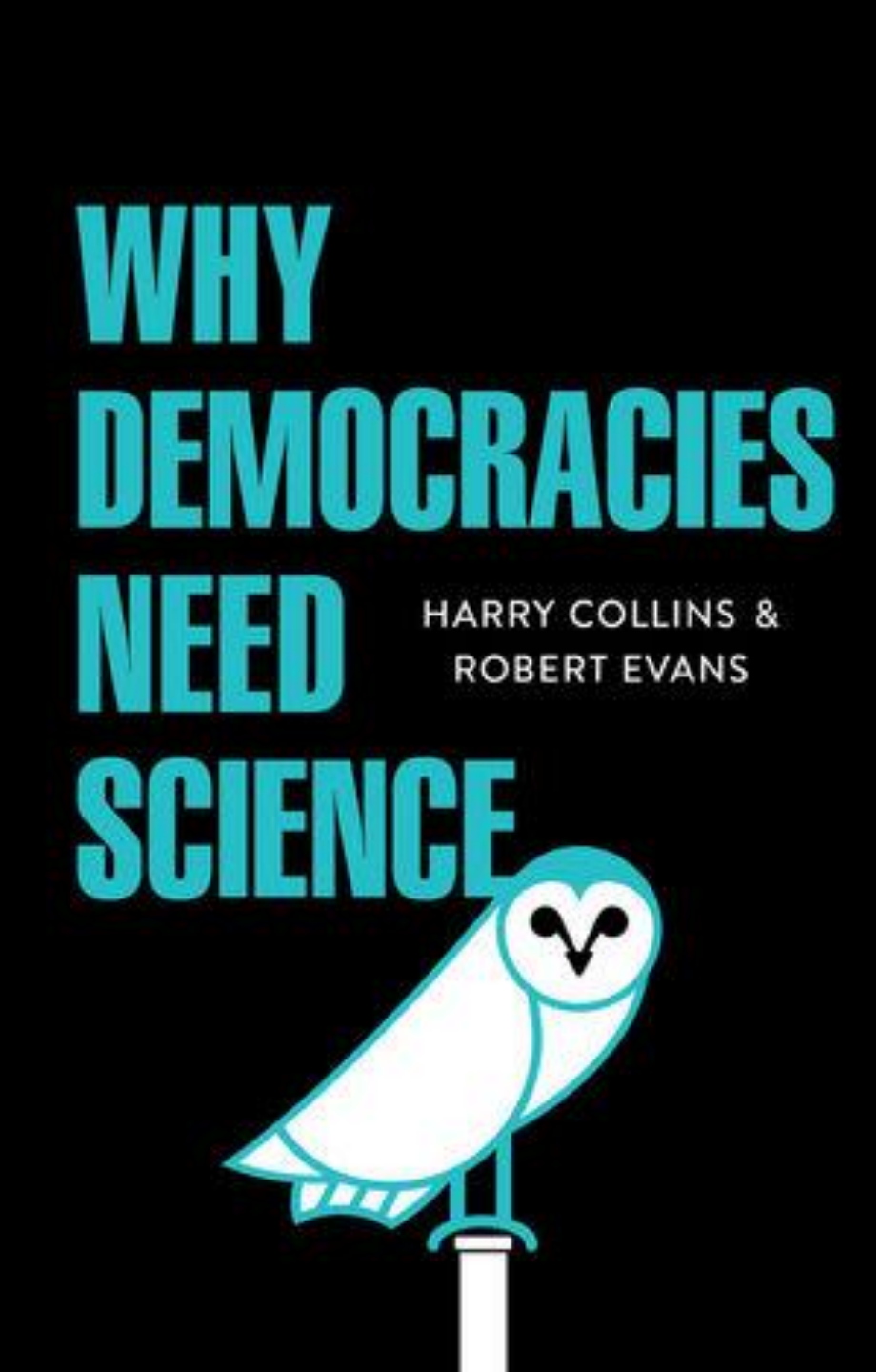


Formative Aspects of Science: Observation

"If one wants to know about some feature of the world, does one prefer to listen to one who has observed that feature of the world, or one who has not viewed that feature of the world?

"We prefer to give more weight to the one who has observed, even though we know that observation is inexact and impure and open to illusions, observer effects, and the influence of the social group."

We must observe directly, but also deeply. We use tools to help us *see* trouble, but we must not allow them to distract or displace us from seeking and finding trouble.



Formative Aspects of Science: Corroboration

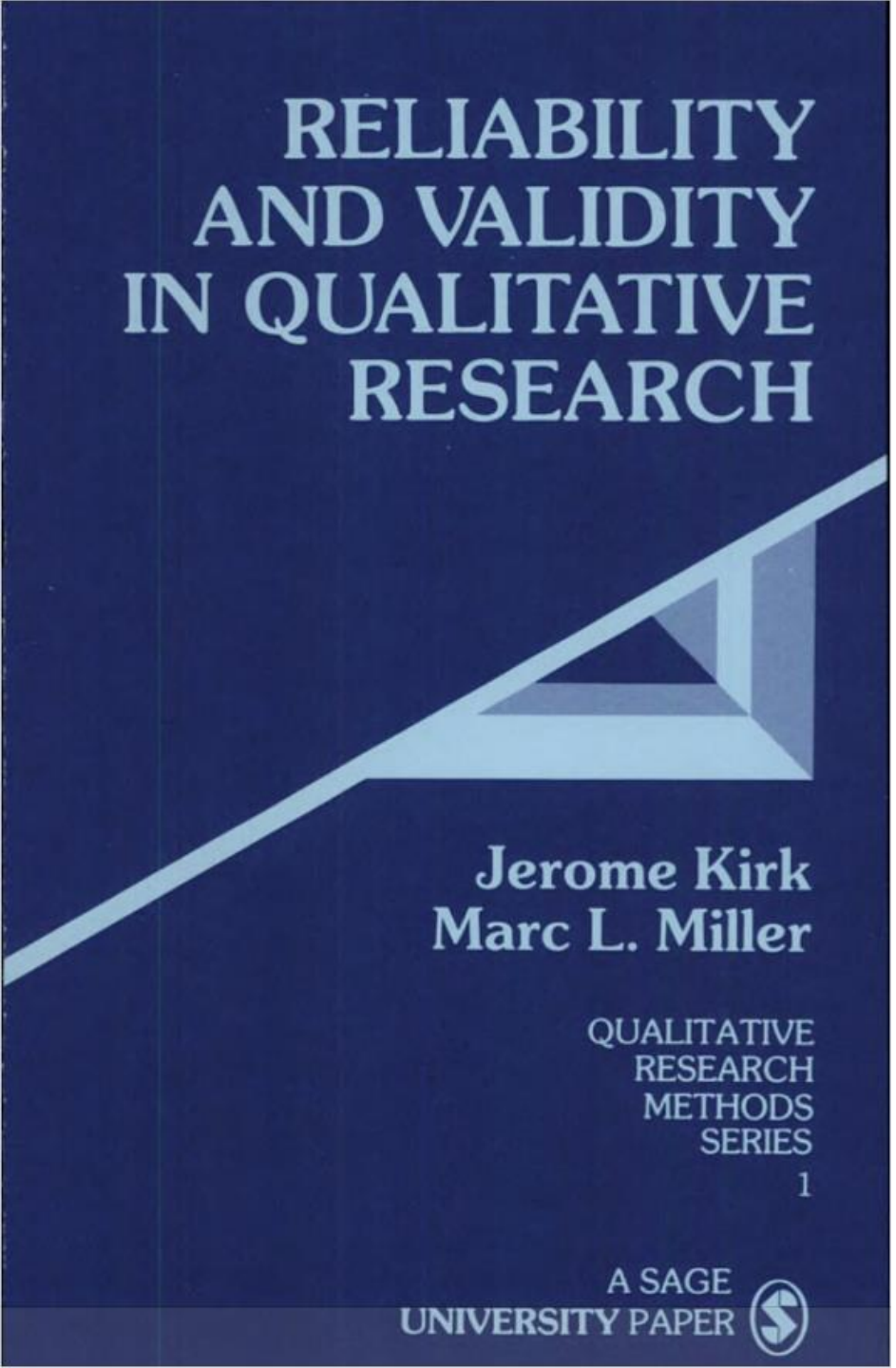
"We prefer to give weight to the outcome of experiments which appear to that have been successfully replicated over those which have not.

"We would give more weight to the opinions of those who believe that it is proper for their findings to be replicated."

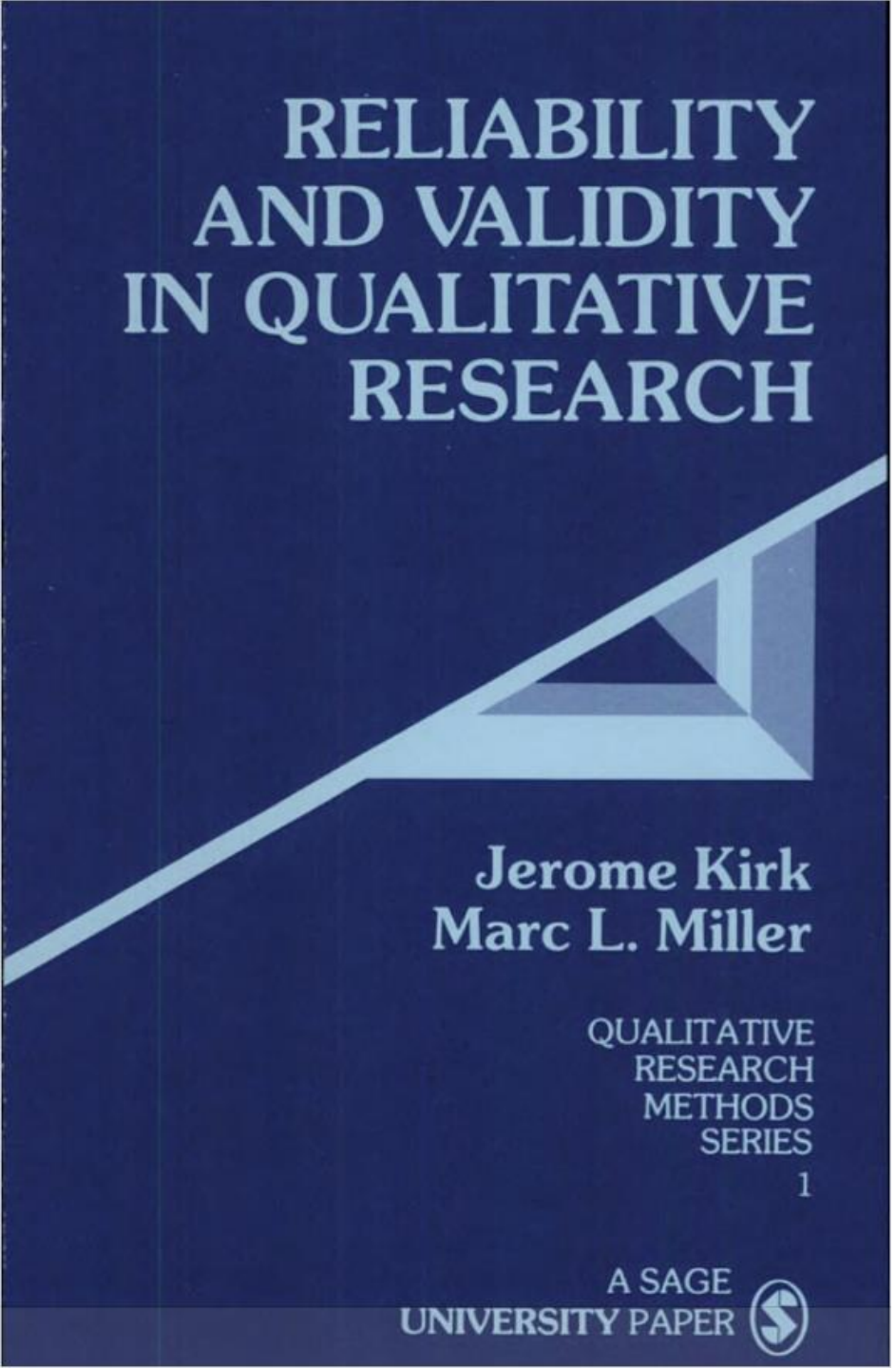
This runs into the problem of *experimenter's regress*.

Yet we choose to believe more in successfully replicated experiments than in one-offs.

BUT we must be careful about repetition, because…



"Most of the technology of 'confirmatory' non-qualitative research in both the social and natural sciences is aimed at preventing discovery. When confirmatory research goes smoothly, everything comes out precisely as expected. Received theory is supported by one more example of its usefulness, and requires no change.



"As in everyday social life, confirmation is exactly the absence of insight. In science, as in life, dramatic new discoveries must almost by definition be accidental ('serendipitous'). Indeed, they occur only in consequence of some mistake."

As testers, we must avoid being drawn into expensive, formal, confirmatory demonstratations that everything is okay — what people often call "automated testing" — *low*-quality *re*assurance.

Problems with “Verification”

“It works” really means *it appears to meet some requirement to some degree.*

“I saw it work” does not mean it **DID** work.

It **DID** work does not mean it **DOES** work.

It **CAN** work does not mean it **WILL** work.

In a system with a non-trivial state space, X + Y = 10 **may be true** ten times in a row, yet **may be false on the next iteration**.

- If you find X + Y = 10 to be true even once, then you have verified that it **CAN** be true.
- From that, you *could* make an inference that it will **PROBABLY** be true next time.
- But unless you check **EVERY POSSIBLE** state of the system, *including possible states that you don’t even know are possible*, you cannot verify that X + Y = 10 will **CERTAINLY** be true.

An inference that something works is fundamentally uncertain. Our clients might want certainty. As testers, we must reveal where the illusion of certainty can be harmful.

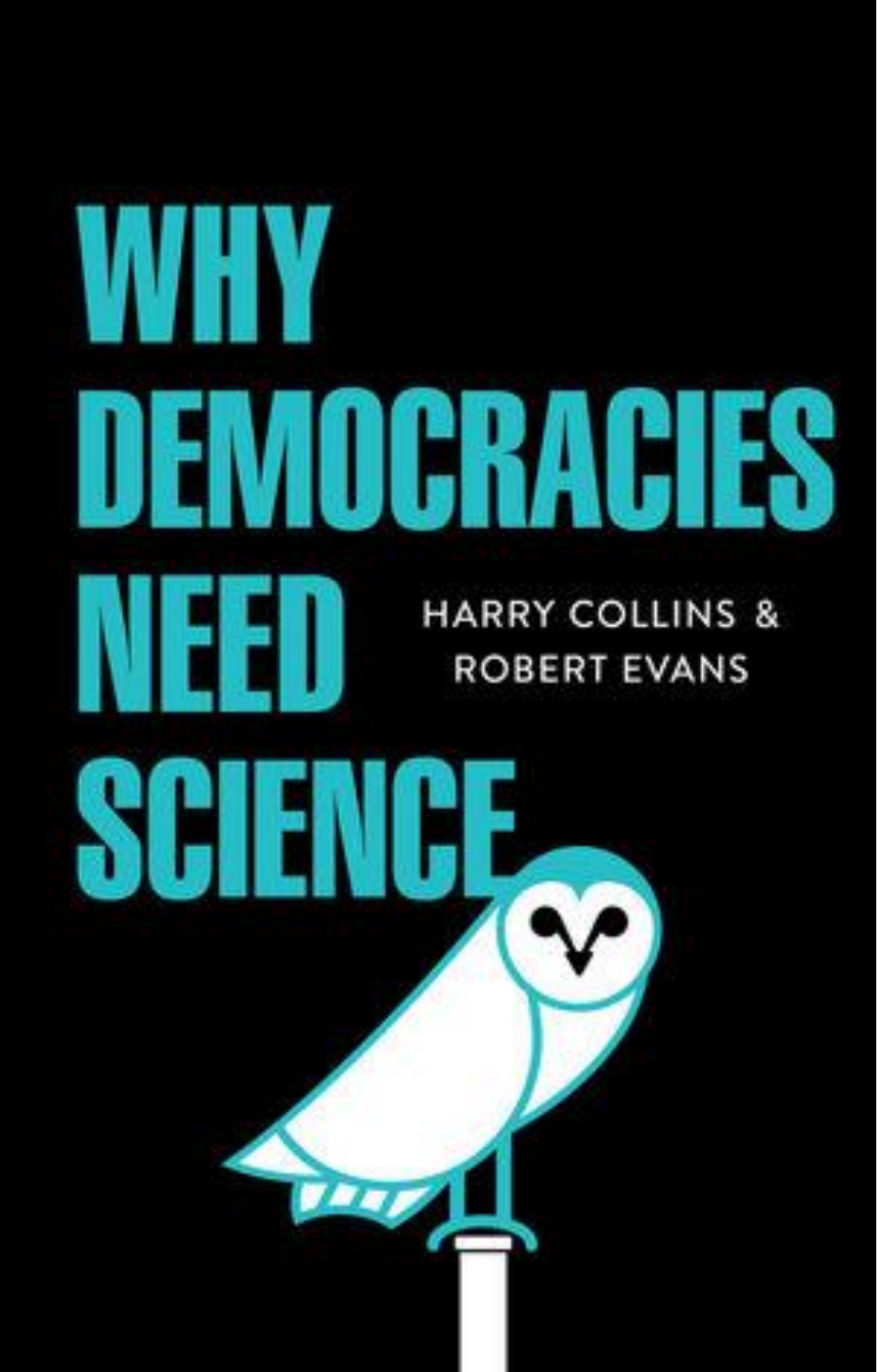
Oracles

An oracle is a fallible means by which you recognize a problem when you encounter one in testing.

“...it works”

really means

“It appeared to meet some requirement to some degree.”
“I saw it seem to work, once. Umm… on my machine.”
“None of the oracles that I applied alerted me to a problem.”



Formative Aspects of Science: Falsification

*"What is to be preferred---a world in which those who claim to make observations are willing to set out the conditions under which they can be shown to be wrong, and **actively invite the test**, or a world in which they would consider this unnecessary or inappropriate?"*

(my emphasis)

Testing can't just be a set of demonstrations to show that things *can* work. We must show where the product doesn't work; how it fails; and how it might fail.

What makes AI (and LLM/GPTs) problematic?

Algorithmic Obscurity

This stuff isn’t written by intentional, socially aware people; it’s both generated and selected by algorithms.

Obscured relationships between input and output mean we can't fully know what its capabilities OR its problems are.

This reduces *epistemtic testability* (that is, roughly, the size of the gap between what we know and what we need to know).

Radical Fragility

Due to algorithmic obscurity, we can’t fix machine learning models at their core.

ML models cannot be easily repaired or hardened against surprising regression bugs, further reducing epistemic testability.

Wishful Claims

Tacit or explicit claims of "thinking like a human" can be invalidated, but are impossible to verify.

Like all software, anything that works wonderfully in one context can be vulnerable to big trouble — even a single-bit change in the context.

Social Intrusiveness

AI — we even label it as “intelligence” — suggests something designed and competent to participate in the human social order.

But it’s not part of any social contract on its own; it is not and cannot be responsible for itself.

The form of its output also exploits our tendency to anthropomorphize.

Social and Corporate Pressure

There’s enormous social pressure due to investment, hype, and FOMO.

Criticism of what AI is and does is seen as opposition to "progress" itself.

“This is the very latest thing! What are you, a LUDDITE?!”

Obliviousness to Truth

These systems generate plausible-sounding text, but they have neither models of the world, nor social competence, nor an understanding of the difference between what is truthful and untruthful.

They’re not liars; but they do generate bullshit — text uttered without regard to truth.

Expensive to Test

This stuff is *crazy* expensive to train, to use, to fix, and to test.

Not just for the business, but for society at large — because when unreliable software is “tested in production”, real people can be hit with real problems.

The Large Language Mentalist Syndrome

When human beings (through social training and experience) see patterns of text that closely resemble human writing, it is almost irresistable to treat a GPT's output as human — and then to read human intention, interpretation, and intelligence into that.

AI can fool us. Experiment deeply and LOOK CLOSELY.

- "AI saves time." Generation time, maybe. But what about time time required for critical evaluation and correction?
  - Did it perform the assignment we gave it, or does it just look like that at first glance?
  - Did it give correct, complete, and consistent answers?
  - Did it drown us in unnecessary fluff that we have to wade through to notice problems?
  - Did it make stuff up?
  - How much of the happy results required us to nudge and repair the bot when it went off track? Are we just seeing a ship in a bottle without seeing what it took to get there?
  - When it says it corrected itself, did it actually correct itself? Did it add new mistakes?
  - Was the output examined critically by people who are actually experts in the domain?
  - Can we trust it to do this again? Every time? For every similar situation? How about now?

Do not consider a pleasing demo as evidence of good and reliable behaviour.

Try *real experiments* multiple times. Watch results closely.

AI can fool us. Experiment deeply and LOOK CLOSELY.

- Is the answer from a probabilistic text generator better than a curated template?
- How much of the answer are we filling in ourselves?
- Do we understand the entire system? Can we find elements and interfaces that are easier and more reliable to test?
- Are the fanboys and managers gaslighting us by treating actual failure as “limited success”?
- What could go wrong? If things go wrong, how will we know?
- Will a responsible human have the time and the capability to override the system, or will problems get ignored or buried?

AI can fool us. Experiment deeply and LOOK CLOSELY.

- Is the answer from a probabilistic text generator better than a curated template?
- How much of the answer ar
- Do we understan d interfaces th
- Are the f ctual failure as "limited
- What could g g, how will we know?
- Will a responsible human have the time and the capability to override the system, or will problems get ignored or buried?

Keep good notes, track your time, and report the whole truth, even if it might disappoint the bosses who are being pressured or conned.

How do we get deep?

determination
faith in trouble
skills
learning
tooling
testability
team support
time
VARIETY

How do we get deep?

- **Determination:** We won't simply blunder into deep testing. It doesn't happen by accident. We have to *want* it.
- **Skills:** We need **subjective testability**—knowing how to model products, identify testable conditions, and design experiments to evaluate them.
- **Learning:** We need rich and detailed models of **the product** and its **context**, and of **testing itself**. These may be mental models, formal models, or both. Deep learning about the product and how people use it requires experience with it — and **value-related** and **epistemic testability**.

See https://www.satisfice.com/download/heuristics-of-software-testability

How do we get deep?

- **Tooling:** We may need tools to help us cover large areas or to get access to hard-to-reach areas of the product.
- **Testability:** We may need special features built into the product — **intrinsic testability** — that help us observe and control it less expensively.
- **Team Support:** We may need lots of eyes and minds poring over the product; **project-related testability**. Developers can help immensely by exposing the code, and building in testability into the product.
- **Time:** Co-ordinating all this AND performing deep experiments takes time. When we have insufficient time, that's a Severity 0 project risk.

See https://www.satisfice.com/download/heuristics-of-software-testability

How do we get deep?

Variety in our…

- Models of **risk** (to identify potential victims, problems, vulnerabilities, and factors that might trigger them)
- Models of **coverage** (to identify perspectives from which we can examine the product, especially including…)
- **data** (because many bugs are triggered by very specific data, or combinations of data)
- **environments** and test platforms (configured and available for gaining experience with the product via exploration and experimentation)
- **test activities and techniques** (to operate the product in different ways, to find obscure yet important bugs)
- dimensions of **quality criteria** (to identify different ways in which people might obtain value from the product)
- **oracles** (to give us lots of ways to recognize problems)

Again… testers are, ideally

- **Empirical** (instead of wishful)
- **Skilled** (instead of merely intuitive)
- **Different** (instead of redundant)
- **Motivated** (instead of ambivalent)
- **Available** (instead of preoccupied)

Interesting Trends

1. Evolution of laws regarding data governance, security, AI, and product liability could reinvigorate the testing industry — especially in the EU.
2. Increased use of AI in development creates new testing demand.
3. Increased use of AI in products creates new testing demand.

New EU Product Liability Directive

Covers safety (damage to life, property, or data)

Article 7

Defectiveness

1. A product shall be considered defective where it does not provide the safety that a person is entitled to expect or that is required under Union or national law.

2. In assessing the defectiveness of a product, all circumstances shall be taken into account, including:

   (a) the presentation and the characteristics of the product, including its labelling, design, technical features, composition and packaging and the instructions for its assembly, installation, use and maintenance;

   (b) reasonably foreseeable use of the product;

New EU Product Liability Directive

Makes it easier to prove that a product is defective…

Article 10

Burden of proof

1. Member States shall ensure that a claimant is required to prove the defectiveness of the product, the damage suffered and the causal link between that defectiveness and that damage.

2. The defectiveness of the product shall be presumed where any of the following conditions are met:

   (a) the defendant fails to disclose relevant evidence pursuant to Article 9(1);

   (b) the claimant demonstrates that the product does not comply with mandatory product safety requirements laid down in Union or national law that are intended to protect against the risk of the damage suffered by the injured person; or

   (c) the claimant demonstrates that the damage was caused by an obvious malfunction of the product during reasonably foreseeable use or under ordinary circumstances.

New EU Product Liability Directive

Doesn’t let companies hide behind a EULA…

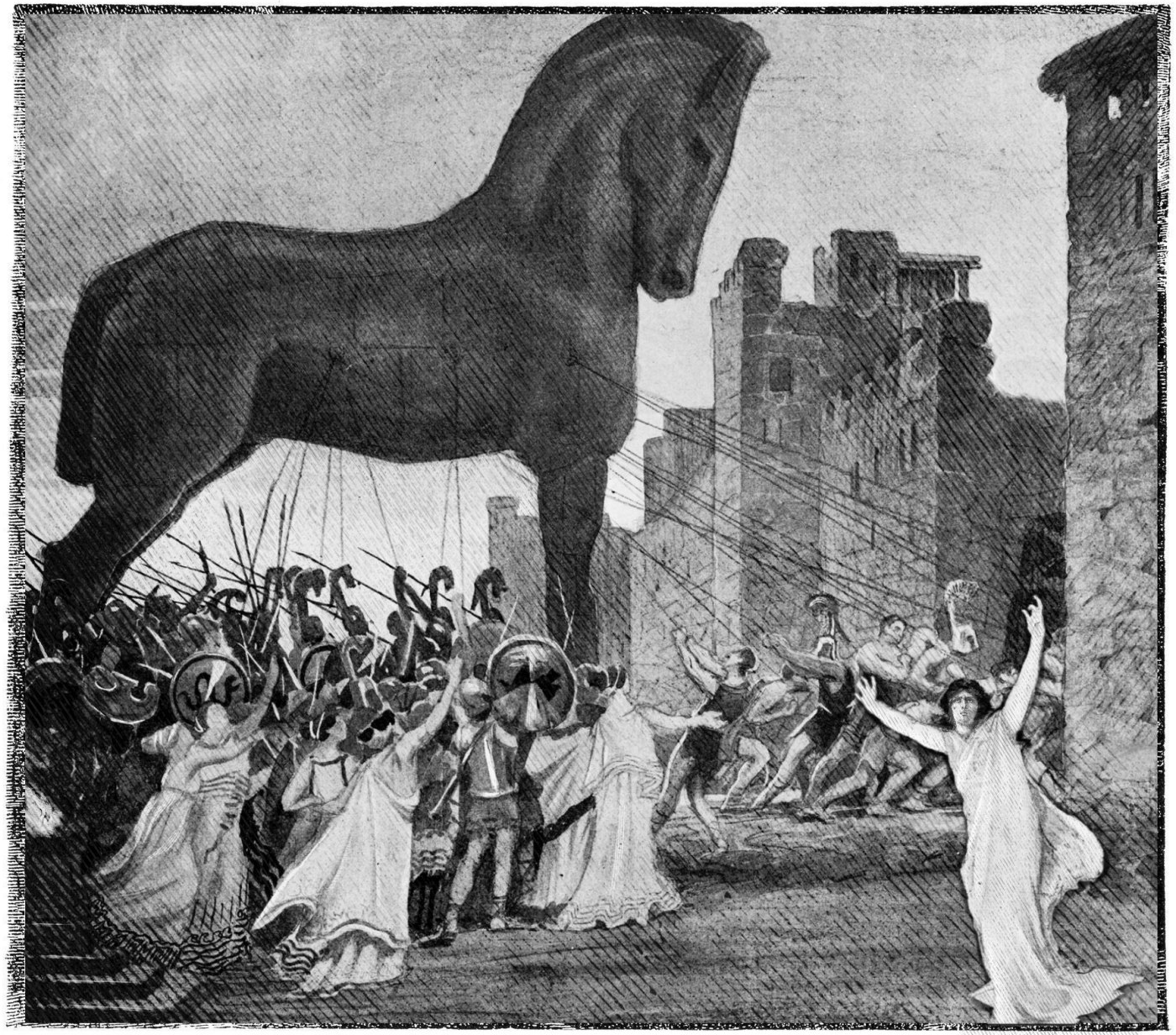
(56) The objective of protecting natural persons would be undermined if it were possible to limit or exclude an economic operator’s liability through contractual provisions. Therefore no contractual derogations should be permitted. For the same reason, it should not be possible for provisions of national law to limit or exclude liability, such as by setting financial ceilings on an economic operator’s liability.

https://data.consilium.europa.eu/doc/document/PE-7-2024-INIT/en/pdf

Things You Could Do

- Read the regulations. Study them. Make notes on them. They're long. They're hard. They can be boring and repetitive. *You'll be the only one who knows them.*
- Practice and perform testing. Expland your models of oracles and coverage.
- Work with data, not just with functions. Actually test the danged product.
- Focus on finding problems that matter — that present risk to the business, to its customers, and to society, before it's too late.
- Learn to describe and explain your work.

Talking about product trouble brings *social* trouble.



Beware of geeks bearing grifts.

And be brave.

Run towards the risk.

Speak up.

And let’s take care of each other.

Final point to ponder

The first step towards making things better
is to embrace the idea
that there might be problems in them.

When we find those problems, responsible
people can start to address them.


expoQA25
MADRID
May 20th,
21st & 22nd
2025
Thank you for attending
expoqa.eu