

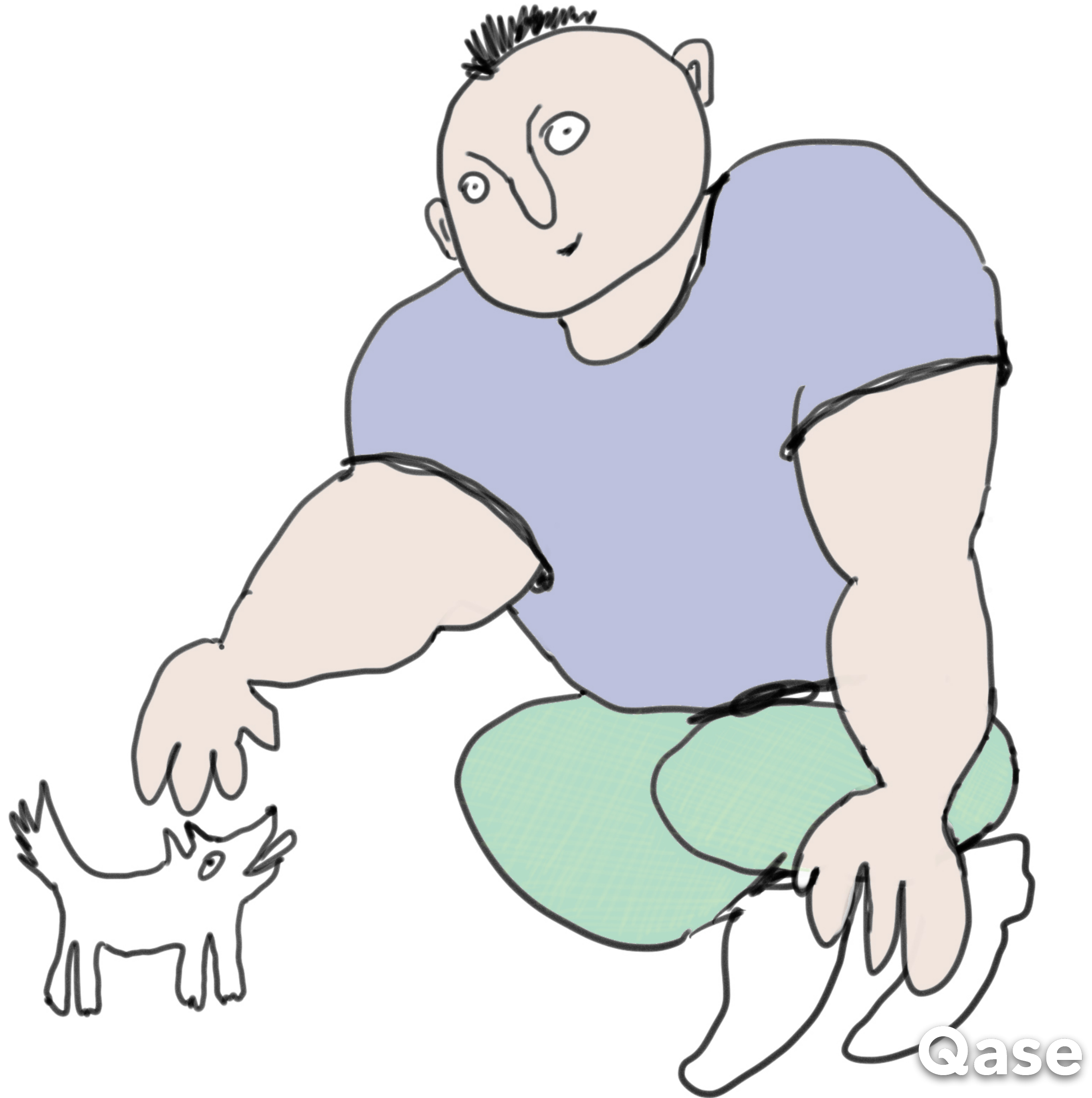
Value, Money, and Costs

The Ultimate Common Language for the Testing Strategy

Sharovатов, 2025

Vitaly Sharovатов

- Developer, QA, EM, mentor, dev 🥑
- Quality Enthusiast
- Animal lover



Tech debt

I was drawing an analogy between the debt financing of a growing company and the incremental intellectual investment in a growing program. I used the idea of debt metaphorically, a metaphor that has been lost on pretty much everyone.



Story points

Story Points Revisited

© May 23, 2019 • [[Agile-Related](#), [Dark-Agile](#), [Dark-Scrum](#), [SAFe](#), [success](#)]

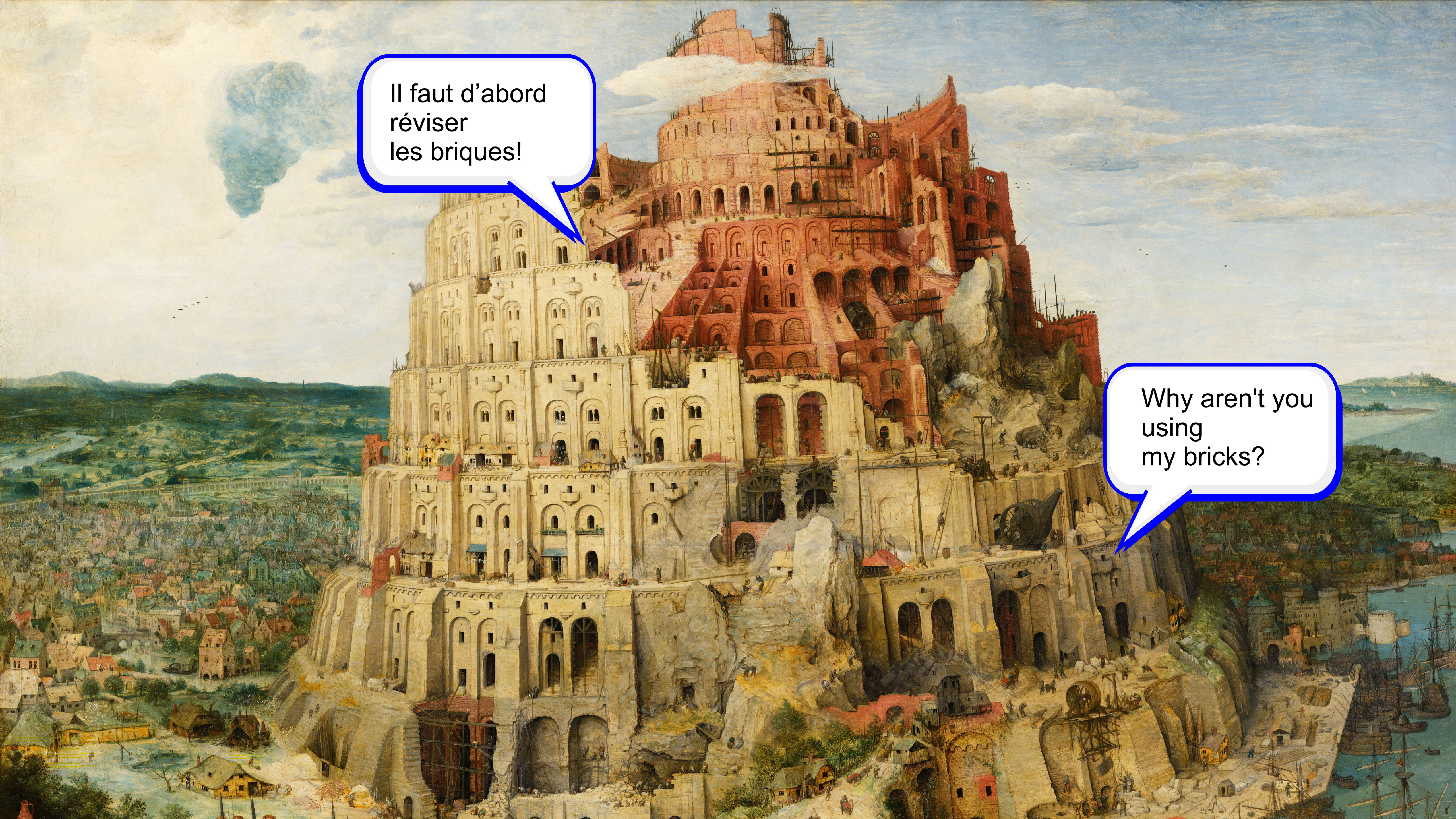
I like to say that I may have invented story points, and if I did, I'm sorry now. Let's explore my current thinking on story points. At least one of us is interested in what I think.

The loss of meaning

Happens way too often

- MVP
- DevOps
- Mocks and stubs

Distorting the communication



Il faut d'abord
réviser
les briques!

Why aren't you
using
my bricks?

Different languages

- **Engineers:** “we need more tests”, **Managers:** “customers don’t ask for this”
- **Engineers:** “we need code reviews”, **Others:** “they slow us down”
- **Managers:** “let’s release now”, **Engineers:** “No, code is still bad!”
- **Managers:** “let’s ask devs to write autotests”, **Engineers:** “Don’t want to”
- **Engineers:** “no deploys on Fridays”, **Managers:** “Why?”

The result is real

- 2012: **\$400M** loss for Knight Capital due to an uncaught regression bug
- 2024: **\$10B** loss for Crowdstrike outage due to the lack of testing
- 2024: **\$60B** loss for Boeing due to insufficient MCAS testing

Everyone tests but...

Economics as the common language

- We're all interested in doing **the right things**
and doing them **the right way**
- Both are about ethics, care for users and ...



Economics of testing

- Testing consumes resources
- Testing doesn't generate revenue
- Testing outcome is *lack of issues*, it's hard to reason about the lack of smth
- How can we justify more resources for smth *not happening*?
- Testing doesn't give 100% guarantees, so smth might happen even if we get more resources

Testing is an investment

Easy to reason!

- *“any asset obtained at a cost, on the expectation that it will deliver future value greater than its initial cost”*
- Testing is an **investment** made **today** to **reduce the probability and impact** of costly failures **tomorrow**
- The **risks** of not doing it **>** than the investment cost

What's next?

Where and how to invest to reduce the risks?

1. What risks do we have?
2. Which ones matter most?
3. How to invest to reduce them?

1. Define the risks

- Brainstorm the risks
- Peak into ISO 25010 Quality Characteristics

Functional Suitability

Security

Performance Efficiency

Maintanability

Compatibility

Flexibility

Interaction Capability

Safety

Reliability

1. Define the risks

Fears to risks examples

- “Clients can’t pay” — Functional suitability → completeness, Compatibility → Interaction capability
- “User data leaked” — Security → Confidentiality
- “Too slow on a mobile device” — Performance efficiency → Time behaviour
- “The device kills the operator when the power goes out” — whole Safety

2. Prioritise the risks

For your context

- Risk for each QC **to have a score** based on **projected loss of money**
- Weighted list of risks based on scores

Risk	Score
Safety	10
Reliability	8
Functional Suitability	7

3. Defining how to invest

For your risks and context

- Choose testing approaches
 - ISO 29119 Testing Methods
 - Team's expertise and skills
- Map testing approaches **M:N** to risks
- Choose the optimal*

Levels of testing

Unit testing
Integration testing
System testing
System integration testing
Acceptance testing

- User acceptance testing
- Operational acceptance testing
- Factory acceptance testing
- Alpha testing
- Beta testing
- Production verification testing

Test practices

Model-based testing
Scripted testing
Exploratory testing
Experience-based testing
Manual testing
A/B testing
Back-to-back testing
Mathematical-based testing
Fuzz testing
Keyword-driven testing
Automated testing

- Capture-replay driven
- Data-driven

Types of testing

Functional testing
Accessibility testing
Compatibility testing
Conversion testing
Disaster/recovery testing
Installability testing
Interoperability testing
Localization testing
Maintainability testing
Performance-related testing

- Performance
- Load
- Stress
- Capacity
- Recovery

Portability testing
Procedure testing
Reliability testing
Security testing
Usability testing

Static testing

Reviews (ISO/IEC 20246)
Static analysis
Model verification

Test des techniqu measur

Specification-based:

- Equivalence p
- Classification t
- Boundary valu
- Syntax testing
- Combinatorial
 - All combin
 - Pairwise
 - Each choic
 - Base choic
- Decision table
- Cause-effect g
- State transitio
- Scenario testin
- Use case testin
- Random testin
- Metamorphic
- Requirements testing

Structure-based:

- Statement test
- Branch testing
- Decision testin
- Branch condit
- Branch condit
- combination t
- MC/DC testing
- Data flow test
- All-definiti
- All-C-uses
- All-P-uses
- All-uses te
- All-DU-pat

Experience-based:

- Error guessing

3. Define how to invest

Incomplete example of the M:N mapping

Risk	Score	Testing Approach
Safety → fail safe	10	Fault Injection Testing Safety requirements-based testing
Functional Suitability → completeness	8	Requirements-based testing Acceptance testing End-to-end testing
Reliability → recoverability	7	Backup testing Fault Injection Testing Stress Testing

3. Defining how to invest

Choosing the optimal methods

1. **Higher score** = more costly risks ‘covered’
2. **Mapped to more risks** = investing once in covering more risks
3. **Cheaper to do** = proactive/reactive

So what about unit tests coverage?

Impact of unit testing

1. A survey showed that people believe unit tests help mitigate ALL the risks
2. My experience shows that without TDD unit tests only help with:
 - Maintainability → Modularity
 - Maintainability → Reusability
 - Maintainability → Modifiability



4. Reassessing the investments

- **Review cadence:** period of time, signals from the market, incidents, ?
- Questions to ask:
 - How well are we mitigating the risks?
 - How much did it cost?
 - Can we do it more efficiently?
 - Reduce Cost Of Delay by moving things left
 - Reduce wait time by decreasing the number of handovers

Common language

For the testing strategy

- Testing is an investment in reducing risks
- “The **risks** of not doing it $>$ than the investment cost”
 - The risks come first, the cost — second
- Good common understanding of the vocabulary — better collab

Let's practice!!!

7 minutes: Let's pick two risks



5 mins: Brainstorm testing methods for them

- Investing in what testing methods would cover the risks we chose?
- Which methods *do we think* are more optimal and why?

10 mins: Define investment reassessment

For the chosen testing methods

- When to reassess?
- Which data to collect for the reassessment?
- *How to reason about proactive and reactive approaches?*

Common language

For the testing strategy

- Testing is an investment in reducing risks
- “The **risks** of not doing it $>$ than the investment cost”
 - The risks come first, the cost — second
- Testing strategy is a *dynamic investment*, it needs revision
- **Everyone should participate in economics decisions**
- Money is the easiest common language

THANKS!

LinkedIn



BeyondQuality community

