

Exploratory Testing

Rediscover the art of exploratory testing



Ingo Philipp

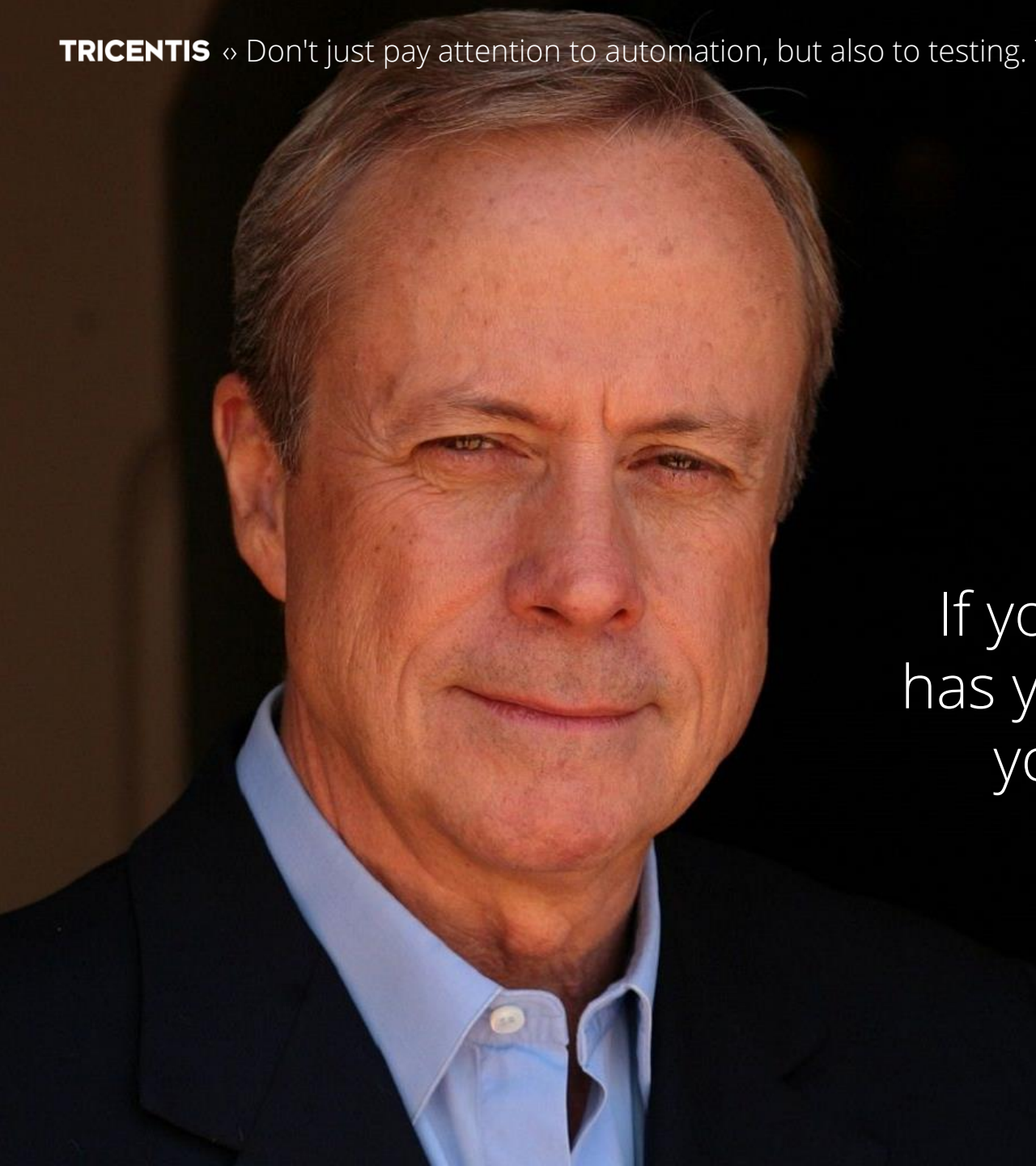
TRICENTIS ◊ How your management looks like when you stop talking about automation and start talking about testing.



TRICENTIS ◊ When you are at a release party and nobody wants to talk about the beauty of testing. Keep evangelizing testing!



TRICENTIS ◊ Don't just pay attention to automation, but also to testing. Trust me, testing will grab your attention sooner or later.

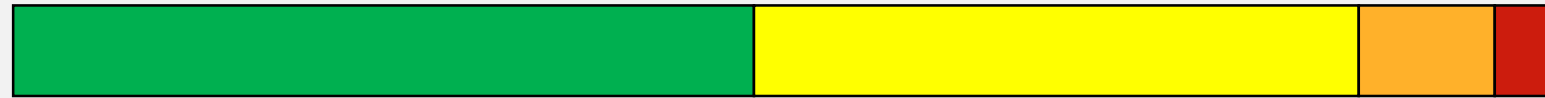


If you don't pay **attention** to what has your attention, it will take more of your attention than it deserves

:: David **Allen** :: Getting Things Done

Software Testing Timeline

Testing? There's only automation!
We automate everything

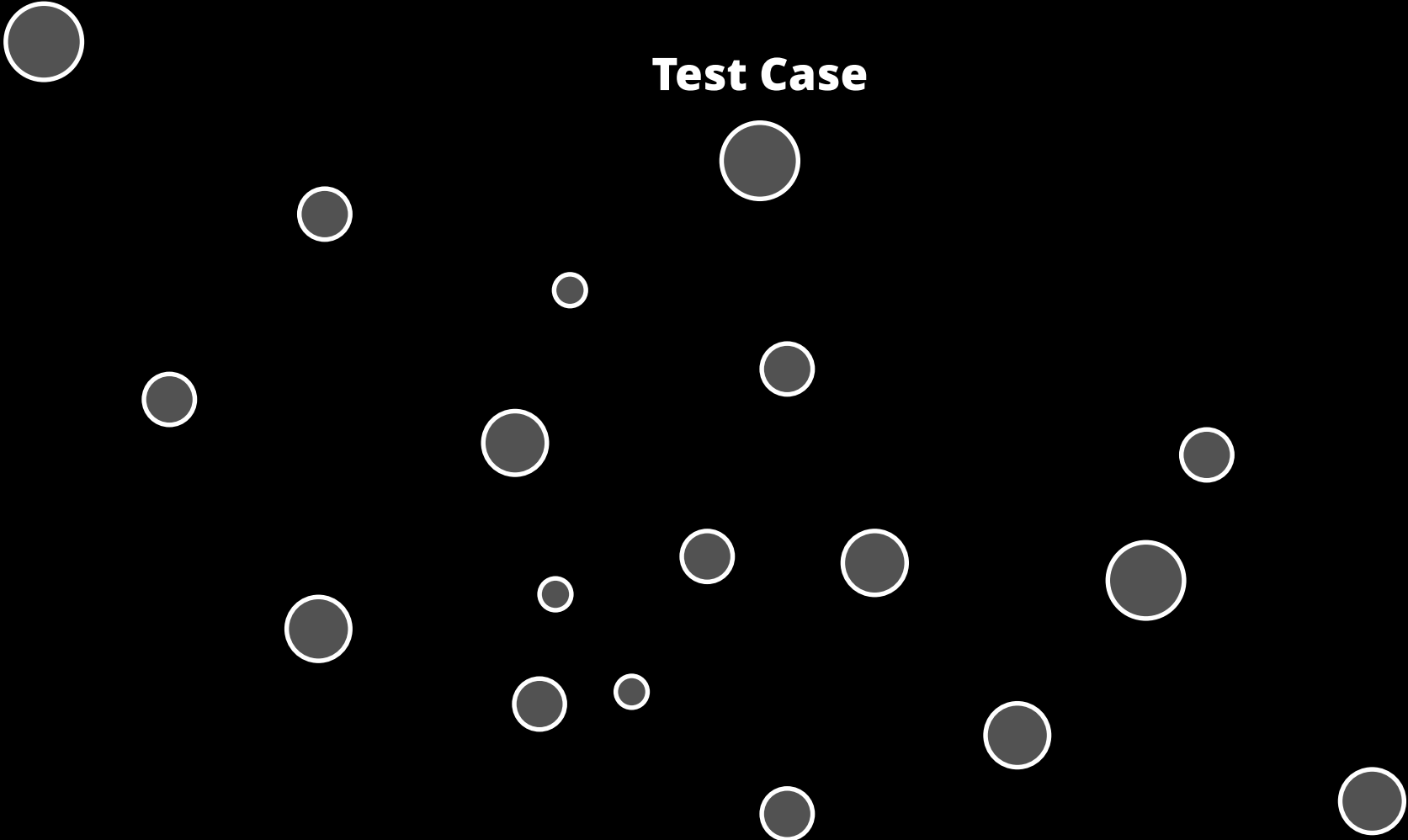


Oops

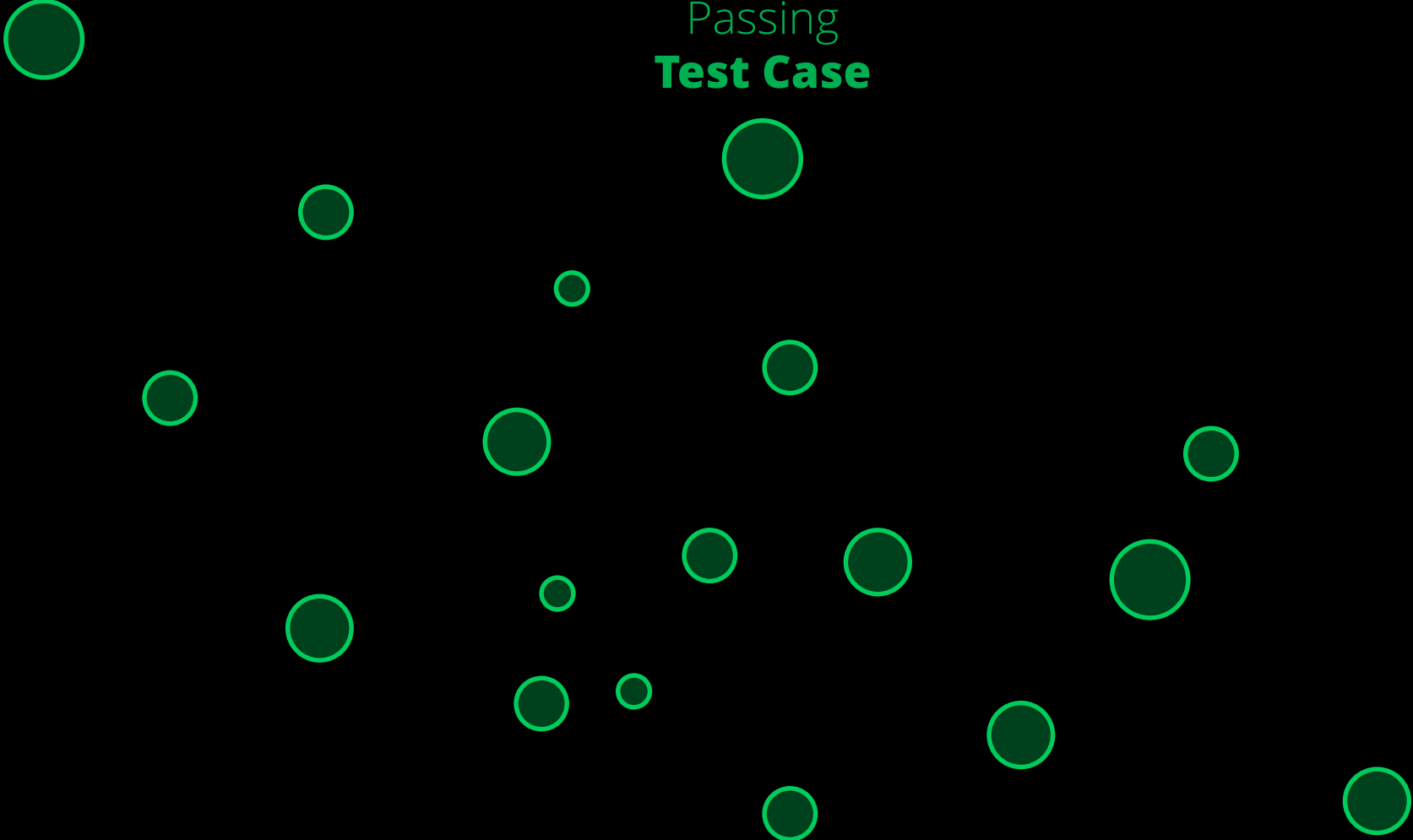
Damn!

OK, there's testing, and testing
is probably more than just automation.
We're just not convinced that we really need it

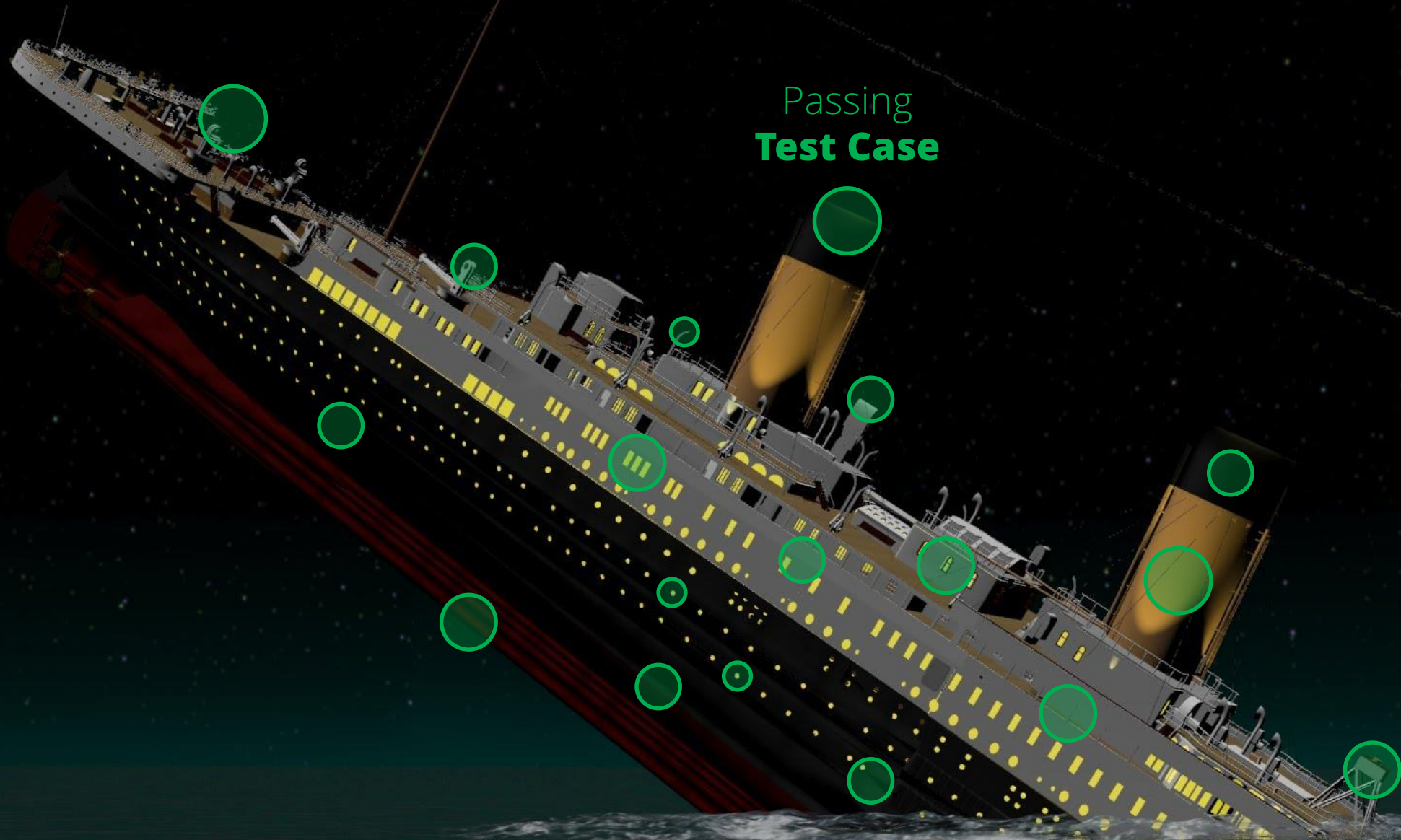
Test Case



Passing
Test Case



TRICENTIS ◊ Maybe you're stuck in mitigating risk because you're pushing a door that says pull.



Passing
Test Case



TRICENTIS ◊ Why do we press harder on the remote when the batteries are dead?



TRICENTIS ◊ Why does the dentist talk to you when you can't respond?





Exploratory Testing



Manual Testing



TRICENTIS ◊ If you don't understand modern art, it's not your fault.



:: Maurizio **Cattellan**

Everyone **talks** about it.
Nobody really **knows** how to do it.
Everyone thinks everyone else is **doing** it.
So everyone **claims** they are doing it.

TRICENTIS ♦ Software development is a lot like wrestling in the mud with a pig.



Testing is exactly like **washing** a pig. Because it's messy. It has no rules. No clear beginning, middle, or end. It's kind of a pain in the ass, and when you're done you're not sure if the pig is really clean or even why you were washing a pig in the first place.

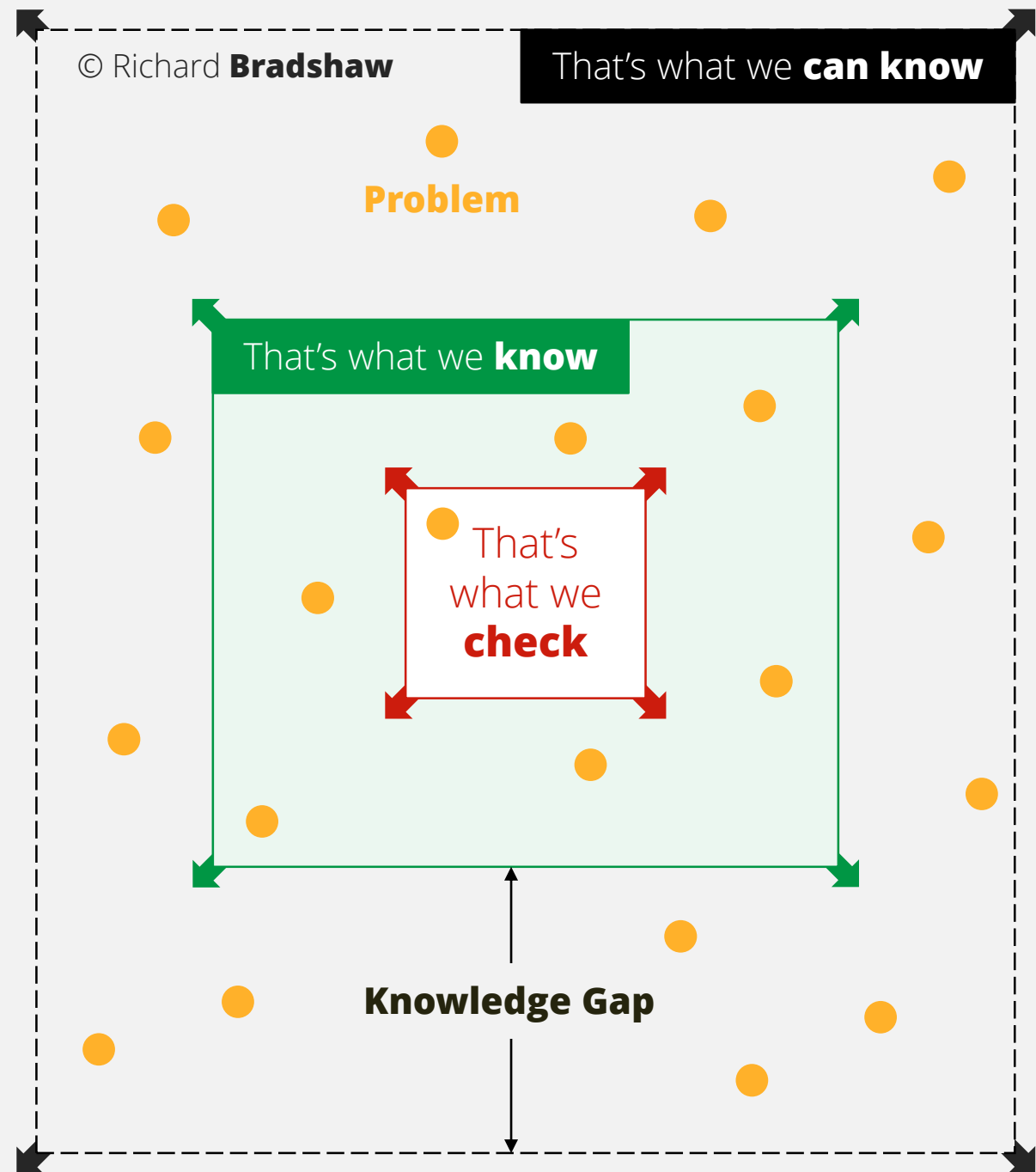


Testing is exactly like **washing** a pig. Because it's messy. It has no rules. No clear beginning, middle, or end. It's kind of a pain in the ass, and when you're done you're not sure if the pig is really clean or even why you were washing a pig in the first place.



That's what we **can know**

Testing is exactly like **washing** a pig. Because it's messy. It has no rules. No clear beginning, middle, or end. It's kind of a pain in the ass, and when you're done you're not sure if the pig is really clean or even why you were washing a pig in the first place.



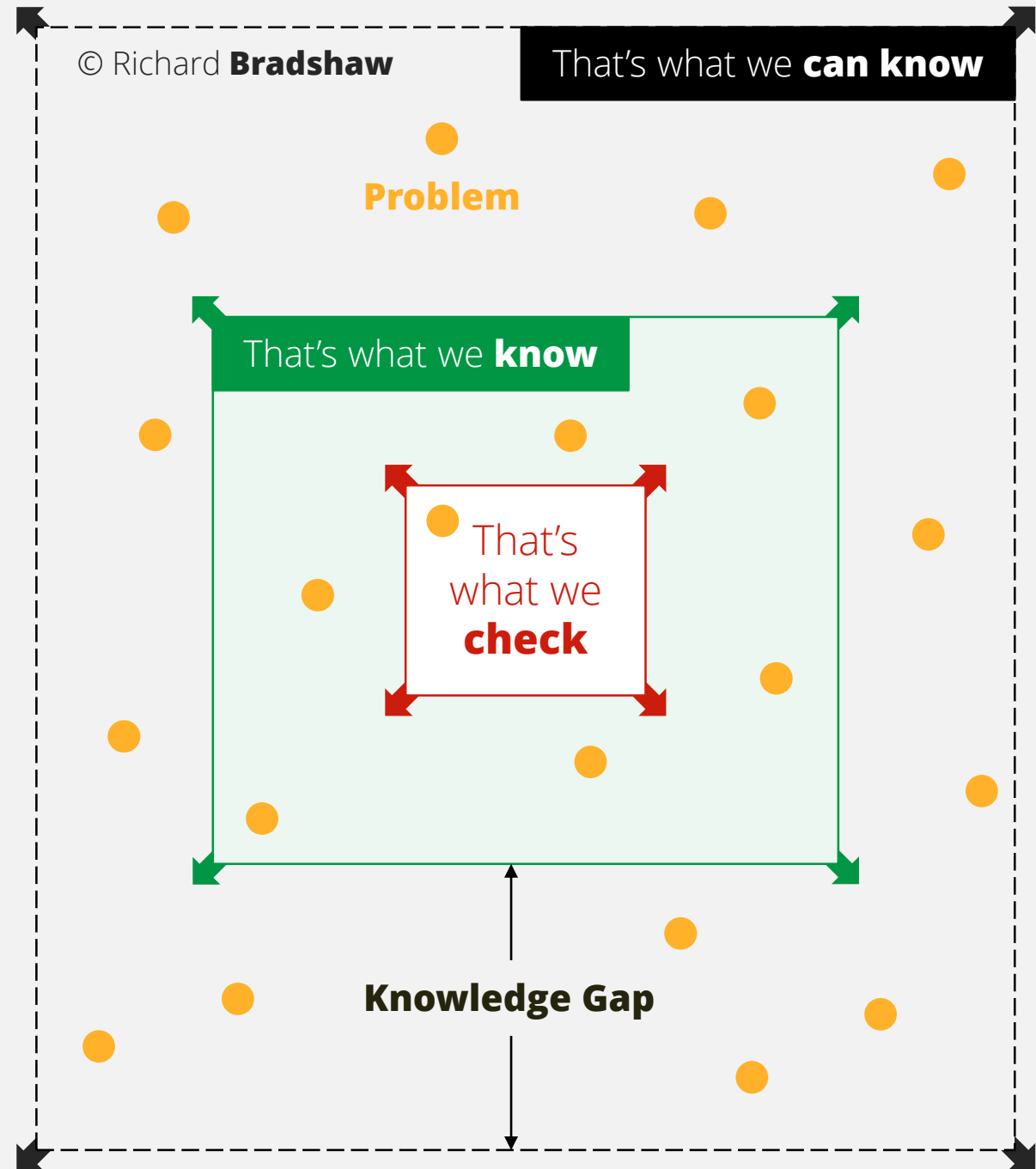
The purpose of testing is to close the **knowledge** gap



The goal is **information**, not gratuitous automation

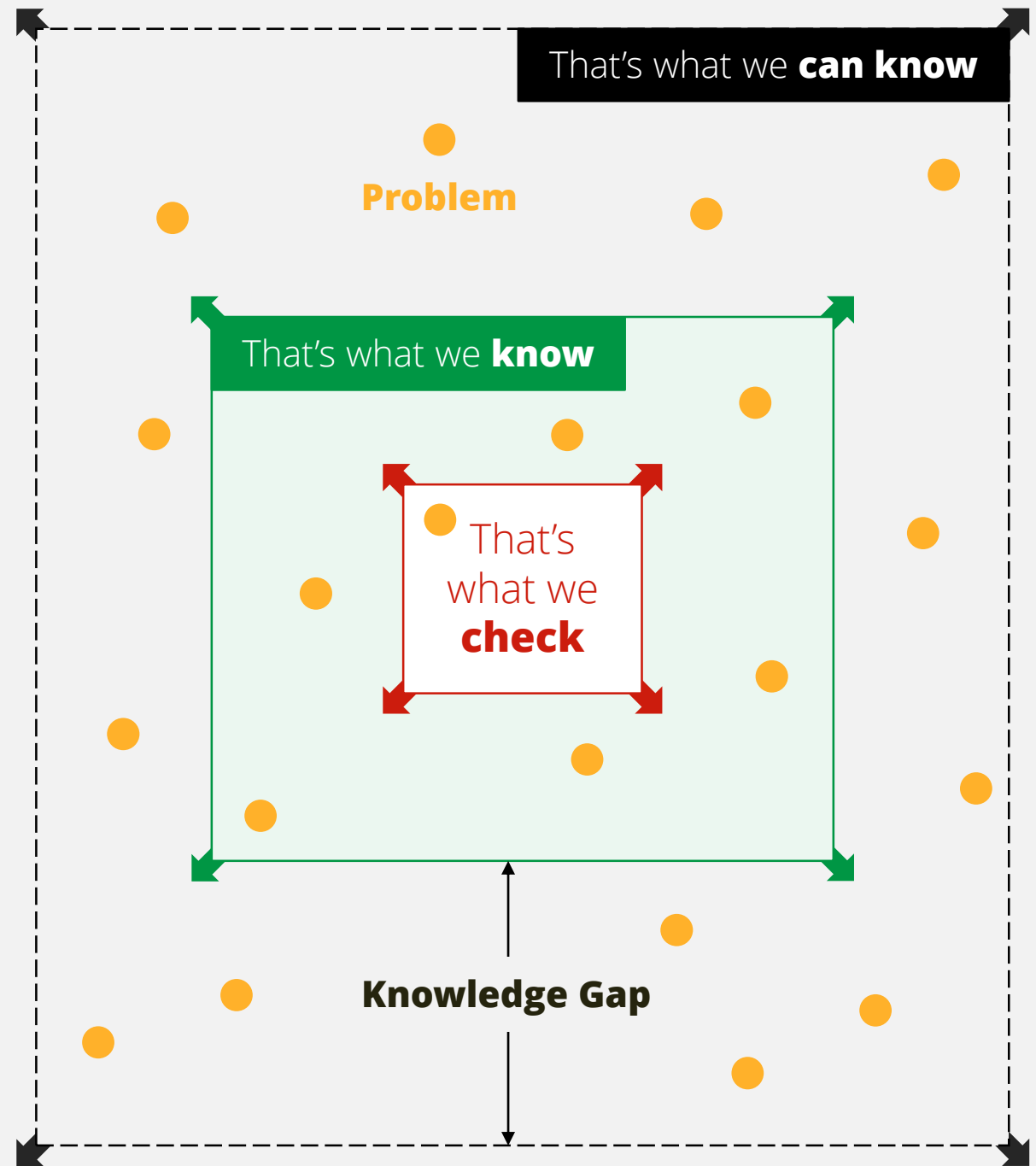


We enable people to make better **decisions** based on the information we provide





We go to crime scenes
(**software**) and search for
evidence (**risks**) to enable the
police (**developers**) to arrest
(**fix**) the culprits (**bugs**)



Change Detector



Low Information Value
Check what you have already learned

Monitor Known Risks
Confirm what you already know

Mechanical Testing
Process pre-defined data in pre-designed steps



Confirmation
« Demonstrate your **depth** of knowledge »

Problem Detector



High Information Value
Learn something new

Analyze Potential Risks
Focus on the things you don't know

Creative Testing
Create new test ideas based on what you have learned



Exploration
« Demonstrate your **breadth** of knowledge »

Change Detector



Evaluate a product by applying **algorithmic** decision rules to specific observations of a product

Checking

« Requires **Processing** »

Problem Detector

High Information Value
Learn something new

Analyze Potential Risks
Focus on the things you don't know

Creative Testing
Create new test ideas based on what you have learned

Exploration

« Demonstrate your **breadth** of knowledge »

Change Detector

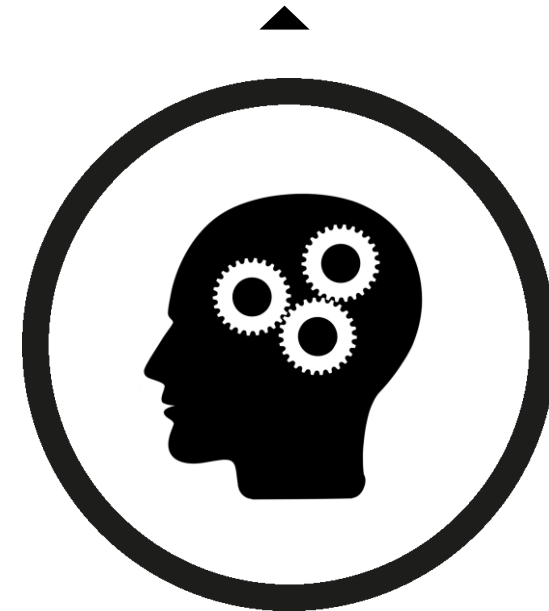


Evaluate a product by applying **algorithmic** decision rules to specific observations of a product

Checking

« Requires **Processing** »

Problem Detector



Evaluate a product by **learning** about it through **exploration** and **experimentation**

Exploring

« Requires **Thinking** »

Goal. Monitor Known Risks

Verify through
Instructions

Pay attention to
Deviations

Create
Test Cases

Follow
Procedure

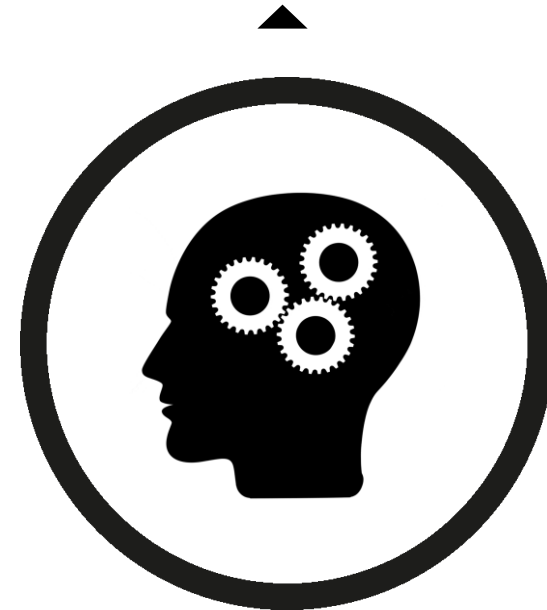
Examine
Requirements

Factory
Process

Checking

« Requires **Processing** »

Problem Detector



Evaluate a product by **learning** about it through **exploration** and **experimentation**

Exploring

« Requires **Thinking** »

Goal. Monitor Known Risks

Mechanical Process

Verify through
Instructions

Pay attention to
Deviations

Create
Test Cases

Follow
Procedure

Examine
Requirements

Factory
Process

Checking

« Requires **Processing** »

Goal. Analyze Potential Risks

Cognitive Process

Investigate through
Experiments

Pay attention to
Oracles

Create
Test Ideas

Follow
Clues

Examine
Risks

Adaptive
Investigation

Exploring

« Requires **Thinking** »

Agile

Testing Equation

Checking

Efficient **Confirmatory Testing**

+

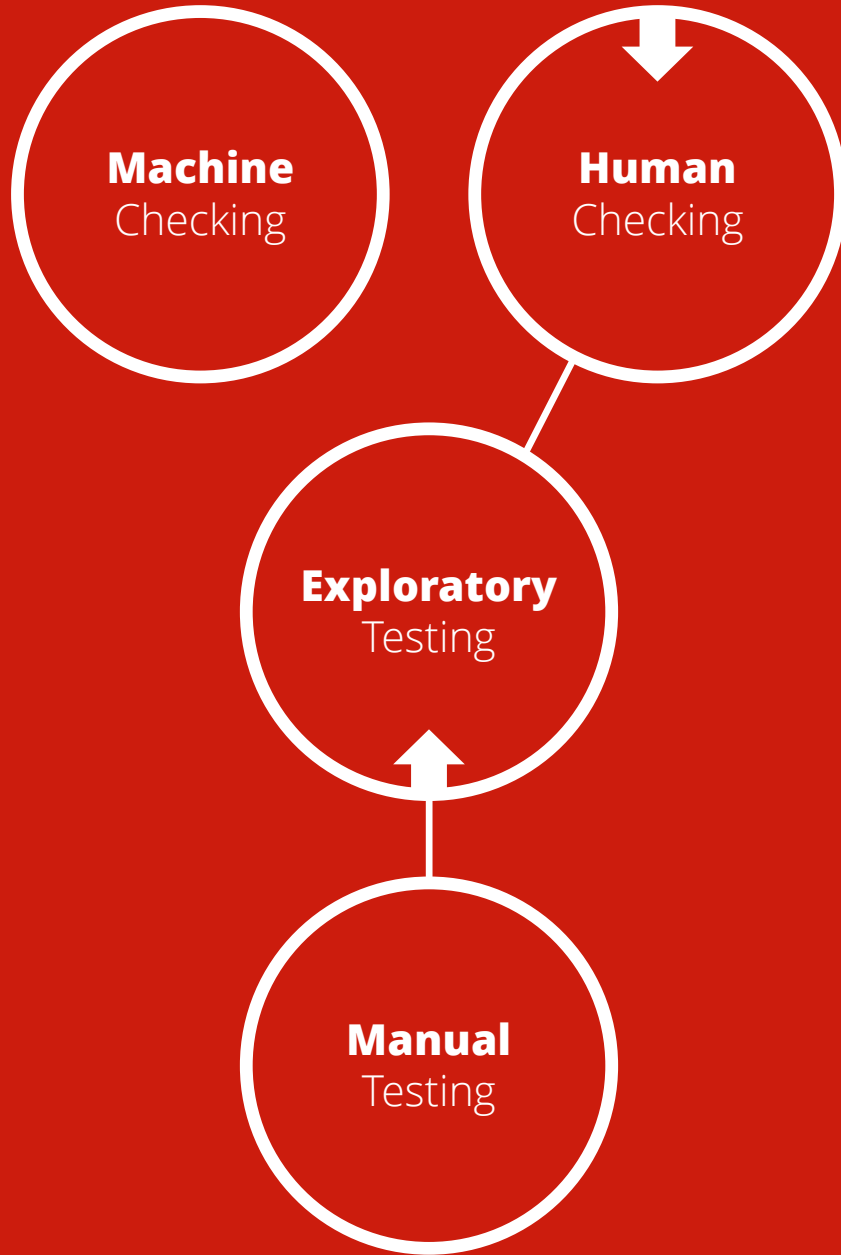
Exploring

Effective **Exploratory Testing**

=

Testing

Thorough Testing



Checking

Efficient **Confirmatory Testing**

+


Exploring

Effective **Exploratory Testing**

=

Testing

Thorough Testing

A man in a dark suit, white shirt, and red tie stands against a light-colored stone wall, looking upwards with a concerned expression. He is holding a large, rectangular cardboard sign in front of him. To his right, a young girl with long brown hair, wearing a white t-shirt with colorful heart patterns, looks down with a sad expression. The sign has the text "ROBOT TOOK MY JOB WILL WORK FOR FOOD" written on it in large, black, hand-drawn capital letters.

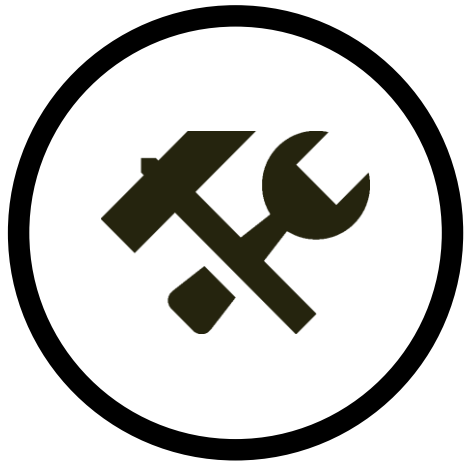
ROBOT TOOK MY JOB
WILL WORK
FOR FOOD



Exploratory testing is not a talent,
it's a set of **skills** that can be learnt



Ingo Philipp



Technique



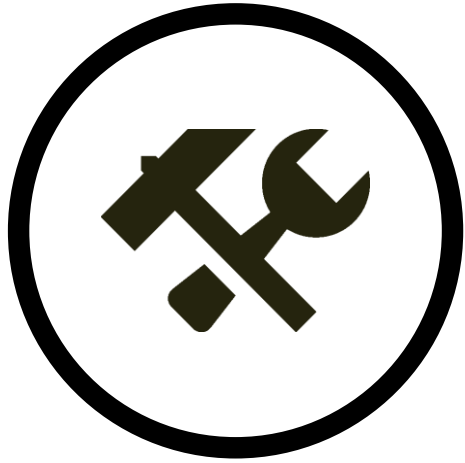
Provides Systematic Procedure



Approach



Provides Orientation



Technique



Provides Systematic Procedure

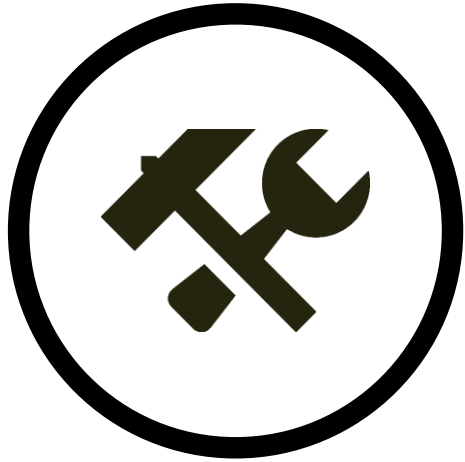
1

2

3

4

5



Technique



Provides Systematic Procedure

Session-Based Testing

Structure exploratory testing to allow large-scale implementations

2

3

4

5



Straightjacketed **Imagination**

Session-Based Testing

Structure exploratory testing to allow large-scale implementations

2

3

4

5

Chartered
Uninterrupted
Reviewable

Session

Session-Based Testing

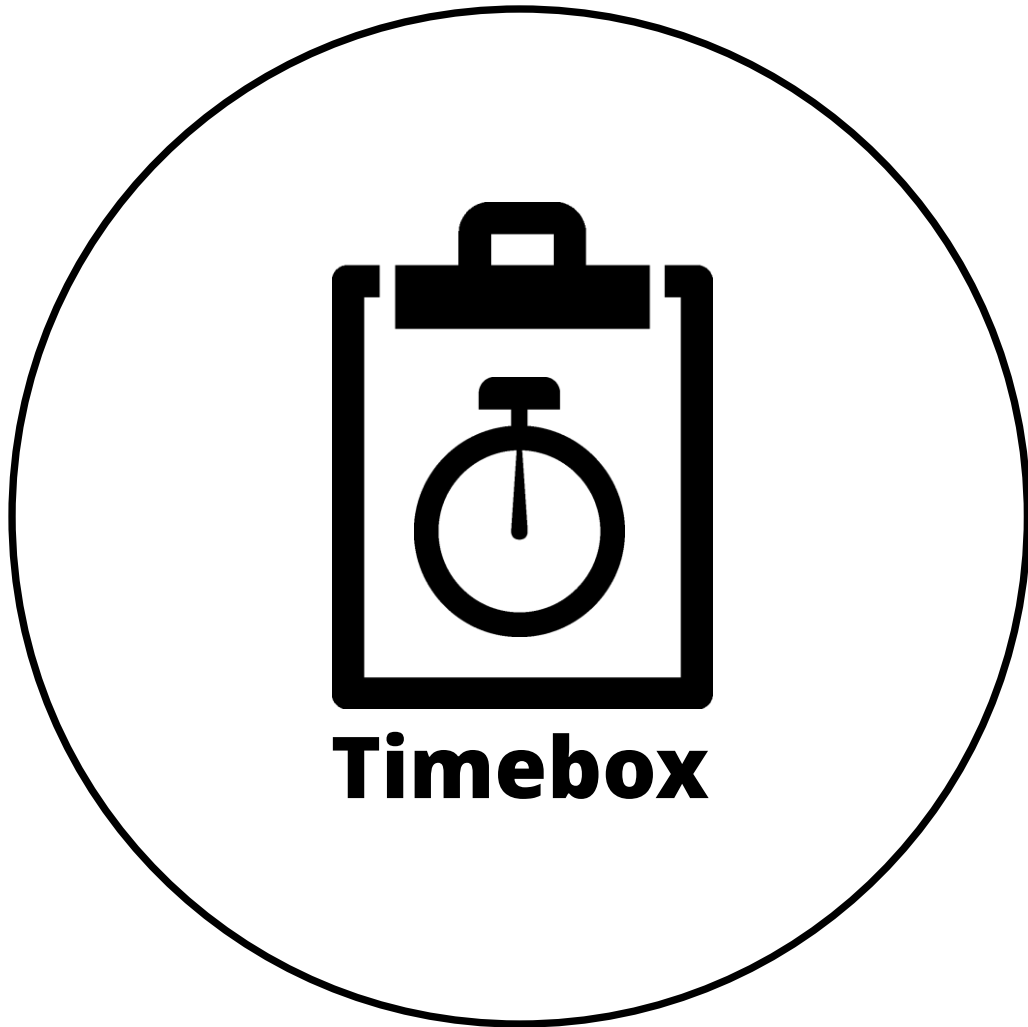
Structure exploratory testing to allow large-scale implementations

2

3

4

5



Timebox

Session-Based Testing

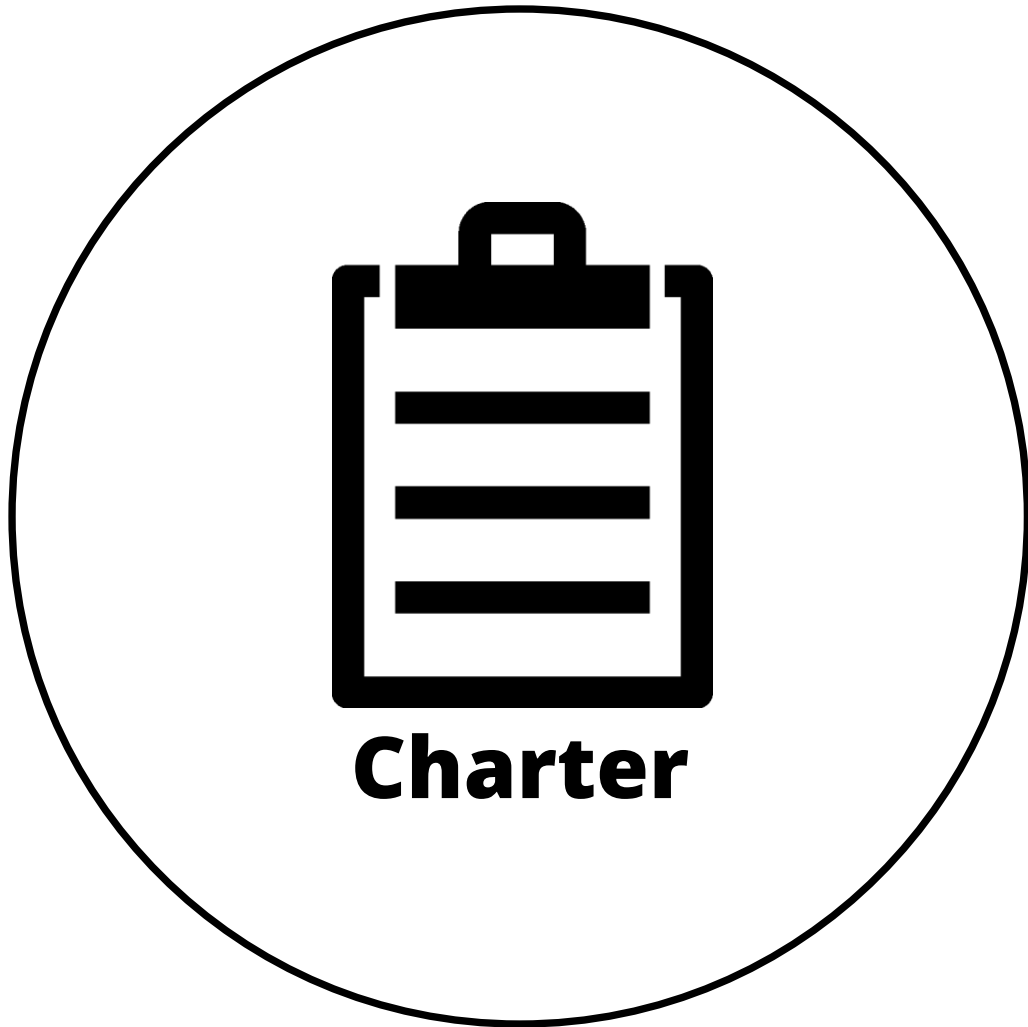
Structure exploratory testing to allow large-scale implementations

2

3

4

5



Charter

Session-Based Testing

Structure exploratory testing to allow large-scale implementations

2

3

4

5



Scope

Straightjacket

Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

Limit the scope to make it manageable

3

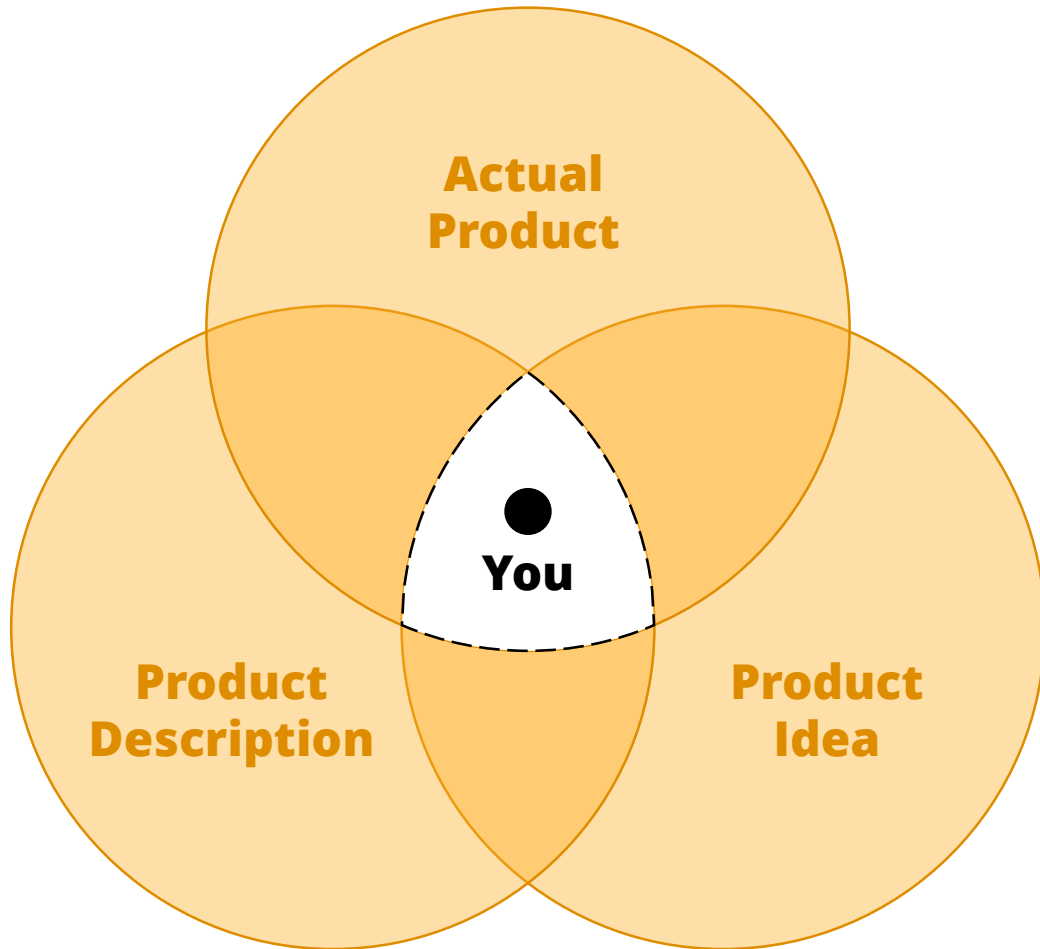
4

5

TRICENTIS ◊ Testing is more than just checking the specification.

Validation

Does our software meets the user's needs?



Verification

Does our software meet the specification?

Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

Limit the scope to make it manageable

3

4

5



BDD is the art of using examples in conversations to illustrate behavior



Liz Keogh

Session-Based Testing

Structure exploratory testing to allow large-scale implementations

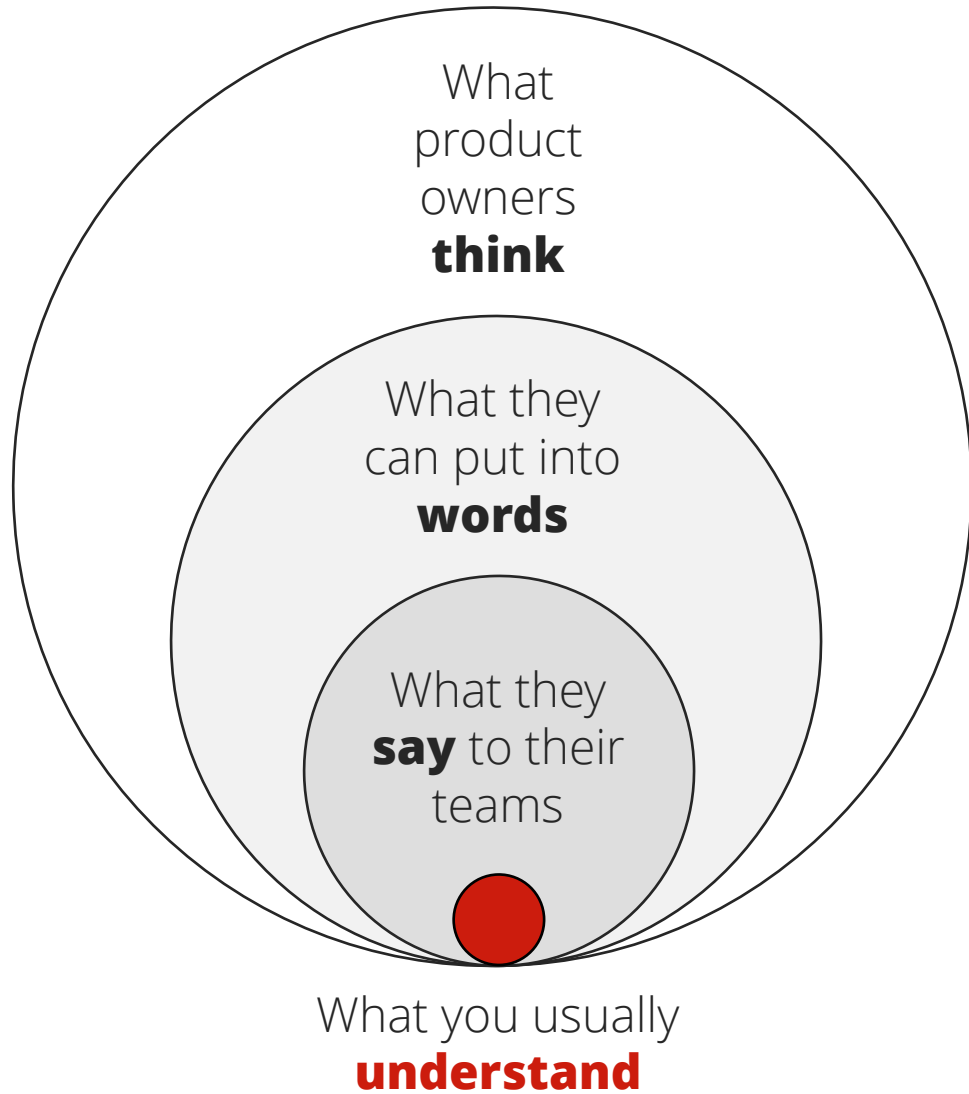
Requirements-Based Testing

Limit the scope to make it manageable

3

4

5



Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

Limit the scope to make it manageable

3

4

5

R » **Recent**

What parts of the product changed recently?

C » **Core**

What critical parts of the product must continue to work?

R » **Risky**

What parts of the product are inherently risky?

C » **Configuration**

What parts of the product depend on environment settings?

R » **Repaired**

What parts of the product changed to address defects?

C » **Chronic**

What parts of the product chronically break?

Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

Limit the scope to make it manageable

3

4

5

S » **Structure**

Test what the product is made of.

F » **Function**

Test what the product does.

D » **Data**

Test what the product processes.

P » **Platform**

Test what the product depends upon.

O » **Operations**

Test how the product is used.

T » **Time**

Test how the product is affected by time.

Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

Limit the scope to make it manageable

3

4

5



Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

Limit the scope to make it manageable

Tour-Based Testing

Set concrete goals to provide a clear focus

4

5



@speed
Quality



It's some value to some person

Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

Limit the scope to make it manageable

Tour-Based Testing

Set concrete goals to provide a clear focus

4

5

Quality is inherently **subjective**



Different stakeholders

will perceive the same product as having different levels of quality



We must look for **different things** for different stakeholders



We must **diversify** testing

Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

Limit the scope to make it manageable

Tour-Based Testing

Set concrete goals to provide a clear focus

4

5



Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

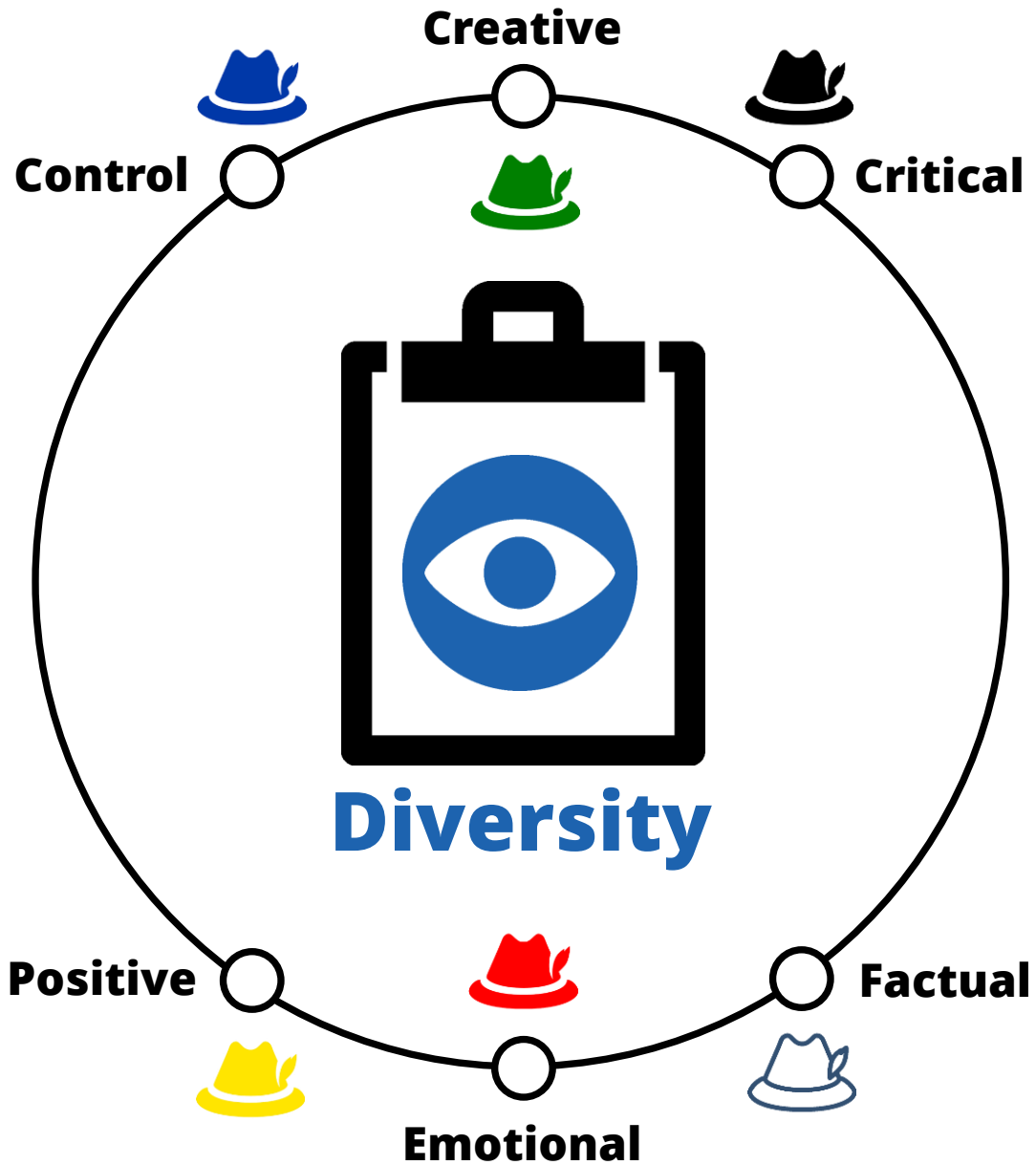
Limit the scope to make it manageable

Tour-Based Testing

Set concrete goals to provide a clear focus

Polychrome Testing

Explore the product from different viewpoints to diversify testing



Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

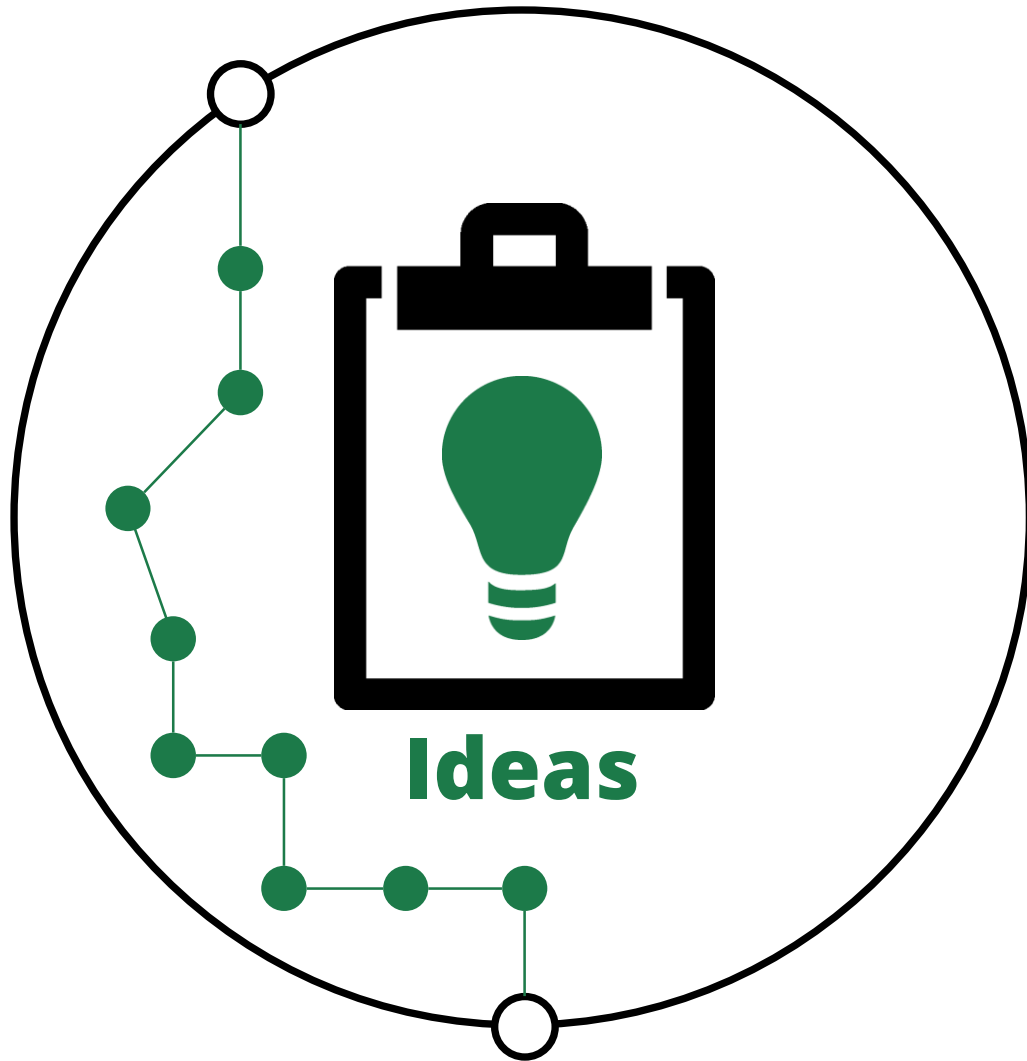
Limit the scope to make it manageable

Tour-Based Testing

Set concrete goals to provide a clear focus

Polychrome Testing

Explore the product from different viewpoints to diversify testing



Session-Based Testing

Structure exploratory testing to allow large-scale implementations

Requirements-Based Testing

Limit the scope to make it manageable

Tour-Based Testing

Set concrete goals to provide a clear focus

Polychrome Testing

Explore the product from different viewpoints to diversify testing

Scenario-Based Testing

Capture each test idea to make it reviewable

**LESSONS
LEARNED**

JUST AHEAD

TRICENTIS ◊ The term exploratory testing is redundant. Exploratory testing is testing. All testing is exploratory in nature.



Asking about **exploratory testing** is like asking about vegetarian cauliflower or metallic copper



Michael Bolton

TRICENTIS ◊ Speaking in terms of "test cases that must be developed" is a misleading way to discuss testing.



Testing is not about creating test cases,
it's about performing **experiments**



James Bach

TRICENTIS ◊ Your test success is not defined by the number of test cases you create, it's defined by the quality of your test ideas.



The test doesn't find the bug. A **human** finds the bug, and the test plays a role in helping the human find it



Pradeep Soundararajan



Automated checks **miss** the same
obvious things every single time



Ingo Philipp



If you want to become better at **testing**, then don't just hire somebody who is better at coding



Steve Watson



Testing is not so much a thing you **do**,
it's far more a way you **think**



Michael Bolton

TRICENTIS ◊ Do not expect answers to questions that are not asked.



Questions

The show is **over**. It's your turn.