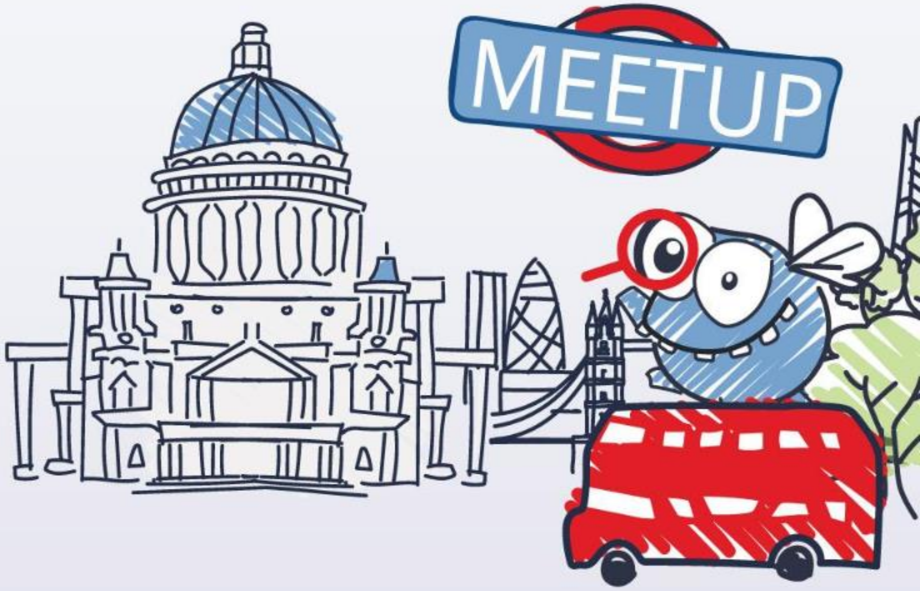
Post Trade Complexity and Disruptive Testing

Marina Kudryavtseva
20th June 2018



Build Software to Test Software
exactpro.com

EXITUS ACTA PROBAT

Three principles to test technology platforms

NO TRUST

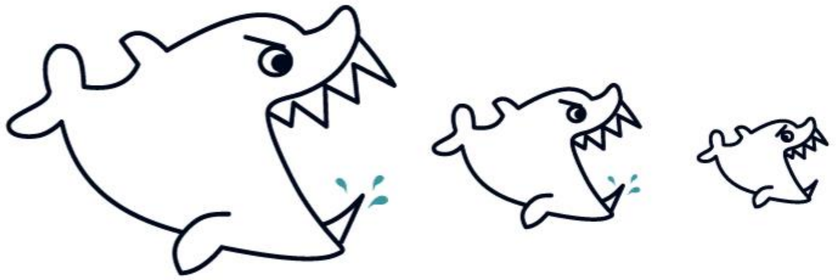
NO FEAR

NO BEGGING

What is the main difference between incumbent and disruptive?

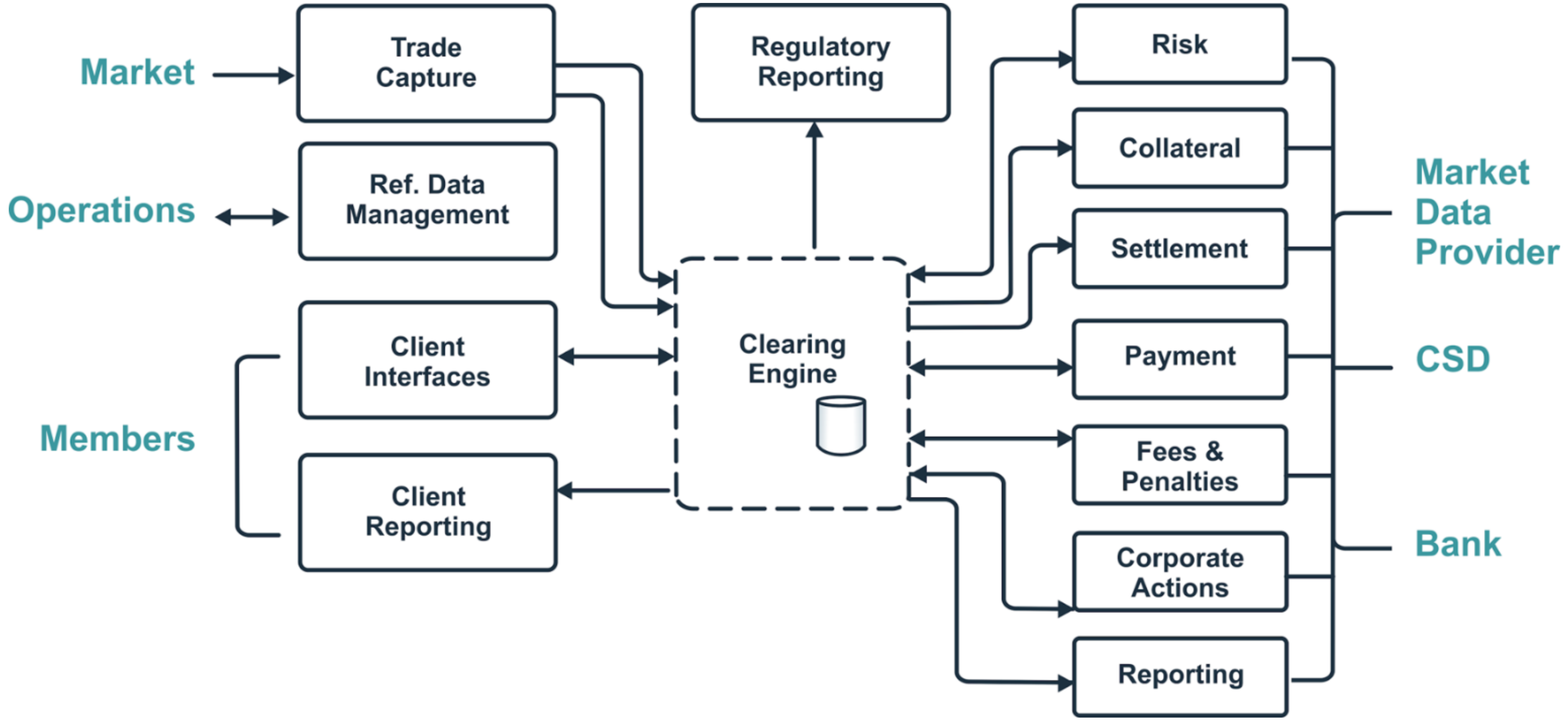
EXTENT CONFERENCE

Incumbent Technology

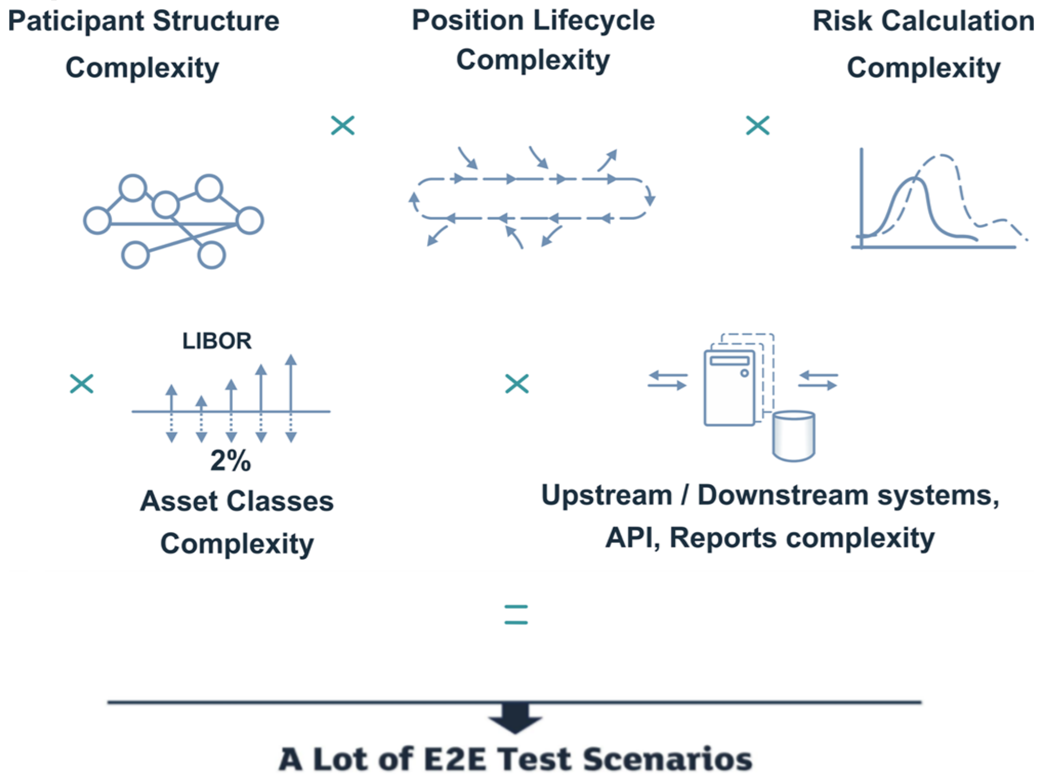


Disruptive Technology

Components of complex Post-Trade systems



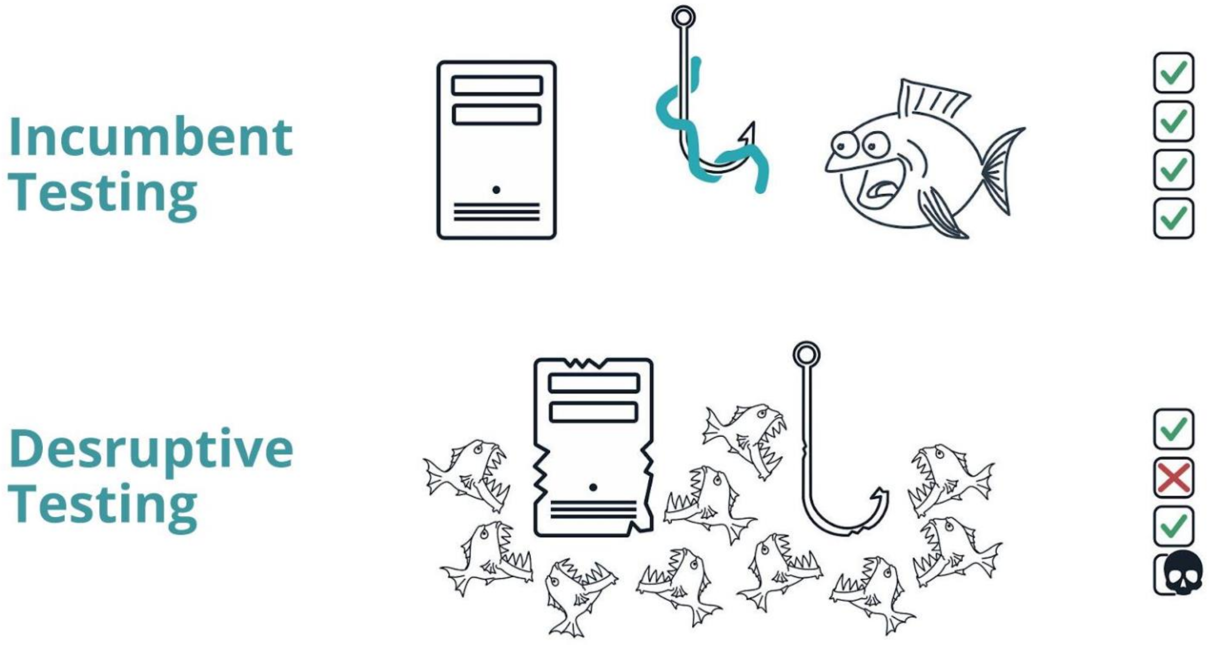
Key challenges in providing QA for Post-Trade platforms


Paticipant Structure Complexity
×
Position Lifecycle Complexity
×
Risk Calculation Complexity
× LIBOR
2%
Asset Classes Complexity
×
Upstream / Downstream systems, API, Reports complexity
=
A Lot of E2E Test Scenarios

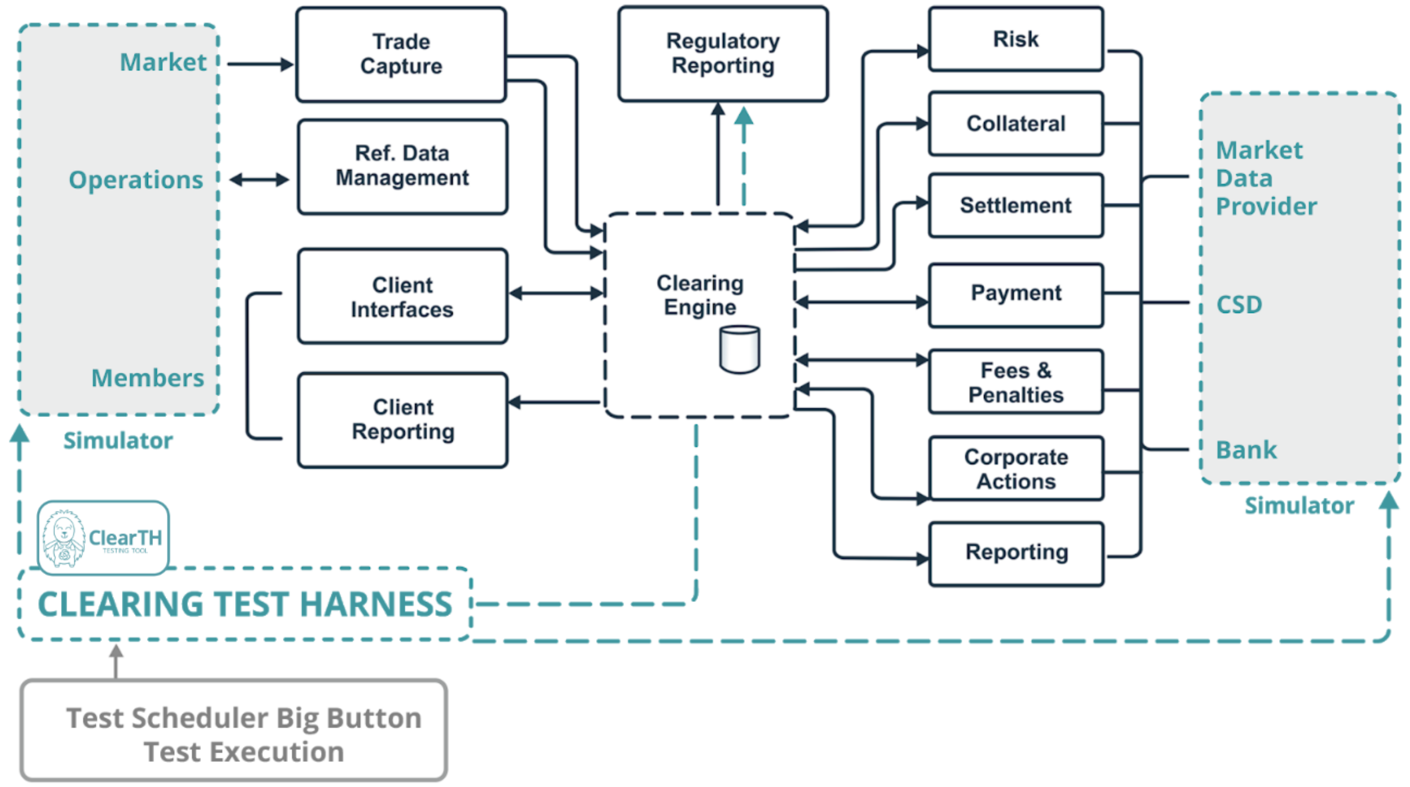
Test library parametrization

Get Off the Hook

With incumbent testing, you are on the hook of pre-conceptions of how your system under test is supposed to work.
It's only when you do disruptive testing that you really gain new knowledge about the system and learn from it.

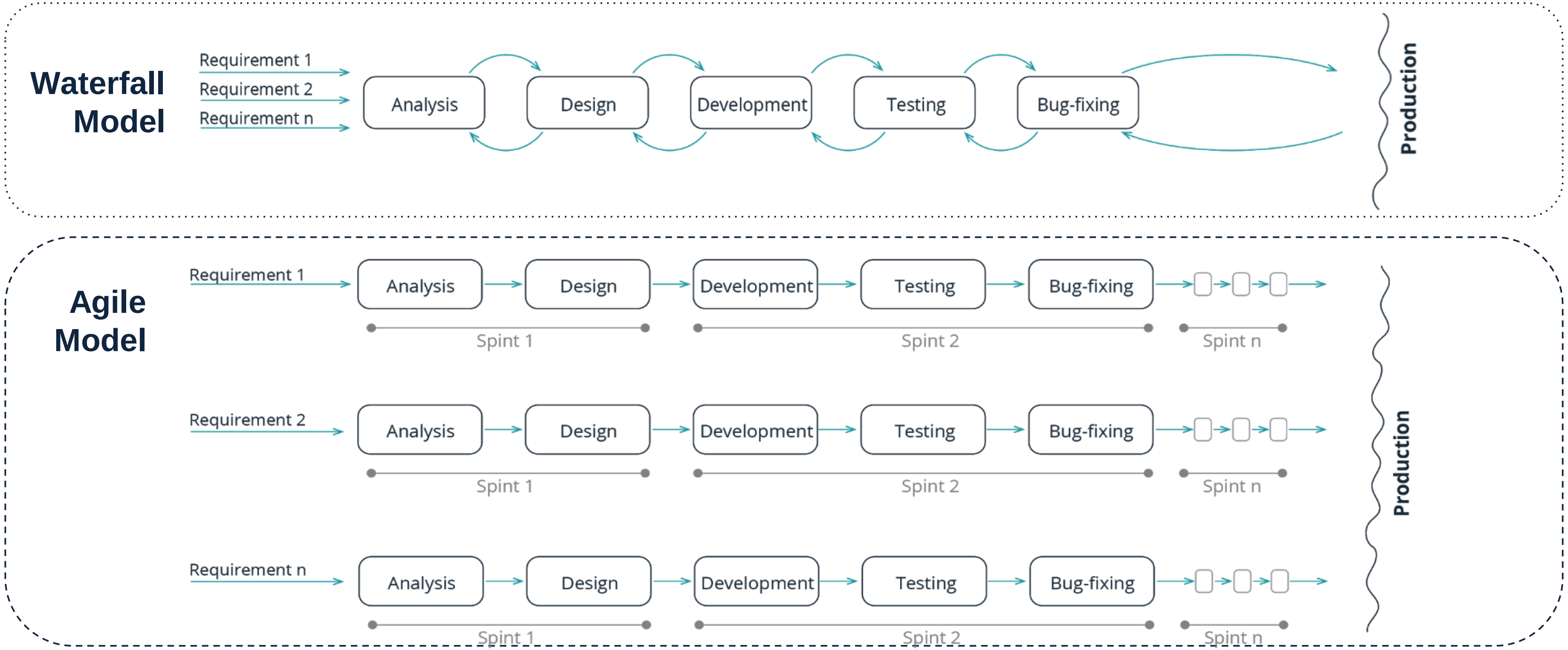


Holistic Integrated Automation Test Framework



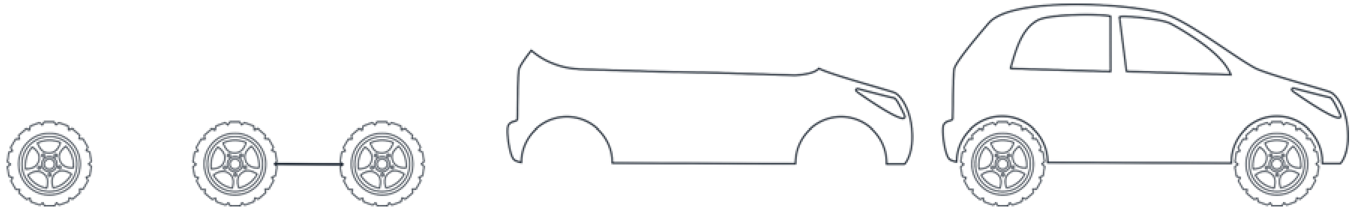
Agile Transformation

Most of the large financial sector organization are going through an Agile transformation.

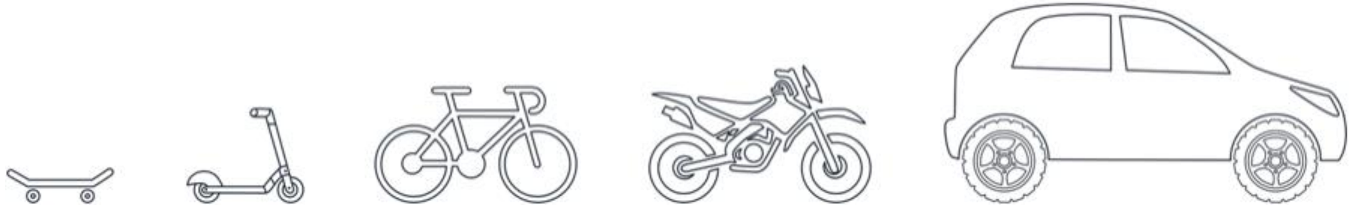


What Agile Development Should Be… and Not

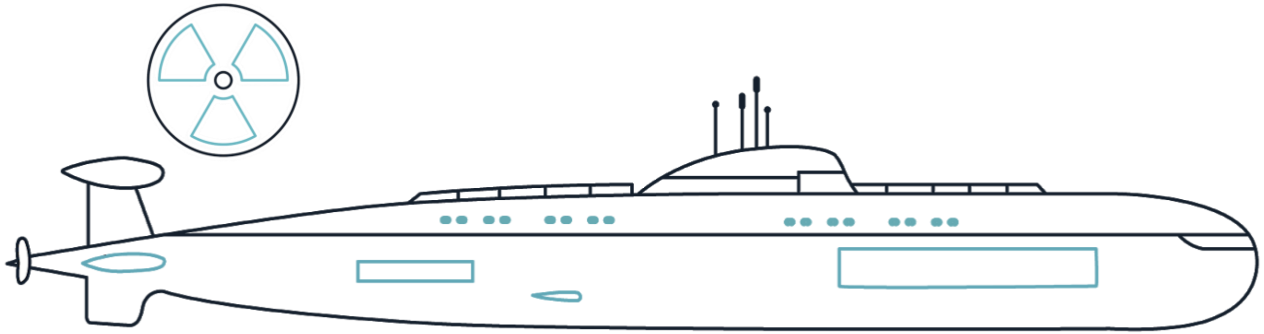
NOT LIKE THIS



LIKE THIS

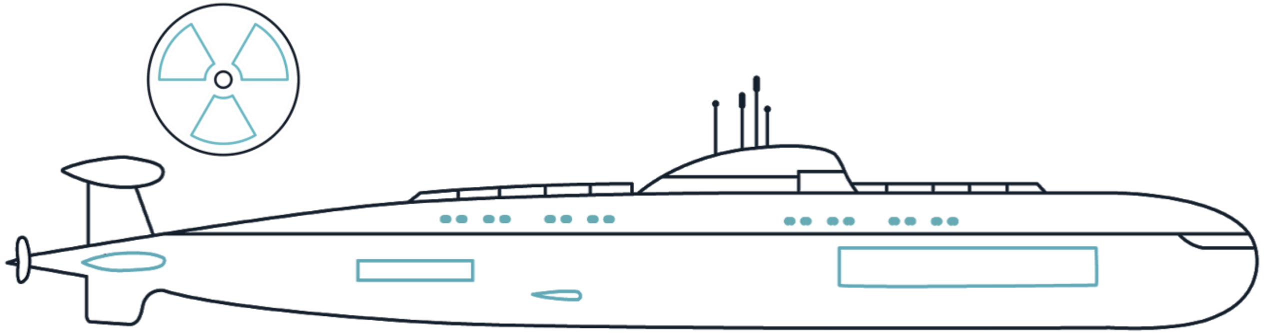


Testing Critical Infrastructures



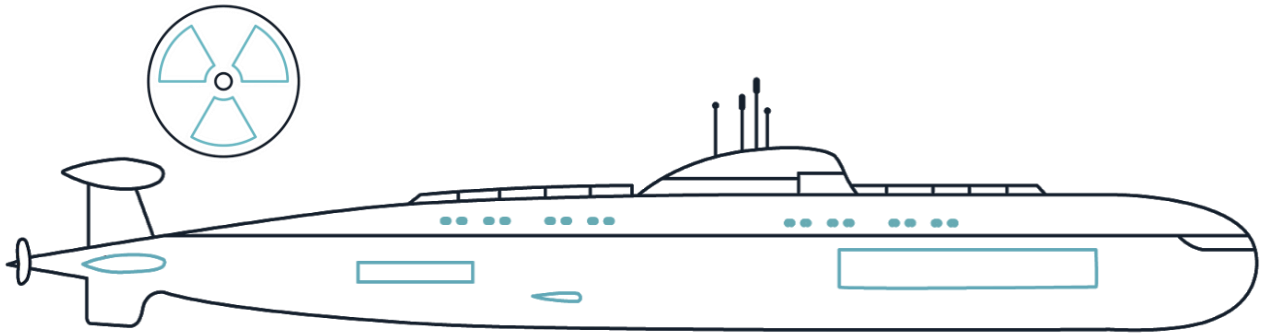
Testing Critical Infrastructures

Safety Rule #1 with Submarines: don't open portholes when underwater!



Testing Critical Infrastructures

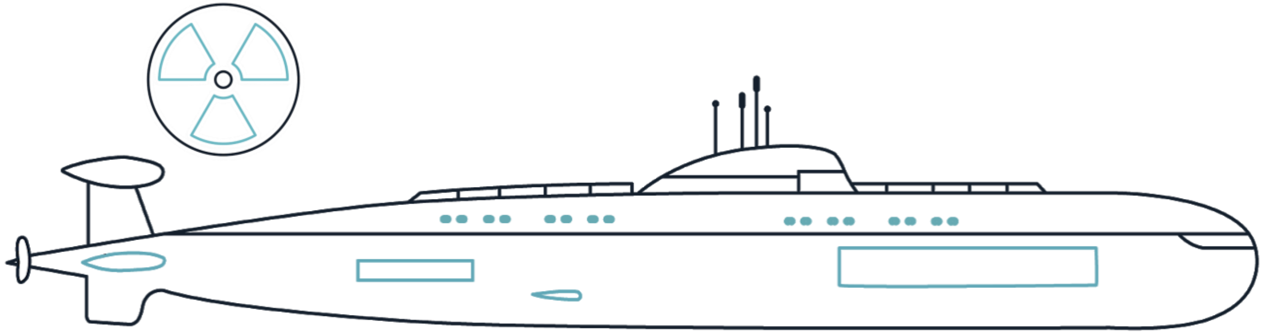
Safety Rule #1 with Submarines: don't open portholes when underwater!



Functional testing: iterate through a finite number of scenarios to prove that the porthole won't open
Non-Functional testing: iterate through a smaller number of scenarios to prove that it won't open by brute force

Testing Critical Infrastructures

Safety Rule #1 with Submarines: don't open portholes when underwater!



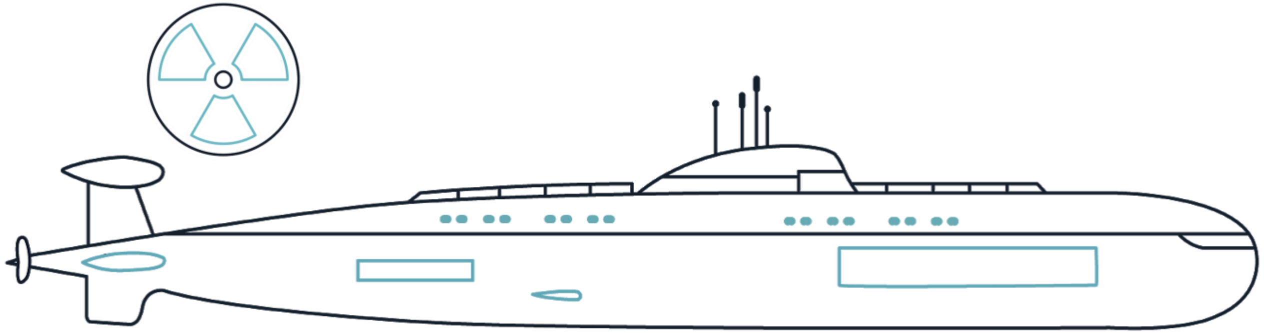
Functional testing: iterate through a finite number of scenarios to prove that the porthole won't open
Non-Functional testing: iterate through a smaller number of scenarios to prove that it won't open by brute force

Disruptive testing:
1) iterate through a huge number of random diverse scenarios under load to prove that it won't open

Testing Critical Infrastructures

EXTENT CONFERENCE

Safety Rule #1 with Submarines: don't open portholes when underwater!



Functional testing: iterate through a finite number of scenarios to prove that the porthole won't open
Non-Functional testing: iterate through a smaller number of scenarios to prove that it won't open by brute force

Disruptive testing:
1) iterate through a huge number of random diverse scenarios under load to prove that it won't open
2) open the porthole

EXTENT CONFERENCE

Thank you!