Generator-based Testing: A State by State Approach

Chris Struble

PNSQC 2020

About me

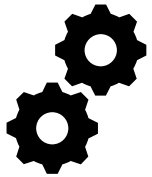
Chris Struble

- Senior Software Development Engineer in Test (SDET)
- Over 25 years in software industry
- Joined Meteorcomm (MCC) in 2019
- Currently focused on functional test automation
- I live in Renton, Washington with my wife, daughter and two cats
- I play guitar and computer strategy games, and write music and fantasy fiction

Key points

- What is Example-based testing (EBT)?
- What is Generator-based testing (GBT)? How is it different from EBT? Why use it?
- What are three examples of GBT practices? How are they different? What are the advantages of each?
- How can I master GBT and apply it in my career?

Evolution of Software Testing

	Test Design	Test Execution
Past		
Present		

A manually designed test case…

- Given I have a locomotive ***travelling at 80-mph***
- When it approaches a curve with ***a 30-mph speed limit***
- Then an overspeed warning message should be sent to the operator console
- And the operator does not slow the train ***within 30 seconds***
- And I should see the locomotive slow safely to a stop automatically

…that uses a concrete scenario…

… to suggest a general claim is

- Given I have a locomotive **travelling at some speed**
- When it approaches a curve with **a lower speed limit**
- Then an overspeed warning message should be sent to the operator console
- And the operator does not slow the train ***in time***
- And I should see the locomotive slow safely to a stop automatically

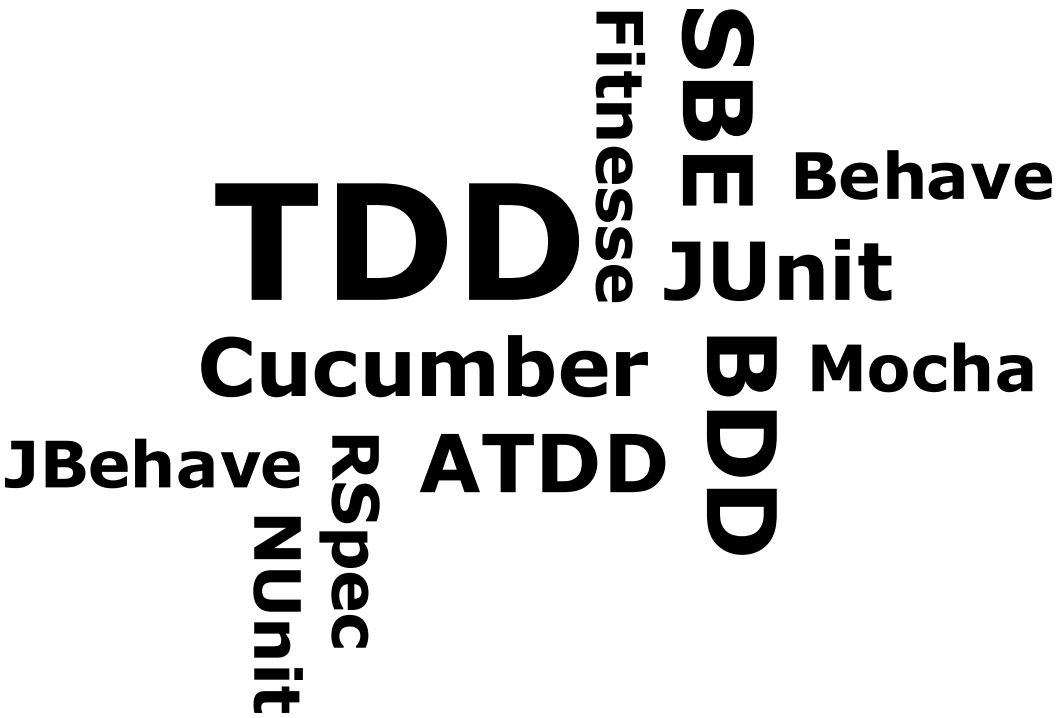
… an “example-based test case”.

Example-based testing (EBT)

Any software testing practice where:

- Test cases are manually designed one example at a time
- With the goal of suggesting that the software works
- By demonstrating automatically that each example works

EBT practices and tools



EBT is good. But…

Limitation

- It takes MANY concrete examples for people to be confident about a general claim
- People are not good at thinking of many examples

Consequence

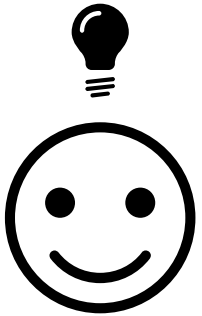
- Defects escape
- Lack of imagination (“no one thought of that”)

Generator-based testing (GBT)

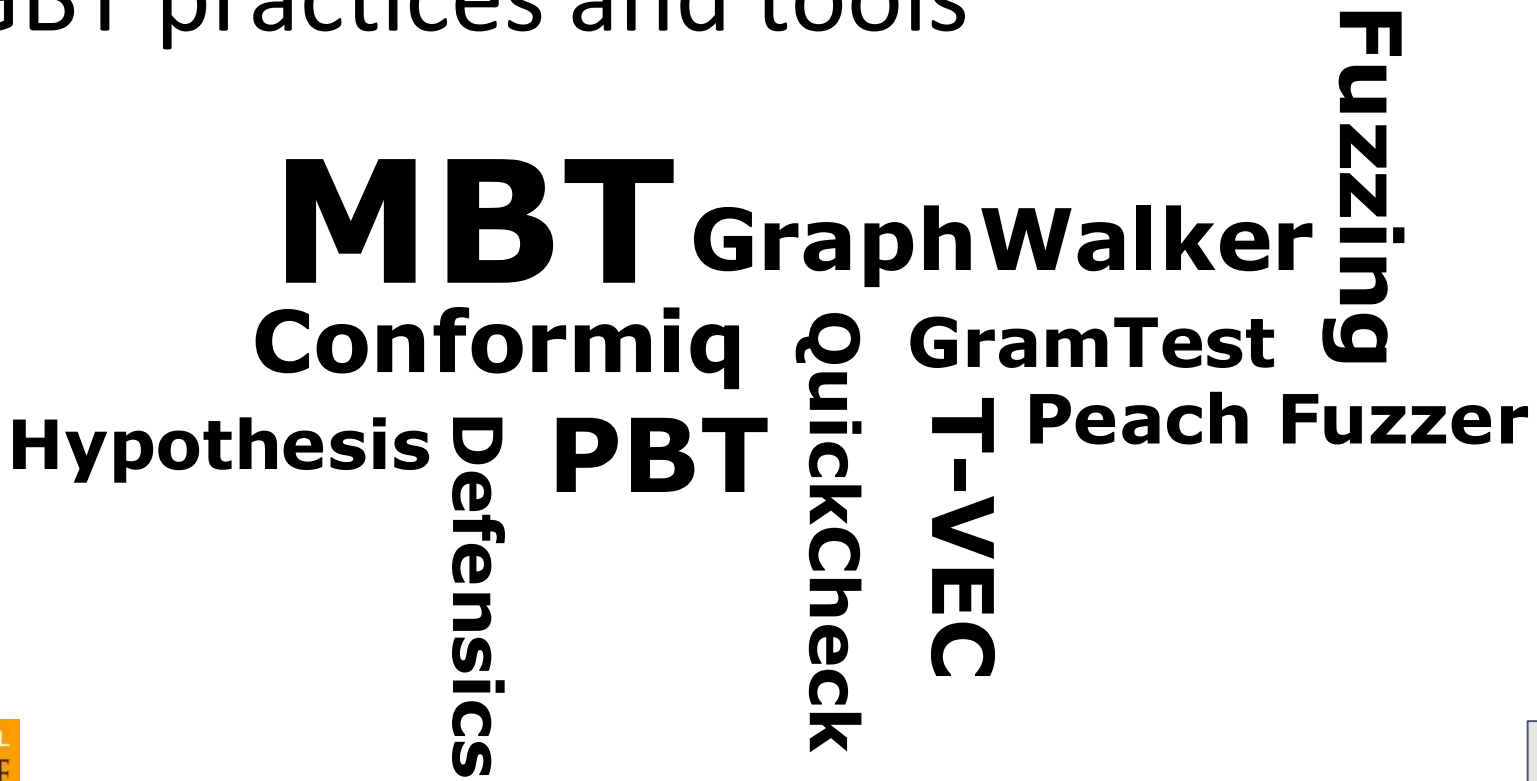
Any software testing practice where:

- Test cases are generated automatically from a description of behavior
- With the goal of discovering defects in the software
- By demonstrating automatically if each generated test case works

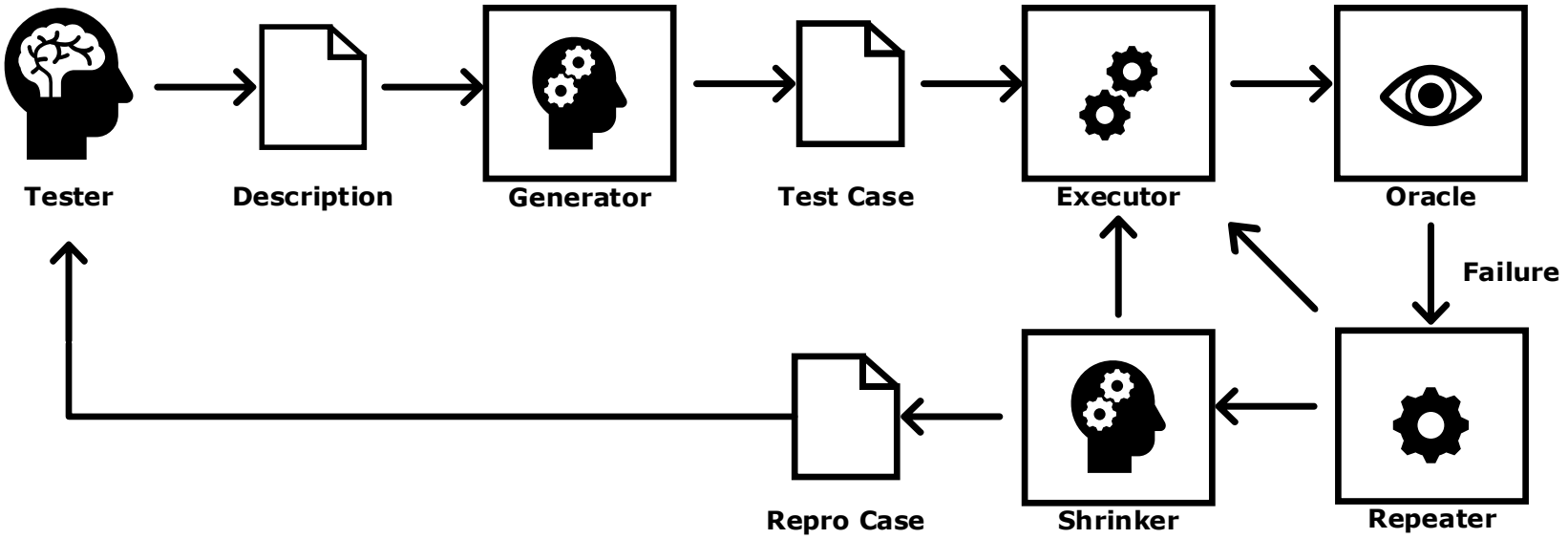
What if software could “think of” more examples?



GBT practices and tools



GBT process


Tester
Description
Generator
Test Case
Executor
Oracle
Failure
Repeater
Shrinker
Repro Case

Three GBT practices

- Model-based Testing (MBT)
- Fuzzing
- Property-based Testing (PBT)

Each has been used for at least 20 years

Each has found major escaped defects

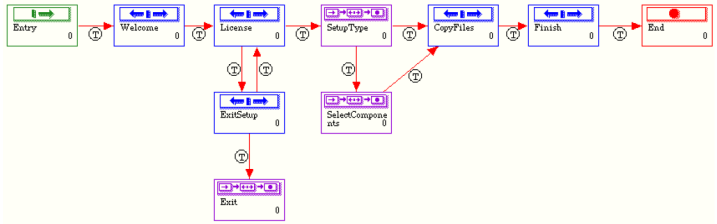
Model-based testing (MBT)

- Description is a directed graph of the possible software behaviors
  - State machine
  - Process flow diagram
- Generator uses a search algorithm to traverse graph
- Used since late 1990s
  - Protocols, UI, workflows,
  - ISTQB Model-based Tester Certification – since 2015

"Automate your brain, not just your hands." - Harry Robinson

My MBT Journey – 2000 - 2020

- In 2000 I used the TestMaster MBT tool to generate tests for HP printer driver installer
  - In six months I found over 100 defects
- In 2008 I released Hanno MBT tool for web application testing in Java
- In 2012 I used Microsoft Spec Explorer MBT tool to test a .NET web application



Mars Polar Lander Crash - 1999

- Probe crashed during landing
- NASA suspected a software defect
- Team used T-VEC MBT tool to model and generate tests for the Touchdown Monitor system
- Detected the fault in a few weeks
- Engines shut down 40 meters above the surface

MBT ✓

Fuzzing (Fuzz Testing)

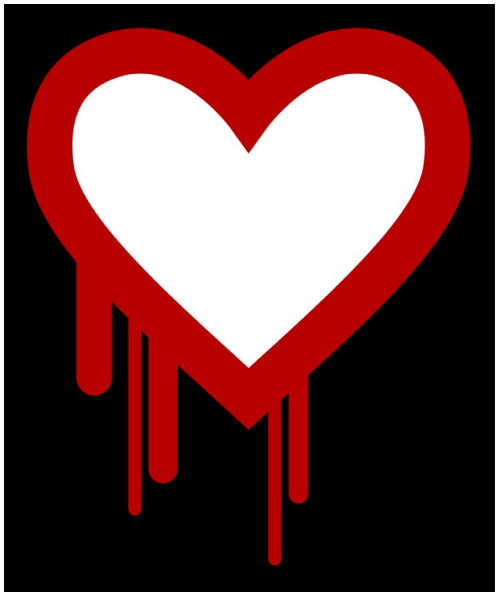
- Description of interface only
- Generator (fuzzer) creates random or malformed input data until a crash or exception occurs
- Shrinks automatically on failure
- Used since 1990
  - Parsers, grammars
  - Web security testing
  - User input devices

Fuzz (v): to envelop in a haze

HeartBleed Vulnerability - 2014

- OpenSSL vulnerability since 2012
- Possible to penetrate computer systems without leaving a trace
- Discovered in April 2014
  - Neel Mehta of Google, using code inspection
  - Synopsis team in Finland, using Defensics fuzzing tool
- Still unpatched systems today

Fuzzing ✓



Property-based Testing (PBT)

- Description is a “property” or “general claim”, written in code
- Generator uses random input data until it finds a falsifying example
- Shrinks automatically on test failure
- John Hughes creates QuickCheck in 1999
  - Haskell -> 35 languages
- David MacIver creates Hypothesis in 2015
  - Python -> Java, Ruby

```
from hypothesis import given
from hypothesis.strategies import text

@given(text())
def test_decode_inverts_encode(s):
    assert decode(encode(s)) == s
```

“Don’t write tests. Generate them!”
- John Hughes

Volvo Emergency Braking - 2015

- Volvo hired John Hughes' company (Quviq) to test the AutoSAR embedded system in its vehicles
- The Quviq team in Sweden used the QuickCheck PBT tool to define properties and generate tests
- Found over 200 new defects
- Fault allowed emergency braking system to be given a lower priority than adjusting the volume

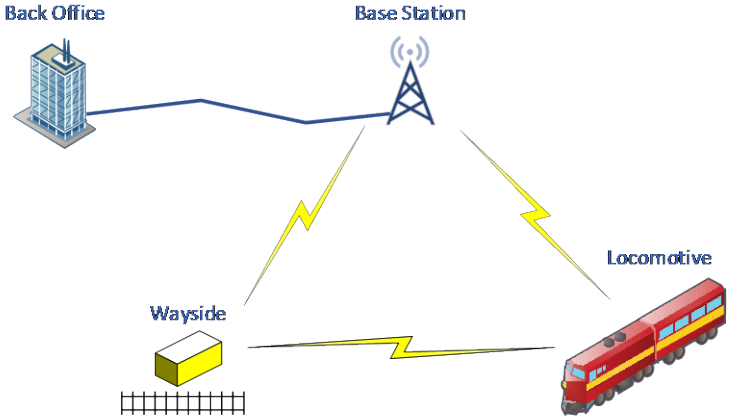
PBT ✓

GBT is not widely adopted

Search Term	LinkedIn Worldwide Job Results
Software test quality	15513
Software test quality “automated”	5669
Software test quality “TDD”	823
Software test quality “BDD”	622
Software test quality “ATDD”	72
Software test quality “MBT”	3
Software test quality “Fuzz”	2
Software test quality “PBT”	0

About Meteorcomm

- Telecommunications company based in Renton, Washington
- Develops Interoperable Train Control Messaging (ITCM), a messaging system for railroads
- Positive Train Control (PTC), a safety system mandated by the Rail Safety Improvement Act of 2008



Introducing MBT at Meteorcomm

- In July 2019 I decided to introduce MBT at Meteorcomm
- I wanted to finish in a two-week “innovation sprint”
- I wanted to use Ruby because we use it for our EBT tests
- I used GraphWalker, an open source MBT tool written in Java, with a REST API
- I wrote Test Generator, a Ruby program that uses GraphWalker to generate and execute Ruby test code

Demos

- Test Generator – Ruby MBT tool I wrote
  - Show an example model for an ITCM test component
  - Generate, run and rerun a functional test case

- Rantly – open source PBT tool for Ruby
  - Start with EBT RSpec unit tests
  - Create a PBT unit test, generate, fail, shrink

Getting Started with GBT

- Start with your existing EBT functional test cases
- Create a script to run them in random order
- Your EBT tests will be more reliable
- Increase your comfort level with randomness

"Can you let randomness into your life?" – TJ Usiyan, *Property-Based Testing with SwiftCheck*

Contact Me

LinkedIn

- https://www.linkedin.com/in/chrisstruble/

Blog

- https://chrisstrublewrites.blogspot.com

Email

- cstruble@meteorcomm.com

Questions