Advanced Test Automation and Analytics

Derrill Sturgeon

derrill.sturgeon@devicepros.net

Testing includes wide and deep problems

Wide

- Problem created by large surface area to test:
- Interface & protocol testing - large array of command and interactions between outside world and the device under test
- Don't go it alone, leverage as much as possible - 3rd party tools and labs

Deep

- Read and write, data related test
- Small surface area, but very deep due to enormous state space
- Requires white and black box approaches

Test Automation is essential to solve both these challenges

If test results look like this

Machine	Start Time	End Time	Status
TS052	8/8/16 16:34	8/8/16 17:39	FAIL
TS053	8/8/16 16:36	8/8/16 17:41	FAIL
TS054	8/8/16 16:38	8/8/16 17:43	FAIL
TS055	8/8/16 16:40	8/8/16 17:45	FAIL
TS056	8/8/16 16:42	8/8/16 17:47	FAIL
TS057	8/8/16 16:44	8/8/16 17:49	FAIL
TS058	8/8/16 16:46	8/8/16 17:51	FAIL
TS059	8/8/16 16:48	8/8/16 17:53	FAIL
TS060	8/8/16 16:50	8/8/16 17:55	FAIL
TS061	8/8/16 16:52	8/8/16 17:57	FAIL
TS062	8/8/16 16:54	8/8/16 17:59	FAIL
TS063	8/8/16 16:56	8/8/16 18:01	FAIL

Or even like this

You are probably not too worried about test effectiveness

Machine	Start Time	End Time	Status
TS052	8/8/16 16:34	8/8/16 17:39	PASS
TS053	8/8/16 16:36	8/8/16 17:41	PASS
TS054	8/8/16 16:38	8/8/16 17:43	FAIL
TS055	8/8/16 16:40	8/8/16 17:45	PASS
TS056	8/8/16 16:42	8/8/16 17:47	PASS
TS057	8/8/16 16:44	8/8/16 17:49	PASS
TS058	8/8/16 16:46	8/8/16 17:51	PASS
TS059	8/8/16 16:48	8/8/16 17:53	FAIL
TS060	8/8/16 16:50	8/8/16 17:55	PASS
TS061	8/8/16 16:52	8/8/16 17:57	PASS
TS062	8/8/16 16:54	8/8/16 17:59	PASS
TS063	8/8/16 16:56	8/8/16 18:01	FAIL

But for test results like this

It is not obvious how to distinguish between high product quality and poor tests.

Machine	Start Time	End Time	Status
TS052	8/8/16 16:34	8/8/16 17:39	PASS
TS053	8/8/16 16:36	8/8/16 17:41	PASS
TS054	8/8/16 16:38	8/8/16 17:43	PASS
TS055	8/8/16 16:40	8/8/16 17:45	PASS
TS056	8/8/16 16:42	8/8/16 17:47	PASS
TS057	8/8/16 16:44	8/8/16 17:49	PASS
TS058	8/8/16 16:46	8/8/16 17:51	PASS
TS059	8/8/16 16:48	8/8/16 17:53	PASS
TS060	8/8/16 16:50	8/8/16 17:55	PASS
TS061	8/8/16 16:52	8/8/16 17:57	PASS
TS062	8/8/16 16:54	8/8/16 17:59	PASS
TS063	8/8/16 16:56	8/8/16 18:01	PASS

Using Stress metrics as a quantitative feedback

- Evaluating Test Effectiveness
- What does stress look like?
- When to measure stress
- Implications to an effective Test Automation System

Evaluating test effectiveness

- **Specific Feature tests**: Intent is clear and focus is relatively narrow. Code review is likely to shed light on how comprehensive the test is. With some effort, effectiveness can be established.
- **Power cycle tests**: More difficult since state space is large and preconditions may not be trivial and cycle time is relatively long.
- **Access pattern tests**: Similar to power cycle in terms of state space and setup time.
- **Random IO**: Difficult to evaluate quantitatively. But it is a mistake to assume all (so called) random techniques have similar effectiveness. Random IO is highly effective on immature codebases but reaches plateau. Inefficient for issues that require an elaborate set of preconditions.

Quantitative stress metrics can remove a lot of the guess work.

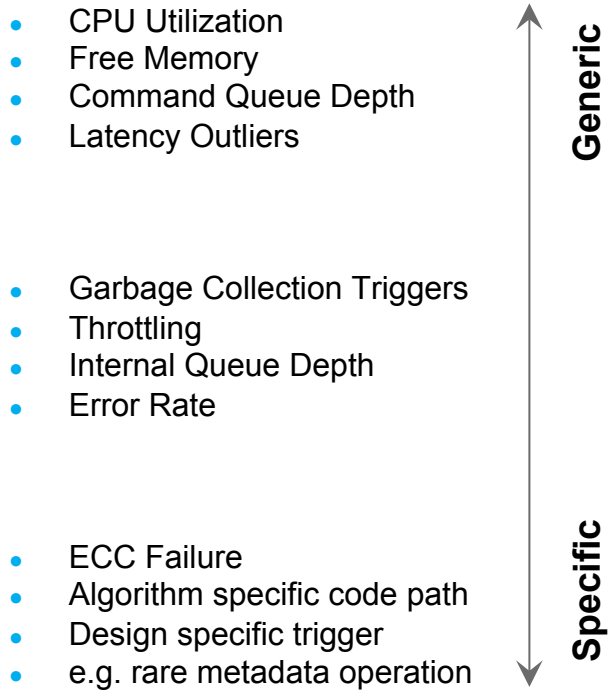
What does stress look like?

- Complex code paths
- Exception handling
- Low memory conditions
- Low stack space, high stack depth
- Saturated performance (IOPS/latency)
- ...lots of other things design specific states and events

Examples

Great tests cause stress and it is objectively measurable. Imagine two tests that run for 1 hour:

- **Test A causes 30 GC events and 3 file system compactions.** Several GC events occur while stack depth is extreme.
- **Test B causes 3 GC events and 0 file system compactions.** Stack depth is moderate throughout the test.
- **Both tests result in a PASS outcome.** But, is there any doubt about which of these tests gives you more confidence?



When to measure stress

Never (seems to be what most people do)

- No way to compare relative effectiveness of tests
- Given scarcity of target hardware, reduces chances of product success

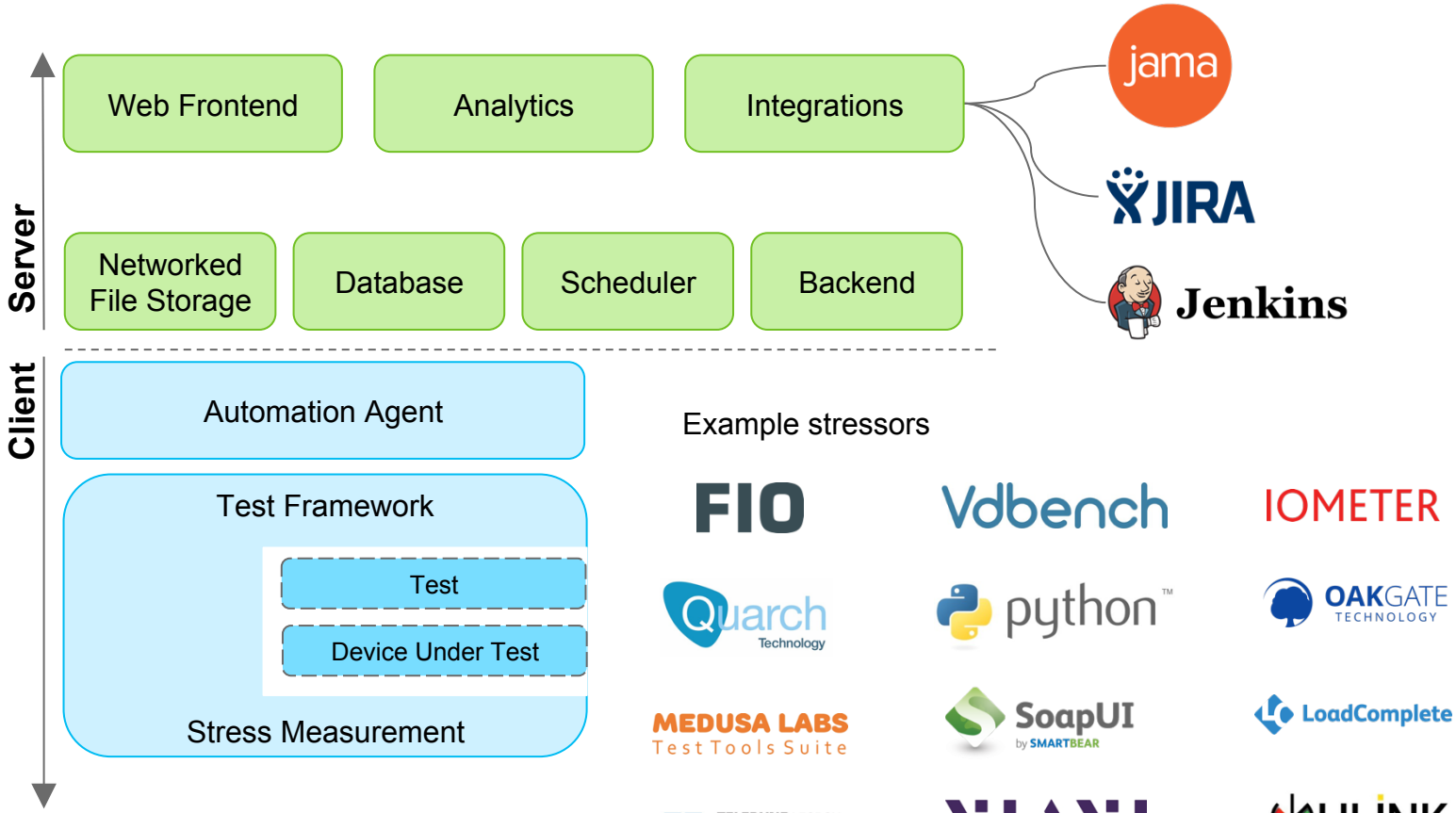
During Test Development

- Evaluation of stress indicators while tests are under development will result in greatly improved effectiveness
- Ineffective tests never make it to the lab

Continuously

- Ability to detect when changes in target code reduce test effectiveness. Can respond quickly with new tests rather than having false sense of security from ineffective passing results.

Automation Overview



Requirements Management Integration

- Executable Test Plan documents
- Traceability from Requirements to Test Planning to Test Execution to Results to Issue Tracking and Releases.
- Select testing focus by Requirements. Useful in development stages.
- Associate test runs and failures with requirements
- Monitor test stress by requirement

Devicepros

Straight Forward

Contact:
derrill.sturgeon@devicepros.net

- Our company motto embodies simplicity, efficiency, and transparency. We adopt best practices ruthlessly enabling speed, security and control as needed;
- Led by storage industry veterans, Derrill Sturgeon and Andy Tomlin. Based in Bay Area with development center in Minsk, Belarus providing expertise in Test Automation and Flash based systems to the Storage industry;
- Passionate about helping our customers achieve their goals! We aim to delight our customers with our services, not merely satisfy;
- Excellence in communication is one of our top priorities. Engineering Services requires robust and frank conversations from inception through execution.