

Paris, 16-18 October 2018



Organizer:  **TESTING
SOLUTIONS
& SERVICES**

How testing enables new technology advancements

Helping you find clarity in the face of complexity

© All rights reserved **Presented by Jeff Warra**

Hmm... Testing
enabling
technology...



Like saying which
came first the
Chicken or the
egg..

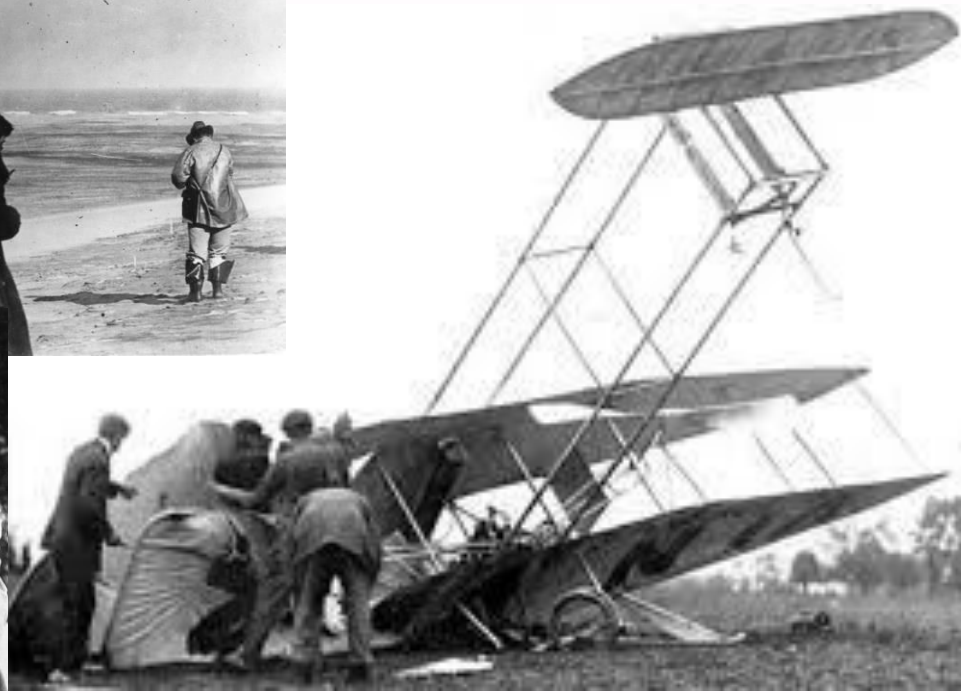
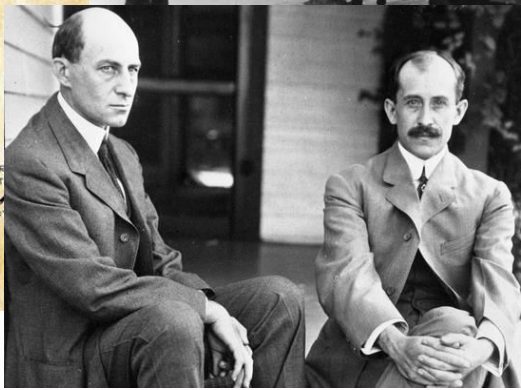
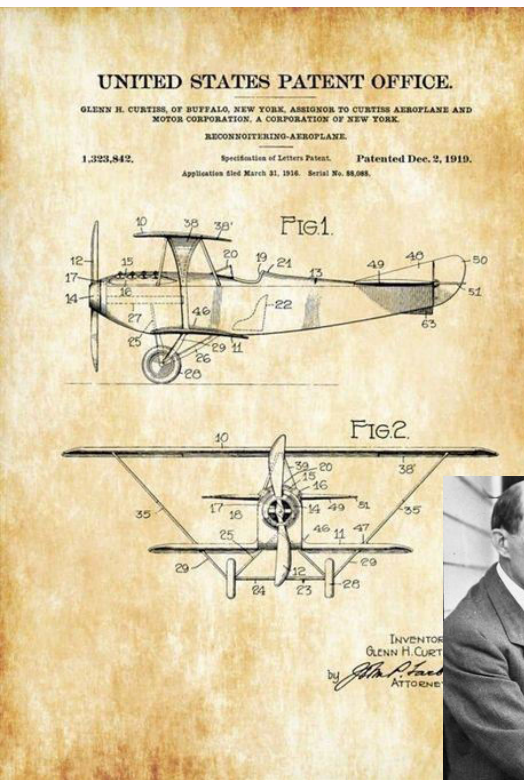
We can debate
this forever.. But..

Inventions by accident

- Matches
- The color dye, Mauve aka, Purple
- Penicillin
- Microwave Oven
- Plastics
- Potato Chips
- X-Rays
- Safety glass (by French artist/chemist Edouard Bénédictus)
- Viagra
- Chocolate chip cookies
- Post-It Notes
- Pacemaker - Wilson Greatbatch
 - wrong resistor...in recording device



How much testing did these guys do?



Smart Grid

Avionics

Smart Cities / Pub safety

Industrial Automation

Banking

Professional Audio &
Video

Construction Equipment

Automotive

Healthcare/Imagery

Military Equipment

Telecom

Agriculture

Heavy Trucks

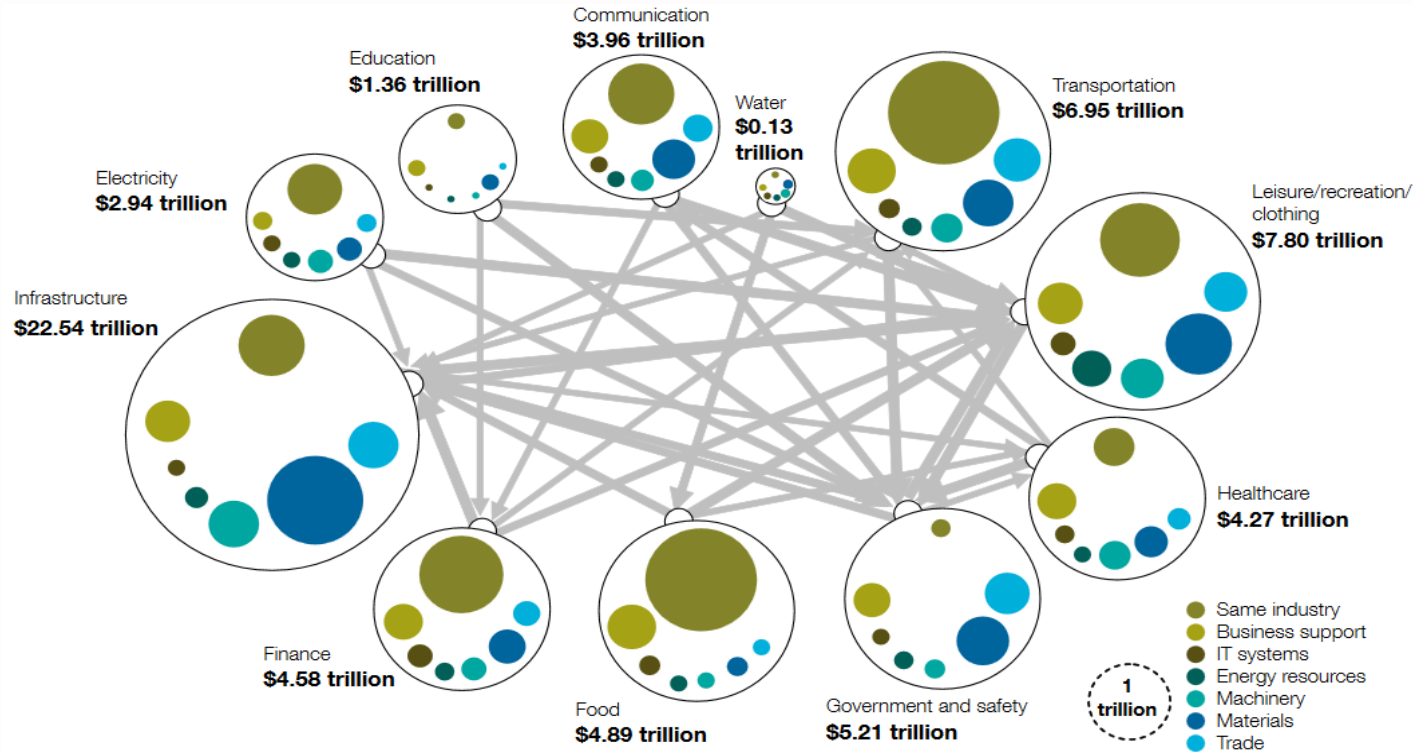
Oil & Gas

Smart Manufacturing

Testing the technology that supports, many verticals

The world's 4 trillion dollar challenge

Using a system-of-systems approach to build a smarter planet



Note: Size of bubbles represents systems' economic values. Arrows represent the strength of systems' interaction.

Source: IBM Institute for Business Value analysis of Organisation for Economic Co-operation and Development (OECD) data.

What is commonly needed to advance technology in these sectors?



Smart Grid

Avionics

Smart Cities / Public Safety

Industrial Automation

Banking

Navigation

Telecommunications

Automotive

Agriculture

Manufacturing

Oil & Gas

How much data is generated EVERY minute?

Email users send more than 204 million messages

Mobile Web receives 217 new users

Google receives over 2 million search queries

YouTube users upload 48 hours of new video

Facebook users share 684,000 bits of content

Twitter users send more than 100,000 tweets

Consumers spend \$272,000 on Web shopping

Apple receives around 47,000 application downloads

Brands receive more than 34,000 Facebook 'likes'

Tumblr blog owners publish 27,000 new posts

Instagram users share 3,600 new photos

Flickr users, on the other hand, add 3,125 new photos

Foursquare users perform 2,000 check-ins

WordPress users publish close to 350 new blog posts.

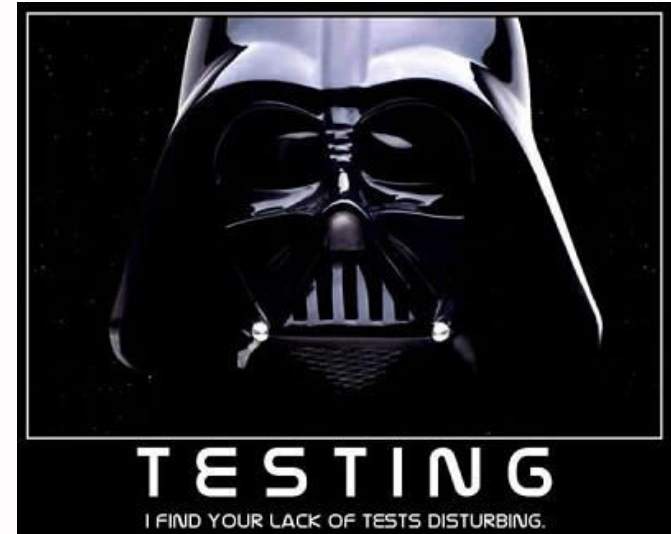
Scale Fail – The Dark Side of Online Ticket Sales

Unprecedented and simultaneous demand for Star Wars Episode VII: the Force Awakens® pre-sale tickets crashed numerous web servers for two days for several movie houses within minutes of becoming available.

“We spun up 40 simultaneous servers... fingers crossed... the massive simultaneous users exposed an unforeseen flaw in the infrastructure.. Which was unable to fix on the fly.. ”

Don't leave scales testing to chance, you have to scope and test your systems beyond expected number of users

Bench testing this by using a test case for millions of users is what needed to be done not, just 40 servers to meet such demand..

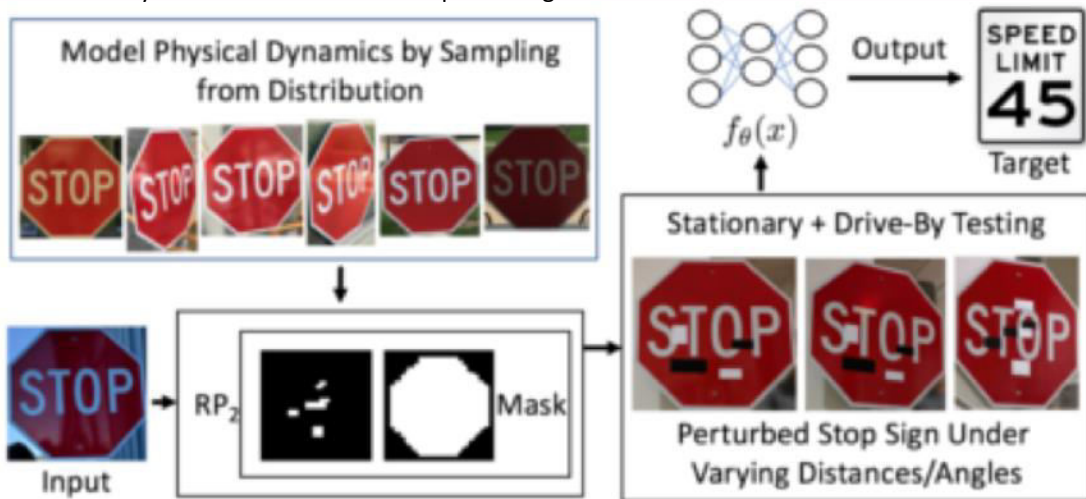


Hacking - Artificial Intelligence

- Deep learning Visual Classifiers – UofM research project – Atul Prakash
 - Presented at ESCAR, June 21, 2018

Illustrates how an AI can be manipulated to fool the algorithm to thinking the stickers represent a 45MPH traffic sign vs a Stop sign

Robust Physical-World Attacks on Deep Learning Visual Classifiers – Atul Prakash UofM Ann Arbor



Building Confidence - Interoperability

Because of no one wants to have a 1 way conversation.

What is interoperability?

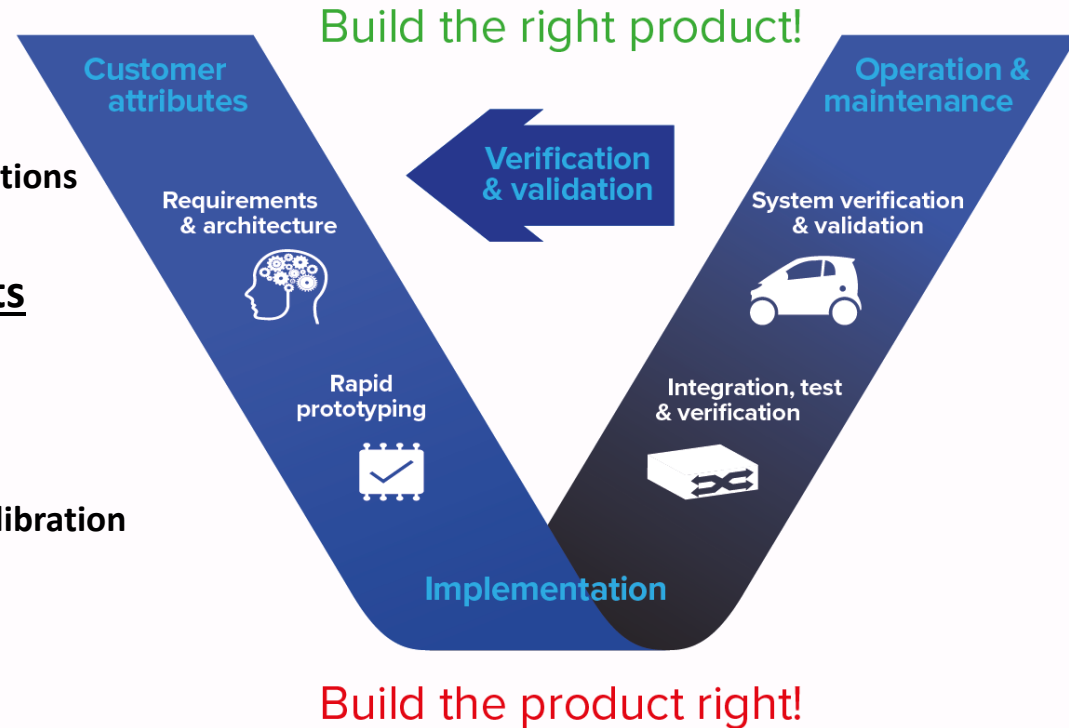
The ability of devices to **exchange** data and **interpret** that shared **data correctly**



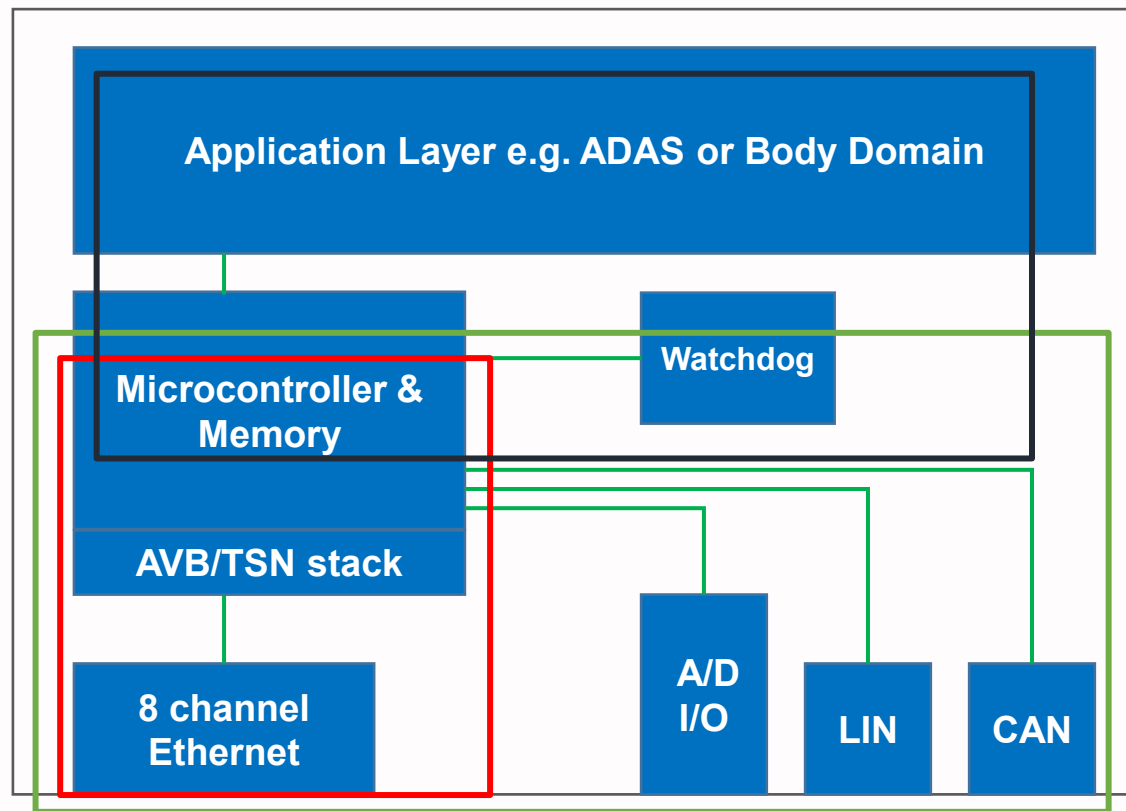
Standards help ensure interoperability

Build it right the first time – V Cycle

- **Develop System Requirements**
 - Left side: requirements, scope, concept, inventions
- **Validate - Unit Test against requirements**
 - Bottom: coding and unit integration
- **Verify requirements**
 - Right side: verification, system integration, calibration
- **Refine and update requirements**
 - Top side: certification, calibration



Gateway/Switch Block Diagram Application Example



Multiple Functional Layer:

1. Ethernet switching
2. Gateway
3. Vehicle related control

Challenge:

Shared usage of μ C & Memory

Store and Forward vs Cut-Through

Types of packet switching modes

Store and Forward

Forwards a packet to the device internal Microprocessor then forwarded on to destination

Pro: Allows packet inspection, packet manipulation, used in Gateway functions, gPTP timing

Con: Consumes more processor utilization and memory, increases latency and adds jitter

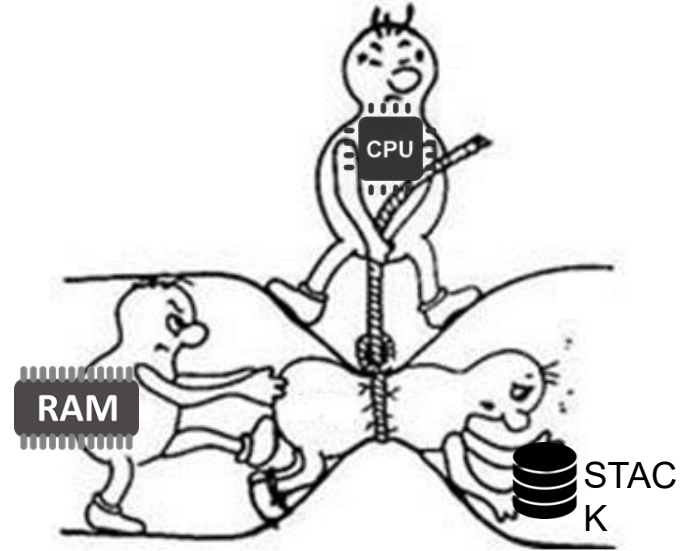
Cut-Through

Involve techniques to begin transmission of frame before the whole frame has been received

Pro: Less memory and processor utilization, lowest latency

Con: No packet inspection

1. All the processing in the world will not solve a RAM or stack issue
2. All the RAM in the world will not solve a stack issue
3. Constantly swapping RAM hinders the best stacks
4. Overtaxed processors hinder the best stacks



Overview of TSN specification

IEEE 802.1AS-Rev - Enhanced Generic Precise Timing Protocol

Adds support for Performance, Redundancy, Aggregation

IEEE 802.1Qci - Per Stream Filtering & Policing

Assigning flows to policer

IEEE 802.1Qbv - Time Aware Shaper

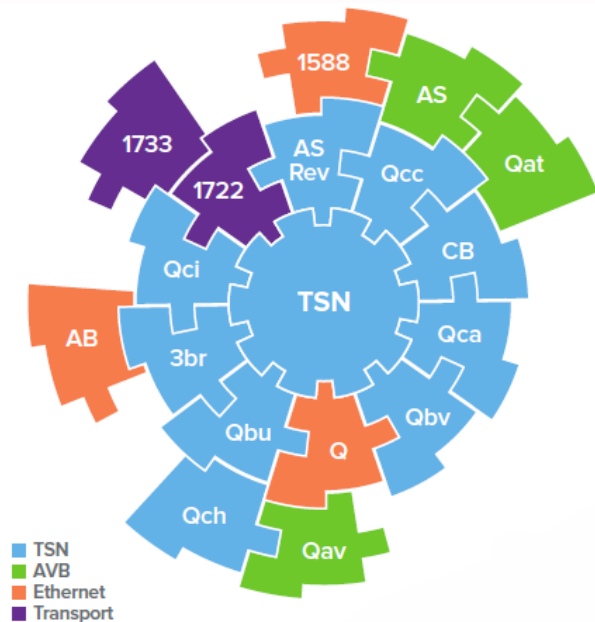
Achieve the theoretical lowest possible latency in engineered networks

IEEE 802.1Qbu & IEEE 802.3br – Packet Pre-emption

Reduce latency of time-sensitive streams in non-engineered networks

IEEE 802.1CB - Frame Replication & Elimination

Support zero switch over time when a link fails or frames are dropped (aka: Seamless Redundancy)



Overview of TSN specification (continue)

IEEE 1722-Rev - Enhance the Stream Transport Protocol

IEEE 1722.1-Rev - **Device Discovery, Connection Control**

Enhance configuration support TSN specifications

IEEE 802.1Qch - **Cyclic Queuing & Forwarding**

Support known latencies, no central controller needed, limits hops

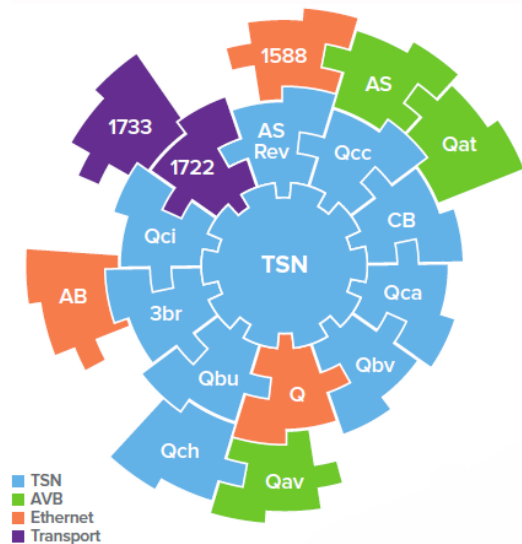
IEEE 802.1Qcr - **Asynchronous Traffic Shaping**

Supports zero congestion loss for asynchronous traffic

Supports deterministic latency without using network topology information

IEEE 802.1Qcc - **Enhanced Stream Reservation Protocol**

Adds support class configurations, shaper and replication



Why should we do Protocol Testing

To ensure safety, security and reliability of a network

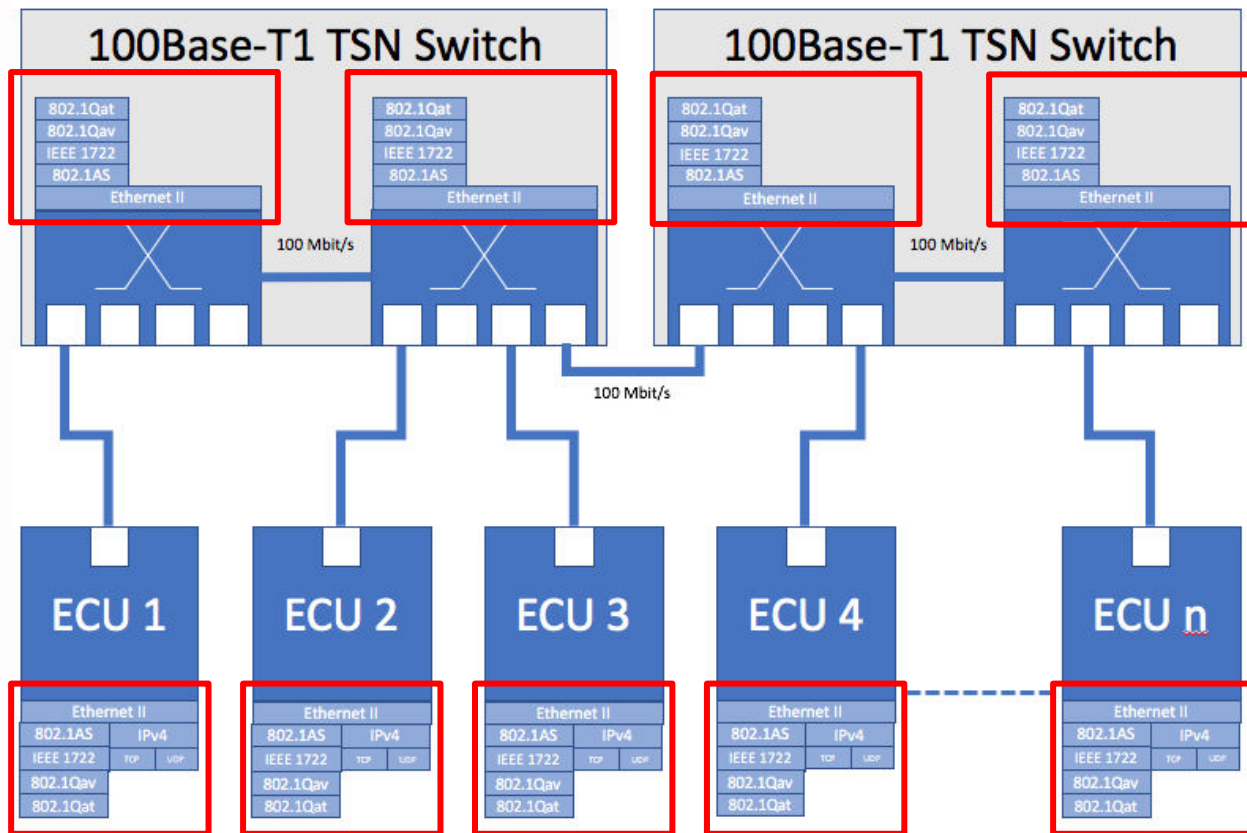
And... to avoid this..

Frustration, delays, time and money..

Brand image and customer quality !



Protocol Conformance testing – Example OEM X



All devices with implemented protocols

In general all end-points

Switches and taps, if specific functionalities are implemented

Can any device be excluded from Conformance Testing?

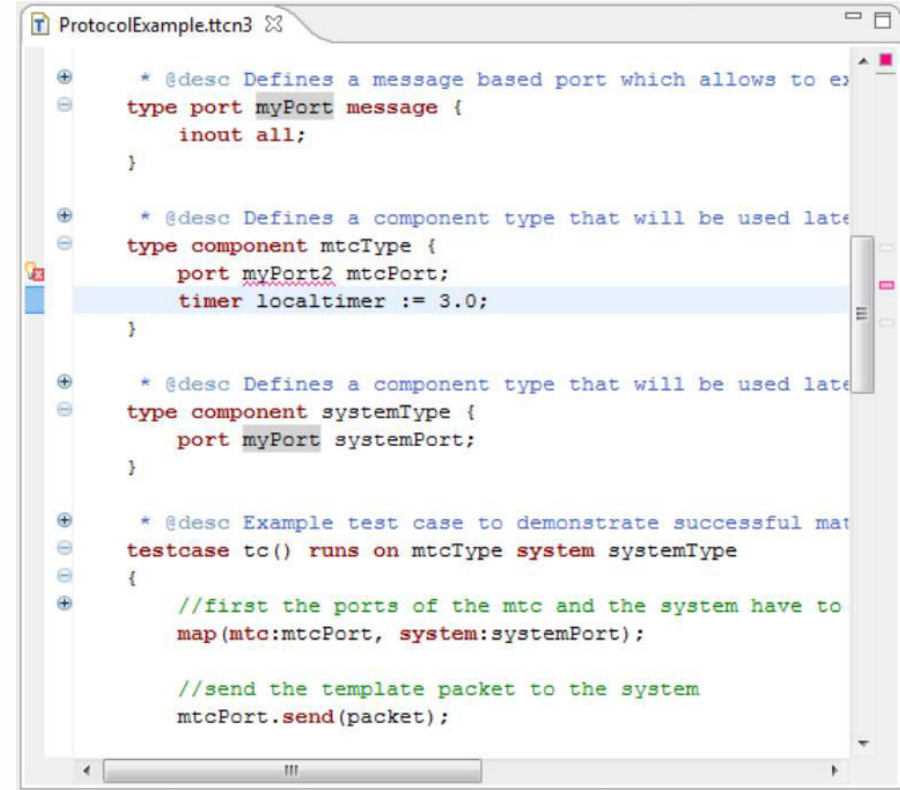
No.. Why?

Promise. Assured.

How to transform manual testing to automated testing

Testing and Test Control Notation programming language v.3 – TTCN-3

- TTCN-3 is an Test Automation Language not a conformance Test Language
- Flexible and non commercial
- Multi vendor tool support
- Sectors: Telecom, Automotive, Medical, Utilities, Financial, Avionics, Railways,
- Accepted and in use with multiple standard organizations:
ETSI / ITU-T / 3GPP / OMA / TCCA / EUROCONTROL-FAA / MOST Cooperation / AUTOSAR / Car2Car CC



```
ProtocolExample.ttcn3

* @desc Defines a message based port which allows to ex
type port myPort message {
    inout all;
}

* @desc Defines a component type that will be used late
type component mtcType {
    port myPort2 mtcPort;
    timer localtime := 3.0;
}

* @desc Defines a component type that will be used late
type component systemType {
    port myPort systemPort;
}

* @desc Example test case to demonstrate successful mat
testcase tc() runs on mtcType system systemType
{
    //first the ports of the mtc and the system have to
    map(mtc:mtcPort, system:systemPort);

    //send the template packet to the system
    mtcPort.send(packet);
}
```



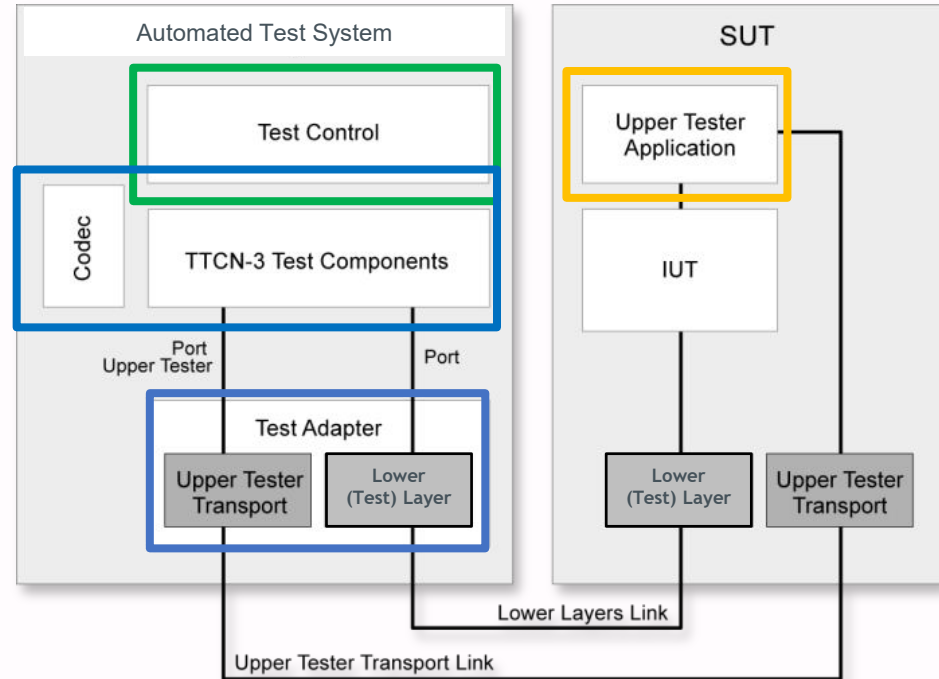
How to simplify the test effort for conformance testing

Improving quality and reliability + reducing cost and time to market

Answer: Fully automated testing instead of manual test case execution

Requirements:

- Automation test framework
- Own test case implementation and/or use of commercial test case suites
- Test Adapter e.g. Physical or Software Interface (Ethernet, CAN, MOST etc.)
- Upper tester integration within the “System under Test”

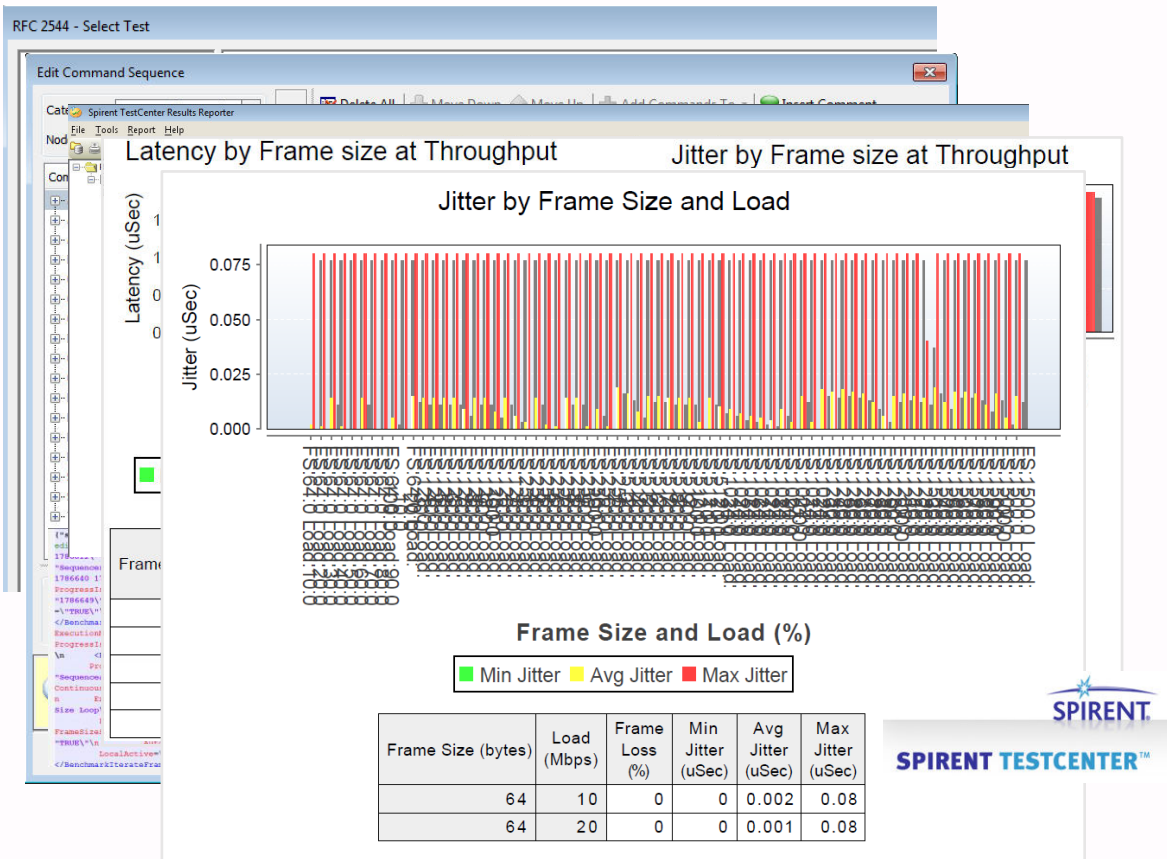


Conformance + Performance

- One chassis enables both conformance and performance at the same time
- Multi-user access up to 16 users per chassis
- Multiple Test suites and plug-in available for Ethernet and V2X technologies
- Build on proven technology platform IDE, eclipse and TTCN-3 automation language



Example – Performance Automatic Benchmarking Reports using RFC test methodology



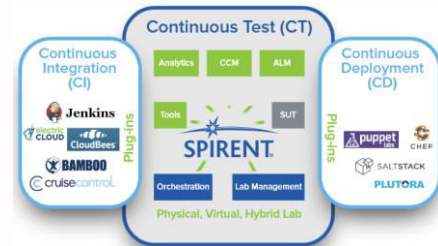
Paris, 16-18 October 2018



Dev/Ops complete solution

iTest and Velocity

Organizer: **TESTING SOLUTIONS & SERVICES**



Test Automation Case Studies

Increase Efficiency

Increased automated test cases from **<10% to 75%** **7X** increase in testing efficiency

Major equipment manufacture streamlined and accelerated release validation for 700% productivity gain

Reduce Costs

65% of test cases automated **\$500K** annual cost **saving**

Major NEM improved time-to-market in spite of increasingly complex testing to reduce costs by 40%

Improve Competitiveness

Cut test execution time from **60** to **6** minutes **Over 10X** speedup in regression testing

Major North American semiconductor company improved release timeliness via test case reuse for 1000% efficiency gain

Improve Quality

Testing time cut **5 hours** to **5 minutes** **98%** increase in test coverage

European service provider NOC developed predictive task automation for 320% capacity increase

Session profiles, APIs & interfaces

All needed session included

CLI

Telnet, SSH, CMD, Serial, Process, File, Syslog, TCL Shell, Python Shell, PowerShell

GUI

Web, Java, Flex, VNC, Selenium/Ranorex, TestPlant

Network Traffic Generator

Spirent (STC, Landslide, Avalanche, CloudStress, Smartbits)
Ixia (IxLoad, IxTrfffic, IxNetwork, N2X)

Enterprise, Virtual & Cloud

VMware, OpenStack, Database, Web Services (RESTful/SOAP/ XM-LRPC)

Protocol & Communications

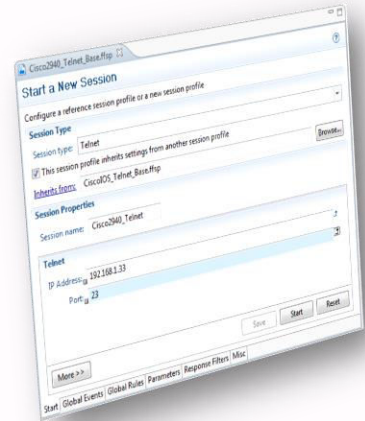
SNMP, NETCONF, HTTP, UDP, XMPP/Chat, SMTP/Email, POP3, Wireshark

Mobile

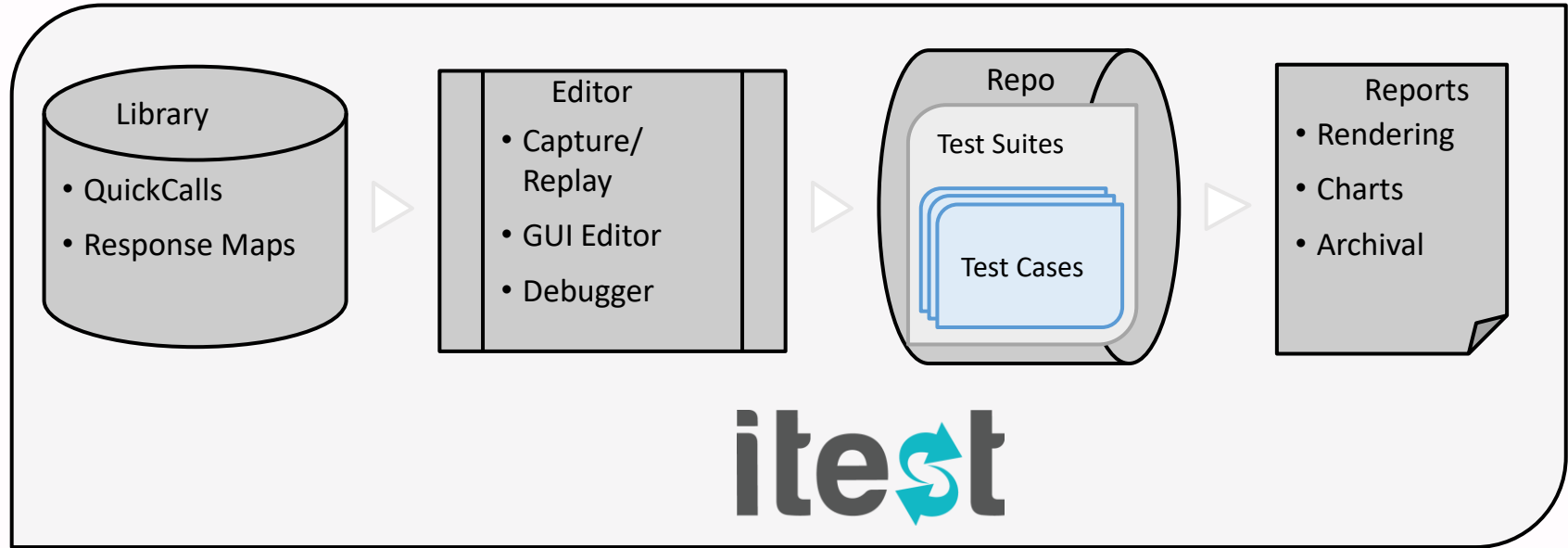
ADB, VNC, Ranorex (APPs), TestPlant

Desktop App

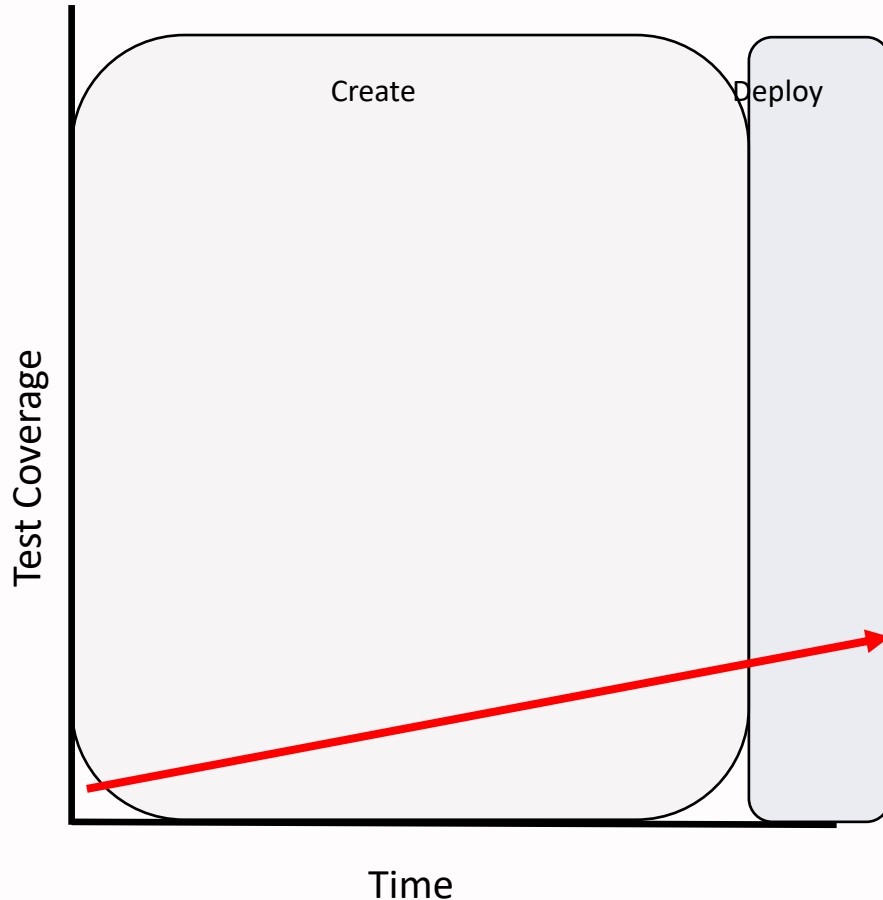
Ranorex (Windows), TestPlant (Any GUI), VNC



iTest: the Complete Network Testing tool

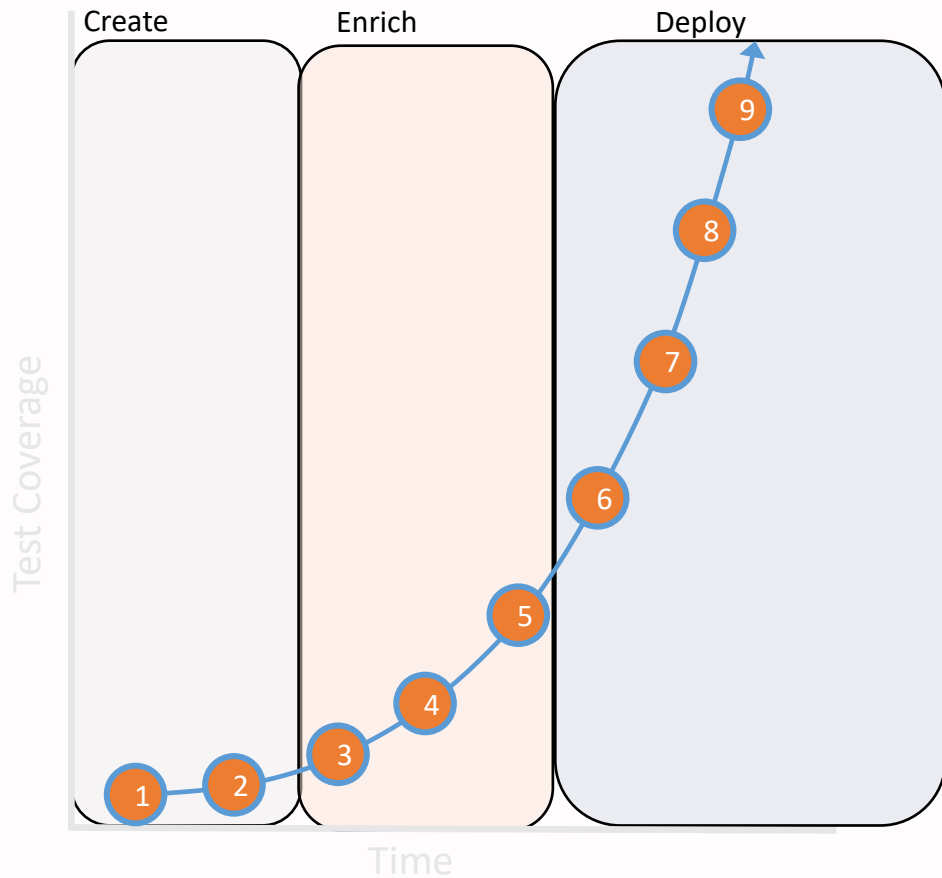


Traditional Test Automation Approach



- Test assets take long time to be built – and have short useful lifespan
- Test repository grows slowly and linearly

New Methodology – Reusable tests



- Test assets built rapidly and reusably – long lifespan

- Test repository grows quadratically

- 9 Validate at scale with distributed Automation Agents
- 8 Enable DevOps Continuous Testing
- 7 Leverage Python frameworks
- 6 Interpret with rich Reporting
- 5 Publish - Share - Consume
- 4 Portabilize via Abstraction
- 3 Extend with Analysis
- 2 Build out Automation with Capture/Replay
- 1 Start with Spirent Developer Community

Start with Spirent Developer Community

Find and Import Existing Test Automation

ADTRAN

JDSU



android



HUAWEI

Calix

JUNIPER
NETWORKS

ciena

NOKIA

CISCO

SPIRENT

CYTEC

vmware

Spirent Developer Community - iTest

Welcome to the Spirent Developer Community! You will find a wide variety of useful automation assets for your iTest projects and/or Velocity deployments. This repository is specific to iTest. The Velocity repository is located [here](#). All Spirent Developer Community assets are available for your immediate download and use.



This portion of the Spirent Developer Community contains a selection of iTest projects. They are useful for gaining the greatest value from your test equipment, devices, and frameworks. The assets are rich in reusable QuickCall libraries and Response Maps. Both are key capabilities that will help you create your own automation most efficiently. They will help you whether or not you are using Python or an Automation framework (e.g., Robot).



All assets in are categorized into three levels of maturity/testing:

- Certified: provided and tested by Spirent
- Reference: provided by Spirent with intent to serve as "blueprint" for your project's structure and usage
- Community: submitted by community or Spirent personnel with no review or testing by Spirent

Technical Benefit

- Reuse existing proven test assets

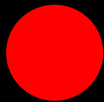
Economic Benefit

- Save time and effort
- Reuse what is available and works

Build out Automation with Capture/Replay

Test Engineers Create Automation Following Their Natural Workflow

CAPTURE



REPLAY



Session ID	Action	Description
Today		
2 Sessions		
s2		ls.rest1.ffsp
s2.1	open	project://iTest7.0.0/session_profile
s2.2	RetrieveTestSession	
s2.3	ConfigureTsGroup	
s2.4	ConfigureTestcaseFavoriteParameters	
s2.5	Run	
s2.6	ShowRunningTestCriteria	
s2.7	StopRunningTest	
s2.8	close	
MME.9		
MME.9.1	open	topology1:MME#ssh
MME.9.2	command	ssh to 10.108.38.84:22
MME.9.3	command	RTRV-NE-ST;
MME.9.4	command	RTRV-NE-ST;ENB;
MME.9.5	command	RTRV-MS-INF:IMSI=310120017639027
MME.9.6	command	exit
	close	ssh to 10.108.38.84:22
Yesterday		
0 Sessions		
Wednesday		
2 Sessions		

- View As Capture Report
- Save As Capture Report
- Delete Selected Sessions Delete
- Collapse All
- Add to Python Script
- Add to iTest Test Case
- Copy as Python
- Replay
- New Form Map
- New Response Map

Technical Benefit

- Capture every action during manual test
- Replay captured steps into automated tests

Economic Benefit

- Save time and money (Domain experts follow natural workflow)
- Increase quality and predictability (automated test cases exactly match test procedure)

Extend with Analysis

Validate with Spirent Patented Response Maps

The screenshot displays the Spirent MME Validation tool interface. The main window shows a test procedure with the following steps:

Action	Session	Description
procedure	main	
1 open	t1	project://Test7.0.0/session_profiles/ls.rest1.ffsp
2 open	MME	device:MME#ssh
3 command	MME	RTRV-NE-STSc;
4 RetrieveTestSession	t1	-library 0 -name mme.nodal.1
5 ConfigureTsGroup	t1	-testHandle \$handle0 -tsGroupIndex 0 -tsId 2
6 ConfigureTestCaseFavoriteParameters	t1	-testHandle \$handle0 -tsGroupIndex 0 -testCaseIndex 0
7 Run	t1	-testHandle \$handle0
8 command	MME	RTRV-NE-STSc:ENB;
9 command	MME	RTRV-MS-INF:IMSI=310120017639024;
10 ShowRunningTestCriteria	t1	-runningTestId \$runId0
11 StopRunningTest	t1	-runningTestId \$runId0
12 close	t1	
13 command	MME	exit
14 close	MME	

The 'Step Properties' panel at the bottom shows the 'ShowRunningTestCriteria' step. The 'Result' section displays the following JSON:

```
{
  criteria : [
    {
      description : "PASS if (ts0::tc0-Test Summary-Sessions Established == 10)"
      id : 0
      failureCount : 0
      status : PENDING
    }
  ]
}
```

A context menu is open over the 'ShowRunningTestCriteria' step, showing options: Copy, Select All, Show most recent response regardless of test report name, Preferences..., Open Response Map, Copy Query, Copy XPath, Add Analysis Rule... (highlighted), and Quick Analysis Rule.

Technical Benefit

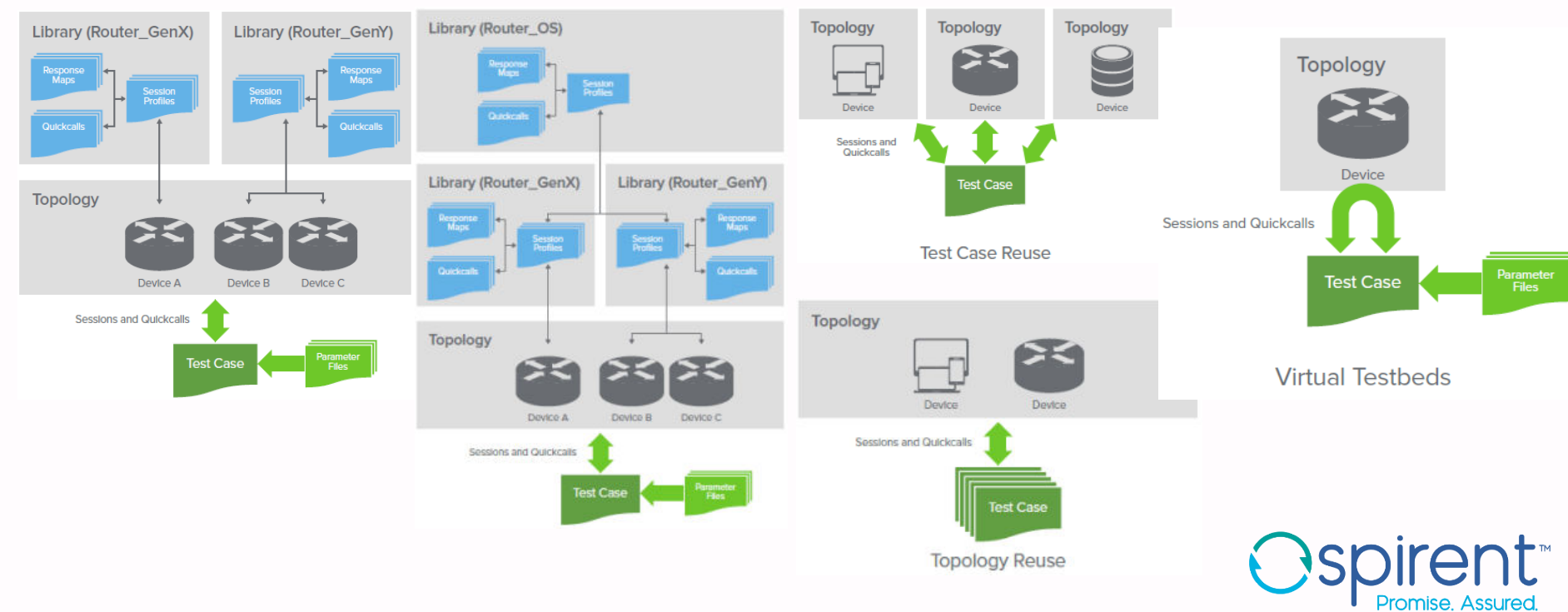
- Extract relevant data from responses for measurements and validation

Economic Benefit

- Save time on low value pattern matching that is better applied to expanding test coverage

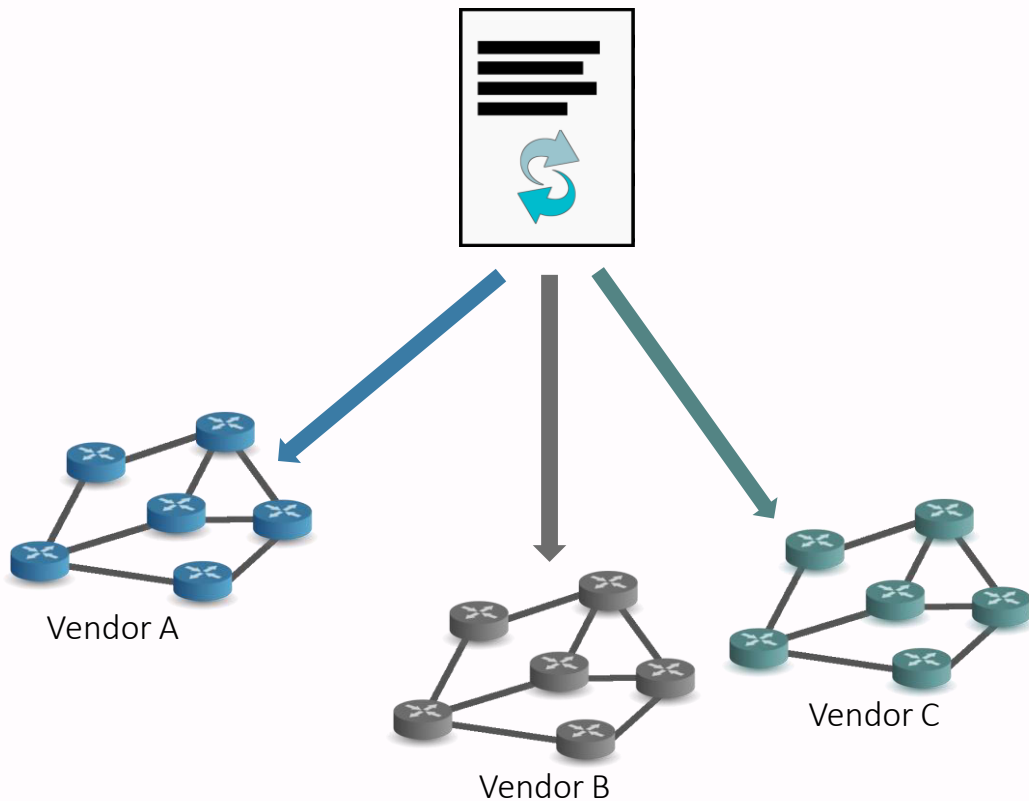
Modular Development Environment

*Creating reusable assets to accelerate test creation
(Libraries, QuickCalls, Custom Sessions)*



Portable via Abstraction

Run the Same Test Cases Against Different Testbeds



Technical Benefit

- Abstract the device specific interface in QuickCalls

Economic Benefit

- Reduce maintenance costs

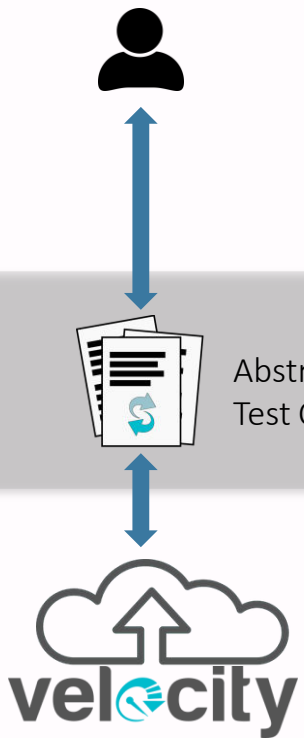
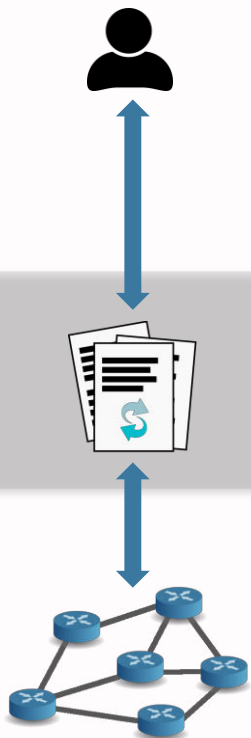
Publish - Share - Consume

Share Test Cases with All Teams

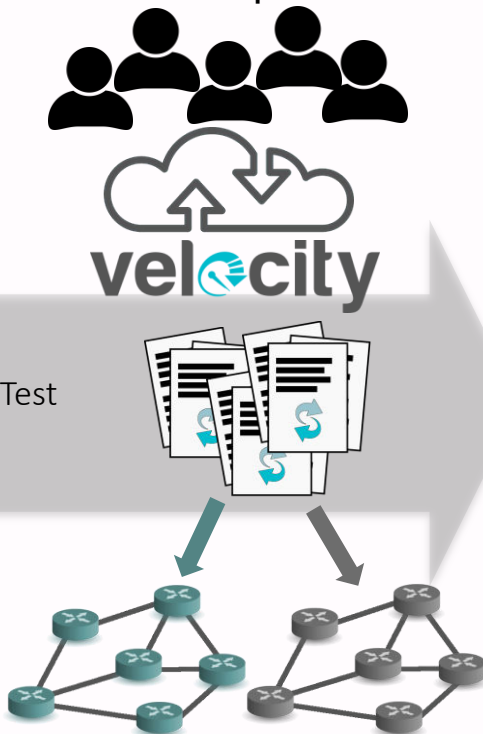
Test Case
Creation

Test Case
Publishing

Test Case
Consumption



Abstract iTest
Test Case



Technical Benefit

- Publish tests for wide consumption both in labs and production

Economic Benefit

- Increase productivity and reduced cost (full team empowered to consume automation from central portal)

Interpret with Rich Reporting

Quickly Find Failure Points Using a Consolidated Test Report

The screenshot displays the Velocity test reporting interface. The top navigation bar includes 'INVENTORY', 'LIBRARY', 'SCHEDULE', 'VIRTUAL', and 'REPORTS'. The main header shows the path 'Execution Reports > mme.validation.1.fttc'. On the left, a sidebar contains a 'FAILED' status indicator (a red circle with an exclamation mark), a search bar, and filters for 'Search By', 'Message Severity', and 'Step Action'. The main content area is titled 'Steps' and shows a table of test steps. The first step, 'Action: getMobileSubscriberDetails', is highlighted with a red '1' in a blue box. Below it, the 'COLLAPSE RESPONSE' section shows a JSON object: `{"imsiStatus": "NOT_CONFIRM"}`. The second step, 'Action: command', is also highlighted with a red '1' in a blue box. Below it, the 'COLLAPSE RESPONSE' section shows a detailed log entry: `[LAB_NRT2_MME1] LAB_NRT2_MME1 2014-05-20 TUE 17:33:55 M6150 RETRIEVE MOBILE SUBSCRIBER INFORMATION - RADIO AND HSS INFORMATION - [1 /1] : COMPLED IMSI = 310120017639024 RESULT = OK IMSI = 310120017639024 MSISDN = 6222316479 IMEI = 9900033711679701 STATUS : NOT_CONFIRM GUTI = MCC(310) MNC(120) MMEG(H'8006) NMEEC(H'21) M-TMSI(H'E8009A D51`. The right sidebar provides additional details for the test asset 'mme.validation.1.fttc', including its location, owner, report detail level, execution start and end times, duration, and host.

Step #	Action	Start Time	Duration
4	Action: getMobileSubscriberDetails Command: -imsi 310120017639024	00:00:00.814	00:00:00.284
4.1	Action: command Command: RTRV-MS-INF:IMSI=310120017639024;	00:00:00.817	00:00:00.267

Technical Benefit

- Leverage detailed reports to perform root cause analysis
- Capture every single test step in detailed, unified customizable report

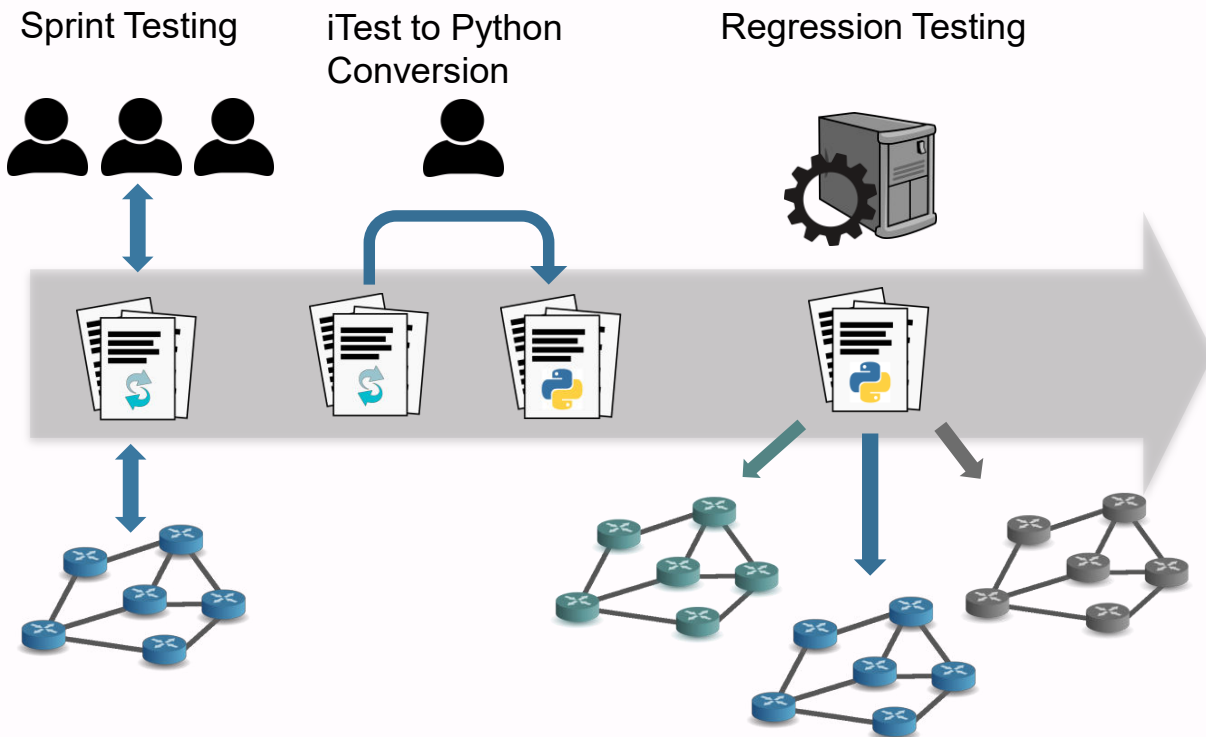
Economic Benefit

- Save time in determining root cause of failures
- Increase quality and customer satisfaction

Integrate with Python frameworks

Include Python code and libraries natively into iTest Test cases

Convert iTest to Python for Automated Regression Testing



Technical Benefit

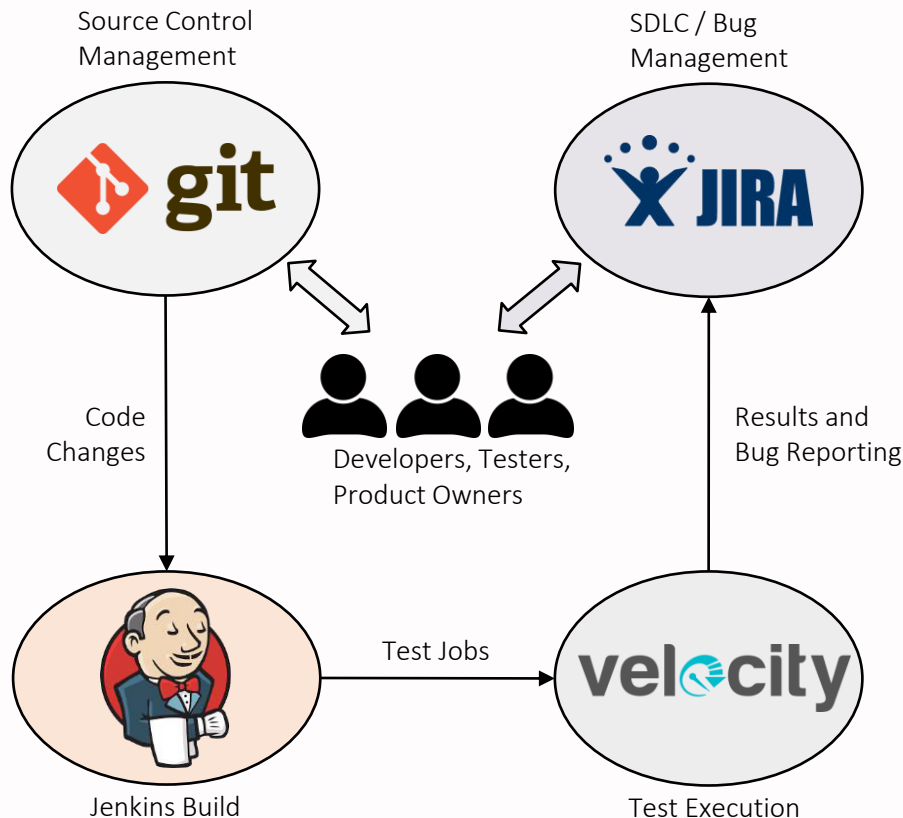
- Use existing Python code in iTest
- Turbocharge Python code with iTest's powerful QuickCalls and Response Maps
- Publish automation to Python test regression systems

Economic Benefit

- Save time and money
- Increase developer productivity

Enable DevOps Continuous Testing

DevOps Tool Chain Integrations Enable Continuous Test and Immediate Feedback



Technical Benefit

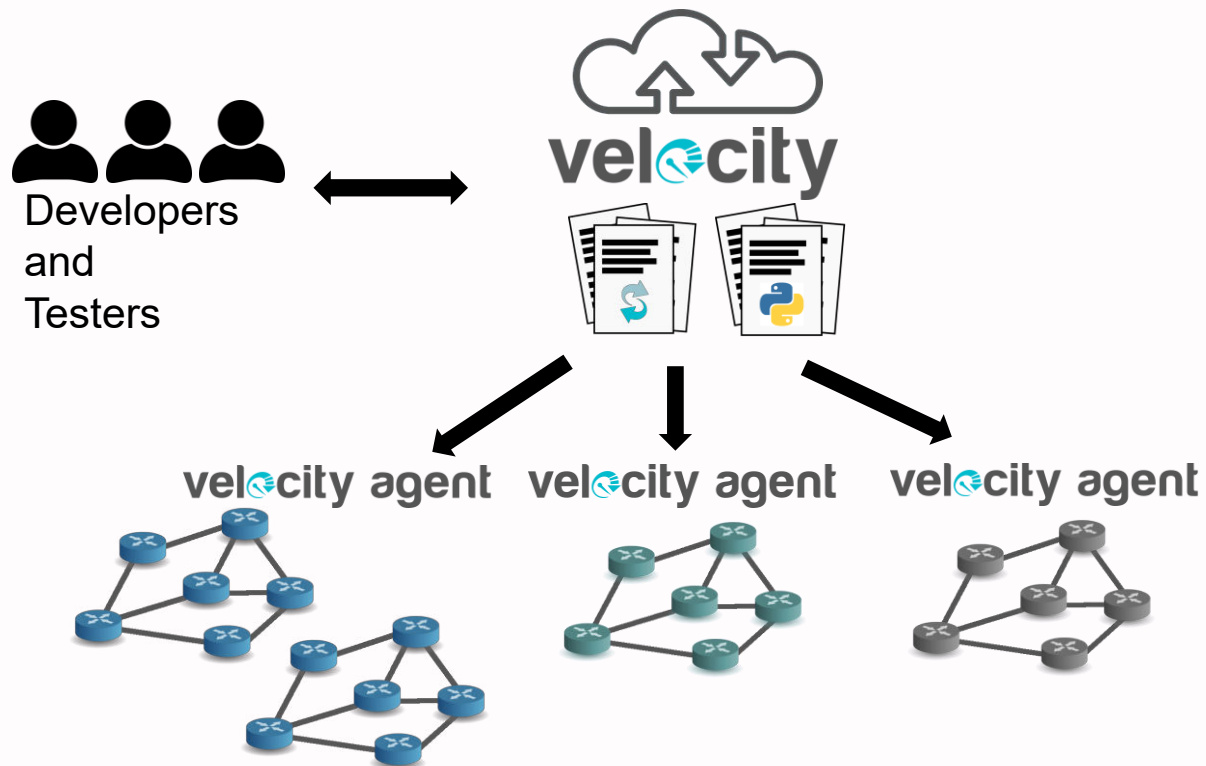
- Integrates with DevOps tool chain applications
- Identify defects *Immediately after* software builds

Economic Benefit

- Increase quality and customer satisfaction

Validate at Scale with Distributed Agents

Deploy Test Agents Near Devices Under Test



Technical Benefit

- Execute at scale with intelligent, high performance, distributed automation agents
- Parallelize test execution locations

Economic Benefit

- Accelerate time to market
- Increase product/service quality

Paris, 16-18 October 2018



Organizer:



Questions & Answer

Thank you..