Open for All.

OCP
GLOBAL
SUMMIT

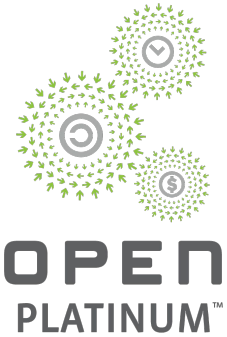
OSF

Open Source Firmware Testing at Facebook

If you don't test your firmware, your firmware fails you

Andrea Barberio (barberio@fb.com)
Production Engineer, Facebook

Marco Guerri, (marcoguerri@fb.com)
Production Engineer, Facebook



Agenda

OPEN SYSTEM FIRMWARE

- Problem statement
- Requirements
- Solution
- Architecture

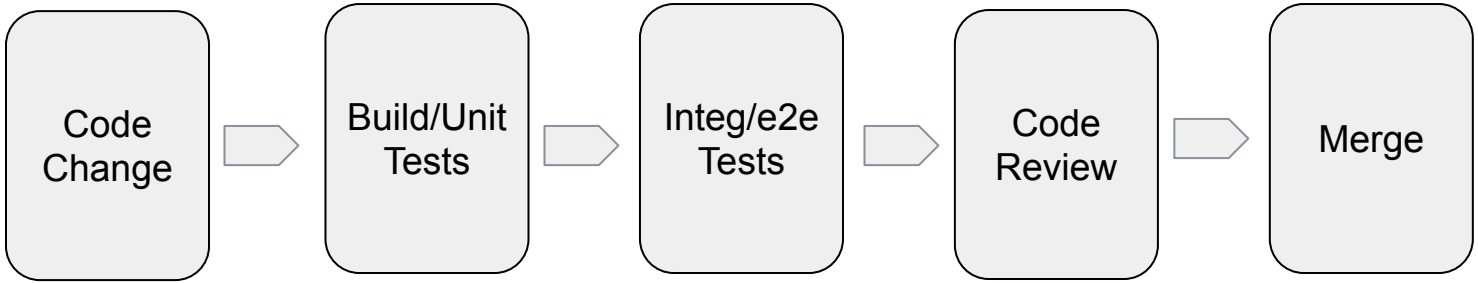
Problem statement

OPEN SYSTEM FIRMWARE

- We run OSF in production[1]
- Development happens upstream (GitHub, Gerrit)
- Process:
  - develop
  - build
  - **integration and end-to-end tests**
  - review
  - release
  - debug

[1] https://engineering.fb.com/data-center-engineering/f16-minipack/

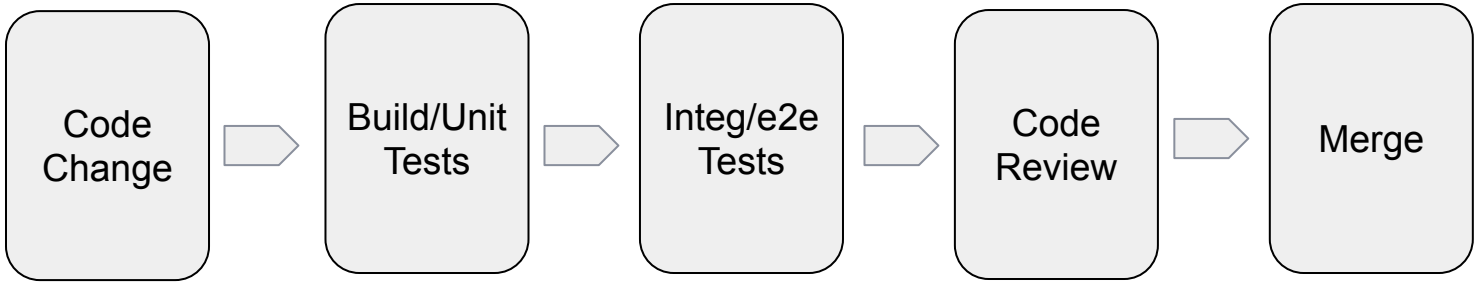
Development timeline

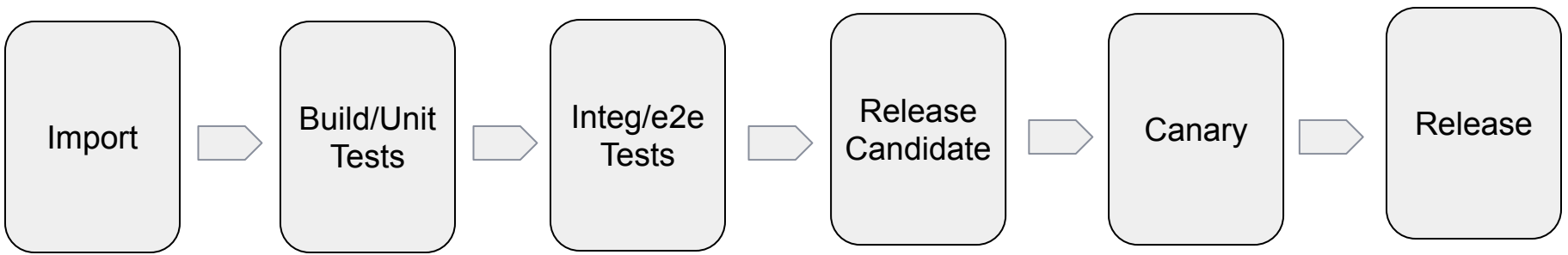


OPEN SYSTEM FIRMWARE

Development timeline

OPEN SYSTEM FIRMWARE





Why testing firmware?

OPEN SYSTEM FIRMWARE

- Pretty obvious with software. But firmware?
- Bugs can brick many devices. Reduced capacity
- Rolling out firmware takes longer than software
- Firmware influences the machine’s behaviour and performances

Requirements (1/2)

OPEN SYSTEM FIRMWARE

We want a firmware testing system that is

- **Robust:** minimize failures in prod, detect errors early
- **Generic:** can work in any infrastructure
- **Scalable:** can run at datacenter scale
- **Simple by design:** easier to reason with, and to understand
- **Flexible:** assembled from independent components

Requirements (2/2)

OPEN SYSTEM FIRMWARE

- **Easy to set up and maintain:** single binary, simple DB
- **Easy to use:** configuration, not code
- **Open-source first:** together is better!
- **Working with OSF,** but not limited to them

What about existing systems?

OPEN SYSTEM FIRMWARE

- We looked at several existing systems
- Difficult to meet all the requirements. Mainly:
  - hard to set up
  - hard to maintain
  - complex to use
  - DUT-only test cases
  - too scoped functionalities

Enter ConTest

OPEN SYSTEM FIRMWARE

- **Con**tinuous and on-demand integration and e2e **Test**ing
- Single binary plus SQL database
- Written in pure Go for ease and memory safety
- Can do more than firmware testing
- https://github.com/facebookincubator/contest

Job Descriptor

OPEN SYSTEM FIRMWARE

```
{
   “JobName”: “My test job”,
   “Runs”: 3,
   “Tags”: [“firmware”, “ocp”, “minipack”],
   “TestDescriptors”: {
        …
   },
   “Reporting”: {
        …
   }
}
```

Test descriptors

OPEN SYSTEM FIRMWARE

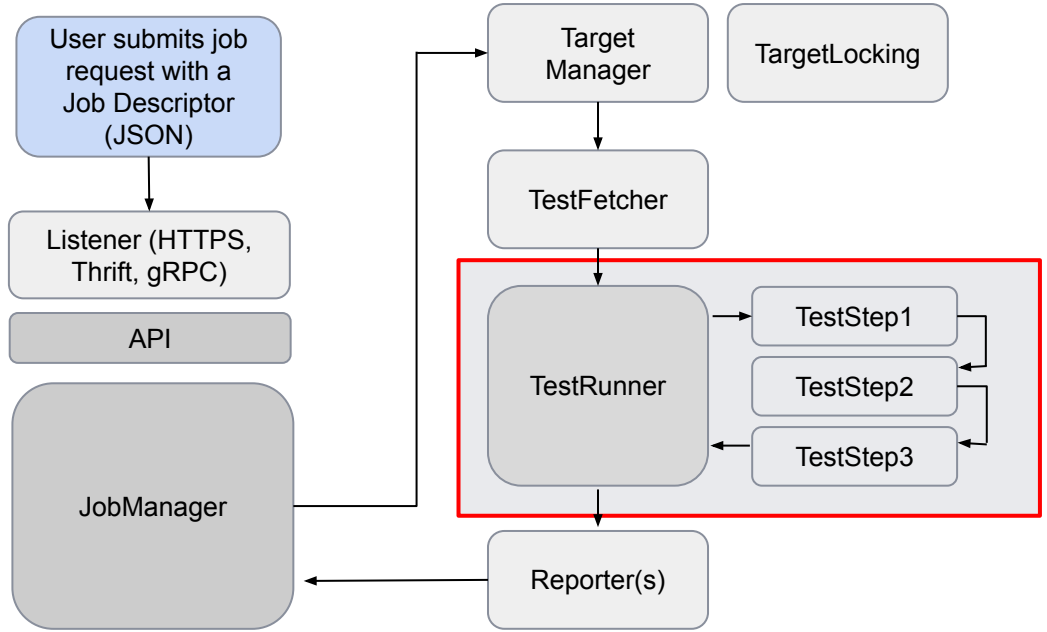
```
"TargetManagerName": "URI",
"TargetManagerAcquireParameters": {
   "URI": "https://example.org/targetmanagers/my-test.json",
},
"TestFetcherName": "literal",
"TestFetcherParameters": {
   "Steps": [
      {"Name": "sshcmd", "host": "jump.example.org", "executable": "ls"},
      {"Name": "sshcmd", "host": "jump.example.org", "executable": "flashrom"},
   ]
}
```

Reporting

OPEN SYSTEM FIRMWARE

```
"Reporting": {
  "RunReporters": {
    {
      "Name": "TargetSuccess", "Parameters": {"SuccessExpression": ">=95%" },
    }
  },
  "FinalReporters": {
    {
      "Name": "AverageTime", "Parameters": { },
      "Name": "Outliers", "Parameters": {"start": "RebootStart", "end": "RebootEnd"},
    }
  }
}
```

Architecture - Overview

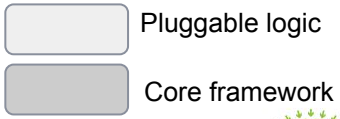


ConTest instance acquires ownership of targets

Fetch a description of the test steps and associated parameters.

Based on the description of the test, a pipeline is setup. The TestRunner orchestrates the flow of Targets through the various steps.

A reporter(s) are invoked to generate custom description(s) of the outcome of the test.



OPEN SYSTEM FIRMWARE

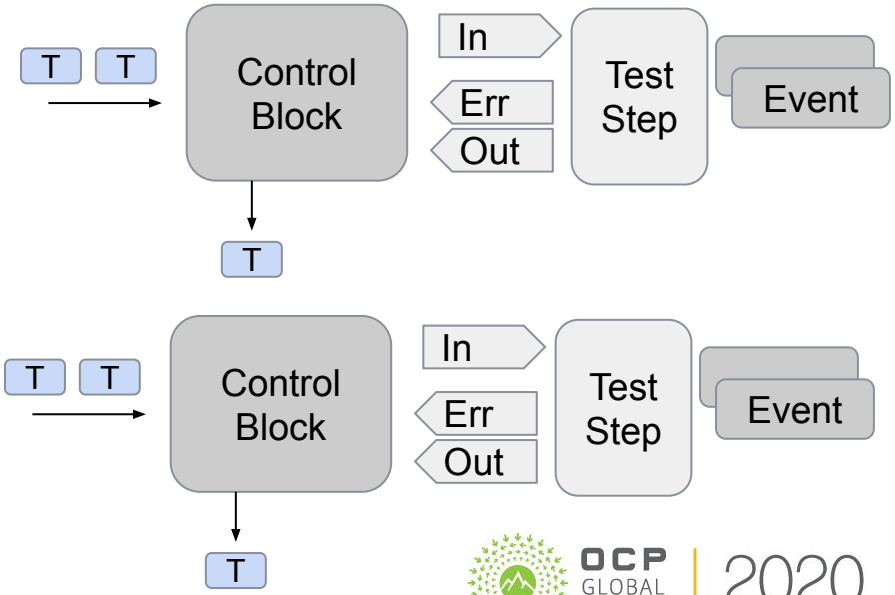
Architecture - Test Runner

OPEN SYSTEM FIRMWARE

The TestRunner controls the flow of Targets through the TestSteps.

A ControlBlock is associated to each TestStep to monitor the behavior of the plugin:

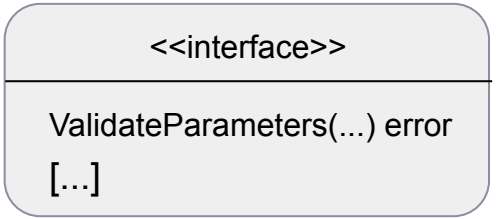
- Records success or failure of a Target via out and err channels
- Records Targets ingress and egress timestamps
- Enforces that targets fed to the TestStep must be returned in output
- Enforces that targets fed in input must be accepted with a timeout



Interfaces and plugins

OPEN SYSTEM FIRMWARE

- Plugins must implement interfaces and meet requirements for I/O on channels, return values, timeouts, etc.
  - ConTest enforces that a job is terminated when a plugin does not comply with the requirements
- Interfaces are designed to allow for early validation of parameters
- Components are easily swappable, integration tests can use custom components that validate the logic of the framework



Call to Action

- **Get involved**! https://github.com/facebookincubator/contest
- **Try it** in your own infrastructure, or even at home
- Help us set up a public testing infrastructure
- Report bugs, implement new plugins, or suggest improvements

Open System Firmware:

- https://www.opencompute.org/projects/open-system-firmware
- https://ocp-all.groups.io/g/OCP-OSF

Contact us:

- Andrea Barberio <barberio@fb.com>
- Marco Guerri <marcoguerri@fb.com>

Open for All.

OCP GLOBAL SUMMIT

MARCH 4 & 5, 2020 | SAN JOSE, CA