Fuego Test System

Roadmap, Priorities and Jamboree Wrap-up

July 2019

Tim Bird

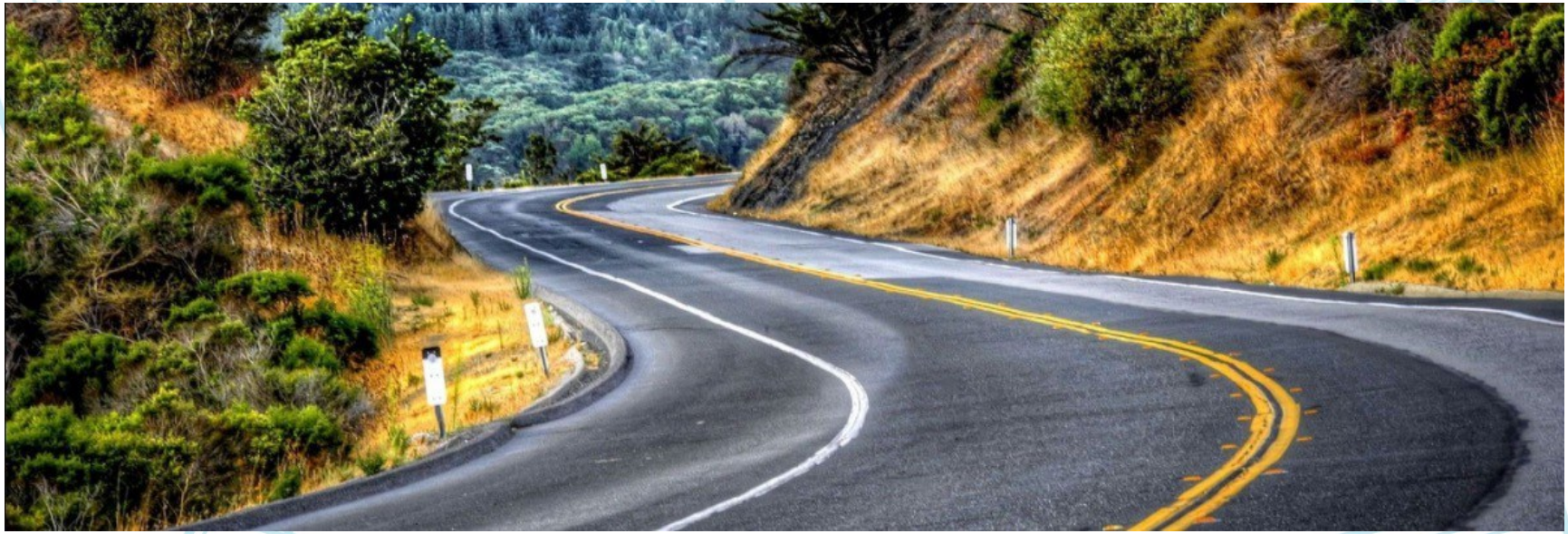
Fuego Maintainer

Sony Electronics

Outline

- Old roadmap
- What to work on next?
- Projects in progress
  - Tim's working on...
  - Daniel's working on...
  - Liu's working on...
  - Wang's working on...
- Discussion

Roadmap



Recent past → Near Future → Long Term

Roadmap

- Recent past:
  - Batch tests
  - Fuego distro, Jenkins upgrades
- Near future:
  - Provisioning support
  - Monitors
  - Test and test improvements
  - Continuing integration with other systems

Provisioning support

- Install of software under test
  - e.g. AGL image deploy, LTSI kernel update, etc.
  - Has been out-of-scope for Fuego
  - Now time to add it
  - Still interested in support for integration with external systems (LAVA)
- Set of batch tests to implement provisioning steps
- Requires board management API

Monitors

- A monitor is a separate “test” that gathers information from some place, while the test is running
  - Has start, stop, gather-data phases
  - Can retrieve data from:
    - External machine source (like power meter)
    - On-board source (like ftrace)
- Intent is to allow test to have pass criteria based on monitor data
  - e.g. power test result is based on monitor data, not test log
- Can be used for stress testing
- The batch test framework is designed with Monitors in mind
  - Is a work in progress to support asynchronous test execution (next page)

Sub-test concurrency

- API:
  - start_parallel <group-name>, end_parallel - declare a block of commands for concurrent execution
  - stop_parallel <group-name> - stop concurrent tasks
    - Issues a signal to each task in the group to stop and record their data
  - wait_for_parallel <group-name> - wait for all concurrent tasks in a group to complete

Concurrency example1 - monitors

Monitor power, kernel_log and ftrace, while running cyclictest

```
def test_run() {
        start_parallel my_monitors
                start_job run_test Monitor.power
                start_job run_test Monitor.kernel_log
                start_job run_test Monitor.ftrace –s function_latency
        end_parallel

        run_test Benchmark.cyclictest

        stop_parallel my_monitors
}
```

Concurrency example2 – multi-tests

Start two tests and wait for them both to complete

```
def test_run() {
        start_parallel some_tests
                start_job run_test Functional.linux_stress
                start_job run_test Functional.LTP
        end_parallel

        wait_for_parallel some_tests
}
```

Concurrency example3 - stressors

Start some stressors in the background, then run a test

```
def test_run() {
        start_parallel stress1
                start_job run_test Stress.filesystem
                start_job run_test Stress.network
        end_parallel

        TESTCASE_NAME="realtime_stress1"
        run_test Benchmark.cyclictest

        stop_parallel stress1

        start_parallel stress2
                loop_job run_test Functional.iozone –s stress
                loop_job run_test Functional.dbench4
        end_parallel

        TESTCASE_NAME="realtime_stress2"
        run_test $NODE_NAME Benchmark.cyclictest

        stop_parallel stress2
}
```

Target package cache

- Target package = archive of test binary, script and data, ready to be deployed on the target board
- Can build a cache of target packages
- Added new phase ‘makepkg’ (letter ‘m’)
  - Simulates a ‘deploy’, but puts output into a tar file, for later transfer to board
- Have a place designated for target packages
  - /fuego-rw/cache
- See fuegotest.org/wiki/Target_Packages

Anticipated uses

- Speed:
  - If test source doesn't change, can just deploy the cached binary package
- Build pre-test:
  - Can build all test packages ahead of time
  - Check dependencies ahead of time
  - Already have a script
    - fuego-core/scripts/make_cache.sh
- Test without toolchain or test source
  - Put target packages on fserver
  - host or target uses downloads target packages, rather than build

Tests

- Fujitsu's command/service tests
- Sony's Linux distribution tests
- LTP visualization improvements
- kselftest improvements

Integration with other systems

- Test sharing
  - Integration with a central test repository
- Results sharing
  - Integration with different results back ends
- Board management API
  - Board power control is just the start
  - Need APIs for button pushes, serial control of bootloader, sdcard muxing, etc.
- All this is part of test standards work also

Roadmap (cont.)

- Long-term
  - Build artifacts
    - Utilize external artifact server
  - Test store (fserver)
  - Distributed test network
  - Hardware testing

Deferred or partially completed

- ftc set-criteria
  - Easy customization of pass criteria from command line
- Target test package binary cache
  - Limited form of build artifact cache
  - Could help with LAVA integration
- Documentation in reStructured text
- Tutorials

What to work on next?

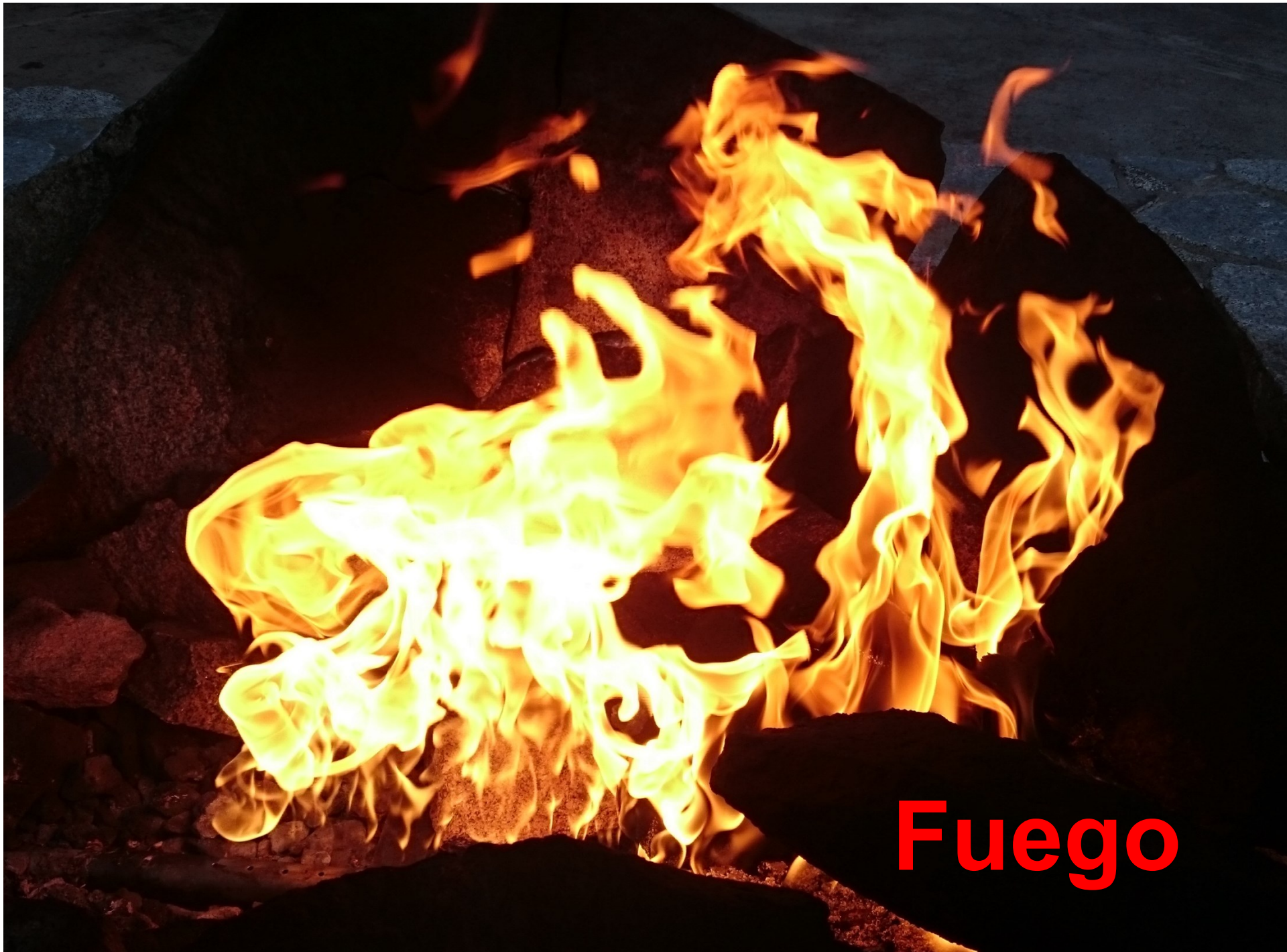
- Fuego 1.5 release
  - Must fix Jenkins plugins issue
- Fuego 1.6 priorities
  - Fujitsu tests
  - Issue tracker
    - Idea: can we switch to gitlab?
      - Would require putting test repository (tarballs) somewhere else (maybe fuegotest.org?)
      - Combine fuego-core into Fuego git repository?

Daniel's projects

- Daniel will be on leave with a new child for several months
  - No Fuego projects planned for the near term

Discussion

- (use the wiki page to record information)
- http://fuegotest.org/Fuego_Jamboree_3
- Some ideas during the code review and brainstorming:
  - Maybe submit patches as plain text, as well as in attachment (to make it easier to apply)
  - Maybe test patches using fserver? virtual machine?
- Should make a ‘fujitsu’ parser
  - Add to common.py, and specify in fuego_test.sh?
  - Autodetect format?


Fuego

Other Priorities

- LAVA integration
  - We have everything needed for transport integration
  - Need test-level integration
    - Separate build phase (done)
    - Deploy to LAVA server
    - Create LAVA test that does:
      - Execute test on board
      - Collect results