



Systems Testing

Michael Starch
NASA Jet Propulsion Laboratory
October 23, 2024



Copyright © 2024 California Institute of Technology.
Government sponsorship acknowledged.



Jet Propulsion Laboratory
California Institute of Technology



Overview

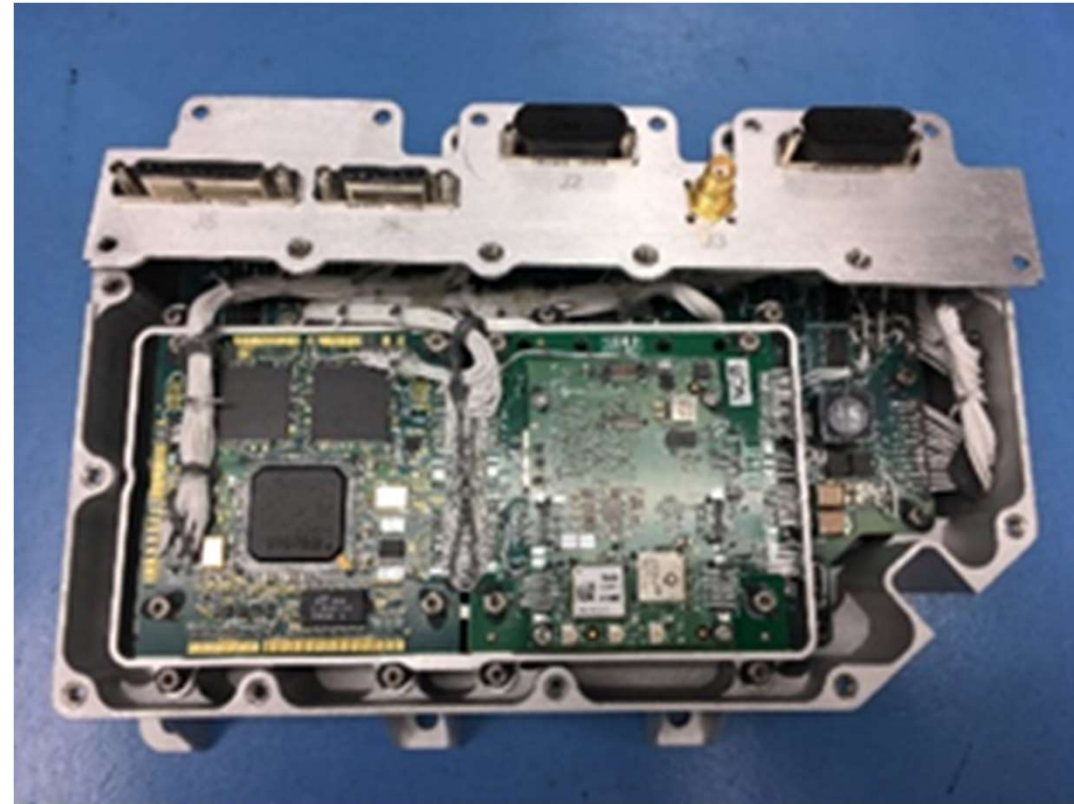
- Introduction to Systems Testing
 - What is system testing?
 - Why do we do system testing?
 - System testing tools
- Types of System Testing
 - Manual systems testing
 - Automatic system testing
 - Continuous integration
- System Testing with F'



Introduction to System Testing

Systems Testing

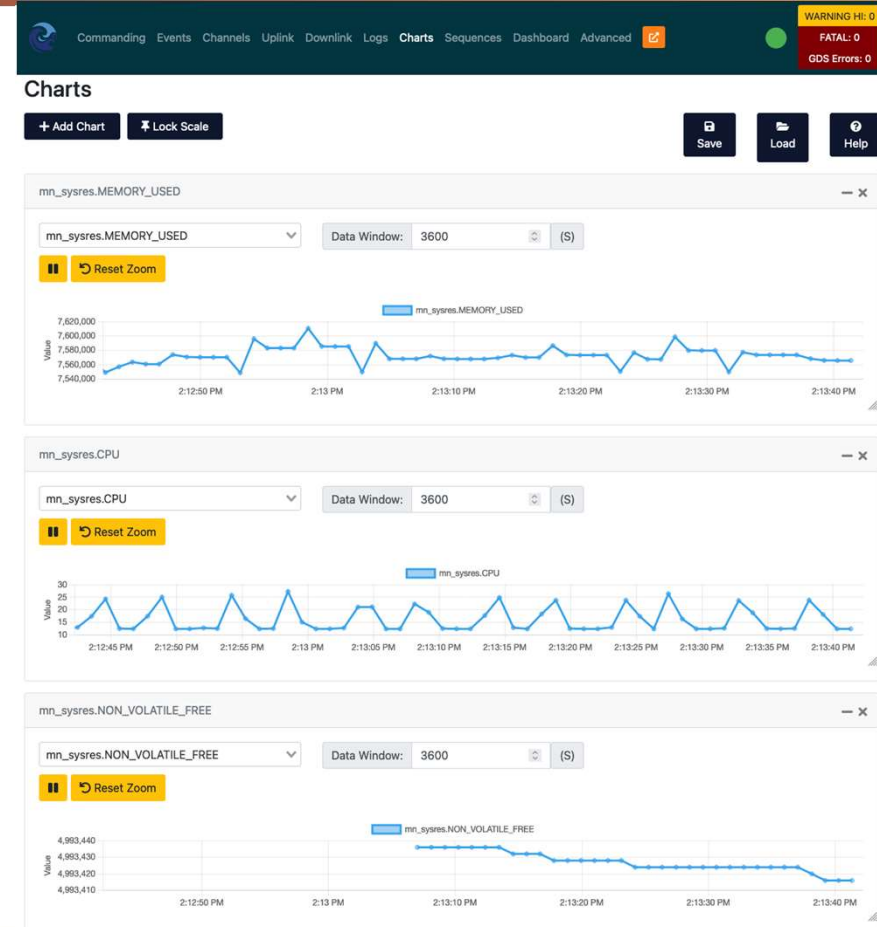
- Testing software within the full running system
- Includes flight-like hardware and ground support
- Also called *integration testing* and similar to *regression testing*
- “Test as you fly, fly as you test”



JPL/Caltech

Why Systems Testing?

- A system is more than the sum of its parts
- Some requirements can only be verified at system level
- Hardware operations need testing on hardware
- “Eat our own dog food”



Thundering Herd



Public Domain



System Testing Tools

- Ground Data Systems
 - Operate spacecraft from earth
 - May be used to conduct testing
- Testing Frameworks
 - Allow for automated testing
 - Reporting, verification, reproducibility
- Hardware Simulations
 - Allows system testing without physical hardware
- Command Sequences
 - How the flight spacecraft is operated
 - May be used to construct tests

Commanding Events Channels Uplink Downlink Logs Charts Sequences Dashboard Advanced

WARNING HI: 0
FATAL: 0
GDS Errors: 0

Command Sequencer

sequence_1.seq ☒ Uplink Build Save As Open

Command Builder

Mnemonic: seq_Primary.CS_RUN

Description: Run a command sequence file

Arguments

fileName: The name of the sequence file /seq/cause_woes.bin ☒ block: Return command status when complete or not SEQ_BLOCK

Clear Arguments

Command String

seq_Primary.CS_RUN /seq/cause_woes.bin SEQ_BLOCK

```
1 ; Entry point to the sequence announcing presence
2 R00:00:00 mn_cmdDisp.CMD_NO_OP
3 R00:00:00 mn_cmdDisp.CMD_NO_OP_STRING "<h1>Starting Sequence</h1>"
4
5 ; Setup fault handling
6 R00:00:00 mn_faultProtection.SET_FAULT_RESPONSE CRITICAL_FAULT "/seq/fix_my_woes.bin"
7
8 ; Run a subsequence
9 R00:00:00 seq_Primary.CS_RUN "/seq/cause_woes.bin" SEQ_NO_BLOCK
10 R00:00:00 mn_cmdDisp.CMD_NO_OP_STRING "Waiting for many seconds"
11 R00:00:30 mn_cmdDisp.CMD_NO_OP
12
13 ; Join on the running sequence
14 R00:00:00 seq_Primary.CS_JOIN_WAIT
15 R00:00:00 mn_faultProtection.CLEAR_FAULT_RESPONSE CRITICAL_FAULT
16 R
17
18
```

Parsing error: Parsing error: Line 16: Each line must contain a minimum of two fields, time and command mnemonic

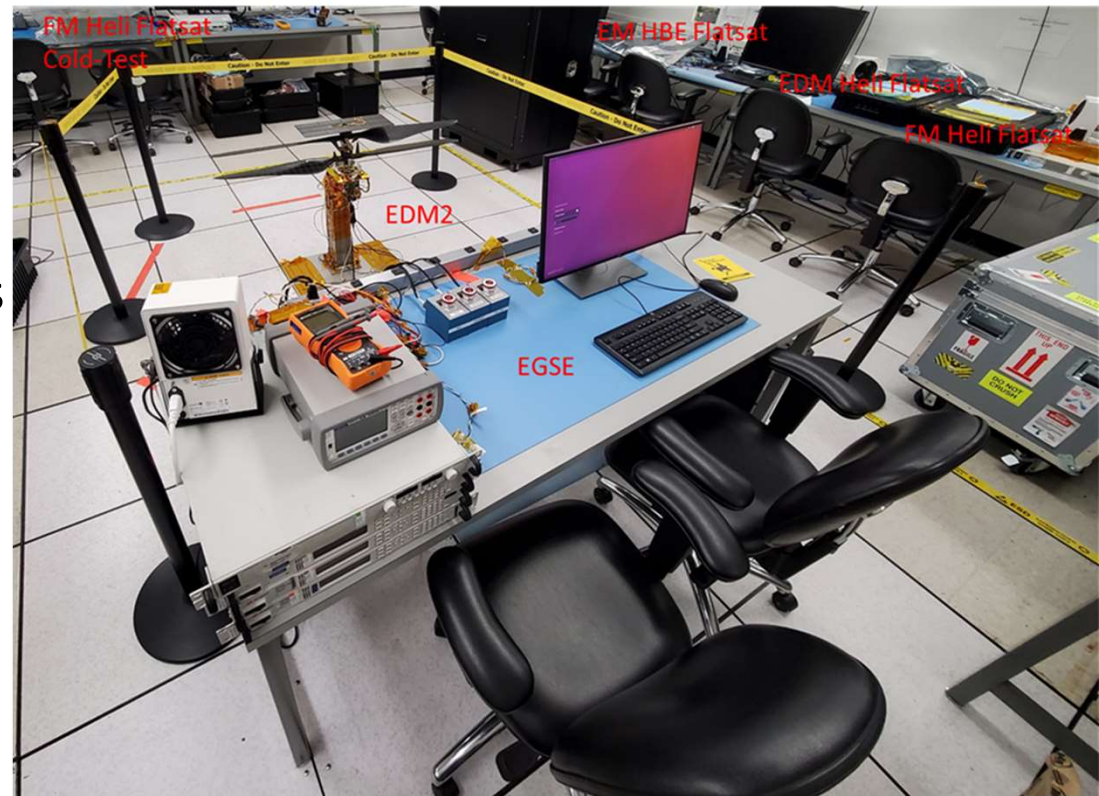
Sequence compilation output



Types of System Testing

Manual Systems Testing

- Operating the spacecraft by-hand to verify requirements
- Running sequences and commands and verifying results
- Ground system is used to send commands
- More complete tests often require full test team



JPL/Caltech



Automated System Testing

- System testing performed by software
- Uses automated testing support from ground station
- Automatically logs data, results, and verification products
- May not require dedicated test team

```
===== test session starts =====
platform darwin -- Python 3.10.5, pytest-6.2.5, py-1.11.0, pluggy-1.0.0 -- /Library/Frameworks/Python.framework/Versions/3.10/bin/python3.10
cachedir: .pytest_cache
rootdir: /Users/mstarch/code/fprime-infra/fprime-sw
collected 0 items

test/int/ref_integration_test.py::TestRefAppClass::test_is_streaming ERROR [ 12%]
test/int/ref_integration_test.py::TestRefAppClass::test_send_command ERROR [ 25%]
test/int/ref_integration_test.py::TestRefAppClass::test_send_command_args ERROR [ 37%]
test/int/ref_integration_test.py::TestRefAppClass::test_send_and_assert_no_op ERROR [ 50%]
test/int/ref_integration_test.py::TestRefAppClass::test_bd_cycles_ascending ERROR [ 62%]
test/int/ref_integration_test.py::TestRefAppClass::test_active_logger_filter ERROR [ 75%]
test/int/ref_integration_test.py::TestRefAppClass::test_signal_generation ERROR [ 87%]
test/int/ref_integration_test.py::TestRefAppClass::test_seqgen ERROR [100%]

===== ERRORS =====
ERROR at setup of TestRefAppClass.test_is_streaming

self = <fprime_gds.common.transport.ThreadedTCPSocketClient object at 0x1067154b0>, connection_uri = '127.0.0.1:50050', incoming_routing = <RoutingTag.GUI: b'GUI'>
outgoing_routing = <RoutingTag.FSW: b'FSW'>

    def connect(
        self,
        connection_uri,
        incoming_routing=RoutingTag.GUI,
        outgoing_routing=RoutingTag.FSW,
    ):
        """Connect to host at given port and start the threaded recv method

        Connects to a running ThreadedTCPServer at the given connection_uri expected in the format tcp://host:port or
        simply host:port. This will also register with the ThreadedTCPServer using the incoming

        Arguments:
            connection_uri (string) -- connection uri of the form (tcp://)host:port
            incoming_routing (RoutingTag): routing tag for incoming data, used to register client to server
            outgoing_routing (RoutingTag): routing tag applied to outgoing data when sent
        """
        try:
            self.dest = outgoing_routing.value
            connection_uri = connection_uri.split("://", 1)[-1] # Remove leading tcp
            host, port = connection_uri.split(":", 1)
            port = int(port)
        except:
            self.sock.connect((host, port))
        ConnectionRefusedError: [Errno 61] Connection refused

/Library/Frameworks/Python.framework/Versions/3.10/lib/python3.10/site-packages/fprime_gds/common/transport.py:1213: ConnectionRefusedError

During handling of the above exception, another exception occurred:

cls = <class 'ref_integration_test.TestRefAppClass'>

    @classmethod
    def setup_class(cls):
        try:
            cls.pipeline = StandardPipeline()
            config = ConfigManager()
            filename = os.path.dirname(__file__)
            path = os.path.join(
                filename,
                "../../build-artifacts/{}/dict/RefTopologyAppDictionary.xml".format(
                    platform.system()
                ),
            )
            logpath = os.path.join(filename, ".logs")
            cls.pipeline.setup(config, path, "/tmp")
            cls.api = IntegrationTestAPI(cls.pipeline, logpath)
            cls.case_list = [] # TODO find a better way to do this.
            cls.dictionary = path
            cls.pipeline.connect("tcp://127.0.0.1:50050")
        except:
            pass


test/int/ref_integration_test.py:32:
-----
self = <fprime_gds.common.transport.ThreadedTCPSocketClient object at 0x1067154b0>, connection_uri = '127.0.0.1:50050', incoming_routing = <RoutingTag.GUI: b'GUI'>
outgoing_routing = <RoutingTag.FSW: b'FSW'>

    def connect(
        self,
        connection_uri,
        incoming_routing=RoutingTag.GUI,
    ):
```







Continuous Integration

- Extension of automatic system testing
- Triggered by software change request or other criteria
- Requires no human interaction
- Implements full regression testing

 **Some checks were not successful** [Hide all checks](#)
12 successful and 3 failing checks

✓		CI / Framework (pull_request)	Successful in 13m	Details
✓		Format Python / Format (pull_request)	Successful in 16s	Details
✓		MacOS-CI / MacOS-Framework (pull_request)	Successful in 27m	Details
✓		RPI-CI / RPI (pull_request)	Successful in 3m	Details
✓		Spell checking / Spell checking (pull_request_target)	Successful in 44s	Details
✓		CI / Ref (pull_request)	Successful in 3m	Details




 **All checks have passed** [Hide all checks](#)
13 successful, 1 skipped, and 2 neutral checks

✓		CI / Framework (pull_request)	Successful in 15m	Details
✓		Format Python / Format (pull_request)	Successful in 22s	Details
✓		MacOS-CI / MacOS-Framework (pull_request)	Successful in 27m	Details
✓		RPI-CI / RPI (pull_request)	Successful in 2m	Details
✓		Spell checking / Spell checking (pull_request_target)	Successful in 36s	Details
⌚		Spell checking / Report (pull_request_target)	Skipped	Details



System Testing with F'


- Provides GDS capabilities out-of-the-box for F' projects
- Designed for system testing early in project development
- Enables developers to:
 - Send commands
 - Receive events and telemetry
 - Uplink and downlink files
 - Design and upload sequences
 - Compose custom dashboards


[Commanding](#)
[Events](#)
[Channels](#)
[Uplink](#)
[Downlink](#)
[Dictionaries](#)
[Charts](#)
[Logs](#)
[Sequences](#)
[Dashboard](#)
[Advanced](#)



WARNING: 0
 FATAL: 0
 GDS Errors: 0

Channels

Filters:

First
 <
 0 - 25
 25
 >
 Last
 

Last Sample Time	Channel Id	Channel Name	Channel Value
2022-09-20T20:11:19.864Z	0x100	blockDrv.BD_Cycles	609
2022-09-20T20:04:25.622Z	0x200	rateGroup1Comp.RgMaxTime	794 us
2022-09-20T20:02:41.329Z	0x300	rateGroup2Comp.RgMaxTime	182 us
2022-09-20T20:07:34.186Z	0x400	rateGroup3Comp.RgMaxTime	518 us
2022-09-20T20:10:47.414Z	0x500	cmdDisp.CommandsDispatched	4
2022-09-20T20:11:14.848Z	0x4501	comBufferManager.CurrBufs	1
2022-09-20T20:11:18.861Z	0x4b00	systemResources.MEMORY_TOTAL	33551536 KB
2022-09-20T20:11:18.861Z	0x4b01	systemResources.MEMORY_USED	29354888 KB
2022-09-20T20:11:18.861Z	0x4b02	systemResources.NON_VOLATILE_TOTAL	1953902876 KB
2022-09-20T20:11:18.861Z	0x4b03	systemResources.NON_VOLATILE_FREE	1071927616 KB
2022-09-20T20:11:18.861Z	0x4b04	systemResources.CPU	10.62 percent
2022-09-20T20:11:18.861Z	0x4b05	systemResources.CPU_00	37.00 percent
2022-09-20T20:11:18.861Z	0x4b06	systemResources.CPU_01	1.00 percent
2022-09-20T20:11:18.861Z	0x4b07	systemResources.CPU_02	19.00 percent
2022-09-20T20:11:18.861Z	0x4b08	systemResources.CPU_03	0.99 percent
2022-09-20T20:11:18.861Z	0x4b09	systemResources.CPU_04	22.77 percent
2022-09-20T20:11:18.861Z	0x4b0a	systemResources.CPU_05	0.00 percent
2022-09-20T20:11:18.861Z	0x4b0b	systemResources.CPU_06	21.00 percent

Edit View



F' Integration Test Framework

- Framework designed for integration testing with F' GDS
- Allows users to write automated tests in Python
- Tests dispatch commands and assert on events and telemetry
- Creates excel test logs
- Can be run within continuous integration system

```
def test_active_logger_filter(self):
    self.set_default_filters()
    try:
        cmd_events = self.api.get_event_pred(severity=EventSeverity.COMMAND)
        actHI_events = self.api.get_event_pred(severity=EventSeverity.ACTIVITY_HI)
        pred = predicates.greater_than(0)
        zero = predicates.equal_to(0)
        # Drain time for dispatch events
        time.sleep(10)

        self.assert_command("cmdDisp.CMD_NO_OP")
        self.assert_command("cmdDisp.CMD_NO_OP")

        time.sleep(0.5)

        self.api.assert_event_count(pred, cmd_events)
        self.api.assert_event_count(pred, actHI_events)

        self.set_event_filter(self.FilterSeverity.COMMAND, False)
        # Drain time for dispatch events
        time.sleep(10)
        self.api.clear_histories()
        self.api.send_command("cmdDisp.CMD_NO_OP")
        self.api.send_command("cmdDisp.CMD_NO_OP")

        time.sleep(0.5)

        self.api.assert_event_count(zero, cmd_events)
        self.api.assert_event_count(pred, actHI_events)
    finally:
        self.set_default_filters()
```




Jet Propulsion Laboratory
California Institute of Technology

jpl.nasa.gov



Jet Propulsion Laboratory
California Institute of Technology