Notes on task for FPrime Integration Test Framework

1. Add Test ID, Title row w/ date, and auto-log evr's in test log (color indicates severity)
2. Add event severity to event predicate and arguments
3. Modify test logger to have font name and timestamps as arguments for now.
4. Continue Unit Testing the API:
    a. Event Await Functions
    b. Event Assert Functions
    c. Dictionary Functions
5. Refactor history searches to use a private helper (put await logic in one place)
6. Remove absolute paths from start script in ref demo.
    a. This needs more time and thought. Coordinate with Michael.

Implementation Brainstorming

1. Test Case Thoughts (not for this week).
    a. Decorator/annotations. Make a decorator for F' Test and gets a subtest.
    b. Decorator support for grouping tests by requirement.
2. Open ended thoughts (not for this week)
    a. Getting user feedback
    b. What should these tests be to give good context to what the design can do?
3. Investigate different types of history:
    a. queued history that will drop oldest elements for long tests
    b. logging histories
4. Incorporate V&V ID tags
5. Introspect library to add to the assert message

Implementation Issues

1. Current pydoc comments do not provide strict types.

Test API Requirements

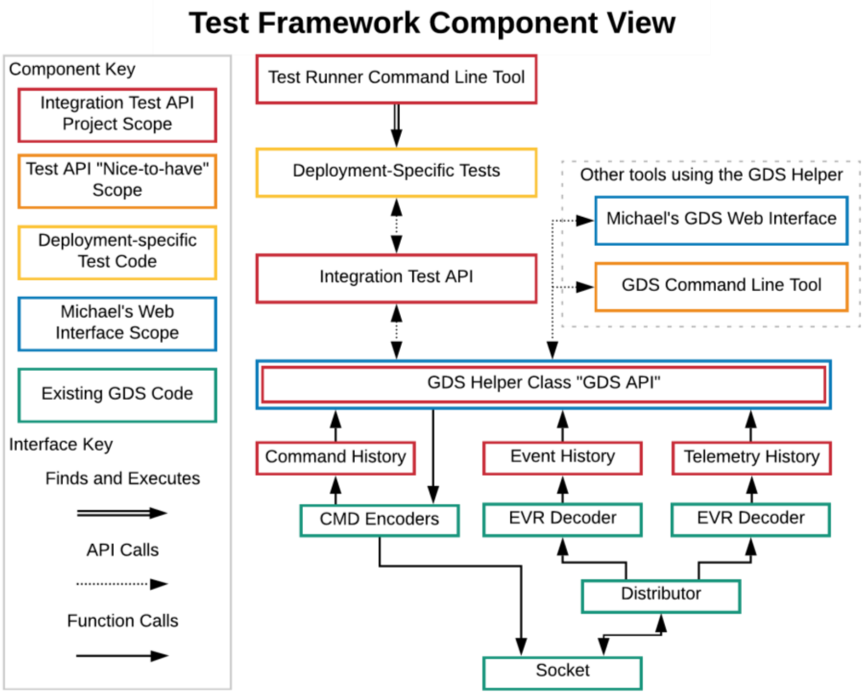
- Requirements on Python written for the Integration Test Framework
    - Python code shall be written to be compatible with both Python 2 and 3.
        - As of 6/25 this requirement has been tabled. Will likely remain this way.
    - Python code shall use named tuples if/when returning multiple results.
    - Python code shall use pydoc-compatible commenting to define parameters, operation and returns.
- Requirements on the GDS Test API
    - The Test API shall support the following functionality for both events and telemetry:
        - Awaiting a message
        - Asserting a message is in a history
        - Asserting a message is in a history or is received before a timeout. (Await assert)
    - The Test API shall provide basic assertions on event and telemetry messages.
    - The Test API shall provide asserts on FSW-created timestamps.

    - The Test API shall limit redundancy through the use of predicate functions for asserts.
    - The Test API shall create a detailed test log while it is being used by the test cases.
    - The Test API shall be performance tested to identify if it drops messages and at what point it drops messages.
    - The Test API asserts shall search efficiently: Optimized for the data structures in the history and better than O(n^2).
- Requirements on the GDS helper layer
    - The GDS helper layer shall provide the ability to initialize the GDS.
    - The GDS helper layer shall provide the ability to Send commands.
    - The GDS helper layer shall provide access to FSW Dictionaries.
    - The GDS helper layer shall provide and expose histories for EVRs, Telemetry and Commands.
- Requirements for the Integration Test Runner
    - The Test Runner shall collect artifacts to record the condition of the tests.
        - History logs
        - Copies of the FSW dictionaries
        - A copy of the FSW binary
    - The Test Runner shall collect files to record the results of the tests.
        - Test logs
        - Test reports
    - The Test Runner shall support specifying a deployment directory to discover and run Integration Tests.
    - The Test Runner should provide usability features to aid in CI/CD setup.
    - Test configurations, Command Line Interface
- Requirements on the integration tests:
    - Have a high-stress case that will find the boundaries on the API.

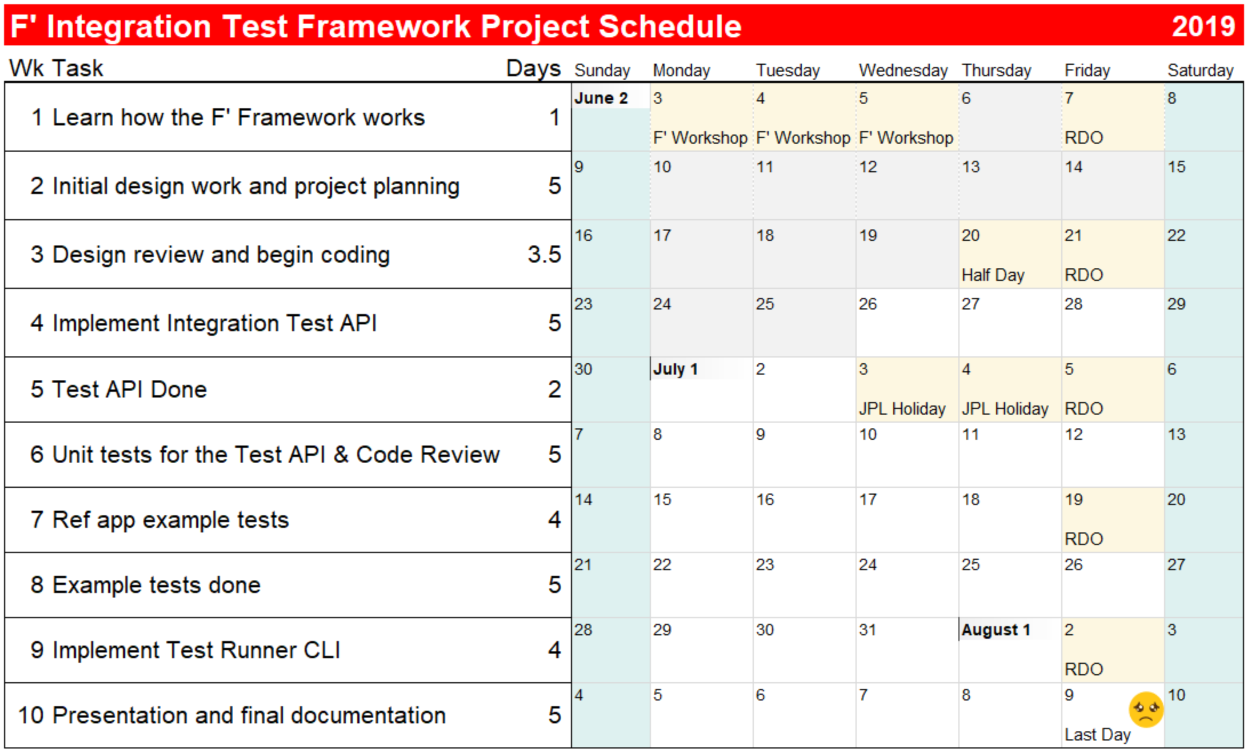
Test Framework Project Nice-to-have's

    - API/GDS: Being able to run multiple instances of the API for multiple F' Connections. -Len
    - GDS: Some sort of CLI tool or script to run command sequences through the GDS
        - Tim had some examples of command sequences in Ref/scripts.
        - This idea of a file interpreter could be used to script tests.
    - API: Break point hook in the API the ability to modify an assert or something like that.
    - GDS: Store histories in a file for long-term logging.
    - API/Runner: Post-processing long test logs to provide more focused feedback when things go wrong.
    - API: Investigate iterative test case generation or Expect functionality "Assert, but go on"
        - Soln 1: Add expect versions of asserts.
        - Soln 2: Add a layer to implement this in a library or something.
        - Soln 3: Recommend the user catches the assert and continues with the case.
    - CLI Tool: Command line tool to poke at the GDS
    - GDS: Store Histories in a file.
    - REF APP: Signal Generators to stress test the API.

API Design/Architecture


Test Framework Component View
Component Key
Integration Test API Project Scope
Test API "Nice-to-have" Scope
Deployment-specific Test Code
Michael's Web Interface Scope
Existing GDS Code
Interface Key
Finds and Executes
API Calls
Function Calls
Test Runner Command Line Tool
Deployment-Specific Tests
Other tools using the GDS Helper
Michael's GDS Web Interface
GDS Command Line Tool
Integration Test API
GDS Helper Class "GDS API"
Command History
Event History
Telemetry History
CMD Encoders
EVR Decoder
EVR Decoder
Distributor
Socket

API Proposed Schedule



F' Integration Test Framework Project Schedule **2019**

Wk	Task	Days	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
1	Learn how the F' Framework works	1	**June 2**	3 F' Workshop	4 F' Workshop	5 F' Workshop	6	7 RDO	8
2	Initial design work and project planning	5	9	10	11	12	13	14	15
3	Design review and begin coding	3.5	16	17	18	19	20 Half Day	21 RDO	22
4	Implement Integration Test API	5	23	24	25	26	27	28	29
5	Test API Done	2	30	**July 1**	2	3 JPL Holiday	4 JPL Holiday	5 RDO	6
6	Unit tests for the Test API & Code Review	5	7	8	9	10	11	12	13
7	Ref app example tests	4	14	15	16	17	18	19 RDO	20
8	Example tests done	5	21	22	23	24	25	26	27
9	Implement Test Runner CLI	4	28	29	30	31	**August 1**	2 RDO	3
10	Presentation and final documentation	5	4	5	6	7	8	9 😟 Last Day	10