Software Requirements Specification (SRS) Document

< Satellite Anomaly Injection & Detection Testbed>

<December 10, 2021>

<Version 1>

<By: Martha Caldera, Diana Degiacomo, Gabriel Kutasi, Jae Lee, Michael Morris, Gustavo Torres, Tomas Velarde, Dearo Yam, Rafael Zaragoza>

Table of Contents

1. Introduction

1.1 Purpose: The purpose of this document is to provide information on the anomaly detection, resolution, and injection. It also provides information on the software tools and simulation tools that are used in order to configure and simulate real-time environment anomalies.

1.2 Intended Audience: This Document is intended for:

- Software developers who can review the project's capabilities and understand where more features can be added or improved.
- Project testers who can use this document as a base guideline for their testing needs.
- Project managers can use this document in order to see what has been completed, what needs to be completed, and as a guidance to let developers know what they should do.
- End-users of the application who would like to read about what the project can do
- End-users can read this document in order to see what the application is capable of doing.

1.3 Scope: The scope of this document is the following:

- The software products in this application are **Anomaly Injection, Onboard Anomaly Detection, Ground Base Anomaly Detection, Onboard Anomaly Resolution,** and **Ground Base Anomaly Resolution**. These products will be further explained in 1.3.1, 1.3.2, 1.3.3, 1.3.4, and 1.3.5.
- All the systems will utilize the OSK/COSMOS environment. The software will use the simulated telemetry data and satellite data to inject and detect anomalies.

1.3.1 Anomaly Injection

- Allows the user to inject an anomaly into the cFS. The anomaly Injection will modify the data that is onboard the satellite.

1.3.2 Onboard Anomaly Detection

- Automated software will be able to detect anomalies within the cFS. The onboard anomaly detection will make comparisons with the satellite data and nominal data.

1.3.3 Ground Base Anomaly Detection

- Automated software will be able to detect anomalies found in the telemetry data. The ground base anomaly detection will make comparisons with the telemetry data and nominal data.

1.3.4 Onboard Anomaly Resolution

- Automated software will be able to resolve any anomalies detected within the cFS onboard the satellite.

1.3.5 Ground Base Anomaly Resolution

- Automated software will be able to resolve any anomalies detected within the telemetry data from the ground system.

1.4 Document Conventions, Definitions, Acronyms, and Abbreviations:

1.4.1 Acronyms/Abbreviations

Acronyms/Abbreviations	Definition
cFE	Core Flight Executive
cFS	Core Flight System
DOS	Denial Of Service
DDOS	Distributed Denial Of Service
OSK	OpenSatKit
SB	Software Bus Services
SBE	Single Bit Error

1.5 References:

1.5.1 Aerospace detailed proposal

1.5.2 OSK User's Guide

2. General Description

2.1 System Analysis:

Main Goals of Project:

Detection and Automation of Anomalous Behavior/Data

- Once an anomaly is detected either on board or on the ground system, software will try to automatically resolve the anomalous behavior
- This is done through advanced data analysis techniques which allow the anomalies to be resolved before they become a large issue.

Technical Hurdles:

- Hardware limitations on board system does not allow for large amount of processing power to be put on board system.
- Potential slow connection between ground system and flight system when they are not close to a communication point could lead to technical faults in the onboard software.
- Potential loss of flight system if anomaly isn't resolved in a quick manner.

2.2 Product Perspective:

2.2.1 Anomaly Injection

The anomaly injection will be integrated into the cFS. The anomaly injection will make use of the cFE libraries to be able to connect with the other applications located on the cFS.

2.2.2 Onboard Anomaly Detection

The onboard anomaly detection will be integrated into the cFS. The onboard anomaly detection will make use of the cFE libraries in order to connect with the other applications located on cFS.

2.2.3 Ground Base Anomaly Detection

The ground base anomaly detection will be integrated into COSMOS. The ground base anomaly detection will make use of the cFE libraries to connect with COSMOS and the cFS.

2.2.4 Onboard Anomaly Resolution

The onboard anomaly resolution will be integrated into the cFS. The onboard anomaly resolution will make use of the cFE libraries in order to connect with the other applications located on cFS.

2.2.5 Ground Base Anomaly Resolution

The ground base anomaly resolution will be integrated into COSMOS. The ground base anomaly resolution will make use of the cFE libraries to connect with COSMOS and the cFS.

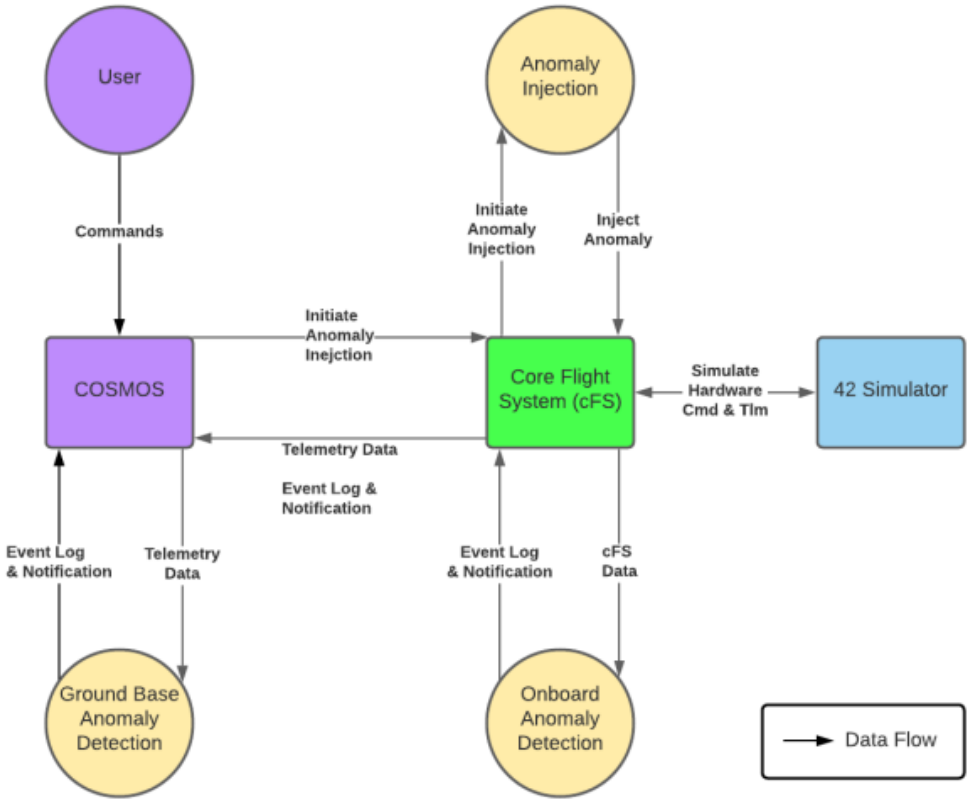

Figure 2-1 Product Data Flow Diagram

2.3 Product Functions:

2.3.1 Anomaly Injection

- Injects the following anomalies:
  - Runaway task - Eats up the CPU of the cFS
  - Memory leak - Eats up the memory of the cFS
  - Denial of service - Occupies the bandwidth of the OSK softwarebus
  - Invalid command sequences - Affects the execution of commands in the cFS
  - Single bit errors - Affects the memory data of the cFS
- The injection software will allow the user to input their anomaly of choice
- The injection software will have an automated process that will wait until a period of time has passed before injecting one of the anomalies at random.
  - The user can turn this feature on and off

2.3.2 Onboard Anomaly Detection

Automated detection software checks for anomalies. The anomalies being checked for are runaway tasks, memory leak, and single bit errors. Once an anomaly has been detected it creates and event log and notification of the anomaly. An event log and notification are queued and downlinked to the ground system next contact.

2.3.3 Ground Base Anomaly Detection

Automated detection software checks for anomalies from the telemetry data. The anomalies being checked for is the denial of service and invalid command sequences. Once an anomaly has

been detected it creates an event log and a notification of the anomaly. The event log and notification are sent to ground.

2.3.4 Onboard Anomaly Resolution

Automated software onboard checks if there are any anomalies that have been detected and it will resolve them. Once an anomaly has been resolved it creates and event log and notification of the anomaly. An event log and notification are queued and downlinked to the ground system next contact.

2.3.5 Ground Base Anomaly Detection

Automated software on the ground system checks if there are any anomalies that have been detected and it will resolve them. Once an anomaly has been resolved it creates an event log and a notification of the anomaly. The event log and notification are sent to ground.

2.4 Operating environment:

- The software will run on the Ubuntu 18.04 LTS operating system, as OSK currently is supported on that specific Linux system. It is expected to have OSK, cFS, COSMOS, and 42 installed to make use of the desired open source software for development.

2.5 Constraints:

- Onboard satellite constraints of limited memory and CPU cycles running on a single thread
- Ground communication limitation of reduced response times and payload limitations

2.6 Assumptions and dependencies:

- Ubuntu Operating System (18.04 LTS)

3. External Interface Requirements.

3.1 User Interfaces

TBD

3.2 Hardware Interfaces

TBD

3.3 Software Interfaces

Open Sat Kit (OSK) – Core Flight System Starter Kit

The project will leverage OpenSatKit (OSK), which combines three powerful open source tools that are currently used in real missions today: Ball Aerospace Corporation's COSMOS ground system, NASA Goddard's core Flight System(cFS) flight software, and NASA Goddard's 42 satellite simulator. See the documentation for OSK in its GitHub Wiki.

Each major software component is described in more detail in the sections below.

Core Flight System – Flight Software

OSK provides a complete desktop solution for learning how to use NASA's open source flight software (FSW) platform called the core Flight System (cFS). The cFS is a reusable FSW architecture that provides a portable and extendable platform with a product line deployment model. The cFS has significant flight heritage, provides a complete set of command and data handling functions required by most spacecraft, and is reliable. A virtual environment with OSK set up will serve as the development environment for learning about flight and ground system communications as well as providing a platform for developing anomalies to be injected into the simulation. OSK comes with the cFS preconfigured for a fictitious satellite called SimpleSat (SimSat).

42 – Spacecraft Simulator

In addition to cFS, OSK uses NASA Goddard's 42 dynamic satellite simulator for simulated hardware command and telemetry. 42 is a comprehensive general-purpose simulation of spacecraft attitude and orbit dynamics. Its primary purpose is to support design and validation of attitude control systems. 42 accurately models multi-body spacecraft attitude dynamics as well as modelling environments from low Earth orbit to throughout the solar system. It also features visualization of spacecraft attitude.

COSMOS – Ground System

OSK implements extensive COSMOS configurations and customizations so COSMOS can serve as the primary OSK user interface. COSMOS is a suite of applications that can be used to communicate with the satellite, monitor its performance and health, and display its data. The systems that COSMOS interfaces with can be anything from test equipment (power supplies, oscilloscopes, switched power strips, UPS devices, etc.), to development boards (Arduinos, Raspberry Pi, Beaglebone, etc.), to satellites.

COSMOS implements a client server architecture with the Command and Telemetry Server and the various other tools typically acting as clients to retrieve data. The Command and Telemetry Server connects to the targets and sends commands and receives telemetry (status data) from them. Targets are the items you are trying to control or get status from

3.4 Communications Interfaces

TBD

4.Requirements Specification

4.1 Functional Requirements

Requirement #	Requirement Description
4.1.1	**Anomaly Injection**
4.1.1.1	The anomaly injection shall inject the following anomalies • Runaway Task • Memory Leak • Denial of Service • Invalid Command Sequence • Single Bit Error Explanation: These are the anomaly examples that were proposed for use.
4.1.1.2	The runaway task shall create a task in the cFS that does not terminate Explanation: A task will be created in cFS that never terminates.
4.1.1.3	The memory leak shall allocate memory on the cFS without deallocating Explanation: This will happen when memory is allocated without properly being deallocated on the satellite.
4.1.1.4	The denial of service shall spam messages to the software bus Explanation: This will happen when there will be an overwhelming amount of messages being sent on the software bus.
4.1.1.5	The invalid command sequence shall send a sequence of commands out of order Explanation: This will happen when commands are going to be sent out of order.
4.1.1.6	The single bit error shall flip a bit in the rewritable memory of the cFS Explanation: This will target rewritable memory as it can crash the whole system if one bit is out of place.
4.1.1.7	The anomaly injection shall have an automated process to inject anomalies at a random time, or the user can manually do it. Explanation: There should be a way to automatically inject anomalies but also the user should be able to manually do this.
4.1.2	**Onboard Detection**
4.1.2.1	There should be automated onboard anomaly checks for the following anomalies: • Runaway Task • Memory Leak • Single bit Error Explanation: These are the anomalies that are found onboard the satellite.
4.1.2.2	The runaway task detection shall acquire the CPU data from the cFS and

	detect anomalous behavior in it Explanation: The runaway task will eat up CPU resources that are not being used, this will be detected and sent for resolution.
4.1.2.3	The memory leak detection shall acquire the memory state from the cFS and detect anomalous behavior in it Explanation: The memory leak will eat up spare memory onboard that is not being used, this will be detected and sent for resolution.
4.1.2.4	The single bit error shall acquire memory data from the cFS and detect anomalous behavior in it Explanation: If there are any bits off in the memory data, the anomaly will be detected and sent for resolution.
4.1.2.5	An event log shall be created onboard, along with a notification of each time an anomaly is detected which can both be sent to the ground system Explanation: This will allow the developers to keep track of all the anomalies that have been detected onboard the satellite.
4.1.3	**Ground Detection**
4.1.3.1	There should be automated ground system anomaly checks for the following anomalies: • Denial of Service • Invalid Command Sequence Explanation: These anomalies are found on the ground system.
4.1.3.2	The denial of service detection shall acquire the connection time between the COSMOS and the cFS and detect anomalous behaviors in it Explanation: If there is any type of slowing down in communication between the COSMOS and cFS, an anomaly will be detected and sent to resolution.
4.1.3.3	The invalid command sequence detection shall acquire the command sequence being sent up to the cFS and detect anomalous behaviors in it Explanation: If any commands are sent out of order it will mean that an anomaly is happening and will be sent for resolution.
4.1.3.4	An event log shall be created in the ground system, along with a notification of each time an anomaly is detected Explanation: This will allow the developers to keep track of all the anomalies that have been detected on the ground system.
4.1.4	**Onboard Resolution**
4.1.4.1	Automated software onboard will resolve runaway task anomalies once they are detected. Explanation: The runaway task will eat up CPU resources that are not being used. Once this is detected, the anomaly will be resolved.
4.1.4.2	Automated software onboard will resolve memory leak anomalies once they are detected.

	Explanation: The memory leak will eat up spare memory onboard that is not being used. Once this is detected, the anomaly will be resolved
4.1.4.3	Automated software onboard will resolve single bit error anomalies once they are detected. Explanation: If there are any bits off in the memory data, the anomaly will be automatically resolved.
4.1.4.4	An event log shall be created onboard, along with a notification of each time an anomaly is resolved which can both be sent to the ground system Explanation: This will allow the developers to keep track of all the anomalies that have been resolved onboard the satellite.
4.1.5	**Ground Resolution**
4.1.5.1	Automated software in the ground system will resolve denial of service anomalies once they are detected Explanation: If there is any type of slowing down in communication between the COSMOS and cFS, the anomaly will be automatically resolved.
4.1.5.2	Automated software in the ground system will resolve invalid command sequence anomalies once they are detected. Explanation: If any commands are sent out of order it will mean that an anomaly is happening and will be resolved.
4.1.5.3	An event log shall be created in the ground system, along with a notification of each time an anomaly is resolved Explanation: This will allow the developers to keep track of all the anomalies that have been resolved on the ground system.

4.2 External Interface Requirements

See Section 3

4.3 Logical Database Requirements

TBD

4.4 Design Constraints

Limited amount of computing power onboard, along with limited CPU cycles and processor running on a single thread.

5. Non-Functional Requirements

5.1 Safety requirements

The main safety requirement is safe practices by everyone using the software and making sure that it only accepts valid templates for anomaly simulations, making sure that there is nothing that can harm the software.

5.2 Security requirements

The main form of software is an open source software, meaning that there is not much security in place.

5.3 Software quality attributes

Most significant issue: software must run on Ubuntu 18.04 LTS operating system; key software components from OSK only support that OS. Hardware limitations should be exceptionally low, allowing for installation and operation on most platforms to support Ubuntu.

5.4 Legal requirements

There are no legal requirements currently in place.