Autonomous Operations System: Development and Application

Jaime A. Toro Medina[1], Kim N. Wilkins[2], Mark Walker[3], Gerald M. Stahl[4]

[1,4]*NASA Kennedy Space Center, Kennedy Space Center, Florida, 32899, United States of America*
jaime.a.toromedina@nasa.gov
gerald.m.stahl@nasa.gov

[2]*General Atomics, 3550 General Atomics Court, San Diego, California, 92121, United States of America*
kim.wilkins@ga.com

[3]*D2K Technologies*
mark.walker@d2ktech.com

ABSTRACT

Autonomous control systems provide the ability of self-governance beyond the conventional control system. As the complexity of systems increases, there is a natural drive for developing robust control systems to manage complicated and emergency operations. By closing the bridge between conventional automated systems and knowledge based self-aware systems, nominal control of operations can evolve into relying on safety critical mitigation processes to support any off-nominal behavior. Current research and development efforts lead by the Autonomous Propellant Loading (APL) project at NASA Kennedy Space Center aims to improve cryogenic propellant transfer operations by developing an automated control and health monitoring system. As an integrated system, the center aims to produce an Autonomous Operations System (AOS) capable of integrating health management operations with automated control to produce a fully autonomous system.

1. INTRODUCTION

Ground Support Equipment (GSE) to support human rated space flight hardware is one of the most robust and safe systems in industry. Such systems undergo the most rigorous scrutiny to ensure nominal and safe operations during a cryogenic propellant transfer from storage to flight vehicle in preparations for a launch. Command and control systems aimed to conduct such operations can be as complex as the GSE and flight vehicle itself. The use of models, simulations and cryogenic lessons learned from past experiences are constant elements being employed on a launch control system. Several technologies have been developed to monitor and assess health of flight vehicle components. For instance, one of these technologies has been applied to the Space Shuttle Main Engine (SSME) to improve Shuttle safety by reducing the probability of catastrophic engine failures during the powered ascent phase of a Shuttle mission (Davidson and Stephens, 2004). Other applications have employed the use of real-time prognostics and health monitoring (PHM) to increase safety and reliability on unmanned aircrafts (Walker, 2010). A wide variety of industrial applications have embraced the capability of health management and prognostics to improve their systems. An integrated approach of these technologies can be combined with modeling of complex system, continuous health monitoring using complex models, and telemetry management. Such capability can be encompassed on an effort to develop knowledge based systems capable of integrating system knowledge with health management. Integrated System Health Management (ISHM) is such capability aimed to integrate data, information, and knowledge to be distributed throughout system elements (Figueroa and Melcher, 2011).

Autonomous control systems are designed to provide the power to self-govern the performance of control functions (Antsaklis et al., 1991). A complex system often requires interdisciplinary knowledge to be physics modeled combined with software tools to produce autonomous control. Some systems can employ architectures that combine models, knowledge system and data acquisition to produce pseudo autonomous systems (Childers, 2007) and autonomous hybrid energy systems (Rafeed Leon et al., 2016).

Autonomous systems that use the advantages of ISHM, automated processes and piece-wise system modeling in a software architecture are powerful concepts to be explored. The Autonomous Propellant Loading (APL) group at NASA Kennedy Space Center has taken the task to explore these concepts and develop an Autonomous Operations System (AOS) capable of using distinct technologies to produce software capable of executing cryogenic propellant transfer operations autonomously.

2. PROJECT OVERVIEW

The Autonomous Operations System (AOS) for the Autonomous Propellant Loading (APL) project is intended to demonstrate the ability to quickly develop a control system that can provide autonomous control and health management capability of cryogenic propellant loading operations using a Commercial-Off-The-Shelf (COTS) software product called G2 (Gensym Corporation) combined with several modifications to a custom application called the ISHM Toolkit (Venkatesh et. al., 2013). To accomplish this, an Integrated System Health Management (ISHM) capability for an Autonomous Propellant Loading (APL) system called Autonomous Operations System has been developed. This application will provide information on the health of every hardware element of the system, such as sensors, actuators, pipes, tanks, valves, etc. It will also include an Automated Control Sequencer (ACS) which executes control sequence steps after evaluating conditions, including health conditions of system elements involved in sequence executions.

The scope of this software project involves the programmatic integration of autonomous sequence control, hardware component commanding and health monitoring in one application. The objective of this software is to perform monitoring, health management, commanding and control of cryogenic system operations for the Universal Propellant Servicing System (UPSS) and Iron Rocket systems. A demonstration of these capabilities has been achieved through the use of a test sequencer that performed autonomous control of cryogenic tank loading and de-tanking operations against simulations of the cryogenics hardware.

3. SOFTWARE DESIGN DESCRIPTION

Some of the main modules that constitute the AOS are the following: Top Level, Engine, Sequencer, NASA Library, Bridge, Domain Object Libraries, Redline Monitoring, AOS Bridge, Application Engine and other supporting modules. Each module is saved as a Knowledge Base (KB) file (a G2 specific format); the KB file format is a proprietary binary file that is created, modified, and used by the G2 COTS tool.

Each module developed in G2 contains all or some combination of the following: rules, methods, procedures, Graphical User Interfaces (GUIs), class definitions and workspaces. G2 is a full-fledge Object-Oriented platform. Rules, procedures and methods set up the behavioral functionality of objects in the real-time environment. Procedures and methods are named objects that execute a sequence of actions including, but not limited to, calling other methods, procedures and setting parameters. GUIs allow the user to interact with the system. Objects represent physical components in the system within the application and class definitions define the attributes of objects and the associated methods. Workspaces are window-spaces where objects exist within the application.

The high level architecture of AOS is illustrated in Figure 1. This architecture includes the deployed application for the APL project composed of the UPSS and Iron Rocket Systems. The AOS architecture is composed of AOS Software Components and Application Software Components.

3.1. AOS Software Components

3.1.1. Top Level

The Top Level module is responsible for the management of the integrated development environment (IDE), initialization of application software components and management of launching procedures to initiate the execution of the AOS module hierarchy, as well as the functionality for creating and closing the telewindows application that is part of running G2.

3.1.2. Sequencer

The Sequencer is the module responsible for executing sequential commands that can be scripted to represent an operation. Steps execute commands when conditions of state and health are met; and alternate plans may be selected autonomously to achieve the final objectives of a plan, in spite of unforeseen anomalies.

3.1.3. Domain Object Libraries

This module contains the domain object elements used to build domain maps. A domain map is a user interface to the application domain model consisting of every element of the system represented as an object. Domain objects are connected according to schematics, and incorporate data, information, and knowledge about the system.

In addition to the domain map, a subcategory of this map is the control map. This map uses information available on the domain map to monitor telemetry obtained from the instrumentation of hardware. The control map enables execution of commands, and controls command-supported elements in the system which includes valves and setpoints. In addition, the control map provides the user with a visual interface to monitor progress of a cryogenic propellant transfer operation.

3.1.4. Engine

The Engine module executes the main rules and procedures necessary to run the application. This module manages streamed to domain object element from the physical system's data acquisition system. In addition, it executes procedures necessary to interact with real-time data streamed to domain objects, processes the end-user operational control map, executes redline monitoring, and manages plotting capabilities.

3.1.5. Redline Monitoring

This module is responsible for redline monitoring on the system. Redline monitors have limit threshold values referenced with a sensor, including virtual sensors. The redline elements perform monitoring activities across the system. The triggering of the redline monitors produce alerts and execution of safing plans. Depending on the redline sensors being used and the type of operation, these redlines can pause a main sequence and execute an advance to shutdown safing plan to protect the system from potential hazards.

3.1.6. Bridge

This module is responsible for establishing communication protocols with external interfaces. This module manages Space Packet Protocol (SPP) data transfer (incoming and outgoing) by means of the G2 Gateway Standard Interface (GSI) by interfacing objects to an external tool called the AOS Bridge. These objects create an interaction with any external system that can manage telemetry in the specified format. In this case, this module contains all the GSI data interfacing object involved in a specific application.

3.1.7. NASA Library

This module is responsible for integrated system health management (ISHM) capability. Within this module a machine state analysis is performed over the command and telemetry of the elements such as valves and sensors to determine health. In addition, several physics localized models are being managed under this module to determine saturation conditions of cryogenic commodity, leaks, valve state assessment, valve consistency, health assessment and failure modes. This module is constantly assessing the conditions of the system to determine an appropriate response. This module can determine instrumentation-only failure according to several approaches, as well as mechanical failures.

3.1.8. AOS Bridge

The AOS Bridge provides the means of communication from external data acquisition systems (DAQ) to the AOS main application code. It runs as an independent process alongside the AOS main application on the same host. The AOS Bridge relies on a set of required callbacks and classes for processing telemetry and commands made possible through the use of two G2 Gateway Standard Interface (GSI) objects instantiated in the main AOS application. The first GSI interface is used to register all telemetry and timestamp variables. These elements are commonly being referred to Compact Unique Identifier (CUIs), which represent a unique identifier for a hardware end item in the current APL application. The Secondary GSI interface supports remote procedure call used for commanding. Each GSI interface operates as a TCP/IP connection on a dedicated common port between the AOS Bridge and the AOS main application. In the sections below, the files composing the AOS Bridge are listed from which further detail is given on the callback structure and classes used.

3.2. Application Software Components

3.2.1. Domain Maps

The domain maps are the central location for the knowledge of the application and supplies information to all the application software components. In addition, they serve as the user interface to the application domain model consisting of every element of the system represented as an object. Domain objects are connected according to schematics, and incorporate data, information, and knowledge about the system.

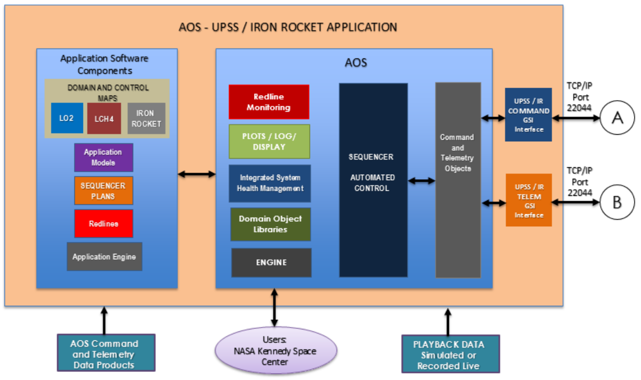

AOS - UPSS / IRON ROCKET APPLICATION
Application Software Components
DOMAIN AND CONTROL MAPS
LO2
LCH4
IRON ROCKET
Application Models
SEQUENCER PLANS
Redlines
Application Engine
AOS
Redline Monitoring
PLOTS / LOG/ DISPLAY
Integrated System Health Management
Domain Object Libraries
ENGINE
SEQUENCER
AUTOMATED CONTROL
Command and Telemetry Objects
UPSS / IR COMMAND GSI Interface
UPSS / IR TELEM GSI Interface
TCP/IP Port 22044
TCP/IP Port 22044
A
B
AOS Command and Telemetry Data Products
Users: NASA Kennedy Space Center
PLAYBACK DATA Simulated or Recorded Live

Figure 1. AOS Software Architecture

3.2.2. Control Maps

These map uses information available on the domain maps to monitor telemetry obtained from the instrumentation of hardware. The control map is capable of executing command and control of command-supported elements in the system which includes valves and setpoints. In addition, the control map provides the user with a user interface to monitor progress of a cryogenic propellant transfer operation.

3.2.3. Application Models

These models can be instantiated and used within specific applications. Generic models from the NASA Library are applied to the domain map representation to produce piping section object models; flow subsystem object models; saturation condition of cryogenic commodity according to application specific sensor pair (pressure and temperature); valve state assessment on application valves and others.

3.2.4. Redlines

The redline monitor generic capability is used, along with user inputs, to determine undesirable conditions of the system in which mitigation actions needs to be taken to protect the system once a threshold violation is triggered.

3.2.5. Sequencer Plans

These plans are derived using the Sequencer generic capability to provide a set of scripted sequential commands focused on nominal and off-nominal operations. Off-nominal plans associated with redline triggering are considered safing plans since they provide a mitigation sequence that returns the system into a safe configuration once an undesired condition is met. In addition, other types of control algorithms are provided by the sequencer to employ the use of sequential commanding in a parallel execution for specialized needs required by the application.

3.2.6. Application Engine

This software component provides an extension of the Engine module. It manages a mapping from the data objects representing the external data to the domain object representation (sensor, valves, etc.). This engine holds application-specific characteristics of the mapping and allows data transfer.

4. AUTONOMOUS OPERATION

The capabilities provided by AOS allow the application developer to develop an application that represents the external hardware for command and control operations. The tools within AOS allow for a domain map representation using generic libraries for components and models used for modeling the real system. These utilities allow for the creation of the application software components needed to deploy a specific application for hardware operations. Along with the application software components, several strains are tasked to execute the application software component in an integrated manner. This coordinated execution of all the AOS Software Components represents the true nature of the autonomous operations.

4.1. Nominal Operations

During nominal operations, the workflow of the autonomous engine behaves as seen on Figure 2. There are three parallel modules executing their functionalities on the application software components.

4.1.1. Automated Plan Execution

The Sequencer uses telemetry from the application software domain object elements to execute the automated script for commands. During the execution of the script, which in this case is a nominal plan, conditions are evaluated prior to command execution. The command execution is supplied to the domain object elements and communication is provided to the external hardware components to execute the command.

4.1.2. NASA Library Models

In a parallel effort, the NASA Library module uses the telemetry and evaluates the domain object elements behavior according to the specified models. These models assess the application and produces an output that is evaluated. For the case of nominal operations, the output of the models provides a favorable assessment and supplies the domain object elements with the nominal state result.

4.1.3. Redline Monitoring

In a parallel effort, Redline Monitoring evaluates domain object element. The nominal state is compared to the previously specified threshold for a redline. Nominal states of domain objects do not trigger any redline conditions and no further action is taken.

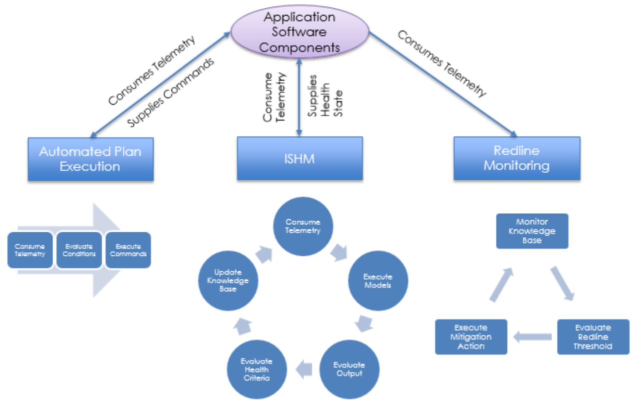


Figure 2. Nominal Autonomous Operations

4.2. Off Nominal Operations

During off-nominal operations, the workflow of the autonomous engine behaves as seen on Figure 3. During this operation there are three parallel modules executing their functionalities upon the application software components.

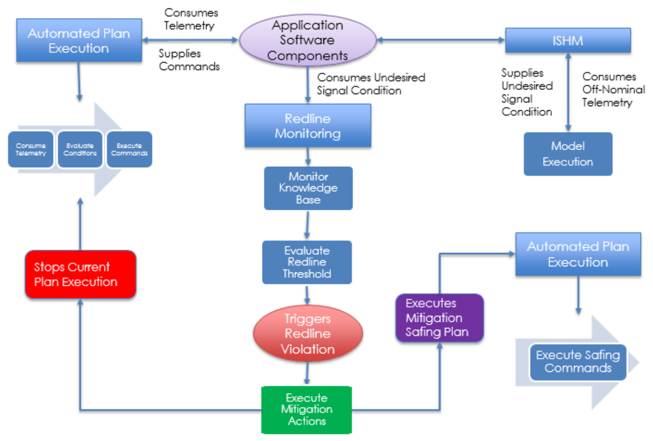

Automated Plan Execution
Consumes Telemetry
Supplies Commands
Application Software Components
ISHM
Consumes Undesired Signal Condition
Supplies Undesired Signal Condition
Consumes Off-Nominal Telemetry
Consume Telemetry
Evaluate Conditions
Execute Commands
Redline Monitoring
Model Execution
Monitor Knowledge Base
Evaluate Redline Threshold
Automated Plan Execution
Stops Current Plan Execution
Executes Mitigation Safing Plan
Triggers Redline Violation
Execute Safing Commands
Execute Mitigation Actions

Figure 3. Off Nominal Autonomous Operation

4.2.1. NASA Library Models

Similar to the nominal operation, the NASA Library consumes telemetry from the domain object elements and executes the models. In the case of off-nominal operations, the telemetry provides off-nominal telemetry values. The models evaluate these values and the output of a model is a signal indicating that an undesired condition is present in the domain objects.

4.2.2. Redline Monitoring

In a parallel effort, the Redline Monitoring is still evaluating telemetry from the domain object. In this case, the telemetry obtained reflects an undesirable signal that is compared to the redline monitor threshold values. Once the redline conditions are triggered, several mitigation actions take place. The first one is to stop the execution of any nominal plan and the second is to spawn a parallel process that executes a mitigation safing plan associated with the triggered redline.

4.2.3. Automated Plan Execution

Similar to nominal operations, the nominal plan is being executed. Once the Sequencer receives the signal from the redline monitor, it stops the execution of the nominal plan and executes the mitigation safing plan. During the execution of the mitigation safing plan, no conditions are evaluated and only safing commands are supplied to the domain object elements.

5. APPLICATION

5.1. Application Description

The APL test site will reside at KSC's Bore Site East. The Development Test Site (DTS) is being built by the 21st Century Space Launch Complex Program to support testing of their Universal Propellant Servicing System (UPSS). One major part of the DTS is the "Iron Rocket" which includes 3 tank stages for both liquid oxygen, LO2, and liquid methane, LCH4 (3000 gallon first stage, 500 gallon second stage, 500 gallon third stage), piping and components. The DTS also includes concrete pads, facility power and lightning protection. UPSS and DTS will be integrated to provide the testing environment for validating APL concepts on a flight-like system.

The Universal Propellant Servicing System (UPSS) includes cryogenic fuel (liquid methane, LCH4) and oxidizer (LO2) mobile tankers, control and relief valves, fluid piping, instrumentation, command and control hardware (see Figure 4). The command and control hardware consists of four Control Chassis and a Mobile Power and Communication Distribution Unit (MPCDU). It also has two skids for each propellant, one that interfaces with the mobile propellant tanker and one that interfaces with the vehicle or iron rocket of the Bore Site Project that interfaces with the Control Center. The control center may evolve over time and includes multiple locations/capabilities including recording, playback, video and associated displays, and communication, remote (Firing Room 4 of the Launch Control Center) vs. local control, and mobile control. Mobile high pressure pneumatic trailers will be provided to support gaseous nitrogen (GN2) and helium (GHe) needs of an integrated configuration. Initial ground testing of UPSS and the Bore Site Iron Rocket will occur using Liquid Nitrogen (LN2) and eventually UPSS will support loading a small class vehicle (SCV) customer's rocket. UPSS is mobile ground support equipment intended to connect with a flight SCV customer's cryogenic propellant tanks and associated fluid systems.

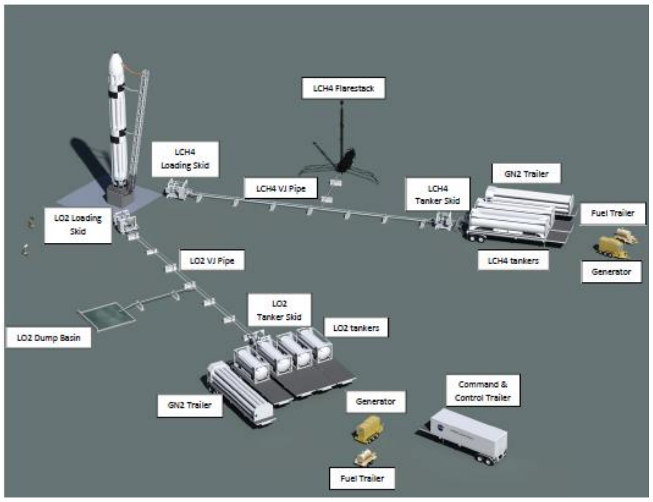


Figure 4. Universal Propellant Servicing System (UPSS)

The DTS which includes two vehicle simulator three stage cryogenic tanks (Iron Rocket), instrumentation associated with these tanks (LO2 and LCH4) and the ground side connections, fluid lines, relief valves, command and control hardware, concrete pads for both tanks and an additional concrete pad for future SCV customers, and structural support for all components (see Figure 5). The DTS is a project that is delivering and installing several unique capabilities, systems, and infrastructure. These capabilities, systems, and infrastructure will be permanently located at the SCV Cryogenic Loading Facility (CLF).

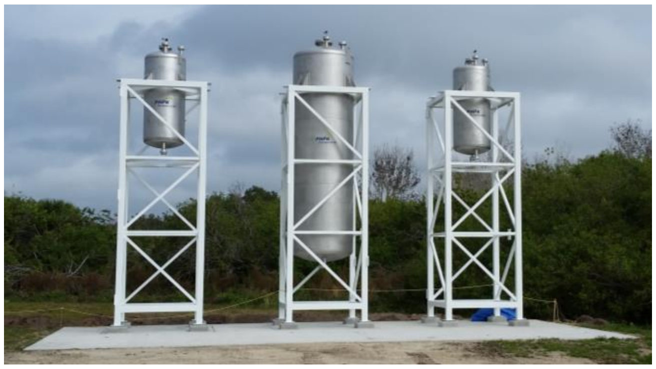


Figure 5. LO2 DTS Vehicle Simulator

5.2. Application Software Model: UPSS/Iron Rocket Simulator

This computer software configuration item (CSCI) simulates the UPSS and Iron Rocket systems for testing purposes. The simulator uses the FlowMaster COTS tool to model the UPSS and Iron Rocket systems (see Figure 6) and generate high-fidelity data that corresponds to various predefined nominal and off-nominal test scenarios. It then uses this data during testing to provide the appropriate telemetry data in response to AOS commands.

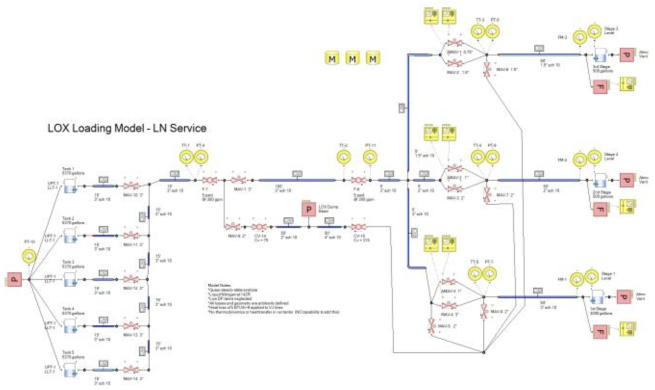


Figure 6. UPSS and Iron Rocket FlowMaster Model

6. TEST ARTICLE ARCHITECTURE

In order to successfully demonstrate the capabilities of Autonomous Operations System, several hardware and software components are used for testing different phases of the development. By means of testing the software in different phases, the AOS will accomplish a validation and verification process that is required to connect with other external systems.

6.1. AOS Application Development System

The AOS Application Development System is composed of a main computer for software development with four monitors for displaying data, visualization, domain maps and programmable code. In this computer system, the main application for controlling and monitoring the cryogenics propellant transfer for AOS is being developed. Within its local configuration, the knowledge of the system is being created by transferring mechanical, electrical and communication schematics into a domain map system. This domain map system receives data for the outside systems (PLC, UPSS/Iron Rocket Simulator, and Gateway) and feeds different subsystems (health management, automated sequencer control, etc.).

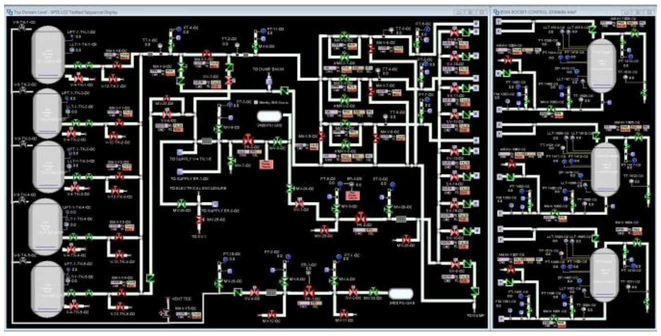


Figure 7. UPSS and Iron Rocket Control Map

The development of several subsystems within the AOS Application includes a visual representation of the domain map for commanding and control (see Figure **7**), an automatic sequencer controller (Figure 8), health management subsystems and plotting features.

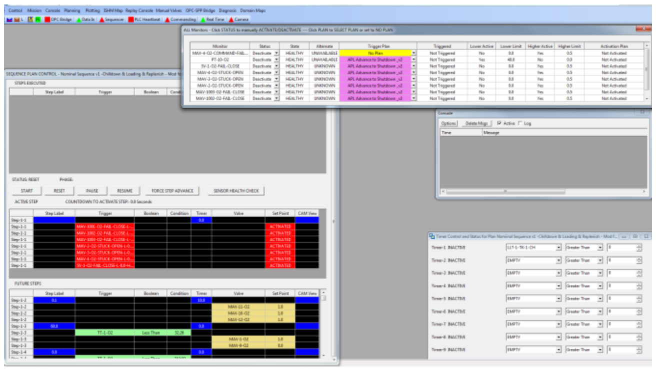


Figure 8. Automatic Sequencer Controller, Redline Monitoring, Timers, Console and Operations Display

6.2. AOS Bridge Development System

The AOS Bridge Development system is composed of an AOS bridge development main computer and a secondary computer used to run a basic User Datagram Protocol (UDP) server to test AOS bridge communications. In the main computer, the bridge code for the AOS Application is being developed. The development of the bridge code includes the capability of receiving telemetry and sending commands to a generic UDP server. The second computer running the generic UDP Server will provide a "test" network connectionless interface similar to the SPP to Common Industrial Protocol (CIP) Gateway code that the AOS Bridge will eventually connect to for PLC (Programmable Logic Controller) data. This generic UDP server will serve as an initial testing server to validate AOS communication code for a future integration with the SPP to CIP Gateway.

6.3. PLC Development System

The PLC Development System is composed of a set of six PLCs and a main computer for development. In this main computer, the code is being developed to send and receive data to the PLCs by means of Compact Unique Identifiers (CUIs). The software RSLogix 5000 is being used to program the PLC. In addition, Labview is being used as a secondary software application to develop a low fidelity simulator capable of supporting telemetry and commanding across the network. The low fidelity simulator allows modification to telemetry values of the different CUIs programmed in the PLC as well as displaying commands being received by the PLC across the network (see Figure 9).

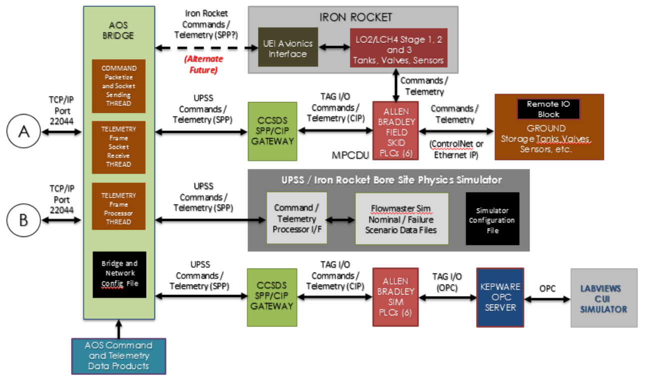

AOS BRIDGE
COMMAND Packetize and Socket Sending THREAD
TELEMETRY Frame Socket Receive THREAD
TELEMETRY Frame Processor THREAD
Bridge and Network Config File
AOS Command and Telemetry Data Products
A
B
TCP/IP Port 22044
TCP/IP Port 22044
Iron Rocket Commands / Telemetry (SPP?)
(Alternate Future)
IRON ROCKET
UEI Avionics Interface
LO2/LCH4 Stage 1, 2 and 3 Tanks, Valves, Sensors
Commands / Telemetry
UPSS Commands / Telemetry (SPP)
CCSDS SPP/CIP GATEWAY
TAG I/O Commands / Telemetry (CIP)
ALLEN BRADLEY FIELD SKID PLCs (6)
MPCDU
Commands / Telemetry
(ControlNet or Ethernet IP)
Remote IO Block
GROUND Storage Tanks, Valves, Sensors, etc.
UPSS / Iron Rocket Bore Site Physics Simulator
UPSS Commands / Telemetry (SPP)
Command / Telemetry Processor I/F
Flowmaster Sim Nominal / Failure Scenario Data Files
Simulator Configuration File
UPSS Commands / Telemetry (SPP)
CCSDS SPP/CIP GATEWAY
TAG I/O Commands / Telemetry (CIP)
ALLEN BRADLEY SIM PLCs (6)
TAG I/O (OPC)
KEPWARE OPC SERVER
OPC
LABVIEWS CUI SIMULATOR

Figure 9. AOS Network Architecture with AOS Bridge connected to LabView Simulator and UPSS/Iron Rocket Simulator

7. TEST RESULTS

For testing the UPSS and Iron Rocket application developed using the Autonomous Operations System, the UPSS and Iron Rocket Simulator was used to provide a physics-based telemetry responses according to application commands. Several strategies for propellant loading transfer were explored to develop a nominal and off-nominal plan for cryogenic propellant transfer. For testing purposes, the physics-based model simulated liquid nitrogen (LN2) as the cryogenic commodity.

7.1. Nominal Operation Test Case

7.1.1. Childown

During the chilldown phase, liquid nitrogen is used to decrease the temperature of the system to a cryogenic liquid nitrogen temperature range (-321 °F). During this phase, liquid nitrogen is transferred from the three storage tanks to the vehicle interface valve. Boil-off gas generated during the cool down process is relieved through valves that are redirected to the exhaust line on the system. Once the main tank downstream temperature sensor (TT-1-O2 from Figure **7**) reaches cryogenic temperature, the main block valve (MAV-1-O2 from Figure **7**) is opened to allow cryogenic flow across the piping system from storage tanks up the vehicle interface (see Figure 10).

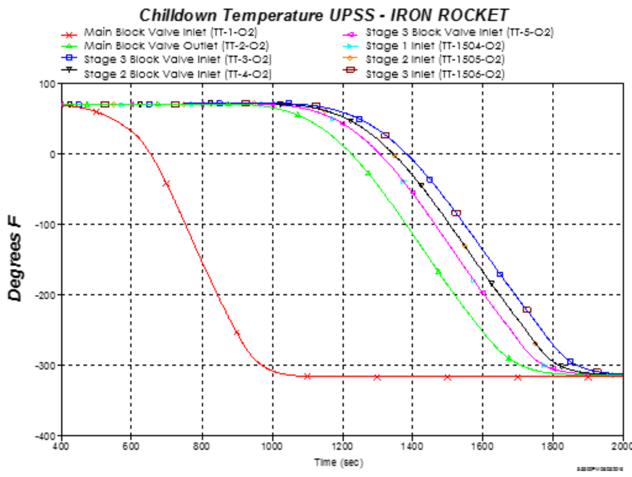


Figure 10. Chilldown Temperature: UPSS - Iron Rocket

7.1.2. Slow-Fast Fill

During this phase, the ullage pressure on the cryogenic storage tank is increased by means of a Pressure Building Unit (PBU) up to 50 psig to pressure-feed the system and increase the flow rate to the Stage 1 vehicle tank. The fill process for the vehicle tanks is a serial process. Once the Stage 1 is up to 100% full (see Figure 11), the other tanks follow the fill process (starting from Stage 2 followed by Stage 3) and Stage 1 enters into a replenish state.

7.1.3. Replenish

During this phase, a maximum tank fill level is achieved by the cryogenic commodity and a replenish algorithm is enabled (see Figure 12). This replenish algorithm monitors the Stage 1 tank level and commands a replenish valve (AMAV-3-O2 from Figure **7**). This replenish valve is controlled by a replenish algorithm that is executed by the Sequencer. This algorithm commands the replenish valve to open to 50% once the liquid level sensor (LLT-1604-O2) falls below 99.5%. This decrease in liquid level is due to boil-off of cryogenic commodity being excessed across the vent valve of the vehicle tank.

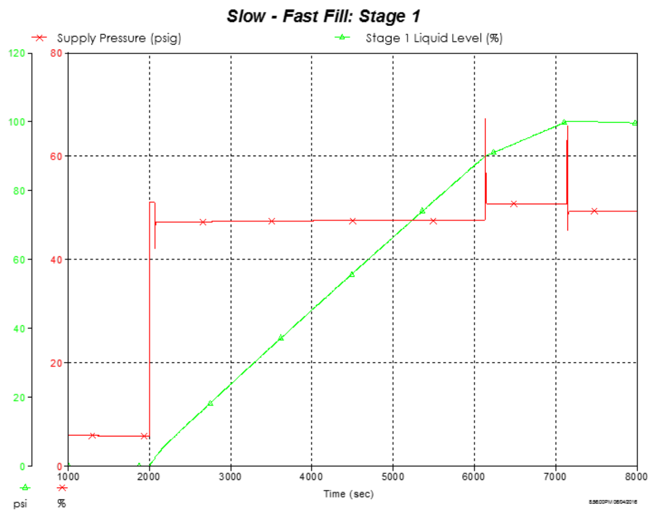


Figure 11. Slow-Fast Fill: Stage 1

Opening the replenish valve allows more cryogenic commodity into the vehicle tank. On the other side of the spectrum, if the liquid level surpasses beyond 100.5%, the replenish valve is commanded to close (0%). Over time, the loss in mass due to the venting of boil-off gases produces a decrease in liquid level of the vehicle tank and the replenish algorithm enters a new cycle. During nominal operations, the vehicle tank for Stage 1 undergoes for 1 cycle of the replenish state.

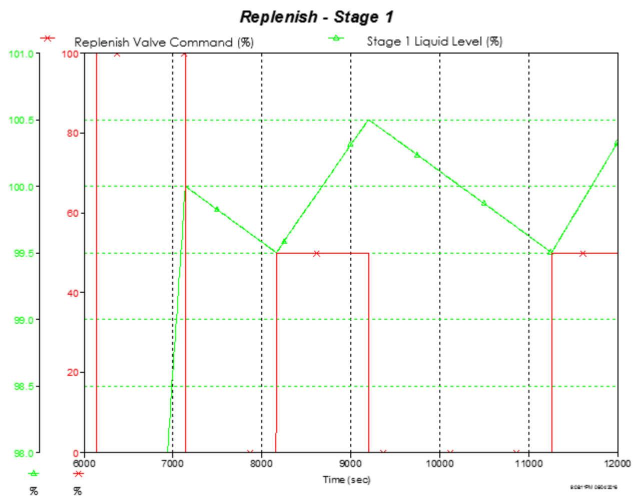


Figure 12. Replenish - Stage 1

This cyclic process is dependent on the conditions for the nominal operations. This process mimics a real launch operation where a real vehicle might stay in the launching platform for hours while maintaining liquid level conditions until vehicle launch. A typical launch vehicle undergoes several cycles to maintain required launch conditions while other parallel operations take place in preparation for launch.

7.2. Serial Loading Vehicle Stage 1, 2, and 3

Similar to stage 1, the cryogenic propellant loading process for the stage 2 and stage 3 transitions to a replenish state once the liquid level for both tanks reaches 100%. In a serial manner, the stage 2 reaches 100% liquid level after the stage 1. Once on the replenish state, stage 2 executes 7 cycles during the nominal loading process. The difference in the amount of cycles is due to the volumetric capacity of the tanks: Stage 1, 2 and 3 have a capacity of 3000, 500 and 500 gallons respectively. The difference on specific heat for the 3 cryogenic tanks produces a difference in boil-off rate. For the smaller tanks (stage 2, 3), a higher boil-off rate is produced which generates a faster decrease in liquid level. The Sequencer commands the replenish valve for all the stages in a parallel execution algorithm which is constantly monitoring all the liquid level sensors (LLT-1604-O2, LLT-1609-O2, and LLT-1614-O2).

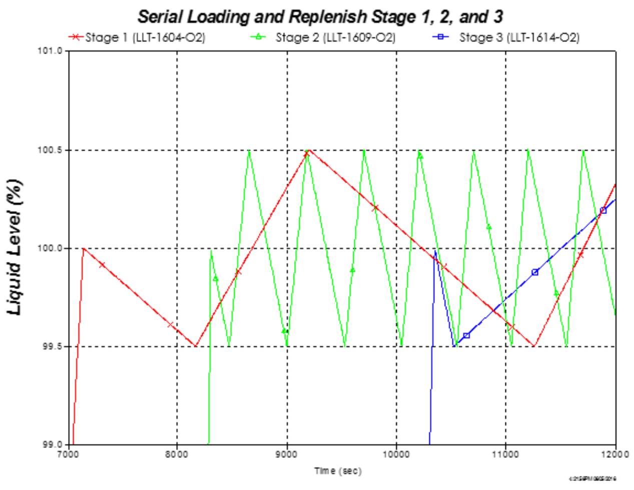


Figure 13. Serial Loading and Replenish for Stage 1, 2, and 3

7.3. Off-Nominal Operation Test Case

These tests will consist of a nominal loading operation followed by inserting preplanned anomalies into the simulation. Similar to previous off-nominal tests executed during the technology maturation of AOS in its first use on the Cryogenics Test Laboratory, common failure modes using a 3-stage loading sequence utilizing fully autonomous sequencing have been identified. As part of the off-nominal case scenarios for instrumentation-only failures, APL will identify the failure and allow the loading operation to continue; for failures requiring safing, APL will safe the system while identifying the fault condition to the operator. The simulator and system testing with UPSS/DTS will confirm these critical mitigation functions are working properly.

7.3.1. Non-safety Critical Failure: Instrumentation Failure

Non-safety critical failure can be categorized as hardware component failure that does not cause a catastrophic damage to the mechanical system or pose a risk to the mission. For instrumentation failure, the open indicator for the stage 1 inlet valve (MAV-1001-O2) is tested. This discrete open position indication is channelized with two different telemetry data objects. One signal provides a primary reading and another signal provides secondary (redundant) reading. The domain object elements have been modeled to correctly represent the physical behavior of a redundant signal system. In the real-hardware, the sensor is connected to two sets of PLCs. In this particular case, an instrumentation failure provides indication of a failure of one PLC. For this case, the primary reading was selected to fail by a signal loss action which produces a discrepancy in telemetry for the discrete position open indicator.

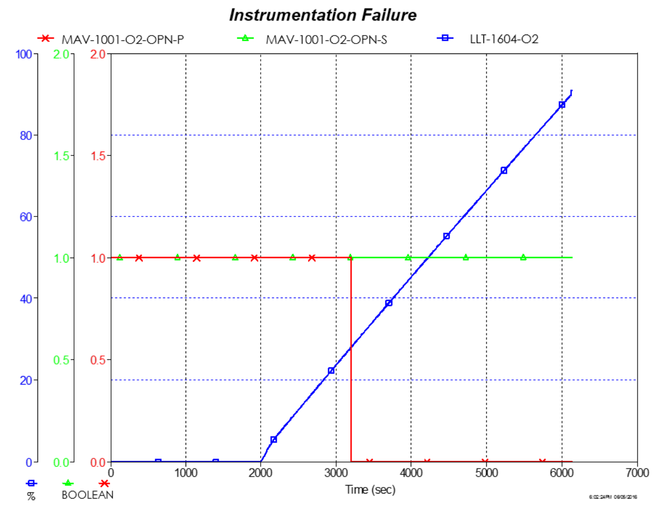
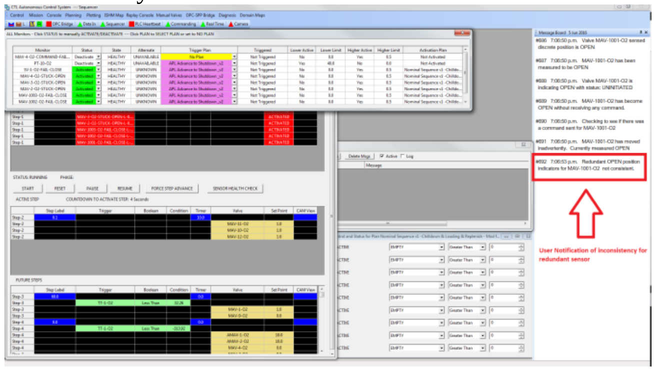


Figure 14. Instrumentation Failure. MAV-1001-O2-OPN-P: Primary Signal, MAV-1001-O2-OPN-S: Secondary Signal, LLT-1604-O2: Stage 1 Liquid Level Sensor

In Figure 14 the primary signal for the position indicator of MAV-1001-O2 shows an open state (Boolean Flag = 1.0). Similarly, the secondary signal for the position indicator initially shows an open state. During the loading of the stage 1 vehicle tank represented by LTT-1604-O2, the primary position indicator shows a close state (Boolean Flag = 0.0) while the secondary indicator shows an open state when the stage 1 tank is about 30% full.

The redundancy models, which are being executed on an asynchronous execution, are triggered once an event is detected. An evaluation of the telemetry produced several conclusions. First, the valve has moved inadvertently due to a change in the open indicator. Second, a redundant open position indicator for the valve in question is not consistent (see Figure 15). The mitigation procedure for this injected failure is to notify the operator of the instrumentation failure only and continue with nominal operations since instrumentation failure is being mitigated by a secondary sensor. The continuation of the nominal operations plan can be reflected on Figure 14 by observing that the stage 1 liquid level keeps increasing beyond 30% after the time segment (about 3200 seconds) that the open position telemetry inconsistency was found.


Message Board 5 Jun 2016

#686 7:06:50 p.m. Valve MAV-1001-O2 sensed discrete position is OPEN

#687 7:06:50 p.m. MAV-1001-O2 has been measured to be OPEN

#688 7:06:50 p.m. Valve MAV-1001-O2 is indicating OPEN with status: UNINITIATED

#689 7:06:50 p.m. MAV-1001-O2 has become OPEN without receiving any command.

#690 7:06:50 p.m. Checking to see if there was a command sent for MAV-1001-O2

#691 7:06:50 p.m. MAV-1001-O2 has moved inadvertently. Currently measured OPEN

#692 7:06:53 p.m. Redundant OPEN position indications for MAV-1001-O2 not consistent.

Figure 15. AOS response to instrumentation failure during nominal operations.

7.3.2. Safety Critical Failure: Main valve mechanical failure

Safety critical failures can be categorized as hardware component failures that cause catastrophic damage to the mechanical system and poses a risk to the mission if being operated during a nominal operation. For this failure, the main fill valve for stage 1 tank (MAV-4-O2) has been selected. MAV-4-O2 has several telemetry components. A valve command signal which received the desired valve command and executes the command, a command response telemetry which acknowledges that the command signal is being received by the PLC, an open indicator, and a closed indicator.

During a Failure Mode Effect Analysis (FMEA), MAV-4-O2 was identified as a critical component during the cryogenic propellant transfer operations. Several failure modes have been identified associated with this valve. The selected failure mode present in this paper is a mode where the valve fails to respond to a given command, which can cause a catastrophic failure during cryogenic loading operation.

For the selected failure mode, the valve is commanded to a closed position after being open during a fast fill phase. Several responses are expected after a close command for this type of valve:

1. PLC acknowledges that the command closed has been received
2. Open indicator shows a 0.0 value in telemetry. This open indicator is a limit switch that turns on or off once the valve stem travels up or down,

respectively. Open indicator equals 1.0 when the valve is open and 0.0 when valve is closed.

3. Closed indicator shows a 1.0 values in telemetry. This closed indicator is a limit switch that turns on or off once the valve stem travels down or up, respectively. Closed indicator equals 1.0 when the valve is closed and 0.0 when valve is open.

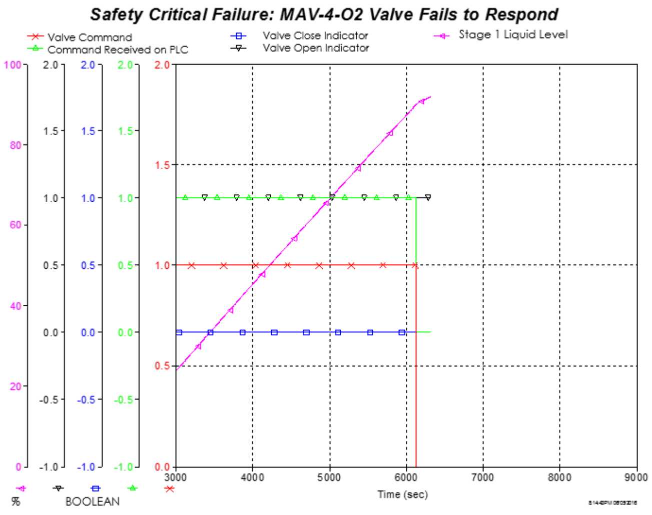


Figure 16. Safety critical failure for main fill stage 1 valve

In Figure 16 the valve in question is commanded closed at the end of the stage 1 fill phase. In nominal operations, after stage 1 reaches 100% on the liquid level sensor the main fill valves closes and the nominal plan transitions to a replenish state. In this particular scenario, the following telemetry is received after the main fill valve is commanded closed and a nominal valve travel time have elapsed:

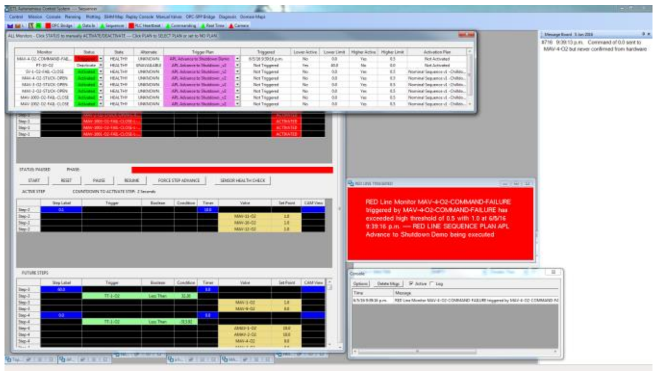
1. PLC acknowledges that the command ‘CLOSE’ has been received

2. Open indicator shows a 1.0 value in telemetry. This indicates the valve is open.

3. Close indicator shows a 0.0 values in telemetry. This indicates the valve is open.

Based on the failure analysis, the valve has failed to close and all telemetry indicates that even if the command was received by the PLC, the valve remains in its original position. In this case, if valve does not close, there exists a potential to overfill the stage and send liquid into the vent line which may result in a catastrophic failure.

This failure scenario has been modeled in AOS and a response to protect the system has been programed. Figure 17 shows the system response of AOS to the described failure. Once the failure is detected, the model triggers a flag which is being monitored by Redline Monitoring. The redline monitoring evaluates the received flag with the established threshold and triggers the redline. The redline violation generates an automated response to stop any current plan execution and proceeds to execute any mitigation plan associated with the redline monitor. For this particular case, the mitigation plan represents a series of commands to be executed with the purpose of shutting down the nominal operation and protecting the system from any catastrophic failure. In addition, the operator receives a notification that shows a brief description of the event, time of trigger and automated mitigation plan to be executed.



RED Line Monitor MAV-4-O2-COMMAND-FAILURE triggered by MAV-4-O2-COMMAND-FAILURE has exceeded high threshold of 0.5 with 1.0 at 6/5/16 9:39:16 p.m. — RED LINE SEQUENCE PLAN APL Advance to Shutdown Demo being executed

Figure 17. AOS response to safety critical failure of main stage 1 fill valve not responding to commands.

8. CONCLUSIONS

The AOS has proven to be a fully autonomous control software capable of integrating automated control with ISHM to command and control cryogenic propellant loading operations under a safe environment. The APL has given AOS the incentive to evolve from its previous version meant for laboratory only operations and expand its capabilities to encompass the necessary components and features needed to command and control ground support equipment (GSE) that is currently undergoing Class B Safety Critical certification for future application on flight hardware.

The complexity of GSE capable of supporting flight hardware increases the need to improve robustness on a command and control system. In addition, it motivates the development of a real-time response system capable of ensuring safe operations even on the unforeseen circumstances where a critical failure may occur.

AOS has made possible an increase in the NASA Technology Readiness Level (TRL) from validation in laboratory environment (Level 4) to validation in relevant environments (Level 5). Current efforts are aimed to increase the TRL to produce a product relevant to ground or space environment (Level 6) and generalize the capabilities to be applicable to a wide variety of industrial systems. In addition, modifications and redesign of AOS are aimed to grant a software safety critical Class B classification under a new software called Autonomous Operations Mission Development Suite (AO MDS).

ACKNOWLEDGMENTS

The authors of this paper will like to thank Mark Turowski and Justin Junell for providing a UPSS and Iron Rocket model and communications simulator used to validate operations performed with AOS. Recognition is due to all the people who worked in development and validation of the technology reported in this paper: John (Jay) Gurecki, Justin Youney, Johanna Johnson, Caylyne Shelton, Scott Ditto, David Moyer, Fernando Figueroa, and Robert Johnson.

Support for this work was provided by NASA Advanced Exploration Systems under the leadership of Jason Crusan and Richard McGuinnis.

NOMENCLATURE

ACS	Automated Control Sequencer
AOS	Autonomous Operations System
AO MDS	Autonomous Operations Mission Development Suite
APL	Autonomous Propellant Loading
CIP	Common Industrial Protocol
CLF	Cryogenic Loading Facility
COTS	Commercial-Off-The-Shelf
CSCI	Computer Software Configuration Item
CTL	Cryogenics Test Laboratory
CUI	Compact Unique Identifier
DAQ	Data Acquisition
DTS	Development Test Site
FMEA	Failure Mode Effect Analysis
GHe	Gaseous Helium
GN2	Gaseous Nitrogen
GSE	Ground Support Equipment
GSI	G2 Gateway Standard Interface
GUIs	Graphical User Interfaces
IDE	Integrated Development Environment
ISHM	Integrated System Health Management
KB	Knowledge Base
LCH4	Liquid Methane
LN2	Liquid Nitrogen
LO2	Liquid Oxygen
MPCDU	Mobile Power and Communication Distribution Unit
PHM	Prognostics and Health Monitoring
PLC	Programmable Logic Controller
SCV	Small Class Vehicle
SPP	Space Packet Protocol
SSME	Space Shuttle Main Engine
TCP/IP	Transmission Control Protocol/Internet Protocol
TRL	NASA Technology Readiness Level
UDP	User Datagram Protocol
UPSS	Universal Propellant Servicing System

REFERENCES

Davidson M., Stephens J., (2004) *Advanced Health Management System for the Space Shuttle Main Engine*. 40th AIAA/ASME/SAE/ASEE Join Propulsion Conference and Exhibit 11-14 July 2004, Fort Lauderdale, Florida.

Walker M., (2010). *Next generation prognostics and health management for unmanned aircraft*. IEEE Aerospace Conference, 6-13 March 2010 Big Sky, Montana.

Figueroa F., Melcher K. (2001). *Integrated Systems Health Management for Intelligent Systems*. Infotech@Aerospace 2011, 29-31 March 2011, St. Louis Missouri.

Antsaklis P. J., Passino K. M., Wang S. J., (1991). *An Introduction to Autonomous Control Systems*. IEEE control systems 11(4)

Rafeed Leon H. M., Shoeb A., Rahman S., Ahmed M. U., Islam S., (2016). *Design and economic feasibility analysis of autonomous hybrid energy system for rural Bangladesh*. 2016 4th International Conference on the Development in the Renewable Energy Technology (ICDRET) 7-9 Jan. 2016, Dhaka

Venkatesh, M., Kapadia R., Walker M., Wilkins K., (2013). Integrated Systems Health Mangement (ISHM) Toolkit. NASA Tech Briefs, January 2013; 23.