Ground Operations Autonomous Control and Integrated Health Management

James Daniels
NASA Kennedy Space Center
Computer Engineering
KSC FO Summer Session
29 07 2014

Ground Operations Autonomous Control and Integrated Health Management

James Daniels[1]
Jackson State University – Jackson Campus, Jackson, MS 39217

Nomenclature

COTS	=	commercial-off-the-shelf
CPU	=	central processing unit
CV	=	control valve
EU	=	engineering units
G2	=	Application Software
HMI	=	human-machine interface
ISHM	=	Integrated System Health Management
I/O	=	input/output
KGCS	=	Kennedy Ground Control System
LVDT	=	Linear Variable Differential Transformer
MCC	=	Motor Control Cabinet
ODBC	=	Open Database Connectivity
PC	=	personal computer
PLC	=	programmable logic controller
RIO	=	remote input/output
SCADA	=	supervisory control and data acquisition
SPLS	=	Simulated Propellant Loading System

I. Abstract

The Ground Operations Autonomous Control and Integrated Health Management plays a key role for future ground operations at NASA. The software that is integrated into this system is called G2 2011 Gensym. The purpose of this report is to describe the Ground Operations Autonomous Control and Integrated Health Management with the use of the G2 Gensym software and the G2 NASA toolkit for Integrated System Health Management (ISHM) which is a Computer Software Configuration Item (CSCI). The decision rationale for the use of the G2 platform is to develop a modular capability for ISHM and AC. Toolkit modules include knowledge bases that are generic and can be applied in any application domain module. That way, there's a maximization of reusability, maintainability, and systematic evolution, portability, and scalability. Engine modules are generic, while application modules represent the domain model of a specific application. Furthermore, the NASA toolkit, developed since 2006 (a set of modules), makes it possible to create application domain models quickly, using pre-defined objects that include sensors and components libraries for typical fluid, electrical, and mechanical systems.

II. Introduction

The purpose of the software, the ISHM, is to provide information on the health of every element of the system, such as sensors, actuators, pipes, tanks, valves, etc. The Autonomous Control (AC), which shall be carried out via the G2 software application, executes control sequences after evaluating conditions, including health conditions of system elements involved in sequence executions. The reason for the usage of this software is the integration of Autonomous Control for integrated control, monitoring, and processing of system elements. The objective of this software is for the G2 System, alongside with the Allen-Bradley (AB) Programmable Logic Controller (PLC) system, to perform monitoring, health management, and autonomous control. This includes the cryogenic system being commanded from G2 or AB user interfaces, to issue higher level commanding (autonomy) that resides in the G2 system, and to systematically increment the levels of autonomy in order to achieve operational status in a short time.

[1] 2014 Summer Intern, System Hardware Engineering Branch (NE-C4), NASA Kennedy Space Center, Jackson State University.

III. Assumptions, Constraints, and Project Deliverables

The assumptions on which the software project is based are from the utilization of the ISHM toolkit, which is the framework developed for using the G2 application to monitor cryogenics systems. This toolkit will be applied to achieve the desired capabilities at the Cryogenics Test Laboratory.

The product for this project will deliver the G2 software application with the ISHM toolkit to perform autonomous control of the Liquid Nitrogen (LN2) loading operations in the Cryogenics Test Laboratory. A series of tests will be conducted using the same configuration and automated sequence to demonstrate repeatability of the loading process from chilldown through securing operations. Environmental conditions and the initial storage tank liquid level may vary from test to test. This test request will be used for multiple tests. The test number for each test will have a letter (A, B, C, etc.) after the test number in the test number field, and the date code will be updated to reflect the date that each test is conducted. The test data will all be stored in the same directory in Share Point. The replenish duration is set for 30 minutes. At least two tests will be conducted with a full 30 minute replenish duration. A shorter replenish duration, may be used to manually shorten tests if it is necessary to preserve sufficient storage tank liquid level for the next test if liquid delivery is not scheduled. Changes to the automated sequencer will be indicated in red in the sequence description tables.

IV. Software Design Description

The reason for the utilization of the G2 software platform and the NASA ISHM Toolkit is to demonstrate rapid domain model development, systematic augmentation of autonomy with the application of new knowledge, rapid instantiation of autonomy, quick access to explore any part of the system without the need to shuffle through printed schematics, and to demonstrate autonomous Failure Modes and Effects Analysis (FMEA) capability that is applied within the proper context provided by knowledge embedded in the system and delivered to it from physical system components.

V. Software Item Input/Output Description

A. Autonomous Control and ISHM

Autonomous control is achieved through execution of sequence plans by the Sequence Engine. Sequence Plans are created utilizing a menu based window, and are validated using the Sensor Simulator. The Sensor Simulator enables verification of a sequence plan. The Sequence Plan is a sequence of steps that is to be carried out autonomously. 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. The Sequencer Engine (and its Lock) is the software engine that executes the sequence plans.

B. Domain Map (and its Lock)

The 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 (DIaK) about the system. The lock is used to stop real time data and to use historical data or simulated data.

C. Analysis Graphics & Monitors

Analysis Graphics is the component that provides the ability to plot any health signal from the system over time and frequency. Multiple monitors can be defined for any member of a sensor class, including logical sensors. Monitors may be classified as redline monitors, and used as such by sequence plans.

D. Integrated System Health Management Root Cause Analysis

This component includes root-cause-trees (or functional directed graphs) that describe failure modes and effects and make possible diagnostics and prognostics (through FMEA). FMEA are programmed graphically and incorporate context that is part of the application domain model.

VI. Concept of Execution for the ISHM toolkit

The Knowledge Domain Model (KDM) of the AC-ISHM encapsulates complete information and knowledge about the system. The three clients to the KDM are the Testbed client, the Graphics client, and the Sequencer client.

The KDM shall execute in two modes of operation. In the Real-time operation mode, the KDM can monitor health, execute manual user commands, navigate to elements in subsystems that are not visible on the top screens, and execute autonomous commands in sequence. In the Replay operation mode, the KDM can replay a mission exactly as it happened along with the ability to speed up, stop, and explore throughout the domain model in order to examine what was happening in any part of the testbed. Also in this mode, the KDM can operate as if in real-time operation (using historical test data from a file), analyze and develop autonomy even further, and execute exploratory activities.

VII. Requirements Used for the G2 Software Selection

A. Object Orientation Requirement

Object representation of system physical elements and associated process models is the best way to embed DIaK in a systematic and in an organized manner. Object orientation also embodies re-use of software that is modularized into objects and allows a more intuitive understanding of the code and its outcomes.

B. Distribution of Domain Models Within/Across Networks Requirement

Domain Models might be distributed among processors connected to a network, simply because it is necessary to use parallel processing, and/or Domain Models might be created by different people in various geographic locations. As complexity of systems increase, and/or a large number of process models are used in achieving effective ISHM-AC capability, it is not reasonable or manageable to do this with a centralized architecture.

C. Distribution across Multiple Platforms Running Simultaneously Requirement

Since multiple domain models are expected to be active at any given time, the software environments should support execution of multiple copies of the platform with a natural integrating mechanism (object to object communication).

D. Inference Engine Requirement

Many tasks require an inference engine. Reasoning and decision-making leading to anomaly detection, diagnostics, effects, prognostics, and autonomy; require contextual integrity and cause-effect analysis using heterogeneous data and information. The inference engine must allow accurate representation and automation of failure modes and effects analysis (FMEA).

E. Integrated Management of Distributed Data Information and Knowledge (DIaK) Requirement

DIaK must be managed in a way to allow embodiment of systems thinking across elements and subsystems. Often this is enabled by definitions of relationships among elements of systems that can be physically visible (i.e. attached to, belong to a system); or more abstracted relationships, as it relates to involvement in process models (e.g. pressure sensors associated to a particular subsystem, subsystem definitions that change with configuration, etc.).

F. Definition of Dynamic Relationships among Objects for Use in Reasoning Requirement

Often, the framework for reasoning and application of process models changes dynamically with configuration changes, stages of operation, etc. This also means that relationships among objects and processes change dynamically, and must be represented in the domain models. For example, reasoning to detect leaks in a sealed subsystem requires that membership of elements to sealed subsystems must change with valve state changes.

G. Iconic Representation of Systems Objects Requirement

The mix of object orientation and iconic representation of DIaK provides the ability to intuitively visualize interrelationships and dig deep into details of the ISHM-AC system. As complexity increases, graphical programming and visualization become essential.

H. Operation in Real-Time Requirement

The platform must incorporate the ability to operate in real-time, and make inferences and decisions where time, models, procedures, and DIaK in the domain model are considered. Platform reliability, maintainability, and updatability are needed; A COTS platform is desirable, in particular if it is an established platform with a record of successful applications, robustness, consistent upgrades, and customer support.

I. Integration with Sources of Data Requirement

The availability of standard network gateways connecting to systems that provides data to populate the domain models and make possible real time ISHM-AC capability.

VIII. Interface Design & Use of Computer Resources

The Ground Operations Autonomous Control and Integrated Health Management System are interfaced to the Allen Bradley PLC Control System through a G2 OPC Bridge and a Kepware OPC Server that communicates with the Allen-Bradley PLC Control System.

The ISHM system computer hardware resources include monitors for graphical user interface (PC desktop running Windows 7), administrator panels, G2 User workspaces, the G2 OPC Bridge, Kepware OPC Server, Allen-Bradley PLC, real physical valves, real physical sensors, real physical pumps, real physical tanks.

IX. Conclusion

G2 helps to develop an application for ISHM and Autonomous control (ISHM-AC) capability at the Cryogenics Testbed Laboratory (CTL). This application uses a toolkit to implement ISHM capability that was developed at the Stennis Space Center in prior years. This toolkit includes an ISHM engine that provides ISHM functions (anomaly detection, diagnostics, and user interfaces). A new Autonomous Control Engine was developed in order to support the AC functions (creation and execution of sequences and autonomous decision-making when a sensor is determined to have failed). The ISHM engine and the AC engine are reusable and can be applied to any application domain model (e.g. CTL G2 Domain Model).

X. Acknowledgments

I would like to especially thank my mentor, Kelvin Ruiz. I want to thank the System Hardware Engineering Branch: Mr. Vo, Mr. Le, Mrs. Potter, Mr. Svedin, Mr. Niev, Mr. Bond, Mr. James, Mr. McDonough, Mr. Villorin, and Mr. Vickers. I appreciate the experience of being a part of a great team at the KSC.

References

Veillard, Daniel. SymCure User's Guide. Vers. 5.1. Burlington, MA: Gensym Corporation, 2007. Program documentation.

Veillard, Daniel. G2 Reference Manual. Vers. 8.4. Burlington, MA: Gensym Corporation, 2009. Program documentation.

NPR 7150.2, NASA Software Engineering Requirements.

KNPR 7150.2, KSC – NASA Software Engineering Requirements.

EPG-T-3922A, Software Design Description Instruction.

K0000131648, Cryogenics Lab Control System Eletrical Schematic.

700EFW00002, Human Machine Interface (HMI) Programming Guidelines (KGCS).