NASA/TM—2009-215442

Sensor Data Qualification System (SDQS) Implementation Study

Edmond Wong
Glenn Research Center, Cleveland, Ohio

Christopher Fulton and William Maul
Analex Corporation, Cleveland, Ohio

Kevin Melcher
Glenn Research Center, Cleveland, Ohio

April 2009

NASA STI Program . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI Program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NASA Aeronautics and Space Database and its public interface, the NASA Technical Reports Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or cosponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include creating custom thesauri, building customized databases, organizing and publishing research results.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at *http://www.sti.nasa.gov*

- E-mail your question via the Internet to *help@sti.nasa.gov*

- Fax your question to the NASA STI Help Desk at 301–621–0134

- Telephone the NASA STI Help Desk at 301–621–0390

- Write to:
  NASA Center for AeroSpace Information (CASI)
  7115 Standard Drive
  Hanover, MD 21076–1320

NASA/TM—2009-215442

Sensor Data Qualification System (SDQS) Implementation Study

Edmond Wong
Glenn Research Center, Cleveland, Ohio

Christopher Fulton and William Maul
Analex Corporation, Cleveland, Ohio

Kevin Melcher
Glenn Research Center, Cleveland, Ohio

Prepared for the
International Conference on Prognostics and Health Management 2008 (PHM08)
sponsored by the IEEE Reliability Society
Denver, Colorado, October 6–9, 2008

National Aeronautics and
Space Administration

Glenn Research Center
Cleveland, Ohio 44135

April 2009

Level of Review: This material has been technically reviewed by technical management.

Available from

NASA Center for Aerospace Information
7115 Standard Drive
Hanover, MD 21076–1320

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161

Available electronically at http://gltrs.grc.nasa.gov

Sensor Data Qualification System (SDQS) Implementation Study

Edmond Wong
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

Christopher Fulton and William Maul
Analex Corporation
Cleveland, Ohio 44135

Kevin Melcher
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

Abstract

The Sensor Data Qualification System (SDQS) is being developed to provide a sensor fault detection capability for NASA's next-generation launch vehicles. In addition to traditional data qualification techniques (such as limit checks, rate-of-change checks and hardware redundancy checks), SDQS can provide augmented capability through additional techniques that exploit analytical redundancy relationships to enable faster and more sensitive sensor fault detection. This paper documents the results of a study that was conducted to determine the best approach for implementing a SDQS network configuration that spans multiple subsystems, similar to those that may be implemented on future vehicles. The best approach is defined as one that most minimizes computational resource requirements without impacting the detection of sensor failures.

I. Introduction

At NASA Glenn Research Center (GRC), researchers are currently developing a Sensor Data Qualification System (SDQS) for NASA's next-generation launch vehicles. SDQS is being considered for incorporation into the vehicle's avionics system to provide the capability to monitor and detect faulty sensor data. In addition to implementing traditional data qualification techniques such as limit checks, rate-of-change checks and hardware redundancy checks, SDQS can provide augmented capability through additional techniques that exploit analytical redundancy relationships existing within the system's sensors to enable the detection of faulty sensor information with greater speed and sensitivity.

The potential implementation of SDQS in future launch vehicles prompted an investigation to gain insight into how different practical implementation decisions may impact the data qualification software in terms of operation, effectiveness, resource utilization and robustness. The resulting information will help flight software developers select the best approach for in-flight implementation of the SDQS algorithms. This paper describes the study approach, test-beds, test set-up and results that were used to determine the preferred architecture implementation for the SDQS system.

II. Test Study Approach

Fault detection and recovery functions in NASA's future launch vehicles will require health and status data from the vehicles' subsystems in order to properly detect and confirm sensor data used to identify abort situations. These data sources need to be qualified to ensure that the response provided by the FDNR is based upon sound information and not influenced by failed sensors. Conventional data qualification techniques (ref. 1), such as limit checks will be utilized to remove gross sensor failures. However, if subtle sensor faults (e.g., drifts) need to be filtered from the FDNR processing, then redundancy networks will be needed to detect these types of faults. Additionally, SDQS may be required to monitor data spanning several unique subsystems. A variety of implementation questions arise from this requirement:

1. Is there a performance advantage to structuring the higher-level SDQS as a single entity processing multiple distinct data qualification networks or should there be separate, independent SDQS networks?

2. What are the real-time performance and resource requirements of the SDQS run-time architecture for cases where (a) multiple networks are run independently and (b) multiple networks are combined into a single large network?

3. How robustly does SDQS handle the processing of real-world stress conditions such as multiple consecutive sensor failures and how gracefully does its capabilities degrade with the loss of sensor information?

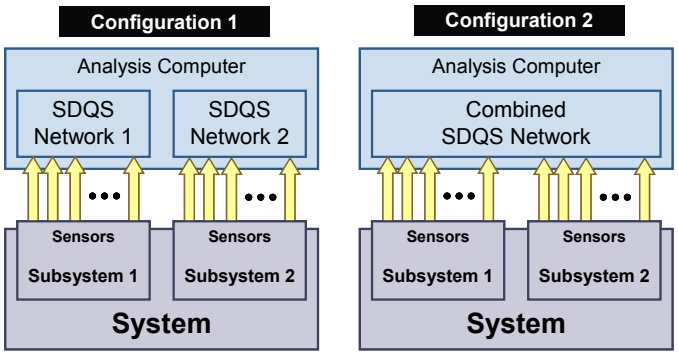

Figure 1.—Two different SDQS configurations.

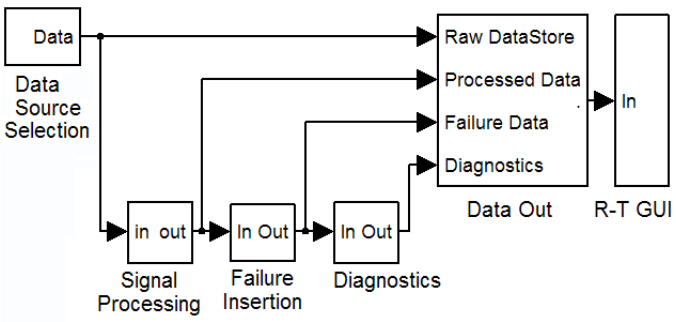

Figure 2.—Simulink test model.

To address these questions, two different SDQS network configurations were conceived, tested and compared for a subject system comprised of two independent subsystems. The first configuration is comprised of two separate, individual SDQS networks running sequentially with each dedicated to processing the sensor's inputs for only one of the subsystems. The second configuration consists of a single SDQS network that processes the combined sensor inputs for both subsystems simultaneously. These two configurations are illustrated in figure 1. Testing was conducted in three separate series of tests. Each test series addressed one of the previously stated study goals and is discussed in detail in subsequent sections of the paper.

A. Diagnostic Algorithms

GRC researchers have proposed and demonstrated an advanced sensor diagnostic approach that augments current state-of-the-art threshold limit checking algorithms with analytical redundancy techniques and a variety of statistical methods to enable more sensitive and timely detection of sensor failures. Central to the development of these advanced data quality algorithms is the SureSense Data Quality Validation Studio (DQVS), a commercial software product developed by Expert Microsystems Inc. in cooperation with NASA Glenn Research Center under a Small Business Innovative Research contract. DQVS is a data qualification development and analysis package that includes a capability for auto-generating run-time application modules that can be executed on target systems. GRC researchers have used DQVS to develop sensor data qualification algorithms and have conducted proof-of-concept testing in real-time using both software and hardware simulated sensor failures (ref. 2).

For the purposes of this study, DQVS was used to build and generate two different network model configurations for a subject test-bed system (fig. 1). Rather than identify a completely new test-bed system and develop new SDQS networks for both the combined and individual network configurations, it was deemed satisfactory for the purposes of this study, to use data from two previous demonstrations. By treating the two individual existing systems as subsystems of a hypothetical larger system, a SDQS network was developed that simultaneously accommodates the combined total number of sensors found in both systems.

To conduct this study, SDQS networks for each of the individual systems and for the combined system were exported from the SureSense DQVS environment to 'C' code. The 'C' code was subsequently integrated into GRC's Portable Health ALgorithms Test (PHALT) system (ref. 3), which is a diagnostic algorithm testing platform based on the Matlab/Simulink environment. Figure 2 shows the Simulink functional block model employed by the PHALT system and is comprised of blocks that handle various functions such as data input, data storage, signal preconditioning, customized failure insertion and diagnostics. The diagnostics functional block is where the SDQS 'C' Code is embedded to enable test data to be applied to the SDQS diagnostic algorithms.

The SDQS 'C' code was also used with the VxWorks compiler/operating system to generate real-time code that was run on a flight-like single-board computer in order to test and analyze the real-time performance.

B. Test-Beds

The first test-bed used in this study's subject subsystems is a prototype Electrical Power System (EPS) Power Distribution Unit (PDU) (ref. 4) test-bed that is representative of the type of system that may be featured on future launch vehicles. This test-bed was constructed at GRC and hardware-in-the-loop testing was subsequently conducted with a DQVS-based SDQS in the fall of 2006. Figure 3 shows a schematic of the test-bed and table 1 shows a list of the active sensors.

The second test-bed is the GRC Cryogenic Component Laboratory 7 (CCL–7) test-bed. CCL–7 is a cryogenic fluid transfer test facility designed to demonstrate liquid level measurement and to characterize propellant management devices within a cryogenic environment. Figure 4 shows a schematic of the CCL–7 test-bed and table 2 shows a list of the six applicable sensors used to develop the SDQS for this subsystem.

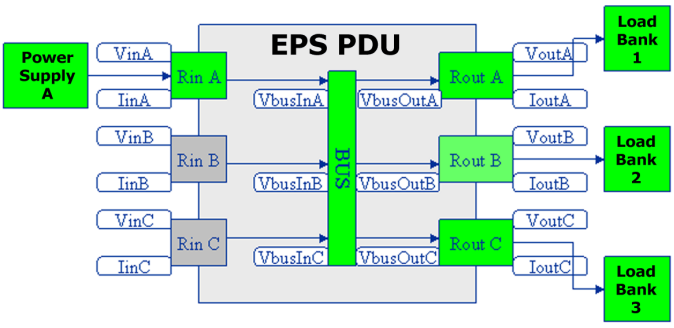

Figure 3.—Schematic of EPS PDU test-bed.

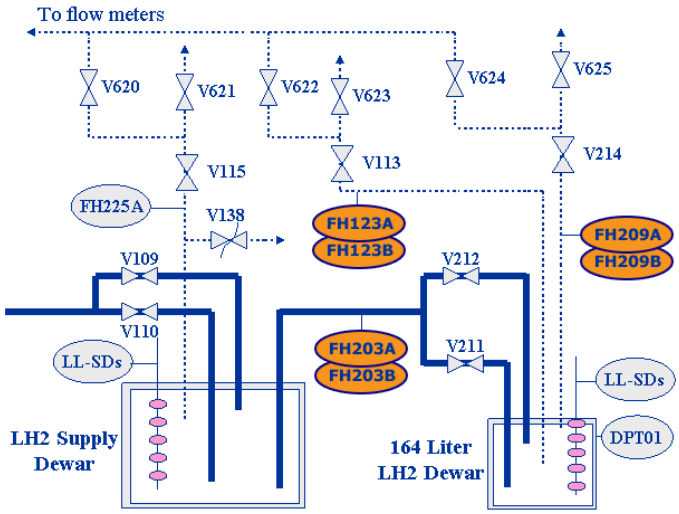

Figure 4.—Schematic of Cryogenic Component Lab test-bed.

TABLE 1.—EPS PDU TEST BED AVAILABLE SENSORS

Sensor ID	Description
VinA	Power Input Relay A Voltage
VoutA	Power Output Relay A Voltage
VoutB	Power Output Relay B Voltage
VoutC	Power Output Relay C Voltage
IinA	Power Input Relay A Current
IoutA	Power Output Relay A Current
IoutB	Power Output Relay A Current
IoutC	Power Output Relay A Current

TABLE 2.—CCL–7 TEST-BED SENSOR LIST

Sensor ID	Description
FH203A	Small Dewar tank supply line pressure—Channel A
FH203B	Small Dewar tank supply line pressure—Channel B
FH209A	Small Dewar tank vent line pressure—Channel A
FH209B	Small Dewar tank vent line pressure—Channel B
FH123A	Small Dewar tank vent line pressure—Channel A
FH123B	Small Dewar tank vent line pressure—Channel B

III. SDQS Configuration-Based Performance Tests

A. Set-Up

This series of tests were used to determine whether or not variations in the configuration of SDQS networks will have an impact on the system's resulting sensor failure detection capabilities. Specifically, the operation of a single SDQS network processing the combined EPS/CCL–7 system was compared with that of separate instances of the SDQS network each operating individually on the EPS and CCL–7 subsystems. To meet this objective, fault detection performance results for the combined EPS/CCL–7 SDQS network were obtained and compared to fault detection results for the two individual EPS and the CCL–7 networks achieved in previous concept demonstration tests. In these earlier tests, sets of input data containing predefined failure scenarios were developed to demonstrate the individual operation of the EPS and CCL–7 networks, respectively. To ensure a relevant comparison, selectively chosen subsets of these same two sets of fault scenarios were used to test the combined network. In this manner, the same sensors that were failed previously in individual testing were failed again for the combined testing. Tables 3 and 4 show the select subsets of fault scenarios chosen for the respective EPS and CCL–7 systems. Each individual scenario from the EPS subset was then paired up with a scenario from the CCL–7 subset to define a new set of combined fault scenarios (as shown in table 5) for simultaneous execution on the combined network.

TABLE 3.—EPS PDU DEMONSTRATION FAULT SCENARIOS SELECTED

Fault name	Sensor failed	Failure Type
M3_01	VoutB	Intermittent—binary mode
M3_02	IoutB	Intermittent—–binary mode
M3_04	VoutB	Intermittent—filtered mode
M3_05	IoutB	Intermittent—filtered mode
M3_07	VoutC	Hard low
M3_09	VinA	Hard high
M3_11	IoutA	Drift—low rate (–0.0043 A/sec)
M3_12	IoutA	Drift—med rate (–0.0067 A/sec)
M3_13	IoutA	Drift—high rate (–0.0167 A/sec)

TABLE 4.—CCL–7 DEMONSTRATION FAULT SCENARIOS SELECTED

Fault name	Sensor failed	Failure type
Fault_01	FH203A	Hard fault (high to 200)
Fault_02	FH123B	Hard fault (low to 0)
Fault_03	FH209A	Drift (0.001/sec)
Fault_16	FH209A	Drift (–0.001/sec)
Fault_06	FH203B	Intermittent binary (0)
Fault_25	FH123B	Intermittent binary (0)
Fault_07	FH123A	Intermittent filtered
Fault_20	FH203A	Intermittent filtered
Fault_13	FH123B	Excessive noise (6.0)

TABLE 5.—SDQS CONFIGURATION-BASED PERFORMANCE TESTS

Test ID	EPS PDU scenario	CCL–7 scenario
P1_01	Nom_Mode_3	Nom_01
P1_02	M3_01	Fault_06
P1_03	M3_02	Fault_18
P1_04	M3_04	Fault_07
P1_05	M3_05	Fault_20
P1_06	M3_07	Fault_02
P1_07	M3_09	Fault_01
P1_08	M3_11	Fault_03
P1_09	M3_12	Fault_16
P1_10	M3_13	Fault_13

B. Analysis

All testing for this phase was performed in accordance with the ten test cases described in table 5. For each test case, the block containing the combined EPS/CCL–7 network stored the results in log files. Analysis of the log files for each test case entailed reviewing the time that it took the combined network to detect the sensor failures (time-to-detect) for each of the subsystems and comparing those values to the corresponding results in the log files previously generated in past testing of the separate EPS and CCL–7 networks. The results of this analysis are summarized in table 6 for the EPS results and table 7 for the CCL–7 results. Since the subsystems considered in this paper have no interconnections, it can be expected that the combined network will not provided any improvements in performance. However, it is important to confirm that there will not be any negative impact on performance as well.

In table 6, each test case is listed along with the corresponding failure scenario used to test the EPS portion of the combined network as well as the name of the affected sensor. The corresponding time-to-detect values for the EPS portion of the combined network are also shown.

TABLE 6.—SDQS TESTING RESULTS FOR EPS PORTION OF COMBINED NETWORK

Test ID	Scenario	Sensor failed	Combined network's EPS time-to-detect (time samples)
P1_01	Nominal	N/A	N/A
P1_02	M3_01	VoutB	2
P1_03	M3_02	IoutB	2
P1_04	M3_04	VoutB	2
P1_05	M3_05	IoutB	2
P1_06	M3_07	VoutC	2
P1_07	M3_09	VinA	2
P1_08	M3_11	IoutA	5691
P1_09	M3_12	IoutA	3743
P1_10	M3_13	IoutA	830

TABLE 7.—SDQS TESTING RESULTS FOR CCL–7 PORTION OF COMBINED NETWORK

Test ID	Scenario	Sensor failed	Combined network's CCL–7 time to detect (time samples)
P1_01	Nominal	N/A	N/A
P1_02	Fault_06	FH203B	2
P1_03	Fault_25	FH123B	2
P1_04	Fault_07	FH123A	222
P1_05	Fault_20	FH203A	2
P1_06	Fault_02	FH123B	2
P1_07	Fault_01	FH203A	2
P1_08	Fault_03	FH209A	876
P1_09	Fault_16	FH209A	1007
P1_10	Fault_13	FH123B	566

The first test case, P1_01, employs a nominal data set featuring no sensor faults to ensure that the combined network operates without introducing false detections on nominal data. Indeed, this outcome proved to be the case as no sensor failures were diagnosed by the combined network for this test run. The results for the remaining test cases show that the combined SDQS network was able to perform accurate fault detection for the EPS portion of the sensors. In fact, the time-to-detect values matched exactly to the values for the standalone EPS network (obtained in previous demonstration testing).

Table 7 lists the test results for the CCL–7 portion of the combined network and shows the fault scenario used in each test as well as the name of the affected sensor. The time-to-detect values for the CCL–7 portion of the combined network are also shown.

As with the EPS subsystem, a nominal data set, featuring no sensor faults, was executed on the CCL–7 portion of the combined network in test case p1_01 to ensure that the combined network did not introduce any false failure detections. The results for the remaining test cases show that the combined SDQS network was able to perform accurate fault detection for the CCL–7 related sensors with time-to-detect values that matched exactly with those obtained by the standalone CCL–7 network (obtained in previous demonstration testing).

Consequently, for the case where the intended application spans multiple subsystems that are completely independent of each other with no interconnections, the results show that the decision to implement SDQS as either a single network processing multiple distinct subsystems or as separate, independent networks has no impact on its ability to perform accurate and timely sensor failure detection. This finding is significant since it points out the fact that SDQS affords the system designer a degree of flexibility in how sensor qualification networks may be implemented for a given application system without concern for impacting its effectiveness. As a result, it is recommended that the decision on SDQS configu-

ration can be made based on other study findings such as those discussed in the following section.

IV. Real-Time Performance and Resource Utilization Tests

A. Set-Up

This series of tests was concerned with real-time operation of the SDQS networks and compared the respective execution times and computer resource usage between the combined EPS/CCL–7 network and the separate networks. The objective was to determine if one configuration or the other would yield an advantage in terms of minimizing CPU execution time or memory usage.

The testing for this phase used the same tests defined for the SDQS Configuration-Based Performance Tests in section III (table 5) with the exception that the execution was performed on flight-like hardware. The same models used previously were exported to 'C' from the DQVS interface and compiled as part of a VxWorks project and downloaded to the Radstone PPC4A–750 VME Single Board Computer. VxWorks functions were utilized in the collection of the results.

B. Analysis

CPU usage.—The code used to determine the CPU usage was adapted from code used for CPU testing during the Propulsion IVHM Technology EXperiment (PITEX) (refs. 5 and 6) project. The associated VxWorks kernel was compiled using the GNU compiler set for optimization level 3 with logging turned off, in order to ensure that debugging or diagnostic information would not adversely affect the results. In addition, the VxWorks Spy utility was used to measure the CPU usage; therefore, it was included in the VxWorks kernel during build time.

A function was used to initialize the Spy interrupt clock to 10,000 ticks per second and to summarize task execution data at 1-sec intervals. The VxWorks spy task tracked and counted the number of times any of the VxWorks tasks were running at the time of each Spy interrupt. At the end of each second, a summary of the tasks' execution (i.e., the number of counts each task ran during the previous second) was printed to a log file on the target machine. Since only one task can run at a time, the sum of all counts for all the tasks should theoretically be equal to 10,000 over any 1 sec summary period. Therefore, if a task was reported as running during 7000 Spy interrupt times, for example, it would have been running about 70 percent of the time or 0.7 sec. Since the SDQS network code was called at a 25 Hz rate, this indicates that the average execution time per execution cycle, or frame, was about 0.028 sec.

This CPU execution information was recorded for each of the individual stand-alone networks and summarized in table 8(a) in terms of average time per frame. The corresponding average time per frame values for the combined EPS/CCL–7 network are summarized in table 8(b) and compared with the summed values for the two individual standalone networks. The difference between the two sets of average time per frame values (Δ) are shown in the fourth and fifth columns of that table in terms of milliseconds and percentage, respectively. These results indicate that the combined network required less average time per frame to execute the same set of scenarios. This reduction in execution time (by 24.83 to 29.02 percent) suggests that the combined network benefits from execution efficiencies in comparison to the standalone networks.

TABLE 8—CPU USAGE SUMMARY

(a) CPU Usage Summary for Separate Networks

Scenario	EPS average time per frame (ms)	CCL–7 average time per frame (ms)
P1_01	1.33	0.78
P1_02	1.29	0.75
P1_03	1.30	0.75
P1_04	1.29	0.76
P1_05	1.30	0.75
P1_06	1.29	0.75
P1_07	1.29	0.75
P1_08	1.31	0.75
P1_09	1.31	0.76
P1_10	1.31	0.75

(b) CPU Usage Comparison Summary

Scenario	EPS + CCL–7 average time per frame (ms)	Combined network average time per frame (ms)	Δ	
			(ms)	%
P1_01	2.11	1.58	–0.52	–24.83
P1_02	2.04	1.48	–0.56	–27.67
P1_03	2.05	1.47	–0.59	–28.52
P1_04	2.05	1.46	–0.59	–28.90
P1_05	2.05	1.46	–0.59	–28.70
P1_06	2.04	1.45	–0.59	–29.02
P1_07	2.04	1.47	–0.57	–27.77
P1_08	2.06	1.48	–0.59	–28.54
P1_09	2.07	1.49	–0.59	–28.42
P1_10	2.06	1.51	–0.55	–26.67

Memory usage.—For each of the networks, two types of memory usage were recorded: static and maximum dynamic. In addition, memory usage was recorded for a network shell to

provide a memory usage baseline. This network shell contains no parameters to qualify and is composed of just the executable object code with only the initialization requirements that are not specific to any application.

Static memory is comprised of three categories: text memory—the number of bytes in memory used to store text characters; data memory—the number of bytes used to store initialized data; and Bss memory—the number of bytes in an uninitialized data segment created and reserved for data not given an initial value (i.e., variables). Static memory usage was recorded through the use of the VxWorks Browser tool and the results (in terms of the three aforementioned categories as well as a total amount) are shown in table 9.

TABLE 9.—STATIC MEMORY SUMMARY

	Baseline	EPS	CCL–7	EPS + CCL–7	Combined network	Δ
Text (bytes)	81680	89024	81728	170752	88896	–81856
Data (bytes)	265	2176	592	2768	2272	–496
Bss (bytes)	8000	8008	8000	16008	8008	–8000
Total (bytes)	89945	99208	90320	189528	99176	–90352

The results show that the baseline network uses a total of 89.9 Kb of static memory. Meanwhile, the EPS network uses 99.2 Kb and the CCL–7 network uses 90.3 Kb of static memory. An SDQS implementation made up of separate EPS and CCL–7 networks would therefore require a total of 189.5 Kb of static memory. Contrast this result with the SDQS implementation that executes a combined network which only uses about 99.2 Kb of static memory. As shown in the Δ column of table 9 (representing the static memory usage difference between the combined network and the sum of the two standalone networks), the combined network provides a usage savings of 90.3 Kb. These results highlight the fact that each SDQS network, regardless of complexity will incur the baseline memory usage requirement in addition to application specific requirements. As a result, an SDQS implementation comprised of separate networks will incur a separate baseline memory "footprint" for each network. In practical terms, the penalty for this memory footprint will increase proportionally to the number of subsystems involved.

To record the dynamic memory usage, the dynamic allocation functions, such as *malloc*, *calloc*, etc, were overridden with custom memory routines that kept track of dynamic memory allocation. During any call to a memory allocation routine, the maximum amount of memory that was allocated for the SDQS task was recorded.

The results for the maximum dynamic memory usage are shown in table 10. From these results, it initially appears that a penalty is incurred by the combined network in terms of the amount of dynamic memory allocated. To determine the cause for the combined network's greater use of dynamic memory than the summed total amount used by the two individual separate networks, a detailed analysis of the SDQS code structure was performed. This analysis determined that the extra dynamic memory allocation is caused by idiosyncrasies in the way the SDQS code defines certain memory allocation multipliers depending on the number of operating phases and analytical relations handled by the SDQS network. The CCL–7 standalone network has relatively few operating phases and relationships. However, in the combined network, the multipliers for the CCL–7 related dynamic allocation were made unnecessarily large due to the greater number of phases and relationships reflective of the EPS portion of the network. This idiosyncrasy can be avoided in a flight code implementation. As a result, the apparent dynamic memory allocation penalty incurred by the combined network can be discounted.

TABLE 10.—DYNAMIC MEMORY SUMMARY

Scenario	EPS (bytes)	CCL–7 (bytes)	EPS + CCL–7 (bytes)	Combined Networks (bytes)	Δ	
					(bytes)	%
P1_01	181728	15448	197176	208232	11056	5.61
P1_02	182880	15760	198640	214032	15392	7.75
P1_03	183952	15760	199712	214032	14320	7.17
P1_04	183128	14920	198048	212984	14936	7.54
P1_05	183128	15760	198888	212320	13432	6.75
P1_06	183456	16040	199496	214640	15144	7.59
P1_07	184760	15200	199960	212320	12360	6.18
P1_08	182296	16040	198336	213016	14680	7.40
P1_09	182896	16880	199776	212752	12976	6.50
P1_10	183704	26400	210104	227904	17800	8.47

V. SDQS Robustness Testing

A. Set-Up

The objective of this portion of testing was to validate, through testing, the robustness and reliability of the SDQS networks. Specifically of interest was the determination of how well the analytical redundancy networks handled multiple consecutive sensor failures and how gracefully SDQS degraded. With the SDQS software's analytical redundancy algorithms, sensor failures are determined by user-provided sensor relationships and system design information. When a sensor fails, that sensor and all relationships involving that sensor are disregarded by the SDQS network, which then determines if enough valid relationships remain to allow accurate qualification of the remaining sensors. If there are not enough valid relationships between remaining sensors to allow qualification, the SDQS network will continue to function, albeit, with a degraded capability that is restricted to traditional limit-checking algorithms.

To validate this behavior, the combined EPS/CCL-7 SDQS network implemented for the SDQS Configuration-based Performance Tests (sec. III) was subjected to tests that involved failing multiple sensors in succession. The resulting

network behavior was analyzed to determine at what point and in what manner the analytical redundancy relationships broke down and whether the SDQS software was able to fall back on the limit-checking algorithms.

To simplify testing and configuration, the scope of limit-check testing was restricted to the CCL–7 portion of the combined network. The sensors considered are shown in table 2 (sec. II) and three distinct tests were performed to verify the networks' robustness. Testing was performed in the Matlab/Simulink environment and executed in accordance with the three test cases described in tables 11(a) to (c). Each test case outlines the insertion of multiple failures sequentially in terms of failure insertion order, failure types and affected sensors. Although the EPS network was not involved in these cases to verify limit-checking, it was used to test an additional performance element key to validating robust operation. After failures were initiated in all the CCL–7 network sensors (failures 1 through 6) for each of the test cases described in tables 11(a) to (c), an additional failure was inserted in one of the EPS sensors (failure 7). The objective of this final step was to ensure that degraded SDQS performance is confined to only the network where relationships fail while the remaining network continues to operate with analytical relationships.

TABLE 11.—TEST CASES FOR COMBINED SDQS NETWORK

(a) Robustness Test 1 (R_T1)			
Failure order	Network	Sensor failed	Failure type
1	CCL–7	FH123A	Noise
2	CCL–7	FH209A	Drift
3	CCL–7	FH123B	Drift
4	CCL–7	FH209B	Hard fault
5	CCL–7	FH203A	Hard fault
6	CCL–7	FH203B	Noise
7	EPS PDU	IinA	Noise
(b) Robustness Test 2 (R_T2)			
Failure order	Network	Sensor failed	Failure type
1	CCL–7	FH123A	Noise
2	CCL–7	FH203B	Noise
3	CCL–7	FH209A	Drift
4	CCL–7	FH123B	Drift
5	CCL–7	FH209B	Hard fault
6	CCL–7	FH203A	Hard fault
7	EPS PDU	IoutB	Hard fault
(c) Robustness Test 3 (R_T3)			
Failure order	Network	Sensor failed	Failure type
1	CCL–7	FH203A	Hard fault
2	CCL–7	FH209B	Hard fault
3	CCL–7	FH203B	Noise
4	CCL–7	FH123A	Noise
5	CCL–7	FH123B	Drift
6	CCL–7	FH209A	Drift
7	EPS PDU	IoutA	Drift

While this study called for multiple sequential sensor faults, none were included in the CCL–7 test data used for the study. Therefore, a nominal data set was used as a base upon which sensor failures were artificially superimposed using a failure insertion block developed in the Simulink environment (fig. 2). This block provided the ability to specify the type and magnitude of the faults as well as the time of insertion.

Analysis

During testing, log files were generated for each test and subsequently analyzed to determine the effectiveness of the SDQS network's ability to identify multiple sensor failures as they were sequentially inserted. Key metrics were identified for each successfully identified sensor failure: the type of detection that identified the failure (analytic relation vs. limit check) and the failure detection time (time-to-detect). The results of this analysis are summarized in tables 12(a) to (c) for each of the test cases.

TABLE 12.—RESULTS FOR COMBINED SDQS NETWORK

(a) Robustness Test 1 (R_T1)						
Fail order	Sensor failed	Failure type	Start	Fail	Time to detect (samples)	Detection method
1	FH123A	Noise	5875	5878	3	Analytic
2	FH209A	Drift	5975	6018	43	Analytic
3	FH123B	Drift	6000	6219	219	Analytic
4	FH209B	Hard fault	6250	6253	3	Limit check
5	FH203A	Hard fault	6300	6303	3	Limit check
6	FH203B	Noise	6500	6563	63	Limit check
7	IinA	Noise	6750	6752	2	Analytic
(b) Robustness Test 2 (R_T2)						
Fail order	Sensor failed	Failure type	Start	Fail	Time to detect (samples)	Detection method
1	FH123A	Noise	5500	5503	3	Analytic
2	FH203B	Noise	5750	5752	2	Analytic
3	FH209A	Drift	6000	6215	215	Analytic
4	FH123B	Drift	6250	6552	302	Limit check
5	FH209B	Hard fault	6750	6753	3	Limit check
6	FH203A	Hard fault	7000	7003	3	Limit check
7	IoutB	Hard fault	7500	7502	2	Analytic
(c) Robustness Test 3 (R_T3)						
Fail order	Sensor failed	Failure type	Start	Fail	Time to detect (samples)	Detection method
1	FH203A	Hard fault	5500	5503	3	Limit check
2	FH209B	Hard fault	5750	5753	3	Limit check
3	FH203B	Noise	6000	6003	3	Analytic
4	FH123A	Noise	6250	6338	88	Limit check
5	FH123B	Drift	6500	6770	270	Limit check
6	FH209A	Drift	6750	8343	1593	Limit check
7	IoutA	Drift	8750	8801	51	Analytic

Test cases R_T1 and R_T2 both begin with failure types that are thought of as being more challenging such as noise and drifts. As each failure is detected, the failed sensors in question are "disqualified" by the network and all relationships involving that sensor are suspended from further consideration in qualifying subsequent sensor failures. The results of these tests show that the CCL–7 network is able to handle up to three sequential faults through analytical redundancy. After the third sensor is failed, the remaining valid sensor relations for this subnetwork do not provide the statistical basis required to support further analytical redundancy-based qualification. Subsequently, the SDQS reverts to hard fault limit checking algorithms for the CCL–7 network to successfully identify further CCL–7 sensor failures. Note that the cessation of qualification based on analytical redundancy, after the loss of the third sensor, is specific to the analytical relationships for this particular network configuration. Another network could be expected to behave differently based on its inherent relationships.

Test case R_T3 begins with two sequential level-change failures. This type of failure is the most easily identified by threshold limit checking algorithms, so the failures were detected first by SDQS through limit checks. Regardless of detection method, these failed sensors are still disqualified and all relationships involving them are suspended from further consideration in qualifying subsequent failures by the SDQS network. The third failure is successfully identified by the analytical redundancy network at which point there are insufficient relationships for further analytical redundancy-based qualification. All subsequent failures in the CCL–7 network are then identified through limit-checking.

Analysis of the time-to-detect values for the more challenging failure types (i.e., drifts and noise) yields another interesting observation. Detection times for these types of failures by the analytical redundancy networks are often, but not always, much faster than those determined by limit checking. For example, the time-to-detect value for identifying the noise failure of sensor FH123A by the analytical network is 3 time samples (see Test R_T1 or Test R_T2). Compare this to a Time-to-detect value of 88 time samples provided by limit checking (see Test R_T3). This result is consistent with the noise limit algorithm's use of a window of data to perform its evaluation and often requires multiple windows for confirmation. Similarly, the analytical network's time-to-detect values for the drift failure of sensor FH209A are 43 time samples (Test R_T1) and 213 time samples (Test R_T2) while limit checking yields a time-to-detect value of 1593 time samples.

Note that the detection times for both the analytical and limit checking algorithms depend on the threshold values set for each algorithm. In addition, for drift faults the time-to-detect for the limit checking algorithm also depends on the value of the sensor measurement relative to the value of the threshold when the drift is applied. If at the time of fault initiation, the signal is far from the detection threshold, the time required to detect the fault will be greater than if the signal had started out close to the detection threshold. For this reason, more so than for the analytical algorithm, the limit-check detection times for drifts can vary substantially. One example in this data is FH123B where detection times are 302 and 270 samples for Tests 2 and 3, respectively.

Also note that for each of the test cases, the final failure was initiated into one of the EPS sensors and SDQS was able to continue detection using analytical algorithms in that network. These results validate that SDQS is able to confine degraded performance to only the network where relationships fail without impacting the remaining network's ability to operate with analytical relationships. Despite potentially inferior performance, the limit checking function provides SDQS with an important backup sensor qualification capability in circumstances where the analytical relationships break down. In real-world real-time applications, where multiple sensor failures are a distinct possibility, this capability to degrade gracefully is crucial to the practical application of SDQS.

Summary and Recommendations

This paper discussed the SDQS implementation study performed at NASA GRC to determine how different configurations of SDQS networks may impact performance in terms of data qualification effectiveness, resource utilization and robustness. Described were the implementation approach, followed by the diagnostic algorithms examined, the test-beds involved and the analysis platforms employed. Discussed next, in detail, was the test approach developed which encompassed three separate studies with each focused on a specific objective: sensor network configuration; real-time performance and resource requirements; and sensor network robustness. The subsequent analysis was then discussed. The major findings resulting from this study are as follows:

(1) Implementing SDQS as either a single network processing multiple distinct data qualification subsystems or as separate, independent networks has no impact on its ability to perform effectively, at least in the cases studied here where the subsystems do not have any interaction.

(2) The SDQS algorithms degrade gracefully. As multiple sensors in a given SDQS sensor network fail and a sufficient number of relationships used for analytical qualification become unavailable, the algorithms continue to qualify data, albeit with reduced capability, by reverting solely to limit checking. However, data shows that degraded performance is limited to the network with a statistically sufficient number of failed analytical relations. Networks with a statistically significant number of valid analytical relationships continue to operate optimally using those relationships. This ability is crucial in practical applications of SDQS since multiple sensor failures are a possibility in real-world systems.

(3) For the case where the intended application spans multiple subsystems that are independent of each other with no coupling—such as those considered in this paper—implementing SDQS as a single combined network will yield performance benefits in terms of less memory usage and faster execution.

This study focused strictly on an implementation perspective in terms of SDQS performance and computational resource utilization. However, it is critical to note that factors related to the Verification and Validation (V&V) process will also greatly impact the choice of SDQS architecture and bears further investigation. Finally, futures studies could be conducted to investigate and characterize other SDQS implementation approaches and their effect on detection and execution performance. One suggested area of focus for future research might be the application of subsystems composed of multiple sample rates and different measurement precisions (e.g., 8-, 12-, or 16-bit). Another would be to investigate how coupling between subsystems impacts performance; and whether or not performance can be improved via a decoupling approach similar to distributed control.

References

1. J.F. Hanaway, R.W. Moorehead, "Space Shuttle Avionics System," NASA/SP–504.
2. R. Bickford, E.P. Liu, "Sensor Data Qualification for CLV Upper Stage," NASA/CR—2007-214948, Dec. 2007.
3. K. Melcher, C.F. Fulton, W.A. Maul, T.S. Sowers, "Development and Application of a Portable Health Algorithms Test System," NASA/TM—2007-214840, Prepared for the 54th Joint Army-Navy-NASA-Air Force (JANNAF) Propulsion Meeting, May 14–18, 2007.
4. W.A. Maul, K.J. Melcher, A.K. Chicatelli, and T.S. Sowers, "Sensor Data Qualification for Autonomous Operation of Space Systems," Proceedings of the American Association for Artificial Intelligence 2006 Fall Symposium on Spacecraft Autonomy, Oct. 13–15, 2006. AAAI Technical Report FS–06–07: 59–66.
5. C. Meyer, C. Fulton, W. Maul, A. Chicatelli, H. Cannon, A. Bajwa, E. Balaban, E. Wong, "Propulsion IVHM Technology Experiment Overview," Paper no. 1481, 2003 IEEE Aerospace Conference, Big Sky, MT, Mar. 8–15, 2003.
6. W. Maul, A. Chicatelli, C. Fulton, E. Balaban, A. Sweet, and S. Hayden, A. Bajwa, "Addressing the Real-World Challenges in the Development of Propulsion IVHM Technology Experiment (PITEX)," AIAA 1st Intelligent Systems Technical Conference, Chicago, IL, Sep. 20–22, 2004.

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
01-04-2009	Technical Memorandum	

4. TITLE AND SUBTITLE
Sensor Data Qualification System (SDQS) Implementation Study

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Wong, Edmond; Fulton, Christopher; Maul, William; Melcher, Kevin

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
WBS 136905.08.05.08.08.01.03

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
John H. Glenn Research Center at Lewis Field
Cleveland, Ohio 44135-3191

8. PERFORMING ORGANIZATION REPORT NUMBER
E-16642

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORS ACRONYM(S)
NASA

11. SPONSORING/MONITORING REPORT NUMBER
NASA/TM-2009-215442

12. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified-Unlimited
Subject Category: 19
Available electronically at http://gltrs.grc.nasa.gov
This publication is available from the NASA Center for AeroSpace Information, 301-621-0390

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The Sensor Data Qualification System (SDQS) is being developed to provide a sensor fault detection capability for NASA's next-generation launch vehicles. In addition to traditional data qualification techniques (such as limit checks, rate-of-change checks and hardware redundancy checks), SDQS can provide augmented capability through additional techniques that exploit analytical redundancy relationships to enable faster and more sensitive sensor fault detection. This paper documents the results of a study that was conducted to determine the best approach for implementing a SDQS network configuration that spans multiple subsystems, similar to those that may be implemented on future vehicles. The best approach is defined as one that most minimizes computational resource requirements without impacting the detection of sensor failures.

15. SUBJECT TERMS
Sensor qualification; Failure detection

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			STI Help Desk (email:help@sti.nasa.gov)
U	U	U	UU	15	19b. TELEPHONE NUMBER *(include area code)* 301-621-0390