AIAA Guidance, Navigation and Control Conference and Exhibit, 20-23 Aug 2007, Hilton Head, SC

A Comparison of GN&C Architectural Approaches for Robotic and Human-Rated Spacecraft

Alejandro D. Domínguez-García[*], Gregor Z. Hanuschak[†], Steven R. Hall[‡] and Edward F. Crawley[§]

Massachusetts Institute of Technology, Cambridge, MA, 02139, USA

This paper compares various architectural philosophies used in the design of GN&C systems for NASA human-rated and robotic spacecraft during the last four decades. This comparison gives insight into the options available for the GN&C systems of the new generation of space vehicles to be developed under NASA's Constellation Program (CxP). This study focuses on the component interconnectivity features behind cross-strapping and channelization, the two main architecting approaches for designing fault- tolerant GN&C systems. The features of theses two approaches, as well as hybrid versions of these approaches, are explained. The paper also discusses their advantages and disadvantages in terms of complexity, reliability, and fault-tolerance. Several systems developed by NASA for human-rated systems and robotic systems are analyzed and compared from this perspective. The final goal of this paper is to lay the foundations for a more in-depth study to assess commonality among different GN&C architectures for the CxP.

Nomenclature

AA:	Accelerometer Assembly
C&C:	Control and Command
CxP:	Constellation Program
CEV:	Crew Exploration Vehicle
CLV:	Crew Launch Vehicle
CMG:	Control Moment Gyro
CSM:	Command and Service Module
FCC:	Flight Critical Computer
FCP:	Flight Critical Processor
FDIR:	Failure Detection, Isolation, and Reconfiguration
FT:	Fault Tolerance
GN&C :	Guidance, Navigation and Control
GPC:	General Purpose Computer
GA:	Gyroscope Assembly
GiA:	Gimbal Assembly
GPS:	Global Positioning System
ICP:	Instrumentation Control Processor
IMU:	Inertial Measurement Unit
ISS:	International Space Station
LSAM:	Lunar Surface Access Module
N_a :	Actuators redundancy level

[*]Post-Doctoral Associate, Department of Electrical Engineering and Computer Science, 77 Massachusetts Avenue, Room 10-082. Member AIAA.

[†]Graduate Research Assistant, Department of Aeronautics and Astronautics, 77 Massachusetts Avenue, Room 33-409. Student Member AIAA.

[‡]Professor, MacVicar Faculty Fellow, Department of Aeronautics and Astronautics, 77 Massachusetts Avenue, Room 33-313. Associate Fellow AIAA

[§]Ford Professor of Engineering, Department of Aeronautics and Astronautics, and Engineering Systems Division, 77 Massachusetts Avenue, Room 33-413. Fellow AIAA

N_c :	Computers redundancy level
N_s :	Sensors redundancy level
MDM:	Multiplexer/Demultiplexer
NE:	Network Element
NEFU:	Network Element Fifth Unit
RAS:	Rudder Actuation Subsystem
RCS:	Reaction Control System
RCSAS:	Reaction Control System Actuation Subsystem
RFAS:	Right Flap Actuation Subsystem
RRAS:	Right Rudder Actuation Subsystem
RGA:	Rate Gyro Assembly
S:	Sextant
SIGI:	Space Integrated Global Positioning System/Inertial Navigation System
SISO:	Single Input System Output
SPS:	Service Propulsion System
SPSAS:	Service Propulsion Actuation Subsystem
ST:	Scanning Telescope
TAS:	Thruster Actuation Subsystem
t :	Time
λ_a :	Actuator failure rate
λ_c :	Computer failure rate
λ_s :	Sensor failure rate

I. Introduction

The Vision for Space Exploration (VSE) of 2004 has provided NASA with a new direction for human spaceflight and exploration. In order to meet the VSE goals, NASA's CxP will have to acquire and operate a number of new human-rated systems, such as the Orion Crew Exploration Vehicle (CEV), the Crew Launch Vehicle (CLV), and the Lunar Surface Access Module (LSAM), along with other elements for crew transportation (e.g., in-space propulsion stages), as well as for lunar habitation and mobility. There will also be lunar robotic orbiter vehicles and robotic lunar landers. Commonality in exploration system hardware, and software elements offers the opportunity to significantly increase sustainability by reducing, both non-recurring and recurring cost and / or risk. The potential benefit of common GN&C avionics and flight software is considerable, not only in the initial development effort, but in validation and verification, and more importantly in the ongoing maintenance efforts and incremental upgrades that will occur over the life cycle of these spacecraft. With commonality of the onboard components of this system, there is more likelihood that ground control and communications systems could be made more common, yielding a multiplier effect. A comparative assessment of robotic and human-rated GN&C system architectural approaches is currently being performed as part of a proactive NASA Engineering and Safety Center sponsored study of CxP GN&C Commonality at the Massachusetts Institute of Technology (MIT). This study effort was driven by the observation, both on the part of NESC and MIT, that GN&C systems for exploration prominently stand out among all the future spacecraft systems, as an area where commonality might be of greatest benefit. This comparative assessment of robotic and human-rated GN&C system architectural approaches was undertaken as a fundamental step towards understanding the opportunities and limitations of GN&C commonality across the CxP flight elements. This paper documents the results of this comparative analysis yielding an understanding of the fundamental differences (historical and objective) between robotic and human-rated mission GN&C systems.

The paper is organized as follows: Section II gives some high-level background regarding similarities and differences between human-rated and robotic spacecraft. Section III of this paper is a review of architectural approaches used in fault-tolerant GN&C systems. Section IV explains the main architectural features used in GN&C systems of NASA human-rated and robotic spacecraft. Section V presents the observations drawn by the authors from comparing the approaches used in GN&C systems for both human-rated and robotic spacecraft. Concluding remarks are presented in Section VI.

II. Top Level Comparison of Human-Rated and Robotic GN&C Systems

In this section, we briefly discuss the similarities and differences in the GN&C design process, and operational environment, for both human-rated and robotic spacecraft. We also discuss unique aspects of each type of spacecraft.

The design processes for human-rated and robotic spacecraft GN&C systems have many similarities. The design processes for both require system-level architecture analyses, trade studies, and fault tolerance and reliability analyses to properly balance mission success (risk), performance, mass, power, and cost. Both types of spacecraft are designed using similar discipline-standard analysis, modeling, and simulation techniques. For example, linear frequency domain stability analyses and time-domain non-linear performance simulations are commonly used. Due to industry consolidation, there are only a limited number of GN&C component vendors. Therefore, human-rated and robotic GN&C systems are both implemented using similar (if not identical) sensors, computer processors, and actuators.

In terms of operation, both human-rated and robotic spacecraft operate in similar environmental conditions. They both must survive demanding launch shock and vibration environments. They both must operate in harsh space radiation and thermal / vacuum environments. They both operate in many of the same mission phases, such as low Earth orbital cruise, entry, descent and landing, rendezvous, etc. Furthermore, both types of spacecraft perform some similar mission functions, such as stellar-inertial navigation, angular rate damping, attitude control, and orbital adjustment propulsive maneuvers.

Human-rated spacecraft GN&C systems differ from those for robotic spacecraft in important ways. The designer of a GN&C system for a human-rated system must always keep the physical safety of the human crew members foremost in mind. Thus, it is almost certain that the GN&C system for a human-rated spacecraft will be required to be tolerant to at least two faults (fail operational / fail safe) in order to meet overall spacecraft safety and reliability requirements. Additionally, most mission phases require an abort strategy that can remove the spacecraft (with its crew) from an unanticipated unsafe state. GN&C systems on most human-rated spacecraft to date have been required to operate over short mission durations (e.g., days to weeks), compared to the multi-year duration of robotic missions. The one obvious exception to the general mission duration rule cited above is the International Space Station (ISS). However, in the case of ISS, certain elements of the ISS GN&C system can be replaced on-orbit, such as the control moment gyros (CMGs) that control the station's attitude. Finally, verification and validation, and re-certification costs of any modified GN&C hardware and software are more burdensome for human-rated spacecraft GN&C applications than for robotic spacecraft GN&C.

There are some aspects of spaceflight unique to human-rated missions. It is likely that future human-rated spacecraft will be reused multiple times over a long multi-year life-cycle period, whereas robotic spacecraft are rarely reused. (However, in CxP, reuse of robotic spacecraft is envisioned.) Therefore, the physical, economic, and safety-related impacts of refurbishing, servicing, and/or replacing GN&C hardware on the ground after each mission must be considered early in the GN&C design process. The cockpit panel displays, monitors and alarms, as well as the hand controllers used for piloting manual inputs, which are typically used on human-rated spacecraft are non-existent on robotic spacecraft. For human-rated spacecraft, specialized GN&C training and simulation is required for both the crew and the ground operations team, whereas GN&C training is only required for the ground operations team on robotic spacecraft missions. Human-rated spacecraft are flown by pilots, whereas robotic spacecraft are not. Therefore, far more than their counterparts working on robotic spacecraft, the designers of GN&C systems for human-rated spacecraft must recognize and address the issue of mode awareness.

There are also some aspects of spacecraft systems unique to robotic spacecraft. Robotic spacecraft GN&C system designers often exploit the advantages of flying dissimilar flight hardware. For example, digital fine sun sensors are often used to backup star trackers for precision attitude determination. Magnetometers can be used for backup attitude determination, and thrusters can be used in place of magnetic torquers to unload excess reaction wheel momentum. Separate and dissimilar processors are used to host and execute digital safe hold mode control laws. Attitude control, line-of-sight pointing, and jitter control performance is often a primary design driver for many robotic science spacecraft. High-accuracy pointing is seldom if ever a requirement for human-rated vehicles.

The authors believe that the most important difference between human-rated and robotic spacecraft GN&C systems is the fault-tolerance requirement. Thus, in the remainder of this paper, we will primarily focus on fault tolerance and reliability-related aspects of GN&C systems for both human-rated and robotic spacecraft.

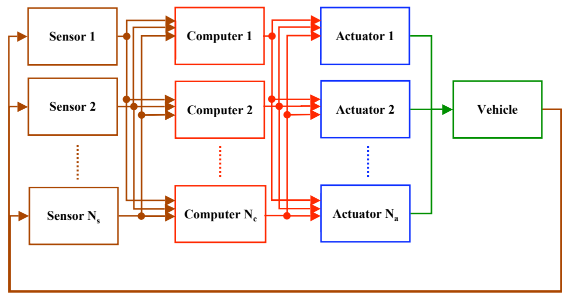


Figure 1. Cross-strapped architecture.

III. Fault-Tolerant System Architecture Approaches for GN&C Systems

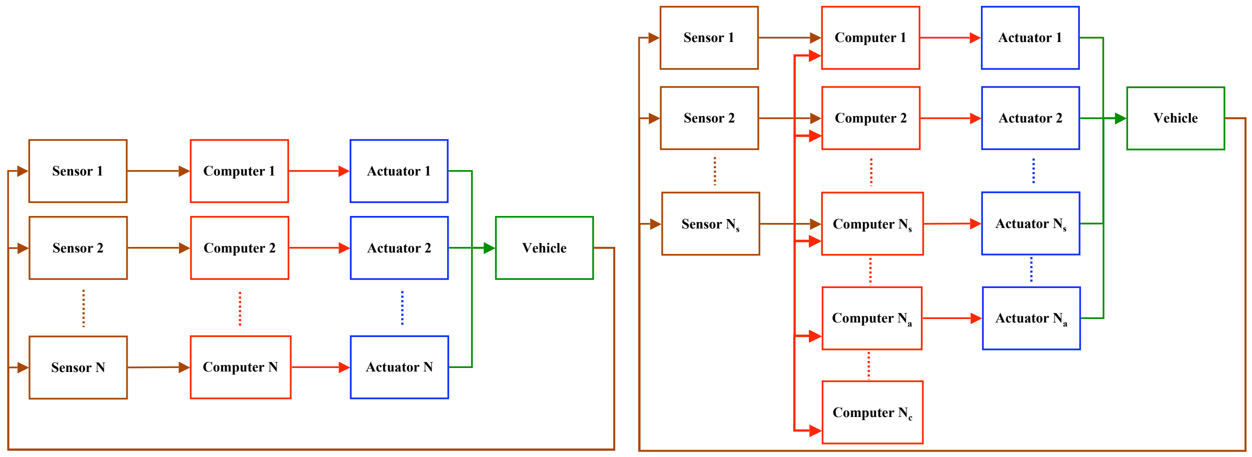
Fault tolerance can be defined as the ability of a system to adapt and compensate for, in a planned and systematic way, random failures of system components that can cause the overall system to fail to perform the function for which it was designed.[1] Fault tolerance is achieved with redundancy (component or subsystem functional replication) and appropriate management of that redundancy through failure detection, isolation, and reconfiguration mechanisms. There are good references in the literature addressing these important topics of fault-tolerant systems; for example, see Ref. 2 for redundancy strategies and their management, Ref. 3 for voting algorithms, and Ref. 4 for failure detection. However, equally important is the interconnectivity between different components in the system. In a GN&C system (and in any closed-loop control system in general), there are three main classes of necessary components: sensors, computers, and actuators/effectors.

From the interconnectivity point of view, there are two major architectural philosophies used to achieve fault-tolerance in a GN&C system: cross-strapping and channelization. In the remainder of this section, we will explain in detail the main features of these two philosophies. This section will discuss these philosophies, comparing their advantages and disadvantages.

III.A. Cross-Strapped Architectures

In a fully cross-strapped architecture (Fig. 1), the output of each component is physically connected to the input of each immediate element in the control loop. Thus, every sensor output is physically connected to every computer, and every computer is connected to every actuator. To simplify the explanation further, let's consider that the vehicle in Fig. 1 can be represented by a single input system output (SISO) dynamic model. Then, every sensor $(1,\dots,N_S)$ is measuring the same vehicle state variable, while every actuator $(1,\dots,N_A)$ can affect the only vehicle control input. The computers $(1,\dots,N_C)$ all implement the same appropriate control law to obtain the desired vehicle dynamic performance. In theory, if just one of the sensors, one of the computers, and one of the actuators is functional, there would be enough functionality to control the vehicle. Thus theoretically, the system could deliver its functionality even if $N_S - 1$ sensors failed, $N_C - 1$ computers failed, and $N_A - 1$ computers failed.

In reality, of course, the system may not successfully tolerate this many failures. Replicating components (adding redundancy) does not ensure fault tolerance. A failed component, if not removed from the control loop, could alter the closed-loop system dynamics and cause the vehicle to become unstable and/or uncontrollable. For example, if $N_S - 2$ sensors have failed, and these failures have been detected and isolated, it will be straightforward to detect the next sensor failure, since there will be a disagreement between the good and bad sensors. However, isolating the failure to the failed sensor is more difficult without additional information, since voting can no longer be used. Built-in-Test equipment can sometimes isolate the failure, but often the probability that built-in test equipment will properly isolate the failure is not as high as would be desired. Thus, the other important aspect of fault tolerance is redundancy management, *i.e.*, the appropriate mechanisms to detect component failures, isolate those failures, and reconfigure the system (making use of the remaining redundant components / subsystems) so that the system remains functional. Therefore, the maximum level of fault tolerance is $\min\{N_S - 1, N_C - 1, N_A - 1\}$, assuming perfect failure coverage of the first $N_S - 1$, $N_C - 1$, or $N_A - 1$ sensor, computer, or actuator failures respectively. If two functional components are necessary to reliably identify the failure of a third, then the level of fault tolerance is $\min\{N_S - 2, N_C - 2, N_A - 2\}$.



(a) Fully channelized architecture. (b) Channelized architecture with computers cross-talk.

Figure 2. Channelized architectures.

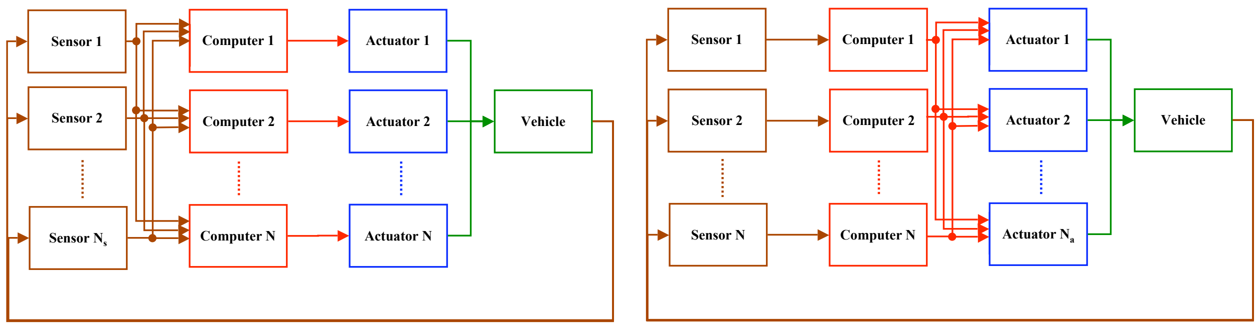
III.B. Channelized Architectures

In a fully channelized architecture (Fig. 2(a)), each component output is connected to the input of a single component. For example, a sensor output $(1, \ldots, N_S)$ is connected to a single computer $(1, \ldots, N_C)$, and the output of each computer is connected to a single actuator $(1, \ldots, N_A)$, where $N_S = N_C = N_A = N$, to create N sensor-computer-actuator strings. As in Section III.A, we consider the case where the vehicle in Fig. 2(a) is a SISO dynamic model. Then, in theory, just one of the N strings would be sufficient for the system to deliver its functionality. It is important to note that in this approach, a single failure of a string component disables the whole string. Thus, the other two components are no longer operational even if they have not failed. Therefore, this architecture can be guaranteed to tolerate at most $N-1$ failures before the system stops delivering its functionality. The level of fault tolerance will be $N-2$ if it is assumed that when two operational strings are left, it is impossible to detect and isolate an additional failure.

There is another type of channelized architecture that results when there is cross-talk between the computers (Fig. 2(b)). In this case, the numbers of sensors, computers and actuators are not necessarily the same. The number of computers $(1, \ldots, N_C)$ is greater than or equal to the number of sensors $(1, \ldots, N_S)$ and/or actuators $(1, \ldots, N_A)$, although, as in a fully channelized architecture, each component output is connected to only one component input. Again, we consider the case that the vehicle in Fig. 2(b) is a SISO system. Then, theoretically, the system can tolerate at most $N_S - 1$ sensor failures, $N_A - 1$ actuator failures and $N_C - 1$ computer failures. Thus the maximum level of fault tolerance is $\min\{N_S - 1, N_C - 1, N_A - 1\}$, and there are scenarios where the system is still functional with up to $N_S + N_C + N_A - 3$ failures (whenever the three remaining non-failed elements are a sensor, a computer and an actuator in the same string). Again, the level of fault tolerance will be $\min\{N_S - 2, N_C - 2, N_A - 2\}$ if there is no possibility of detecting and isolating an additional sensor failure when only $N_S - 2$ sensors are non-failed, and similarly when $N_C - 2$ computers, or $N_A - 2$ actuators are non-failed.

One of the advantages of a channelized architecture with computer cross-talk is that a single sensor (or actuator) failure does not disable the other two components of the string (the actuator (or sensor) and the computer). Therefore, this architecture can tolerate more failures than the fully channelized architecture. However, the channelized architecture with computer cross-talk has the disadvantage of adding an additional layer of complexity to the system design. In order to ensure proper operation of this architecture, it is necessary to implement the necessary algorithms to cope with the Byzantine Generals Problem[5] to ensure that all the computers are receiving the same sensor data through the computer cross-talk.

Asymmetric channelization is another advantage of channelized architectures with computer cross-talk. This means that the number of sensors, computers, and actuators does not need to be the same; it is possible to have fewer redundant actuators and sensors than computers. Therefore, if different levels of fault tolerance are demanded for each of these subsystems, it is possible to implement different levels of redundancy among sensors, computers and actuators.



(a) Hybrid architecture: cross-strapped sensing and channelized actuation.

(b) Hybrid architecture: channelized sensing and cross-strapped actuation.

Figure 3. Hybrid architectures.

III.C. Hybrid Architectures

Hybrid architectures are a blend of the two architecting approaches already discussed — cross-strapping and channelization. In the hybrid architecture displayed in Fig. 3, the sensors $(1, \ldots, N_S)$ are cross-strapped to each computer $(1, \ldots, N)$, but each computer independently controls a single actuator $(1, \ldots, N)$. Figure 3(b) shows another type of hybrid architecture where the actuators are now cross strapped and the sensors are connected to the computers using a channelized approach. An example of a hybrid architecture of the type shown in Fig. 3(a) is discussed in Ref 6.

III.D. Comparison of Different Architectural Approaches for Fault-Tolerant GN&C Systems

Unreliability, or probability of system failure, is one metric for comparing different architectures. In this paper, we define two different unreliability estimates: the unreliability lower bound and unreliability upper bound. The first estimate, the unreliability lower bound, is computed assuming perfect failure detection and isolation of the first $N_S - 1$ sensor failures ($N_C - 1$ computer failures or $N_A - 1$ actuator failures). The second estimate, the unreliability upper bound, is computed assuming that when two components of any class (sensors, computer or actuators) are left, it is impossible to detect and isolate an additional failure of the same class. These unreliability upper and lower bound estimates are collected in Table 1. Combinatorial analysis was used to obtain these estimates[7] with the following assumptions:

- Component failures are independent and exponentially distributed, and they are described in terms of a constant failure rate.
- All components of the same class have the same failure rate. Thus, any sensor, computer, or actuator has a failure rate of λ_s, λ_c, λ_a respectively.
- The values of $\lambda_s t$, $\lambda_c t$, $\lambda_a t$, where t is the time the system is operational, are assumed to be small enough that
  1. $1 - e^{-\lambda_s t} \approx \lambda_s t$, $1 - e^{-\lambda_c t} \approx \lambda_c t$, and $1 - e^{-\lambda_a t} \approx \lambda_a t$; and
  2. the unreliability function is approximately equal to the sum of probabilities of the smallest-size cut- sets (sets of components which, if failed, the system fails).

The unreliability lower bound estimates collected in Table 1 indicate that, for the same level of redundancy of each class of component ($N_s = N_c = N_a = N$), the fully cross- strapped architecture is the most reliable of all architectures, whereas the fully channelized architecture is the least reliable. This can be easily inferred from the inequalities

$$(\lambda_s t)^N + (\lambda_c t)^N + (\lambda_a t)^N < [(\lambda_s + \lambda_c)t]^N + [(\lambda_c + \lambda_a)t]^N - (\lambda_c t)^N < [(\lambda_s + \lambda_c + \lambda_a)t]^N, \tag{1}$$

$$(\lambda_s t)^N + (\lambda_c t)^N + (\lambda_a t)^N < [(\lambda_s + \lambda_c)t]^N + (\lambda_a t)^N < [(\lambda_s + \lambda_c)t]^N + [(\lambda_c + \lambda_a)t]^N - (\lambda_c t)^N, \tag{2}$$

$$(\lambda_s t)^N + (\lambda_c t)^N + (\lambda_a t)^N < (\lambda_s t)^{N_s} + [(\lambda_c + \lambda_a)t]^N < [(\lambda_s + \lambda_c)t]^N + [(\lambda_c + \lambda_a)t]^N - (\lambda_c t)^N. \tag{3}$$

Table 1. Comparison of unreliability upper and lower bound estimates for different fault-tolerant architecting approaches.

Architecture	Unreliability lower bound	Unreliability upper bound	Reliability ranking
Fully Cross-strapped	$(\lambda_s t)^{N_s} + (\lambda_c t)^{N_c} + (\lambda_a t)^{N_a}$	$N_s(\lambda_s t)^{N_s-1} + N_c(\lambda_c t)^{N_c-1} + N_a(\lambda_a t)^{N_a-1}$	1
Fully Channelized. $N_s = N_c = N_a = N$	$[(\lambda_s + \lambda_c + \lambda_a)t]^N$	$N[(\lambda_s + \lambda_c + \lambda_a)t]^{N-1}$	5
Channelized with computer cross-talk. $N_s = N_c = N_a = N$	$[(\lambda_s + \lambda_c)t]^N + [(\lambda_c + \lambda_a)t]^N - (\lambda_c t)^N$		4
Hybrid with cross-strapped sensors. $N_c = N_a = N$	$(\lambda_s t)^{N_s} + [(\lambda_c + \lambda_a)t]^N$	$N_s(\lambda_s t)^{N_s-1} N[(\lambda_c + \lambda_a)t]^{N-1}$	2(3)
Hybrid with cross-strapped actuators. $N_s = N_c = N$	$[(\lambda_s + \lambda_c)t]^N + (\lambda_a t)^{N_a}$	$N[(\lambda_s + \lambda_c)t]^{N-1} + N_a(\lambda_a t)^{N_a-1}$	3(2)

Similar relations can be established for the unreliability upper bounds, but the reliability ranking remains the same. This reliability ranking is summarized in column 4 of Table 1. It is important to note, however, that there are a number of factors that were not taken into account that may impact this ranking. These factors are the failure coverage probability, which it was assumed to be one for either $N-2$ or $N-1$ component failures of the same class; and the assumption that component failures are independent, thus precluding the possibility of common mode failures. These factors must be included if a more detailed analysis of a particular architecture is performed. Furthermore, it assumes that the cross strapping mechanisms are perfectly reliable, which is generally not the case.

Table 2 shows the ability of each architecture to detect and isolate failures. It also compares the testability of failure detection, isolation and reconfiguration (FDIR) mechanisms and the complexity of developing each architecture. In terms of failure detection, all the architectures are equivalent if the appropriate voting algorithms and built-in-test mechanisms are used. In terms of failure isolation, fully cross-strapped architectures are better than others since, in a properly designed cross-strapped architecture, a single component failure can be isolated without disabling other components. This is different than for fully channelized architectures, where a single component failure will disable all the other components in the same string.

Looking at failure containment, cross-strapping is good for isolating individual failures, but it makes the architecture more prone to common mode failures. In this regard, a fully channelized architecture is more immune to common mode failures; a single failure will take out one sensor-computer-actuator string, but is unlikely to affect the remaining strings. The same is true for a fully channelized architecture with computer cross-talk.

In terms of FDIR testability, fully channelized architectures are easier to test than fully-cross strapped architectures, since each string can be tested individually, while a fully-cross strapped system needs to be tested as a whole. This has a direct impact on the complexity faced while developing and implementing each architecture, since fully cross-strapped architectures are more complex to develop and implement than channelized architectures. As mentioned earlier, the exchange of information between computers adds an additional layer of complexity, since it is necessary to implement the appropriate algorithms to handle the Byzantine Generals Problem.

Table 2. Comparison of failure detection, isolation and containment for different fault-tolerant architecting approaches.

Architecture	Failure detection	Failure isolation	Failure containment	FDIR testability	Complexity
Fully Cross-strapped	High	High	Moderate	Low	High
Fully Channelized	High	Low	High	High	Low
Channelized with computer cross-talk	High	High	Moderate	Moderate	High
Hybrid with cross-strapped sensors	High	Moderate	Moderate	Moderate	Moderate
Hybrid with cross-strapped actuators	High	Moderate	Moderate	Moderate	Moderate

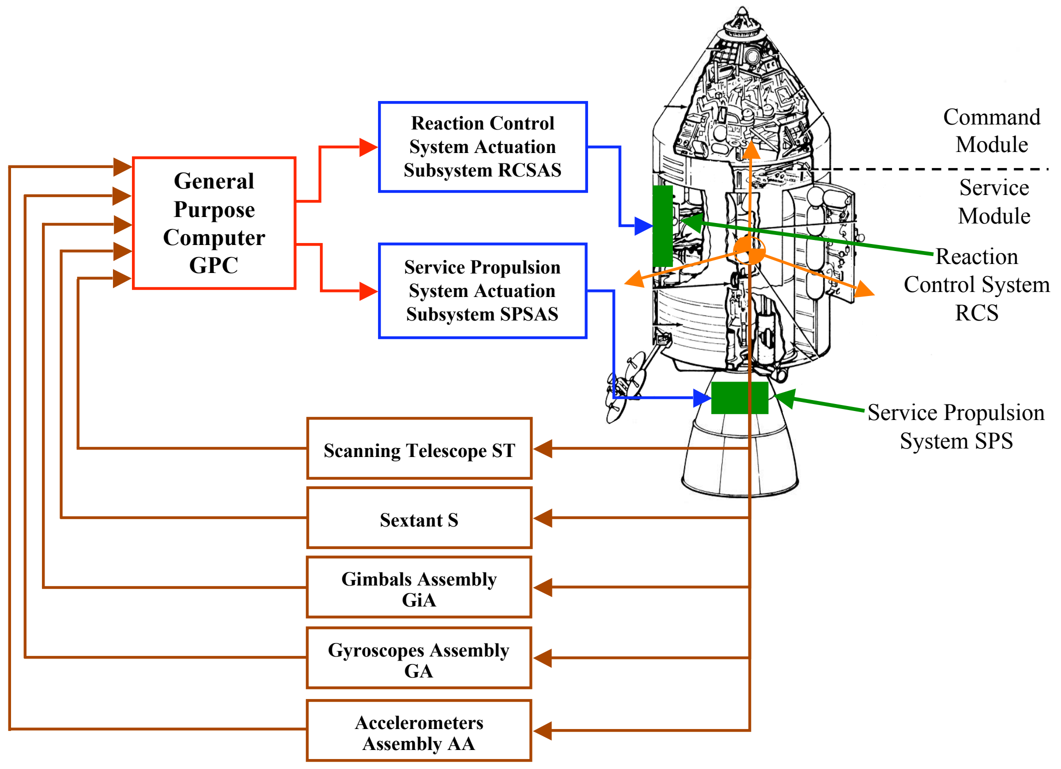

General Purpose Computer GPC
Reaction Control System Actuation Subsystem RCSAS
Service Propulsion System Actuation Subsystem SPSAS
Command Module
Service Module
Reaction Control System RCS
Service Propulsion System SPS
Scanning Telescope ST
Sextant S
Gimbals Assembly GiA
Gyroscopes Assembly GA
Accelerometers Assembly AA

Figure 4. Partial description of the Apollo Command and Service Module GN&C architecture: computer, sensors, reaction control, and service propulsion systems actuation subsystem interconnectivity. Diagram credit: NASA.

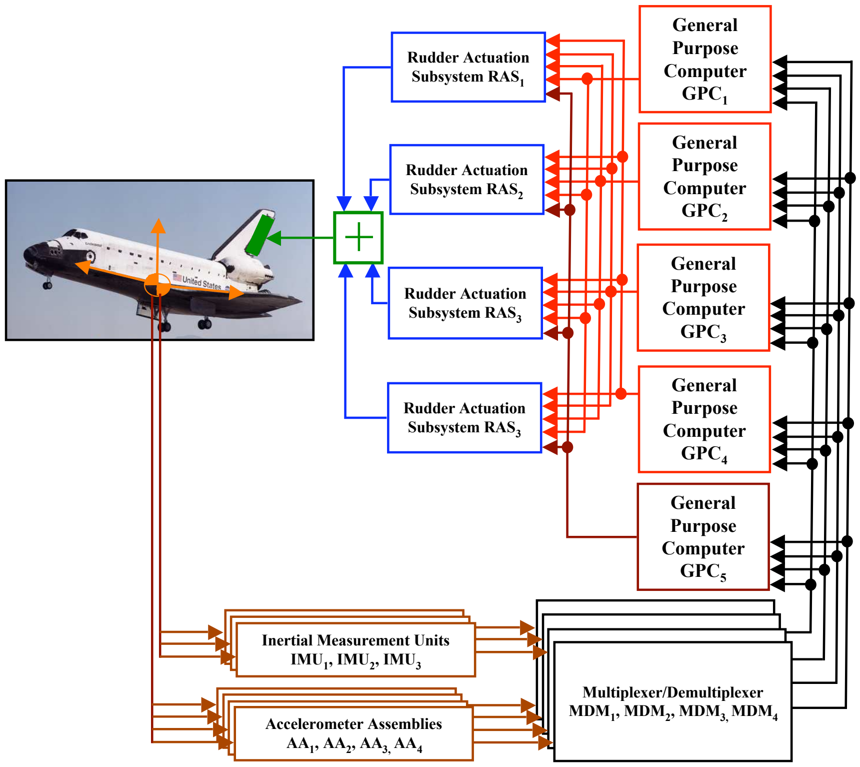
It is important to note that there are additional considerations in terms of cost, weight, and volume that we did not discuss here. Often, it is not immediately clear what the best architectural approach is for a particular GN&C system. For example, consider that the configuration and the complexity of the GN&C components will strongly influence spacecraft power needs. Thus, a structured system optimization design process is necessary to formulate a rationale for allocating scarce resources, such as power and mass. Therefore, it is necessary to carry out quantitative analyses and trade studies for each design and mission application to find the best solution in terms of performance, reliability, cost, weight, or any other important metrics.

IV. A Survey of GN&C Systems for Human-Rated and Robotic Spacecraft

The purpose of this section is to explain the main features of the GN&C system architectures of several human-rated and robotic NASA spacecraft. Rather than giving a detailed description of each architecture, this section focuses on the different design approaches used to achieve fault tolerance. For each of the described systems, several references are provided for the reader interested in further details.

IV.A. Apollo Command and Service Module (CSM): Single String

The Apollo CSM was a largely single-string system, yet its lack of redundancy should not be confused with a lack of fault tolerance. Subsystem specialists on the ground were constantly on the lookout for problems. Although the computer onboard the CSM could, and would, extrapolate the state vector, this extrapolation usually only occurred during time-critical mission phases and when the CSM was unable to communicate with Earth. The majority of state vector computation came from the larger and more accurate computer on the ground at the Manned Space Fight Network. This added accuracy improved the performance of the Apollo CSM. Furthermore, sensors and actuators on the Apollo CSM were rigorously tested, as were the integrated circuits used in its computer (which, to achieve improved reliability, were constructed using only one type of logical gate, see Ref 8 for details). Due to this testing, sensors, actuators, and computer were extremely unlikely to fail.



Rudder Actuation Subsystem RAS_1
Rudder Actuation Subsystem RAS_2
Rudder Actuation Subsystem RAS_3
Rudder Actuation Subsystem RAS_3
General Purpose Computer GPC_1
General Purpose Computer GPC_2
General Purpose Computer GPC_3
General Purpose Computer GPC_4
General Purpose Computer GPC_5
Inertial Measurement Units IMU_1, IMU_2, IMU_3
Accelerometer Assemblies AA_1, AA_2, AA_3, AA_4
Multiplexer/Demultiplexer MDM_1, MDM_2, MDM_3, MDM_4

Figure 5. Partial description of the Space shuttle GN&C architecture: computers, sensors, and control surface actuation subsystems interconnectivity. Photo credit: NASA Dryden Flight Research Center.

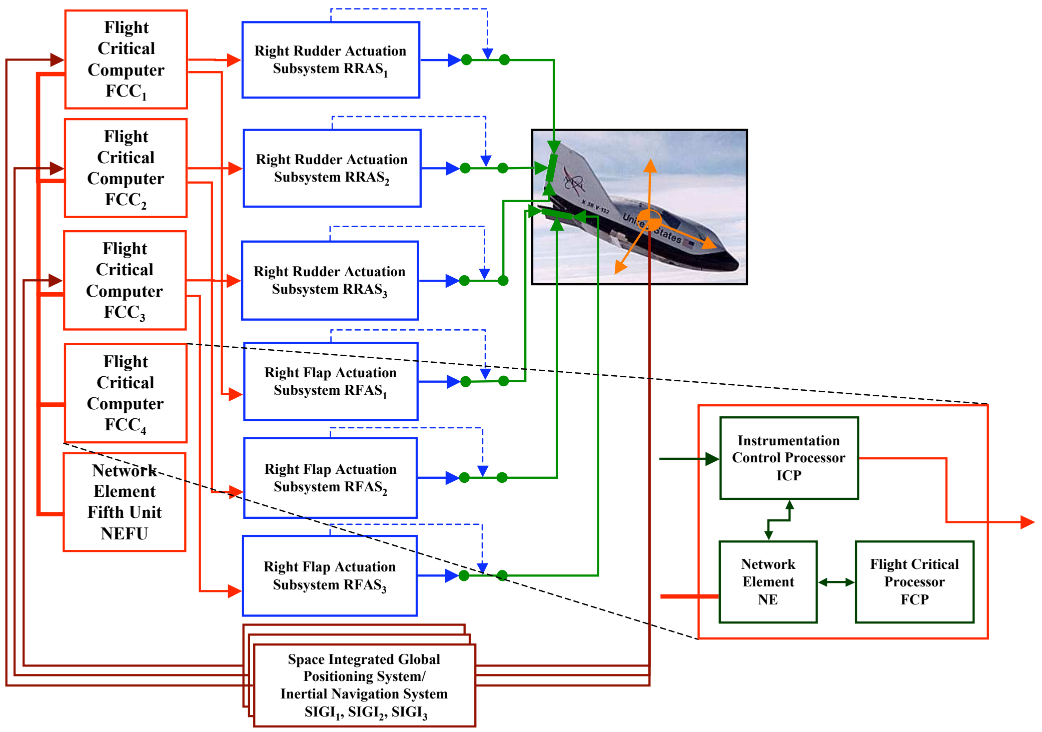
Figure 4 shows a small portion of the Apollo CSM GN&C system architecture. The primary sensor for the Apollo CSM GN&C system was the IMU, part of the inertial subsystem, and its function was to measure altitude, location, and attitude. It included a gyroscope assembly (GA) as well as an accelerometer assembly (AA). Its platform was mounted in a gimbals assembly (GiA) such that translational accelerations and rotation-rates could be measured in all six degrees of freedom. Drift errors, which occurred due to the mechanical nature of the device, were corrected periodically by collecting data with the two sensors in the optical subsystem: the scanning telescope (ST) and the sextant (S).[9]

There was only one general purpose computer (GPC) on the Apollo CSM. It ultimately accepted signals from all sensors (ST, S, GiA, GA, and AA) and hosted the necessary software to compute, based on sensor measurements, the spacecraft attitude. Based on the current attitude, the GPC would compute the appropriate control commands for the service propulsion system actuation subsystem (SPSAS), to be used by the service propulsion system (SPS) and also the appropriate commands for the reaction control system actuation subsystem (RCSAS). The SPS was responsible for pitch and yaw control during powered flight regimes. The main engine of the SPS was gimbaled and the precise direction and duration of main engine firings had to be determined by the computer. The RCS was the system responsible for the spacecraft attitude control during unpowered flight regimes.

A form of fault tolerance can be found in the Apollo CSM GN&C system. The system was designed so that each subsystem (inertial, optical and computer) could be operated independently during an emergency or backup mode. Therefore, the failure of any one subsystem would not disable the entire GN&C system.[10]

IV.B. Space Shuttle: Fully Cross-Strapped Architecture

Two of the requirements imposed during the design of the space shuttle were that the avionics system should remain fully operational after any single failure, and fully capable of a safe return to Earth after any two failures.[11] This meant that any safety-critical onboard system (including the GN&C system) ought to be two-fault-tolerant. Additionally, voting strategies were preferred as a means of failure detection. Thus, to achieve these design requirements, a fully cross-strapped philosophy, with quadruple redundant copies of almost every single component, was used in the design.



Flight Critical Computer FCC_1
Flight Critical Computer FCC_2
Flight Critical Computer FCC_3
Flight Critical Computer FCC_4
Network Element Fifth Unit NEFU
Right Rudder Actuation Subsystem $RRAS_1$
Right Rudder Actuation Subsystem $RRAS_2$
Right Rudder Actuation Subsystem $RRAS_3$
Right Flap Actuation Subsystem $RFAS_1$
Right Flap Actuation Subsystem $RFAS_2$
Right Flap Actuation Subsystem $RFAS_3$
Space Integrated Global Positioning System/ Inertial Navigation System $SIGI_1$, $SIGI_2$, $SIGI_3$
Instrumentation Control Processor ICP
Network Element NE
Flight Critical Processor FCP

Figure 6. Partial description of the Crew Return Vehicle (X-38 V201) GN&C architecture: computers, sensors, and control surface actuation subsystems interconnectivity. Photo credit: NASA Dryden Flight Research Center.

Figure 5 shows a small portion of the shuttle GN&C system architecture. All the necessary sensors for performing GN&C are triple or quadruple redundant; e.g., there are three Inertial Measurement Units (IMU_1, IMU_2 and IMU_3) and four accelerometer assemblies (AA_1, AA_2, AA_3 and AA_4). The information measured by the sensors is gathered by several Multiplexer / Demultiplexer interfaces (MDM_1, MDM_2, MDM_3 and MDM_4) before it is sent to the computers. Every Multiplexer/Demultiplexer is cross-strapped to five general purpose computers (GPC_1, GPC_2, GPC_3, GPC_4 and GPC_5). Four of these computers (GPC_1-GPC_4) perform the main GN&C functions, and the fifth one (GPC_5) is a backup only used to abort the mission if a common failure mode takes down the other four (GPC_1-GPC_4) at the same time. Each GPC implements mid-value selection voting algorithms as a means to detect and mask sensor failures. A failed sensor is taken out of the control loop so it will not cause detection problems when a second sensor fails. Each GPC has exact copies of the GN&C algorithms so, in nominal conditions, they should produce the same output commands for the different actuation subsystems. Quadruple redundancy is used for the actuation subsystems of the orbiter engines' thrust vector control, and aerodynamic control surfaces (Fig. 5 shows the rudder actuation subsystems RAS_1, RAS_2, RAS_3 and RAS_4). Each actuation subsystem receives commands from the four GPCs and issues a command to a four-port hydraulic valve (attached to the main power actuator) that will act as a "mechanical voter". If one of the four-port valve's input commands is erroneous, either by a failure in a GPC or by a failure in one of the actuation subsystems, then the other three will override the erroneous one and produce the right command to the main power actuator. A failed actuation subsystem is taken out of the control loop so a second failure can be overridden by the the two remaining "healthy subsystems". In summary, any GPC failure or any actuation subsystem failure is detected and isolated by the "mechanical voter" four-port hydraulic valve. The reader is referred to Ref. 11 for a more comprehensive explanation of all the features of this system.

IV.C. Crew Return Vehicle (X-38 V201): Channelized Architecture

A channelized architecture was used to design the X-38 V201's GN&C system. Figure 6 shows a portion of that system. The GN&C computer was designed to be two-fault tolerant,[12] however, the GN&C sensors and the control surface actuators were designed with an approach that allows full coverage of single failures and imperfect coverage of second failures.[13,14]

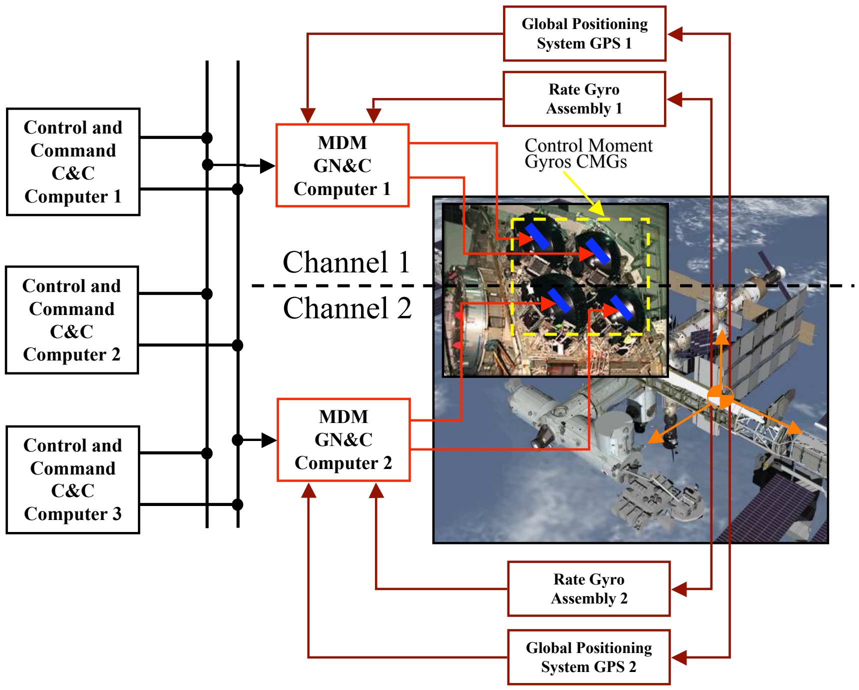
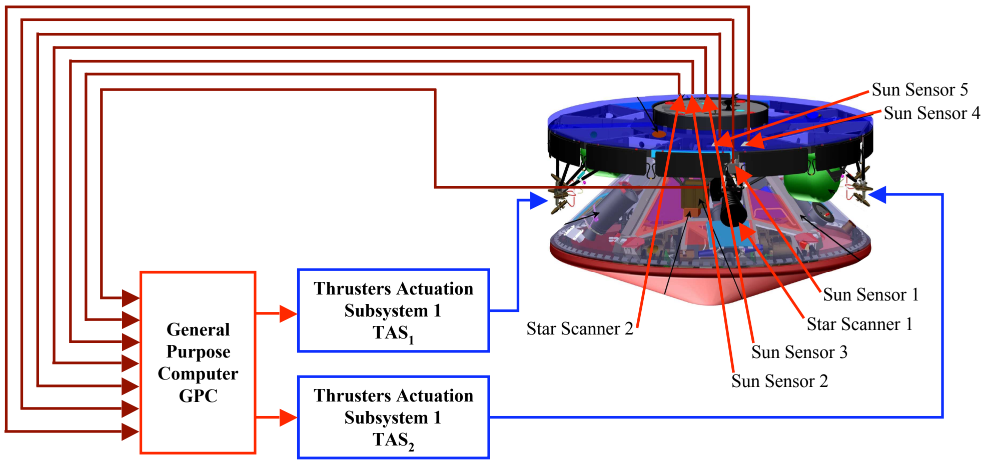

Global Positioning System GPS 1
Rate Gyro Assembly 1
Control and Command C&C Computer 1
MDM GN&C Computer 1
Control Moment Gyros CMGs
Channel 1
Channel 2
Control and Command C&C Computer 2
Control and Command C&C Computer 3
MDM GN&C Computer 2
Rate Gyro Assembly 2
Global Positioning System GPS 2

Figure 7. Partial description of the International Space Station GN&C architecture: computers, sensors, and control moment gyros interconnectivity. Photo Credits: (CMGs) Boeing Integrated Defense Systems, (ISS) NASA.

Three redundant space Integrated Global Positioning Systems / Inertial Navigation Systems ($SIGI_1$, $SIGI_2$ and $SIGI_3$) provide spacecraft position, velocity, acceleration, altitude, and attitude rates. The two-fault-tolerant computer is composed of four Flight Critical Computers (FCC_1, FCC_2, FCC_3 and FCC_4) and a Network Element Fifth Unit (NEFU).[15] Each FCC receives a single set of data from a single SIGI, i.e., the SIGIs are not cross-strapped to each computer. To illustrate this, in Fig. 6, it can be seen that the $SIGI_1$ is connected only to FCC_1, and similarly, FCC_2 and FCC_3 only receive readings from $SIGI_2$ and $SIGI_3$ respectively.

The architecture of each FCC is depicted on the right side of Fig. 6. In addition to a Network Element (NE) card, each FCC contains two processors: a Flight Critical Processor (FCP) and an Instrumentation Control Processor (ICP). The NEFU only contains an ICP and a NE, and it was added to achieve the two-fault-tolerance requirement. The ICP of each FCC is the I/O processor, which gathers information from the SIGI directly connected to it, and also outputs control commands to the control surface actuation subsystems and other GN&C actuation subsystems. The ICP of each FCC passes its own SIGI readings to the FCC-NE and then these readings are sent out to other FCC-NE (and also to the NEFU-NE). The exchange of information is done in two rounds to solve the Byzantine Generals Problem. After this exchange of data each FCC-NE have all three SIGI readings and a voting process is carried out to determine the correct measurements. Once each FCC-NE has determined the correct measurements, these are passed to the corresponding FCP. Each FCP contains the GN&C algorithms, and will compute the appropriate control commands for the actuation subsystems. After this, each FCP sends the computed commands to its own NE and another (one round) exchange of data between each FCC-NE takes place. Then each FCC-NE votes the correct commands and these commands are passed to the corresponding actuation subsystems. This arrangement allows separation of the GN&C algorithm development from the FDIR algorithm implementation.[12]

Triple redundant actuation subsystems are used to position each control surface. There are two rudders and two body flaps. Figure 6 displays the right rudder and right flap and their corresponding actuation subsystems. Each control surface actuation subsystem receives a command from a single computer, e.g., $RRAS_1$ receives a command from FCC_1, $RRAS_2$ from FCC_2, and $RRAS_3$ from FCC_3. Each actuation subsystem is connected to the respective control surface through a clutch. If one of them fails, the other two can overpower it and the faulty actuation subsystem will be removed by disengaging its clutch. Built-In-Test failure detection is used to detect a second actuation subsystem failure.



Sun Sensor 5
Sun Sensor 4
General Purpose Computer GPC
Thrusters Actuation Subsystem 1 TAS_1
Thrusters Actuation Subsystem 1 TAS_2
Star Scanner 2
Sun Sensor 1
Star Scanner 1
Sun Sensor 3
Sun Sensor 2

Figure 8. Partial description of the Mars exploration rovers cruise stage GN&C architecture: computers, sensors, and actuation subsystems interconnectivity. Drawing Credits: NASA Jet Propulsion Laboratory.

IV.D. International Space Station (ISS) US Segment: Channelized Architecture

Channelization was used to design the GN&C system architecture of the ISS US segment. The channelization concept used in the ISS comes from the design baseline of its predecessors, the Space Station Freedom and the International Space Station Alpha.[16,17]

Figure 7 shows a high-level description of the ISS US segment GN&C system. Three dual-pair-processor computers perform the control and command function of the ISS. These three computers can independently control any of the two GN&C system channels thorough a Multiplexer/ Demultiplexer GN&C Computer that receives information about the ISS attitude from a Ground Positioning System (GPS) and from a Rate Gyro Assembly (RGA). Each MDM will generate the appropriate control commands for a pair of Control Moment Gyros (CMGs) that control the attitude of the station.

IV.E. Mars Exploration Rovers Cruise Stage: Single String

Like the Apollo CSM, the cruise stage of the Mars Exploration Rovers was largely single string. There is some redundancy in the sensors. The star scanner has a backup system and there are five sun sensors. There is only one computer, but there are two major fault protection algorithms for command loss and battery charge control.[18] Looking at the actuators, there was some redundancy here as well. The eight 1 lbf thrusters, organized in two clusters of four, were all valved to provide redundancy in case of a leaky thruster.[19] The transportation system of the Mars Exploration Rovers is very similar to that of Mars Pathfinder. See Fig. 8 for a top-level diagram and schematic of the cruise stage.

Table 3. Architecting approaches and fault tolerance level for GN&C systems for human-rated and robotic spacecraft

Spacecraft	Architecture	Computer FT level	Sensors FT level	Actuators FT level
Apollo CSM	Single String	0	0	0
Shuttle	Cross Strapped	2	2	2
ISS	Channelized	1	1	2
X-38 V201	Channelized with computer cross-talk	2	1 (limited second)	1 (limited second)
CEV	TBD (Cross-strapped or single String)	TBD	TBD	TBD
Mars Pathfinder	Single String	0	0	0
Mars Exploration Rover	Single String	0	1	1
Phoenix	Single String	0	0	0

V. Observations

The second column of Table 3 displays the architectural approaches for GN&C systems of different human-rated and robotic spacecraft. Columns 3–5 display a breakdown of the levels of fault tolerance required for GN&C subsystems, specifically, sensors, computers, and actuators.

The human-rated spacecraft discussed in this paper were designed using different architectural approaches. Apollo CSM was essentially single string, with limited fault tolerance. If a failure occurred, the mission would be aborted by means of dissimilar backup mechanisms, with degraded performance compared to the primary system, and the spacecraft and crew could be safely brought back to Earth. The lack of fault tolerance in Apollo can be explained by the fact that fault-tolerant computing was a relatively new and emerging field when the Apollo program was being developed. It was not until 1967 (well into the Apollo Program's hardware development phase) that the concept of fault-tolerant computers was formalized.[20] Alternatively, to ensure a high-level of reliability, the Apollo program used a high-quality-parts production process.

By the time the space shuttle was under development, the fault tolerance field was more mature. Furthermore, maneuvers such as unpowered landing and entry through final approach demanded stringent performance requirements not just on the main systems, but on the backup systems as well. Thus, backup mechanisms with degraded performance were no longer acceptable. This requirement, together with the fact that aborting a mission after one failure was unacceptable, led to the introduction of a fully cross-strapped, two-fault tolerant architecture for all flight critical subsystems.[11]

A channelized architectural approach was used for designing the ISS GN&C system. The requirements on the FDIR mechanisms are not as stringent for ISS as for the shuttle, and therefore the GN&C architecture is only one-fault tolerant. The requirements are less stringent because the ISS has slow dynamics, so real-time failure detection and isolation are not as critical as they are in the space shuttle, where an undetected failure during reentry could be catastrophic. Furthermore, the station crew can perform repair activities in sensors and computers without additional ground support.

The X-38 program was meant to develop a reusable crew return vehicle (CRV) for the ISS. A channelized architecture with computer cross-talk was used for the GN&C system. The computer system is two-fault tolerant, while one-fault tolerance, with limited second failure coverage, was implemented for sensors and actuators. The limited fault tolerance of the primary navigation and control system is supplemented by a steerable parafoil, which provides a dissimilar backup mechanism for controlling the spacecraft attitude after reentry. Furthermore, the space shuttle was responsible for delivering the CRV to the ISS. Thus, one additional design requirement was to comply with the shuttle payload weight and size requirements. This may have also had an impact on the level of redundancy used for controlling the position of the control surfaces.

The crew exploration vehicle (CEV) is still under development, and no final decision has been made on the design baseline for the GN&C system. A summary of the GN&C reference design architecture as of 2006 can be found in Ref. 21. The architecture of the CEV GN&C system is still evolving, but it appears likely that a cross-strapped architecture will be implemented. This architecture will have a yet-to-be determined number of computers implementing lock-step processors (self-checking pairs) to achieve a high-level of failure coverage. Thus, no voting mechanisms will be used for computer failure detection.

Single-string architectures have been used for Mars robotic spacecraft with zero computer fault tolerance, and limited fault tolerance at the sensor and actuator level. Two drivers for using single-string architectures are the stringent weight and volume requirements for these missions. Additionally, as is the case in the Mars Exploration Rover mission, fault tolerance was achieved by sending two separate spacecraft, each with its own rover.

The architectural philosophies that have been used for designing the GN&C systems used for NASA's various human-rated spacecraft over the years are substantially different. This is likely due to two factors. First, there are gaps of many years, and in some cases decades, between the design cycles of NASA's human-rated spacecraft systems. As a result of these gaps, NASA and its industry partners must periodically re-learn the process of architecting human-rated spacecraft. Second, rapid technology developments since the 1960s, especially in electronics and avionics, have strongly influenced the design process, resulting in significant architectural changes over the last four decades.

On the other hand, robotic missions of the same class appear to have a very similar architectural basis. For example, the Mars missions, such as Pathfinder, and the Mars Exploration Rovers have similar architectures. This similarity is likely due to several factors. First, the development cycles are shorter compared to human-rated systems. Second, because of the shorter time between development cycles, there is less technological change between two designs. Third, a wide variety of mission-unique robotic spacecraft GN&C architectures are designed, implemented, and flown each year by NASA and the same teams of contractors. The result is that NASA and its contractors have a substantial and diverse (by mission class) GN&C engineering experience base for robotic spacecraft applications. Thus, legacy designs are often adopted for robotic missions. Finally, it should noted sometimes there are few viable choices for GN&C system components. For example, Pathfinder, Mars Polar Lander, etc., all use the same single board CPU, in part due to the expense of modifying the layout of a commercial chip to make it tolerant to the space environment.

As would be expected, the GN&C architectures for robotic spacecraft are simpler than for human-rated missions in terms of fault tolerance. In human-rated missions, crew safety is paramount, thus imposing a higher requirement on fault tolerance than in robotic missions, even at the expense of cost, weight, and complexity. On the other hand, significantly more risk is acceptable in robotic missions. Furthermore, risk can sometimes be reduced in robotic missions by an extreme form of channelization, namely, flying multiple, non-redundant spacecraft.

VI. Concluding Remarks

This paper summarizes the first steps in our study of commonality in GN&C systems, for both human-rated and robotic spacecraft. As discussed, the fault tolerance and reliability requirements are the main factors driving the architectural differences between the GN&C systems of human-rated and robotic spacecraft. We defined, from a component interconnectivity point of view, the two main philosophies — cross-strapping and channelization — used to achieve fault tolerance, explaining advantages and disadvantages of each of the resulting architectural design approaches for GN&C systems. This definition helped us to classify the GN&C systems of different NASA spacecraft, and to understand some of the driving factors that led the design teams to choose a particular architecture.

Future work will include a more detailed understanding and analysis of the options available to implement the three main GN&C subsystems, namely sensor, computer, and actuator subsystems. This analysis will help us to understand the options for commonality at the subsystem level. Based on the high-level architectural analysis presented in this paper, and the more detailed work to be conducted at the subsystem level, it will be possible to enumerate numerous feasible architectures for the GN&C systems of the new generation of space vehicles to be developed under NASA's CxP. The ultimate task of the ongoing work will be to carry out a more detailed analysis of performance, reliability, and commonality of this set of architectures, to better inform the design of GN&C systems for the CxP vehicles.

Acknowledgments

The authors wish to acknowledge the encouragement and contributions to this paper from Neil Dennehy of the NASA Engineering and Safety Center, especially on the top-level comparison of the the two types of spacecraft GN&C systems as discussed in Section II. The authors would like to thank also John West and Joel Busa at the Charles Stark Draper Laboratory for fruitful discussions of this research. Gregory W. Vajdos from Boeing Integrated Defense Systems and Susan Gomez provided information on the International Space Station GN&C system, which was also appreciated. Robert Rasmussen and Robert Manning of NASA Jet Propulsion Laboratory provided valuable information about the robotic Mars missions.

References

[1]Gai, E. and Adams, M., "Measures of Merit for Fault-Tolerant Systems," Tech. Rep. CSDL-P-1752, The Charles Stark Draper Laboratory, Cambridge, MA, 1983.

[2]Hammett, R., "Design by Extrapolation: An Evaluation of Fault-Tolerant Avionics," *IEEE Aerospace and Electronic Systems Magazine*, Vol. 17, No. 4, April 2002, pp. 17–25.

[3]Pahami, B., "Voting Algorithms," *IEEE Transactions on Reliability*, Vol. 43, No. 4, December 1994, pp. 617–629.

[4]Willsky, A., "A Survery of Design Methods for Failure Detection Systems," *Automatica*, Vol. 12, No. 6, November 1976, pp. 601–611.

[5]Laprie, J., editor, *Dependability: Basic Concepts and Terminology*, Springer-Verlag, New York, NY, 1991.

[6]Domínguez-García, A., Kassakian, J., Schindall, J., and Zinchuk, J., "On the Use of Behavioral Models for the Integrated Performance and Reliability Evaluation of Fault-Tolerant Avionics Systems," *Proceedings of DASC 2006*, Portland, OR, 2005.

[7]Høyland, A. and Rausand, M., *System Reliability Theory*, John Wiley and Sons, New York, NY, 1994.

[8]Williamson, M., "Aiming for the Moon: the Engineering Challenge of Apollo," *Engineering Science and Education*, Vol. 11, No. 5, October 2002, pp. 164–172.

[9]Holley, M., Swingle, W., Bachman, S., Leblanc, C., Howard, H., and Biggs, H., "Apollo Experience Report - Guidance and Control Systems: Primary Guidance, Navigation, and Control System Development," Tech. Rep. NASA TN D-8227, National Aeronautics and Space Administration, Washington, DC, May 1976.

[10]Wilson, R., "Apollo Experience Report - Guidance and Control Systems," Tech. Rep. NASA TN D-8249, National Aeronautics and Space Administration, Washington, DC, June 1976.

[11]Hanaway, J. and Moorhead, R., "Space Shuttle Avionics System," Tech. Rep. NASA SP-504, National Aeronautics and Space Administration, Washington, DC, 1989.

[12]Kouba, C., Buscher, D., Busa, J., and Beilin, S., "The X-38 Spacecraft Fault-Tolerant Avionics System," *Proceedings of the Military and Aerospace Programmable Logic Device International Conferences*, Washington, DC, 2003.

[13]Bedos, T. and Anderson, B., "X-38 V201 Avionics Architecture," Tech. Rep. NASA-20000086667, National Aeronautics and Space Administration, Houston, TX, February 1999.

[14]Goodman, J., "GPS Lessons Learned from the International Space Station, Space Shuttle and X-38," Tech. Rep. NASA/CR-2005-213693, National Aeronautics and Space Administration, Houston, TX, November 2005.

[15]Racine, R., Leblanc, M., and Beilin, S., "Design of a Fault-Tolerant Parallel Processor," *Proceedings of the Digital Avionics Systems Conference*, Irvine, CA, 2002.

[16]Babcock, P., "Channelization: Two-Fault Tolerant Attitude Control function for the Space Station Freedom," *IEEE Aerospace and Electronic Systems Magazine*, Vol. 11, No. 5, May 1996, pp. 9–22.

[17]Smith, J., Suchting, S., McDonald, M., and Schikner, J., "Avionics Architecture for the U.S. Segment of the International Space Station Alpha," *Proceedings of the 10th Computing in Aerospace Conference*, San Antonio, TX, 1995.

[18]Muirhead, B., "Mars Pathfinder Flight System Design And Implementation," *Proceedings of the IEEE Aerospace Conference*, Snowmass at Aspen, CO, 1996.

[19]Muirhead, B., "Mars Pathfinder Flight System Integration and Test," *Proceedings of the IEEE Aerospace Conference*, Snowmass at Aspen, CO, 1997.

[20]Avižienis, A., "Design of Fault-Tolerant Computers," *Proceedings of the Fall Joint Computer Conference, AFIPS Conference*, Washington, DC, 1967, pp. 733–743.

[21]Tamblyn, S., Hinkel, H., and Saley, D., "NASA CEV Reference GN&C Architecture," *Proceedings of the 30th Annual American Astronautical Society Guidance and Control Conference*, Breckenridge, CO, 2007.