DOT/FAA/AR-09/27

Air Traffic Organization
NextGen & Operations Planning
Office of Research and
Technology Development
Washington, DC 20591

Data Network Evaluation Criteria Report

July 2009

Final Report

U.S. Department of Transportation
Federal Aviation Administration

NOTICE

This report is available at the Federal Aviation Administration William J. Hughes Technical Center's Full-Text Technical Reports page: actlibrary.tc.faa.gov in Adobe Acrobat portable document format (PDF).

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
DOT/FAA/AR-09/27		

4. Title and Subtitle	5. Report Date
DATA NETWORK EVALUATION CRITERIA REPORT	July 2009
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Kevin Driscoll, Brendan Hall, Phil Koopman, Justin Ray, and Mike DeWalt	

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
Honeywell International, Inc. 3660 Technology Drive Minneapolis, MN 55418	
	11. Contract or Grant No. DTFACT-05-C-00002

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
U.S. Department of Transportation Federal Aviation Administration Air Traffic Organization NextGen & Operations Planning Office of Research and Technology Development Washington, DC 20591	Final Report
	14. Sponsoring Agency Code AIR-120

15. Supplementary Notes

The Federal Aviation Administration Airport and Aircraft Safety R&D Division COTR was Charles Kilgore.

16. Abstract

Databus and data network technology continues to play an ever-increasing role in aviation digital electronics architectures. The evolution of integrated modular aviation digital electronics architectures with multiple subsystem integration into single and redundant data networks is increasing the influence of the data network at the airframe level. The criticality of data networks has long been recognized in the avionics industry. This previously has led avionics manufacturers and aircraft original equipment manufacturers to design specific aerospace solutions to meet these requirements.

However, the costs associated with such special-purpose designs run contrary to the current cost pressures on aviation digital electronics. In recent years, these cost challenges have led to the adoption of commercial off-the-shelf (COTS) data communication solutions for aviation digital electronics. Although attractive from a cost perspective, the adoption of COTS present issues, particularly as the complexity and increased leverage of technology continues to evolve. Subtleties in data network characteristics may escape the system designer and leave dependability holes. Allied with this trend is the emergence in the consumer marketplace of COTS labeled as "safety-critical COTS." As industries face increasing safety requirements while maintaining low cost due to large volumes, they may develop low-cost, safety-critical data networks that could be applicable to aviation digital electronics. However, it is necessary to ensure that the technology is consistent with real-world requirements for safety-critical aviation digital electronics systems.

This report covers a 2-year research effort aimed at creating evaluation criteria to be used in assessing the design of existing and new data networks for their applicability to safety-critical aviation digital electronics systems. This report contains a list of issues and evaluation criteria created by these activities.

17. Key Words	18. Distribution Statement
Databus, Network, Certification, COTS	This document is available to the U.S. public through the National Technical Information Service (NTIS) Springfield, Virginia 22161.

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	234	

TABLE OF CONTENTS

		Page
EXECUTIVE SUMMARY		xi
1.	INTRODUCTION	1
1.1	Report Organization	1
1.2	Data Network Background	2
1.3	System Safety and Dependability Background	4
1.3.1	Network Evaluation Relative to a System Safety Process	4
1.3.2	Determining Data Network Dependability Requirements	6
1.4	Scope	7
1.4.1	System Network Role	7
1.4.2	Protocol Stack	8
1.4.3	Developmental Time Horizon	9
1.5	Data Network Certification Issues in Context	9
1.5.1	Supported Application Requirements	9
1.5.2	Multiple-Requirement Engineering Trades	10
1.5.3	System Architecture and Design	11
2.	TASK ACTIVITIES	13
2.1	Phase 1	13
2.1.1	Literature Survey	13
2.1.2	Industry Survey	14
2.1.3	Development of Primary Issues	14
2.1.4	Phase 1 Report	20
2.2	Phase 2	20
2.2.1	Literature Survey Update	20
2.2.2	Condensing the Criteria List	20
2.2.3	Evaluation of Criteria by Applying to Networks	21
2.2.4	Criteria Update and Handbook Creation	24
3.	PHYSICAL LAYER	24
3.1	Environment	25

3.2 Probability of Bit Errors 25
3.3 Probability of Electrical Component Failures 29
3.4 Electrical Isolation Properties 30
3.5 Physical Composability 31

4. DATA LINK LAYER 31

4.1 Media Access Control 32

4.1.1 Problems Unique to Each Type of MAC 32
4.1.2 The MAC Replacements 34

4.2 Line-Level Encoding 35
4.3 Message Formatting (Framing) 35
4.4 Error Detection 36

4.4.1 Protocol Violation Error Detection 36
4.4.2 Parity and Frame Check Sequences 36
4.4.3 Interactions Between Line-Level Encoding and Error Detection 37

5. NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT 38

5.1 Network Vulnerability to Addressing Information Failure 39
5.2 Network Vulnerability to Flow Failure 40
5.3 Impact of Intermediate Stages 40

5.3.1 Vulnerability to Intermediate-Stage Failure 40
5.3.2 Vulnerability of Intermediate Stage to Fault Propagation 43

5.4 Network Configuration Data 43
5.5 Start-Up and Recovery 44
5.6 Global Synchronization 46
5.7 Fault Diagnosis 48
5.8 Client Effect on Network Operations 50
5.9 Acknowledgement 50

6. APPLICATION-LAYER SERVICES 51

6.1 Host Interface Management 51

6.1.1 Client Buffer Queue Management 52
6.1.2 Buffer Management Partitioning 52
6.1.3 Buffer Management Performance Considerations 54

6.2 Support for Application-Layer Redundancy 54

6.2.1 Support for Active Replication 54
6.2.2 Support for Passive Replication 55

6.2.3 Support for Increased Integrity 55
6.2.4 Support for Robust Partitioning 56

6.3 Time Service for Time Stamping and Time Interrupts 56

7. FAULT TOLERANCE MECHANISMS 56

7.1 Topological Fault Tolerance 56
7.2 Guardian Schemes 57
7.3 Protocol Logic Fault Tolerance 59
7.4 Local Transmission Monitoring and Self-Checking Schemes 59
7.5 Reconfiguration and Degraded Operation 60
7.6 Latent Failure Detection 60
7.7 Voting, Selection, or Agreement Services and Redundancy Management 61
7.8 Byzantine Fault Tolerance 62

8. DESIGN ASSURANCE 64

8.1 Design Assurance Processes 64
8.2 Availability of Standards and Conformance Evidence 65

8.2.1 Open Specification and Standardization 65
8.2.2 Conformance and Interoperability Testing 65
8.2.3 Protocol Design Correctness 65

8.3 Design Margin 66
8.4 Configuration Table Correctness and Performance Justification 66
8.5 Network Monitoring and Test Equipment 68

9. SECURITY 68

10. DISCUSSION 69

11. RECOMMENDATIONS 70

12. REFERENCES 70

13. GLOSSARY 72

APPENDICES

A—Literature Survey Annotated Bibliography
B—Survey Responses
C—Original Evaluation Criteria
D—Condensed Evaluation Criteria
E—Updated Evaluation Criteria
F—Evaluation Criteria Test Results

LIST OF FIGURES

Figure		Page
1	Eye Pattern	26
2	Error Burst Length Extension due to Encoding	38
3	Schrödinger's CRC	63

LIST OF TABLES

Table		Page
1	The Seven-Layer ISO OSI Model	8

LIST OF ABBREVIATIONS AND ACRONYMS

AFDX	Avionics Full-Duplex Switched Ethernet
ARINC	Aeronautical Radio Incorporated
ASCB	Aircraft systems control bus
BER	Bit error rate
BGP	Byzantine Generals Problem
CAN	Controller area network
CAST	Certification Authorities Software Team
CCA	Common cause analysis
CFR	Code of Federal Regulations
CMOS	Complementary metal-oxide semiconductor
COTS	Commercial off-the-shelf
CRC	Cyclic redundancy check
DC	Direct current
DMA	Direct memory access
EDAC	Error detection and correction
EMI	Electromagnetic interference
FAA	Federal Aviation Administration
FAR	Federal Aviation Regulations
FCS	Frame check sequence
FDIR	Failure detection, isolation, and recovery
FIFO	First in, first out
FMEA	Failure modes and effects analysis
FMECA	Failure modes, effects, and criticality analysis
FTA	Fault tree analysis
HD	Hamming distance
HIRF	High-intensity radio frequency
IC	Integrated circuit
ID	Identification
IEEE	Institute of Electrical and Electronics Engineers
IO	Input/output
ISI	Intersymbol interference
ISO	International Standards Organization
LLC	Logical link control
MAC	Media access control
NASA	National Aeronautics and Space Administration
OSI	Open Systems Interconnect
PCI	Peripheral component interconnect
PHAC	Plan for hardware aspects of certification
PSAC	Plan for software aspects of certification
PPP	Point-to-point Protocol
PSSA	Preliminary system safety analysis
RF	Radio frequency
RMA	Rate monotonic analysis
ROBUS	Reliable Optical Bus

SAE	Society of Automotive Engineering
SERDES	SERializer and DESerializer
SEU	Single event upset
SMV	Symbolic model verification
SNMP	Simple Network Management Protocol
SNR	Signal-to-noise ratio
SOS	Slightly-off-specification
SPIDER	Scalable Process-Independent Design for Electromagnetic Resilience
SWIFI	Software-implemented fault injection
TCN	Train Control Network
TT	Time-triggered
TTA	Time-triggered architecture
TDMA	Time Division Multiple Access
TMR	Triple modular redundancy
TTP/C	Time-triggered protocol, SAE Class C
XOR	Exclusive or

EXECUTIVE SUMMARY

Databus and data network technology continues to play an ever-increasing role in aviation digital electronics architectures throughout the range of aviation markets. The evolution of integrated modular aviation digital electronics architectures with multiple subsystems integration into single and redundant data networks is increasing the influence of data networking. The criticality of data networks has previously led avionics manufacturers and aircraft original equipment manufacturers to design specific aerospace solutions to meet their requirements. In recent years, cost challenges have led to the adoption of commercial off-the-shelf (COTS) communication solutions in avionics. Although attractive from a cost perspective, the adoption of COTS presents certification issues, particularly as the complexity and increased leverage of technology continues to evolve. Subtleties may elude the system designer and leave dependability holes. An example is the interference of the Controller Area Network (CAN) bit-error stuffing mechanism with message cyclic redundancy code coverage. COTS can be adopted as-is, or fixes added so that it is a better fit for dependable avionics requirements, i.e., the adaptation of Ethernet to ARINC 664 part 7. Helping this trend is the arrival of "safety-critical COTS" in the marketplace, particularly in automobile and process-control areas. However, even with designed-for-purpose technology, it is necessary to ensure that the technology has dependability consistent with real-world requirements and redundancy management schemes.

Development and evaluation of aviation digital electronics data networks that are suitable for safety-critical aviation digital electronics is a complex subject area. It requires detailed knowledge of communications systems, aviation communication and application requirements, mechanisms for creating dependable architectures, and certification expectations and assurance strategies. It is also important to note that, with correct architectural mitigation, almost any data network may be used in a certified system. For example, a layer of fault tolerance can be placed above the network to fix any of its shortcomings.

The objective of this task was to develop criteria for evaluating data network technology for use in safety-critical applications. However, this should not be taken to mean that these criteria can be used to rank data networks in some scale of absolute goodness independent of the avionics systems in which they are employed. The operation of a data network is so entangled with the avionics system it supports that it is not possible to make an evaluation of a data network on its own. The goal is to create a sufficient breadth of criteria that can be used to evaluate the widest range of data networks with respect to avionics systems in which they may be employed.

To develop the evaluation criteria, a wide range of networks and communications solutions were considered using personal knowledge, a literature review, and interviews of other knowledgeable people in this area (both informally and via a written survey). The intent was to cover a suitable breadth of behavior such that the key attributes of the communication scheme could be extracted. Many of these criteria and their related issues are overlooked or are underappreciated by many of today's aviation digital electronics designers. An initial set of criteria, in the form of over 200 questions to be asked of the network and how it fits into an architecture, was reduced to a hierarchical form that consisted of 36 criteria, each with a set of supporting questions. These criteria and questions were then applied against seven diverse types of data networks to ensure

that the criteria and their supporting questions met a number of usability metrics. The lessons learned from this application exercise were used to refine the criteria.

This application exercise also confirmed the suspicion that there is a greater diversity in the ways that data networks can be designed than there are for microprocessors. This means that the decision for selecting a data network is even more complex than for a data processor.

This exercise further reinforced that the details cannot be ignored when examining a very complex component that forms a major part of an avionics system's infrastructure.

1. INTRODUCTION.

The purpose of this task was to develop and document objective evaluation criteria for networks to be used in aviation products. Of particular interest were the safety-critical aviation digital electronics applications. The evaluation of databus and networking technology is not a simple matter. It requires a detailed review and analysis of the lowest-level implementation characteristics of the selected technology together with the ability to map significant behaviors and failures up to their architectural relevance. From the list of behaviors, and beginning with the Certification Authorities Software Team (CAST) Position Paper CAST-16 [1] as a departure point, an examination was made of how communications primitives and services can be leveraged at the application layer, and what impacts the behaviors many introduce with respect to certification. Information used for this examination was obtained from previous experience, a literature search, and an industry survey that was conducted during Phase 1. From this information, the primary issues and outline were developed. This outline was refined through several iterations to create a framework for this examination.

For the purposes of this report, the term "evaluation criteria" means the standards on which a judgment can be made regarding the suitability of a data network for use in aviation digital electronics systems, given the characteristics or features of the data network that may have impact on system safety. One cannot definitively say that a particular characteristic or feature would have a safety impact, because the architecture in which the network is used may be insensitive to (e.g., may not need) the particular characteristic or feature that would be a problem for other architectures. Thus, this report will describe all the evaluation criteria that need to be considered, regardless of any particular architecture. The system designer must then determine whether a particular evaluation criterion is applicable to the data network being evaluated and the system being designed.

These criteria form the nucleus of the Data Network Evaluation Handbook [2] produced as the primary output of this task.

1.1 REPORT ORGANIZATION.

Section 1 gives the introduction that provides the basis of understanding of the report.

Section 2 describes the activities undertaken during this task.

Sections 3 to 8 present a detailed discussion of databus and network technology attributes that must be considered when evaluating the technology within aviation digital electronics systems. This information is organized: first, in relation to the hierarchies of communication stack models, for those evaluation criteria that fit well to these models; and second, by themes of special interest that require attention in the system design and deployment. Organizing the criteria along a communication stack model should make them easier to find and to correlate against communication network description documents, which are often organized in this way. However, a pure communication-stack model approach misses essential attributes of data network design. Indeed, it is those areas of special interest that are most likely to be missed. Existing data network technologies are very diverse, so this report organization encompasses a large number of issues that may not be applicable in every case, or may apply differently with

respect to various technologies. Due to this design diversity and the fact that each data network implementation may itself be highly configurable, generating evaluation criteria is more complex than evaluating individual implementation decisions—the synthesis of the entire system must be considered.

Section 9 provides the security backdrop for the use of data networks in aircraft.

Section 10 provides a final discussion.

Section 11 provides recommendations for further work.

Section 12 lists the references.

Section 13 provides a glossary of technical terms used throughout this report.

1.2 DATA NETWORK BACKGROUND.

The reader is assumed to have a basic understanding of networks' physical and architectural characteristics. For the details of the networks listed here and networks in general, extensive bibliographies have been provided as part of this report. The specific references within this report are listed in section 11, and an annotated bibliography resulting from the literature search, can be found in appendix A.

To develop the evaluation criteria, a wide range of network and communication solutions need to be considered. The intent was to cover a suitable breadth of behavior such that the key attributes of the communication scheme can be extracted. The following list is representative of the communication solutions studied to extract these behaviors.

- SAFEbus®[1] (ARINC®[2]-659) [3 and 4] formed the backplane of the world's first certified integrated avionics platform, which is flown today on the Boeing 777. It is a very efficient fault-tolerant, time-triggered protocol.

- ARINC 629 [5] is a time division databus using a waiting room protocol to access the communication medium of terminals. The databus provides a broadcast service on a linear databus and enables raw data rates of 2 Mbps.

- ARINC 429 [6] enables digital communication between a single source and single sink and provides raw data rates of 12.5 and 100 Kbps.

- Time-Triggered Protocol Society of Automobile Engineering (SAE) Class C (TTP/C®[3]) [7] is a fault-tolerant, time-triggered protocol that provides fault-tolerant global time base, a membership service, and clique avoidance to detect and eliminate the formation of cliques in case the fault hypothesis is violated. It will be used in several avionics

[1] SAFEbus is a registered trademark of Honeywell International, Inc.
[2] ARINC is a registered trademark of Aeronautical Radio, Inc.
[3] TTP is a registered trademark of FTS Computertechnik G.m.b.H.

applications; e.g., for cabin pressure control, engine control, and the full avionics system for general aviation aircraft.

- Contoller Area Network (CAN) [8] is an event-triggered, serial broadcast bus widely used in the automotive and industrial industry, which uses a binary countdown protocol for link access. The link access is priority-based, demanding a recessive and dominant physical-layer state. CAN includes some error detection mechanisms.

- TT-CAN [9] is a time-triggered protocol that is built on top of the CAN data link layer.

- FireWire®[4], Institute of Electrical and Electronics Engineers (IEEE) 1394b [10 and 11] allows isochronous and asynchronous traffic transfer. Of particular note is the SAE variant and its use in the Joint Strike Fighter, F-35. It is characterized by high bandwidth and reliable data transfer, and has been constructed to move data to or through systems without disruptions when they are busy with other tasks and applications.

- FlexRay®[5] [12] and Byteflight®[6] [13] are a pair of protocols, with Byteflight being a simpler protocol and FlexRay being an expanded protocol intended for new applications in vehicles. FlexRay contains both event-triggered and time-triggered capability. It is being developed by a consortium including BMW®, DaimlerChrysler®, and Motorola®. It has been gaining acceptance in the automotive industry as a high-speed, fault-tolerant, deterministic communications network.

- The Scalable Processor-Independent Design for Electromagnetic Resilience (SPIDER) [14] Project at National Aeronautics and Space Administration (NASA) Langley Research Center uses a fault-tolerant architecture based on replicated, reconfigurable stars that communicate via a Reliable Optical Bus (ROBUS) using Time Division Multiple Access (TDMA) protocol. This architecture provides several mechanisms for integrating interdependent applications of differing criticality levels, such as clock synchronization, group membership, and interactive consistency.

- RealNet is an optical, real-time, fault-tolerant mesh network developed by Honeywell International, Inc. for vehicle control.

- PI-Bus [15] is the only standard backplane developed for the military. It comes in two different degrees of fault tolerance (fail operational and fail detect). It is the first standard backplane bus to tolerate a failure on any of its signal lines, including all of its control lines.

- MIL-STD-1553B defines the electrical characteristics and protocol for a databus using a Master-Slave principle. It uses time-division multiplexing and can transmit at a raw data rate of 1 Mbps.

[4] FireWire is a registered trademark of Apple Computer, Inc.
[5] FlexRay is a registered trademark of the DaimlerChrysler AG Corporation.
[6] Byteflight is a trademark of BMW AG.

- IntelliBus®[7] [16] is a MIL-STD-1553B derivative databus developed by The Boeing Company with hopes that it would be used for automotive X-by-wire.

- ARINC 664 and Avionics Full-Duplex Switched Ethernet (AFDX) [17] is a data network family derived from Ethernet. AFDX uses replicated full-duplex communication channels and central switches. Some degree of determinism is achieved using rate-limiting of senders and a traffic policing in the switch. AFDX will serve as the avionics backbone on the Airbus A380 and has been chosen for the Boeing 787 as its central communication system.

1.3 SYSTEM SAFETY AND DEPENDABILITY BACKGROUND.

1.3.1 Network Evaluation Relative to a System Safety Process.

The sheer variety of network and databus technology makes it difficult to characterize generic attributes that can be used for a set of all-encompassing evaluation criteria. The details of the implementation of these networks determine their characteristics; they may be serial, parallel, synchronous, asynchronous, external, internal, intersystem or intrasystem wired, or wireless, etc. In addition, the potential failure behavior of the databus or network technology may be mitigated at the system architecture level, for example, by employing multiple independent data paths, design dissimilarity, or enhanced end-to-end integrity mechanisms above the core network behavior. For these reasons, a bottom-up go and no-go checklist is very difficult to elicit at the network level itself. Instead, a holistic view of the entire system is required to ensure that the use of the network technology is sufficient to meet the system-level functional responsibility and safety assumptions. While databus and network technology have traditionally been evaluated on a case-by-case basis against Federal Aviation Regulation (FAR) XX.1309 (the safety-related regulations) and FAR section XX.1301 (the intended function-related regulations) with a detailed review of the implementation mechanisms, the regulations provide very little guidance and information on the basic safety processes.

Pertinent regulations related to this research, adopted and enforced by FAA, are contained in the Code of Federal Regulations (CFR) (FAR) Chapter I (Parts 1-199), Title 14 (Aeronautics and Space, Airworthiness Standards), Part XX, Subpart F (Equipment), Section XX.1309 (Equipment, systems, and installations) and Section XX.1301 (Function and Installation), where XX refers to the particular part (Parts 23, 25, 27, 29, or 33) and where the identification of these parts is as follows:

- Part 23—Small Airplanes (Normal, Utility, Acrobatic, and Commuter Category Airplanes)

- Part 25—Transport Category Airplanes

- Part 27—Small Helicopters (Normal Category Rotorcraft)

[7] IntelliBus is a registered trademark of IntelliBus Network Systems, Inc.

- Part 29—Large Helicopters (Transport Category Rotorcraft)
- Part 33—Aircraft Engines

In addition, Part 33.28 (Aircraft Engines, Electrical and Electronic Engine Control Systems) also applies.

SAE ARP 4754 and SAE ARP 4761 provide a good resource for understanding the detailed analysis and process to demonstrate compliance to the regulations. This process is initially top-down, focusing on functions at the aircraft level that are enumerated in a function list.

The hazards associated with the functional failure conditions are determined for each function at the aircraft level. Note that at the initial stages of the process, designers and evaluators may not know how these functions will be allocated to subsystems. While it can be the common cause for failures in multiple functions, the databus or network has not traditionally been viewed as an airplane-level function; rather, it is a tier design choice for how the functions are provided. Thus, at this point, there is no impact. One or more candidate system architectures for aircraft-level functions are proposed. The system could be a single processing module (analog or digital) with a number of inputs or outputs fed directly to the box or a single box for each function (analog or digital) or any of a number of alternative architectures. This architecture then forms the basis for an aircraft-level fault tree that demonstrates how failure conditions will flow through the architecture.

At this stage, it is not uncommon to start looking at common cause analysis (CCA). CCA consists of three components: (1) particular risk analysis (e.g., lightning), (2) common mode analysis (e.g., all boxes receive cooling from a single source, or data from a shared network), and (3) zonal analysis (e.g., a fire in the wheel well damages wires that pass through the area but are not related to any equipment in the wheel well) at the architecture level (for example, consider the implications of the two mentioned architectures). As the architecture is refined, an airplane-level network may be derived. This will need to be considered as part of the system fault tree analysis (FTA). This process continues iteratively until a detailed component (i.e., line replaceable unit) level design emerges. This iterative top-down process is captured by a preliminary system safety analysis (PSSA), system and subsystem fault trees, and revisitation of the common cause and zonal analysis as appropriate. The lower levels of the fault tree will contain a number of different faults that can be traced to aircraft-level failure conditions. As the architecture is continuously refined, the use of databuses and network technology can appear at any level and feed into the continuously evolving PSSA. When a preliminary complete design emerges, then a bottom-up approach, called a failure modes and effects analysis (FMEA) or a failure modes, effects, and criticality analysis (FMECA) is instituted on the actual design looking at specific failures of components or group of components and their contribution to the aircraft hazards. A failure condition would be phrased as “loss of all braking due to a hardware failure (unspecified),” and an analysis would be conducted to determine all possible failures that could cause the failure condition. The FMEA would start with something like the failure of a power supply and trace it to either a corresponding fault in the PSSA or to a system-level effect if no associated PSSA fault can be identified. Ideally, the top level of an FMEA or FMECA can be identified with the faults from one or more fault trees. Databuses and network technology services may therefore appear in any level of the system design and are required to be analyzed

from both the bottom-up (FMEA and FMECA) and top-down (FTA and PSSA, as well as the CCA). When the iterative process is finished, the safety results are documented in the system safety analysis, including the summaries of the FMEA and FMECAs and the CCA.

For this process to work effectively, it is paramount that the impact of the behavior and potential failure of the databus and network technology is adequately captured and represented in the FTA. For low-complexity network and databus technology, the process above is relatively straightforward. In such cases, the network services assumed by the upper levels of the system behavior are simple and restricted to point-to-point communication primitives only (for example, those concerned with the loss of information, or delay of information, or corruption of information restricted to few nodes). However, as silicon integration increases (enabled by continually decreasing process geometries), the failure modes of integrated devices are getting considerably more difficult to bound. Hence, even in the case of simple communication services, great care is required to ensure that the failure mechanisms and assumptions are suitably captured. In addition, as networking technology has advanced, a number of additional services have been implemented at the network level (for example, acknowledgement, message agreement, global time synchronization, system mode change distribution, fault diagnosis, power distribution, etc). The system-level impact of such services may be significant, and in many cases, the databus or network may form the "intelligence backbone of the system" or entire aircraft. In these cases, a more detailed analysis of network behavior and system logic and assumptions is required. For example, if message agreement or interactive consistency is leveraged by applications operating above the network infrastructure to implement active replication strategies (for example, replica determinism for triple modular replication), the justification of the application-layer behavior needs to address implications of network failures or transient upsets that may affect the coverage of such strategies in the event of a fault or external system upset.

1.3.2 Determining Data Network Dependability Requirements.

When evaluating a data network for a particular aviation digital electronics application, one must begin by establishing the requirements for that data network. The requirements placed on a data network are highly dependent on the aviation digital electronics architecture that will employ the network.

To develop requirements for an avionics data network, results from the following tasks should be captured.

- Establish the most critical system failure condition for each data on the network. Determine failure states of the data create that condition. Establish the associated probability and assurance requirements.

- Define the network functions that are required by the system's applications.

- For each data element on the network, examine network function failure on the following classes of failure:

    - Inability to provide data

- Failure to meet specified criteria (e.g., timing, lack of corruption, latency, sequence, identification, etc.)

- Establish the fault containment and fault tolerance requirements from the system safety analysis for data hosted by the network. This may require high-integrity message integrity checks or redundant paths.

- Determine whether the system requires the data network to coordinate action or consensus between different networked components (e.g., for synchronization, fault diagnosis, or voting).

Such a top-down examination of network usage is needed to establish a context for the detailed analysis of the network lower-level properties and behaviors presented in the following sections. It is emphasized again that without such a systems context, the justification of databus and network suitability or nonsuitability is very difficult to determine.

1.4 SCOPE.

1.4.1 System Network Role.

The evaluation criteria described in the Data Network Evaluation Handbook [2] (the companion document to this report) were selected to help in the creation or selection of safety-critical aviation digital electronics data networks. The safety-critical data networks tend to be system data networks (i.e., data networks that connect a number of subsystems) or data networks that connect the redundant elements of a safety-critical subsystem. These data networks are generally "box-to-box" rather than backplane memory or peripheral extension busses, such as the peripheral component interconnect (PCI). The latter are used to connect cards within a box that implement a single function or form a single fault containment zone within the redundancy set (i.e., one replicant).

Practitioners are beginning to see networks, such as SAFEbus, that are actually system buses implemented in a backplane. What is important is the role of the network, not where or how it is implemented. Backplane system networks can be differentiated from simple extension backplanes by the fact that they provide a layer of behavior and abstraction above the simple data transfer and signaling of a typical backplane. As such, they will provide a different impact on the safety analysis. In some cases, the subsystems may not have knowledge that the communication medium is a backplane. This may result in the interconnection of multiple subsystems rather than just the components of one subsystem.

While development of these evaluation criteria was not intended to cover networks within a single function box, a subsystem, or that connect together nodes within a single fault containment zone, these types of networks could have an impact on safety. For example, a generic failure in a backplane bus used in each copy of a redundant, safety-critical system could cause that system to fail.

If multiple cards and functions are connected by a single backplane network, then the common mode influence and failure of the backplane network needs to be considered when the

availability and integrity of these functions are justified. This is especially true if functions connected by data network infrastructure (be it backplane or box-to-box) are assumed to fail independently. In such cases, some of the evaluation criteria described by the Handbook may be equally applied to these internal networks. However, the scope of these criteria are not intended to wholly cover the case of internal subsystem networks or on board networks, where all network connections lie within a common fault zone.

1.4.2 Protocol Stack.

Data network protocols are often designed to comprise multiple layers of functionality organized within a "stack." Each layer is sufficiently independent of the layer above it and the layer below it such that the layer can be reused in other stacks. Generic models for the stacks have been developed. The most widely known model is the seven-layer International Standards Organization (ISO) Open Systems Interconnect (OSI) model, as shown in table 1.

Table 1. The Seven-Layer ISO OSI Model

Number	Name	Description
Layer 7	Application	This layer is where communication partners are identified, quality of service is identified, user authentication and privacy are considered, and any constraints on data syntax are identified. This layer is not the application itself, although some applications may perform application-layer functions.
Layer 6	Presentation	This layer is usually part of an operating system that converts incoming and outgoing data from one presentation format to another.
Layer 5	Session	This layer sets up, coordinates, and terminates conversations, exchanges, and dialogs between the applications at each end. It deals with session and connection coordination, authentication, and end-to-end encryption.
Layer 4	Transport	This layer manages the end-to-end control (for example, determining whether all packets have arrived) and error-checking. It ensures complete data transfer.
Layer 3	Network	This layer handles the routing of the data (sending it in the right direction to the right destination on outgoing transmissions and receiving incoming transmissions at the packet level). The network layer does routing and forwarding.
Layer 2	Data Link	This layer provides synchronization for the physical level. It furnishes transmission protocol knowledge and management.
Layer 1	Physical	This layer conveys the bit stream through the network at the media and mechanical level. It provides the hardware means of sending and receiving data on a carrier.

The widely used Department of Defense ARPANET, or transmission control protocol/Internet protocol (TCP/IP) suite, does not have an official stack description document. But it is regarded by many as having four or five layers, with the bottom three layers generally corresponding to the bottom three layers of the OSI stack.

For the embedded real-time systems used in the vast majority of safety-critical aviation digital electronics, models with a reduced number of layers have been used to provide lower complexity, latency, and overhead.

The lower levels of the stacks deal with the communication media and the hardware connected to it. The higher levels of the stacks represent functionality that is progressively abstracted further away from the hardware. When developing selection criteria for data networks that will be used in safety-critical systems, a question naturally arises as to which layers need to be evaluated. This question is equivalent to asking: What layers can affect the dependability of the overall communication system? This will depend on where the designers have implemented mitigation means, the type of failures that can be realized at a given layer, as well as the effect on the system safety analysis. In general, the highest layer where communication dependability is considered is usually identified as the transport (or equivalent) layer. Some networks handle dependability issues partly or wholly within layers below the transport layer while others deal with them at the application layer that interfaces to the transport level. In many cases, multiple layers will be needed to provide an acceptable dependability argument. Communication network hardware typically implement stack layers below the transport layer and many of the networks proposed for aviation digital electronics systems only define these lower layers. However, some of these hardware devices also provide special application services that support system fault tolerance.

In general, the scope of the evaluation criteria described in the Handbook extends from the lowest stack layer through the dependability features of the transport layer or to the highest layer that is part of the network standard (or definition if the network is not a standard), if that network does not include functionality up to the dependability features of the transport layer. Safety-critical embedded real-time systems often require services not included in generic protocol stack models; such as a time synchronization service. These special services will be included within the scope of these criteria.

1.4.3 Developmental Time Horizon.

The evaluation criteria were chosen so future data network communication technologies, as well as current and past technologies, could be evaluated.

1.5 DATA NETWORK CERTIFICATION ISSUES IN CONTEXT.

1.5.1 Supported Application Requirements.

When evaluating a data network for a particular aviation digital electronics application, one must begin by establishing the requirements for that data network. The requirements placed on a data network are highly dependent on the aviation digital electronics architecture that will employ the network.

Some questions and associated tasks include:

What is the most critical system failure condition for each data on the network? What failure states of the data create that condition? What are the associated probability and assurance requirements?

- What network functions are required by the system's applications?

- For each data element on the network, examine network function failure on the following classes of failure:

  - Inability to provide data

  - Failure to meet specified criteria (e.g., timing, lack of corruption, latency, sequence, identification, etc.)

- Establish the fault containment and fault tolerance requirements from the system safety analysis for data hosted by the network. This may require high-integrity message integrity checks or redundant paths.

- Does the system require the data network to coordinate action or consensus between different networked components (e.g., for synchronization, fault diagnosis, or voting)?

Such a top-down examination of network use is needed to establish a context for the detailed analysis of the network lower-level properties and behaviors presented in the following sections. It is emphasized once again that without such a systems context the justification of databus and network suitability or unsuitability is impossible to determine.

1.5.2 Multiple-Requirement Engineering Trades.

As with any complex technology, the selection (or creation) of data network technologies requires the evaluation of how well a particular technology alternative meets a large number of requirements, many of which are contradictory. What is more important: size, weight, power, cost, bandwidth, latency, availability, integrity, etc.? Technology trades must consider all requirements simultaneously. To do so means that that the relative importance or weighting of these requirements must be established. After the relative importance rankings of the requirements have been established, ratings of how well a particular technology alternative meets each requirement must be created. Then, the multiple requirement trade-off can be done.

These trades are most often done in a linear compensatory manner. That is, for each alternative, its "goodness" value for a particular requirement is multiplied by the ranking or weight of that requirement; then all these products are summed to get the overall value of that alternative.

The best alternative is the one with the highest sum. This process can be represented by the formula:

$$Va = \sum_{r=1}^{n} w_r v_{ar}$$

where: Va = Value of the a th technology Alternative
w_r = Weight or importance of the r th Requirement
v_{ar} = Value of the a th Alternative with respect to the r th Requirement

Some requirements may have a minimum acceptable level. That is, any alternative that fails to achieve meet this minimum level will be rejected, regardless of how well it does against other requirements. Above the minimums, how well an alternative meets one requirement can be traded for another requirement. However, even the minimum acceptable levels may be adjustable. This is because a data network does not have to carry the sole responsibility for any particular system characteristic. For example, a data network by itself cannot guarantee system safety. It is only one component of the system (although it may be the most important component). One can trade-off the characteristics required for safety, e.g., data integrity, between the data network and any architectural mitigation for that characteristic.

The criteria for accepting data network technologies and component implementations with respect to a certification process constitute only a subset of the requirements typically considered when doing a data network trade study. This can lead to multidimensional trade-offs. For example, the inclusion of a certain level of data integrity in a network may force other characteristics to fail their requirements (e.g., excessive size, weight, and power). But, moving the responsibility for the integrity to some other part of the system design may not incur the same level of problem.

1.5.3 System Architecture and Design.

It is not possible to evaluate data network technologies and components without regard for the specific architecture and system design within which they will operate. Note that "architecture" as used in this context is not synonymous with high-level system design. Here, architecture refers to the set of rules for design. It is meta-design (i.e., the design for the design). Architectural rules can apply in any level in a design hierarchy. For example, an architectural rule may be that triple modular redundancy will be used for fault tolerance. This rule could be applied at a high level where three boxes of electronics are voted or it could apply at a low level where three memory chips are voted.

A particular system design may not need certain features of a data network. Shortcomings in a particular data network may be mitigated by an architecture. Pushed to the ultimate, architectural mitigation could make almost any data network technology work. However, this extreme mitigation may require the use of a large number of replicated networks used in a manner completely outside the original intent of the network. For example, each node in a network could be connected to all other nodes in the system via one-way point-to-point Ethernet links. This would eliminate the nondeterminism of Ethernet because each link would have only

a single transmitter. However, this would be very expensive; for N nodes it would require N^2 Ethernets. And, the topology would not be a standard Ethernet bus or star; it would be a fully connected mesh.

While architectural mitigation may be possible, it may not be the best trade-off. One should be wary of the "pile of bandages" fallacy. This fallacy tries to overcome data network inadequacies by applying architectural mitigation "bandages" that do not have a demonstrable level of coverage. The erroneous mindset is that if you pile enough "bandages" with unknown coverage on top of a problem, eventually you will get sufficient coverage. However, the truth is that the sum of a number of unknown coverages is still unknown. The bottom-line issue for architectural mitigation is that it must have some demonstrable level of coverage.

1.5.3.1 Determinism.

Determinism is a widely discussed characteristic of digital electronic systems and, in particular, data networks. Very often these discussions narrow the definition of determinism to be applicable only to media access control (see section 4.1). While media access control is an important area to demonstrate determinism, it is not the only area. Determinism has a much broader applicability. In general, determinism means that the behavior of a system can be determined a priori. That is, given knowledge about the system's current state and a sequence of events that will affect the system from its environment, one can predict how the system will behave. If one cannot predict the behavior of a system in this way, one cannot claim that it has determinism as a property. In most cases within civil aviation, the property of determinism is needed to contribute to claims in the system safety analysis. Thus, determinism is an essential characteristic of a system that is used in safety-critical applications.

An obvious question is: How accurately must the behavior be known? Most safety-critical aviation applications are real-time systems. That is, for the system's behavior to be correct, its outputs must have correct values with correct timing. There are two types of timing determinism, ordinal and cardinal. Ordinal timing determinism means that the order of events can be determined a priori. Cardinal timing determinism means that the time between events can be determined a priori. The degree of precision required for values and timing is specific to each application.

Given that all avionics data networks are digital, the behavior of which can be modeled as a finite state machine, value determinism can be established in the absence of failure (including "normal" failures such as those caused by intersymbol interference, metastability, or single event upset). An evaluation of a data network must, first, establish the requirements for ordinal time determinism and the required degree of cardinal time determinism. Then, the evaluation must consider if the data network can be proven to meet the required time determinism. Again, this time determinism applies to more than just media access control. For example, the effect of nondeterminism in the arbitration for local memory between a network interface and the processor it supports may make it difficult to prove the correct behavior of the processor or the network interface under all possible timing conditions.

1.5.3.2 Robust Partitioning.

Robust partitioning for avionics is a concept used in the ARINC 650 series of documents. The concept of robust partitioning is such that a function that is robustly partitioned from other functions cannot be adversely affected by those other functions. Partitioning analysis provides assurance that one partitioned function's behavior does not unacceptably affect the behavior of another partitioned functions behavior. With a common data network there are many opportunities for partition violations to occur. Every data network used in a partitioned environment should be rigorously analyzed with regard to maintaining partitioning integrity. Because partitioning is a system property, it is beyond the scope of this report to comprehensively detail the issues associated with it. However, there are specific areas where the impact of some network feature is elaborated to illustrate the potential effect on partitioning.

2. TASK ACTIVITIES.

2.1 PHASE 1.

2.1.1 Literature Survey.

In support of creating evaluation criteria, a preliminary literature survey was conducted. This survey included literature describing existing data networks, comparisons of multiple networks, and papers describing related underlying technology mechanisms and issues. The literature was identified according to a number of criteria relevant to future tasks. Eighty-four relevant papers have been identified and are categorized according to the specific protocol or general category of data network technology to which they pertain.

Publications were included if they satisfied at least one of the following criteria:

- Described an embedded network protocol designed for critical avionic operation or other similar purposes (especially automotive and rail applications, as well as "best practice" research designs)
- Described or evaluates a mechanism used or proposed for use in critical embedded networks (e.g., group membership)
- Described an evaluation technique applicable to critical embedded networks (e.g., fault injection)
- Proposed or applied a set of evaluative criteria to examine one or more critical embedded networks
- Described known protocol faults and vulnerabilities (i.e., lessons learned)
- Presented ideas that the investigators think might be otherwise relevant to the study

2.1.2 Industry Survey.

An industry survey questionnaire was developed. Paper copies of the questionnaire were distributed at the 2005 National Software and Complex Electronic Hardware Standardization Conference and an electronic copy was distributed by the Federal Aviation Administration (FAA). Although the long original questionnaire was reduced in size to entice more responses, very few responses were received. Responses to the industry survey are included in appendix B.

2.1.3 Development of Primary Issues.

When selecting appropriate data network technologies, three major sets of criteria must be considered: performance criteria, safety criteria, and certification criteria. Performance criteria are focused on identifying data network technologies that can fulfill the engineering requirements of the application (e.g., meet bandwidth, throughput, and message timing requirements). Safety criteria are focused on best practices to mitigate design failures and ensure correct operation in the event of component failures. Certification criteria focus on the design issues that must be addressed to make an argument for the correctness of the system in accordance with the certification rules of the FAA (or other governing body). These sets of criteria were further refined and used as an outline to develop the issues and criteria detailed in the remainder of this report.

2.1.3.1 Performance Criteria.

There are many different design choices that can distinguish one network from another, such as choice of physical media, access control mechanism, and message encoding. These choices can interact in complex ways (e.g., certain media are required for certain access control mechanisms). Further, not all networks will have the same set of features or capabilities, making comparison more difficult. While all criteria may not be applicable to all networks, the following list of criteria provides a basis for comparison of various networks.

2.1.3.1.1 Physical Media Dependent Layer.

While network specifications may or may not require a particular type of physical medium (e.g., unshielded, twisted pair, optical fiber), choices of a physical medium can affect performance, cost, and maintainability. The following factors should be considered when examining physical media requirements:

- Isolation—Network drops may provide isolation (e.g., optoisolation or transformer coupling) to prevent faulty nodes from affecting the network or vice-versa.

- Susceptibility to noise or interference—Some physical media may be affected by electromagnetic interference (EMI).

- Network length—The length limitations of a network should be examined to ensure the maximum length is sufficient for the application requirements. This includes lengths of individual links of media and the end-to-end signal paths across network that is not a bus topology.

- Propagation delay—Propagation delay can vary depending on medium, and may affect other aspects of the network, as in the case of binary arbitration schemes, which have data rates limited by the propagation delay of the network.

- Impedance or termination—The network media, taps, stubs, connectors, and terminators must be designed so that the network's voltage(s), current(s), impedance(s), and signal reflections provide an operating point and signal-to-noise ratio that has sufficient margins to meet the assumed communication error rate.

2.1.3.1.2 Topology.

Any network that is being considered should support topologies necessary to achieve safety and certification requirements. Choice of topology can also affect other aspects of the network, e.g., a topology with repeaters may increase message latency or create single points of failure.

2.1.3.1.3 Data Transmission.

Concerning the actual transmission of data over the network, there are a number of factors to consider:

- Raw data rates—The network specification may allow a range of data rates for various applications. These data rates may be affected by factors like choice of physical medium and EMI considerations.

- Throughput—The network should be able to support all message traffic necessary to meet application requirements.

- Fairness or priority—The network should have a mechanism to ensure that all nodes can transmit in a manner that meets their individual message timing requirements.

- Minimum message loop timing—The minimum message loop timing must be fast enough to meet application requirements for control loops while leaving enough bandwidth for all message traffic.

2.1.3.1.4 Node Processing Requirements.

The network specification should describe node processing requirements, i.e., whether the node must process all messages or only those relevant to its operation. Processing requirements may be further broken down into requirements for a network controller and a host processor, which may have different roles and responsibilities depending on network design.

2.1.3.1.5 Efficiency.

Message size can significantly affect the efficiency and throughput of network traffic. Marshalling multiple messages into a single message can improve efficiency, but this approach

increases the complexity of scheduling to meet multiple timing requirements. Two metrics for efficiency are defined as:

- Message efficiency—The total amount of data sent divided by the total number of bits required to send it. Total number of bits sent should include any preambles, start or stop bits, error detection codes, etc. This metric must also factor in any error signals that nodes may send, as well as possible retransmission of corrupted messages.

- Bit efficiency—The number of physical bits required to send a single logical bit. Line-level encoding schemes that map multiple physical bits to a single logical bit effectively reduce the raw data rate on a network.

2.1.3.1.6 Scalability.

When selecting a network, it is important to take into account future expansion of the number of nodes on the network. Two factors that may affect this expansion are:

- Address space—If the network uses an addressing scheme to identify nodes, the maximum number of physical addresses may limit expansion

- Physical medium limitations—The physical medium chosen for or required by a network may affect the maximum number of nodes that can be connected to the network. This includes impedance and loading issues for the trunk medium of a bus and the length and the number of links that can exist, end-to-end, on a nonbus topology.

2.1.3.1.7 Services.

Some networks may offer application-level services that are integrated into the protocol. The services offered must be compatible with the design requirements of the target system. Application services may generate additional message traffic that should be taken into account when calculating design margins and bandwidth requirements. Some services that may be offered are:

- Group membership—This service allows correct nodes to agree on the group of correct nodes, so that messages from faulty nodes can be disregarded. Proper reintegration strategies for transient faults or network blackouts may be an important requirement of a network that offers a group membership services.

- Global clock synchronization—This service allows all nodes to have the same notion of time by synchronizing local clocks (through rate or offset correction), or by providing a synchronized execution state.

- Data value agreement service—This service allows nodes to agree on the values of data sent by other nodes.

2.1.3.1.8 Emulation and Interoperability.

The network may need to be compatible with requirements of other networks to allow bridging or interfacing of legacy systems. In this case, the characteristics of existing networks must also be taken into consideration.

2.1.3.1.9 Design Margin.

When selecting a network, candidate networks should have slack to accommodate additional message traffic that is not considered in the original design proposal. Such unanticipated traffic may be the result of additional nodes or messages that must be added to meet safety requirements. They may also be the result of unanticipated features that are added later in the design cycle or during future expansion of the system.

2.1.3.2 Safety Criteria.

In addition to the numerous ways networks can differ in design and functionality, there are additional requirements that must be considered when these networks are being considered for avionics systems. While actual safety requirements for networks are heavily dependent on the target application and potential for interaction with users and other systems, the following guidelines list some issues that should be considered for all applications.

2.1.3.2.1 Specification Completeness.

The network specification shall be sufficiently detailed and unambiguous, including requirements for interacting with host processors, that components from different vendors built to the same specification will be interoperable.

2.1.3.2.2 Message Corruption.

The network should have the capacity to detect messages corrupted by network noise, component failure, etc., in accordance with the probability requirements of the safety analysis.

2.1.3.2.3 System Parameter Consistency.

The network should have some capacity for nodes to verify the consistency of configurable parameters (e.g., message tables, message identification (ID) lists, static schedules, etc.) across all participating network nodes.

2.1.3.2.4 Replication.

Replication or redundancy in the network should be sufficient to show that the system is survivable under actual operating conditions and failure probabilities in accordance with the requirements specified in the safety analysis.

2.1.3.2.5 Security.

Network design should include analysis of potential security threats, both internal and external, worst-case outcomes of security breaches, and mitigation strategies.

2.1.3.2.6 Fault Tolerance.

Correct operation in the presence of faults is a common requirement of modern embedded systems. Fault models must be rigorously defined and tested. There are several aspects of fault tolerance to consider:

- Failure detection, isolation, containment, and recovery—Procedures for identifying faulty or failing components should be clearly defined. Once failures have been detected and isolated (specific failed components identified), mitigation means can be established at either the system level or the network level as appropriate for a given design approach. At a minimum, mitigation should include fault containment and the recovery of the system to a safe state.

- Reintegration—If reintegration of transiently faulty components is permitted, the reintegration strategy should properly distinguish between permanently and transiently faulty components. Further, the network must handle reintegration in such a way that it cannot cause system instability.

- Upgrade or repair safety—Maintenance and upgrade procedures should include testing or verification of system correctness after maintenance procedures (i.e., replacements) have been performed.

2.1.3.3 Certification Criteria.

Certification is an important step in the design cycle of avionics systems. Being aware of the following certification issues can allow designers to select networks with characteristics that will facilitate final certification arguments.

2.1.3.3.1 Certification Responsibility.

Creating certification artifacts and supporting data for approval of a certification authority is the responsibility of the applicant. To meet these responsibilities, the applicant must examine the supporting data for compliance to regulations and associated interpretative guidance. The responsibility for preparation of this data and demonstration of compliance can be assigned to either a vendor of network components, the integrator of networks or the applicant that is applying for certification. The supporting data can be accepted as satisfactory by the certification authority leading up to making a final certification argument.

2.1.3.3.2 Design Assurance.

Design assurance of components should be conducted according to accepted design assurance guidelines (e.g., DO-178B and DO-254).

2.1.3.3.3 Network Message Requirements.

To meet specification requirements, each of the following requirements must be met. While each requirement must be addressed individually, there may be interaction between the requirements. For example, a protocol may use an error detection scheme whereby detection of corrupted message can be guaranteed to a desired level of probability. However, detection (and rejection) of the corrupted message results in the loss of message, so further steps must be taken to meet requirements for loss of message.

2.1.3.3.3.1 Reliable Message Delivery.

For any message whose loss may produce a hazard, the design should ensure

- that the hardware and software design assurance appropriate to the hazard provides the required level of confidence that there are no design errors that can result in loss of the message.

- that the hardware reliability can support the probability requirements appropriate to the hazard category for loss of message due to random failure.

- that, for the catastrophic hazard failure condition, there should be at least two random failures regardless of probability of occurrence before loss of message occurs.

2.1.3.3.3.2 Reliable Message Timing.

For any message whose incorrect time of arrival can produce a hazard, the design should ensure

- that the hardware and software design assurance appropriate to the hazard provides the required level of confidence that there are no design errors that can result in incorrect time of arrival of the message.

- that the hardware reliability can support the probability requirements appropriate to the hazard category for incorrect time of arrival of message due to random failure.

- that, for the catastrophic hazard failure condition, there should be at least two random failures regardless of probability of occurrence before incorrect time of arrival of message occurs.

2.1.3.3.3.3 Message Integrity.

For any message whose corruption may produce a hazard, the design should ensure

- that the hardware and software design assurance appropriate to the hazard provides the required level of confidence that there are no design errors that can result in corruption of the message.

- that the hardware reliability can support the probability requirements appropriate to the hazard category for corruption of message due to random failure.

- that, for the catastrophic hazard failure condition, there should be at least two random failures regardless of probability of occurrence before corruption of message occurs.

2.1.3.3.4 Distributed Coordination Verification.

Any distributed coordination algorithm provided by a network (e.g., group membership, global clock synchronization) must be analyzed to ensure that it will operate correctly, especially in the presence of faults.

- Agreement algorithms—Any algorithm for distributed coordination or agreement should be verified for correctness (e.g., through formal proof).

- Start-up procedures—Start-up procedures should be validated. Worst-case start-up or restart time should be determined and be within acceptable limits for system functionality.

2.1.4 Phase 1 Report.

At the end of Phase 1, a report was written that contained a discussion of evaluation criteria and their related issues. These discussions were contained in narrative sections that are included in sections 3 through 9 of the combined Phase 1 and Phase 2 Reports. In addition, the Phase 1 Report included a list of evaluation criteria that is also included in this combined report as appendix C.

2.2 PHASE 2.

2.2.1 Literature Survey Update.

The literature survey done in Phase 1 was updated to include 53 new papers. The resulting annotated bibliography for the pertinent papers is included in appendix A.

2.2.2 Condensing the Criteria List.

The evaluation criteria created during Phase 1 were too numerous to be comprehensively used in this combined report (see appendix C). However, just deleting criteria to reach a manageable number would have removed many of the examination-of-conscience questions that really are required. A reasonable number of criteria would be in the 10 to 100 range. As a compromise, the number of actual criteria was reduced to fit that range, with the remaining questions grouped to form a set of support questions in an aiding paragraph under each criterion. Some adjustments to the criteria were also made to help with orthogonality issues. The resulting list of 36 criteria is presented in appendix D and also the Data Network Evaluation Criteria Handbook [2]. The structure follows the Phase 1 Report, which uses the ISO OSI stack layering as much as possible. The format is a criterion number followed by a short title, the main criterion question or

statement (all in bold) and, where appropriate, a paragraph of supporting questions and statements to help in the evaluation for the criterion.

2.2.3 Evaluation of Criteria by Applying to Networks.

2.2.3.1 Data Network Candidates.

The original plan for this task was to validate the criteria against one or possibly two data networks. However, during Phase 1, it was recognized that data networks have a huge range of design with characteristics much more varied than, for example, microprocessors. Evaluating criteria against one or two networks was determined to be insufficient to adequately assess the criteria.

Some networks considered included:

- ARINC 429, 629, 659
- Braided Ring Availability Integrity Network (BRAIN)
- CAN, TT-CAN
- Ethernet family [ARINC 664, GAMA Aircraft Systems Control Bus (ASCB), Ethercat]
- FlexRay, ByteFlight
- IEEE 1149, 1355.2 (Spacewire), 1394 (and SAE 1394)
- IntelliBus
- LonWorks®[8]
- MIL STD 1553 family (1773, STANAG, Notice 5)
- PCI
- PI-Bus
- RealNet
- SFODB
- SPIDER and ROBUS
- Train Communication Network
- TTP/C, LTTP
- VME

The following seven specimen networks were chosen as a sample from the above list: ARINC 629, ARINC 659 (SAFEbus), ARINC 664 part 7/(AFDX), CAN, FlexRay, SAE AS5643/IEEE 1394b (FireWire), and TTP/C. The slashes in this list indicate variations of a candidate, and names for particular implementations are in parentheses.

Of course, it would take much more effort to evaluate all the criteria against seven networks than just one or two. To partially compensate for the much greater effort, not all criteria were evaluated against all networks. Some criteria were not evaluated against some specimen networks if it was obvious to the evaluator that some other specimen networks better exercised the criterion. And, in some cases, redundant findings were observed and discarded.

[8] LonWorks is a registered trademark of the Echelon Corporation.

2.2.3.2 Validation Process.

The criteria validation procedure followed the process that is expected to be applied when the Handbook is used. However, instead of the output of this procedure being an evaluation of each data network per se, the output is a validation of the quality of the criteria in its

- coverage
- orthogonality
- usability

In fact, the data network evaluation criteria results have been deliberately discarded. Including them in this report may have confused readers about the evaluation process' true purpose, which is to validate the criteria themselves, not the data networks. Given that the majority of the criteria need to be evaluated with respect to a particular use, including the results would have been meaningless without including a description of the use. Presenting such results would have introduced a number of dangers here including

- the appearance of advocating a particular use.

- the appearance of being the definitive evaluation of a data network, which cannot be done or divorced from a particular use.

- the appearance of preferring one vendor's network technology above another. Since the study, by necessity and design, made some assumptions about usage and did not evaluate all vendors, all versions of a specific vendors' product, or all attributes of a specific vendor, publishing of this data could not provide any conclusive information relative to specific vendor's offerings and would be misleading at best.

During Phase 1, criteria were created with the anticipation that their application to a data network would produce one of three results:

- OK
- Unsure
- Unsatisfactory

During Phase 2, it was found that the three results provided no ability to carry over an evaluation of a network from one particular use (architecture or system) to a different use. Therefore, the three results were expanded so that now the results of applying each criterion to a data network must state that the criterion is one of these five results:

1. Not applicable to this network in general

2. Applicable to this network in general, but not this use

3. Applicable to this network and this use, and is OK without mitigation

4. Applicable to this network and this use; fails, but has mitigation

5. Applicable to this network and this use; fails with no mitigation, or the result cannot be determined (must assume it fails with no known mitigation)

In addition, for all but result 5, a rationale must be given to justify that result. As one progresses from result 1 through result 5 in the list above, evaluation dependencies between a data network and its application increase. Results 4 and 5 require detailed knowledge of the particular usage. Results 2 and 3 require at least some knowledge of the particular use. Result 1 is independent of use and can be carried over from one use to another. The addition of the rationale also helps in evaluation carryover. The rationale may point out similarities in use that aid in carryover.

At the beginning of the validation process, it became apparent that the definition of mitigation was not clear. The intent of mitigation is to remedy any deficiencies in a candidate network. To clarify the definition, the broadest possible meaning and use of the word mitigation is what is intended. For example, mitigations can take the form of architecture mechanisms that cover a network's shortcomings in its behavior. Or, mitigations can take the form of additional analysis and testing if that is the area of the network's deficiency.

The validation process is really a process to determine the goodness of the evaluation criteria. This goodness measurement itself needs some criteria. The technically correct term for the latter would be metacriteria (criteria concerning criteria). But, this terminology can be confusing. To minimize any possible confusion, the less formal term "goodness metric" will be used instead of metacriteria. The term criteria will be used solely for the network evaluation criteria themselves. The selected six goodness metrics for the evaluation criteria are:

1. Understandable: Would the criterion and its supporting questions be understood by those most likely to use the Handbook?

2. Measurable: Can the statement or the question posed by the criterion be resolved into one of the five results listed above by using information or data that can be obtained by those most likely to use the Handbook?

3. Loopholes: Is there any way that a network can circumvent the intent of the criterion and produce an unsafe situation that the criterion was designed to prevent?

4. Supporting: Are the supporting questions in the paragraph following each criterion adequate? Should additional supporting questions be included? Should some supporting questions be deleted or moved?

5. Orthogonal: To the degree that it is possible, how independent is this criterion from any of the other criteria (i.e., is there minimal overlap between this criterion in all other criteria)?

6. Bins: Are the five results described above the right set of results?

The detailed outputs of the validation process are reported in appendix F. Each subsection of appendix F is dedicated to one network. Within each subsection, the criteria are restated in the same numeric order as they appear in appendix D. Finally, the goodness metrics are listed below each criterion. For each goodness metric, a value of OK means that the criterion immediately above it was found to be adequate for the network of its containing section. Anything other than OK for a goodness metric is some suggestion about how to improve the criteria. While the improvement may have been conceived while applying the immediate criterion to the section's network, the suggested improvement may apply to other criteria. This process inevitably leads to some redundant suggestions, most of which have been eliminated. For those criteria that were not tested for a particular network, the six goodness metrics are replaced by "Untested" or "Untested (partial)" for any criteria that was only partially tested.

2.2.4 Criteria Update and Handbook Creation.

The results of the evaluation and any information that came to the researchers' attention during the latter half of Phase 2 were used to update the criteria and their supporting questions. This includes swapping the order of criteria 20 and 21. The results of these updates are presented in this document as a consolidated list in appendix E and in the Handbook as criteria and ancillary questions interspersed throughout the body of the Handbook.

3. PHYSICAL LAYER.

The lowest level (Layer 1) of most data communication reference model stacks is the physical layer (see section 1.3.2). The function of the physical layer is to send and receive communication symbols via network media. Layer 1 defines mechanical characteristics (such as connector configuration), characteristics of the media, and characteristics of the signal. The physical layer is responsible for transferring individual bits through the communication media. This level is concerned with the following:

- Connector geometry, gender, and pin assignments
- Physical connections to the media and their characteristics
- Media topology
- Media characteristics (attenuation, delay distortion, impedance, noise, etc.)
- Full-duplex or half-duplex transmission
- Signal speed
- Definition of symbols with respect to signal characteristics (e.g., in amplitude and time)
- Physical service data units; serial bits or multiple bits in parallel
- Handshaking
- Notification of physical fault conditions

The laws of physics impose limits on the frequency and the quality of a signal that can be transmitted through a given media, as described by the works of Nyquist [18] and Shannon [19]. Designers of each data network try to create a physical layer that maximizes data rate and quality for a given cost.

Given the physics and cost constraints, some compromises and trade-offs must be made. Users of data networks in safety-critical applications must be aware of how these design choices for the physical layer can impact system safety via the quality of data transmission provided by the data network.

Because the physical layer is the foundation upon which all other protocol layers depend, any failure in this layer will adversely affect all the layers above it unless adequately mitigated. An obvious question that must be answered is, What is the probability of failure in the physical layer? Failures at the physical layer can be grouped into two main sources, bit errors and component failures. The probability of faults in both of these sources depends on the environment.

3.1 ENVIRONMENT.

Data network components must meet the requirements of an aviation digital electronics environment such as those described in DO-160. This means that the data network components not only must survive this environment, but also must simultaneously satisfy all requirements placed on the data network while residing in this environment.

3.2 PROBABILITY OF BIT ERRORS.

Physical layer specifications often state a bit error rate (BER), which gives the probability of error for each bit. This number is typically in the 10^{-6} to 10^{-15} range. The purported source of these errors is the signal-to-noise ratio (SNR). Formulas relating SNR to BER have a form similar to

$$BER = 1/2(1-erf)(Eb/No)^{1/2}$$

where

erf is the error function,
Eb is the energy in one bit,
and *No* is the noise power spectral density (noise power in a 1 Hz bandwidth).

The ratio *Eb/No* is a form of signal-to-noise ratio. The energy per bit, *Eb*, can be determined by dividing the carrier power by the bit rate. As an energy measure, *Eb* is measured in joules. *No* is in power (joules per second) per Hz (seconds), so *Eb/No* is a dimensionless term, or simply, a numerical ratio.

It is important to note that the exact formulas for BER depend on the modulation and encoding schemes used because these schemes, coupled with the physical properties of the media, are important for establishing the so-called "eye pattern." This pattern encloses the space bounded by the minimum upper value, maximum lower value and the minimum spacing between transitions of a signal. Figure 1 shows a typical eye pattern created by the superposition of many symbols and the effects of additional signal noise. The two dashed horizontal lines in the figure represent the minimum and maximum value of the receiver's input threshold, which the receiver uses to determine whether an incoming signal is high or low. The two vertical dashed lines

represent the variation in the time that the receiver samples the input. A receiver's decision about the data value of an incoming signal takes place within the area enclosed by these dashed lines, which is highlighted by the gray box in the figure. The distance between this box and the incoming signal's eye pattern determines the noise margin of the receiver. It is clear that the smaller the area enclosed by the eye pattern, the higher the probability that an error will occur. The size of the eye pattern is determined by the modulation and encoding schemes plus signal noise. Thus, claims of a specific BER without reference to the modulation and encoding schemes and assumed noise amplitudes are worthless for predicting the probability of bit errors. When establishing the actual value for BER, the test patterns used in an evaluation must be those that are actually used by the network, not just the linear feedback shift register-generated, pseudo-random bit sequences used by most BER test equipment. The BER test must also be run in the same noise environment as the actual system will experience.

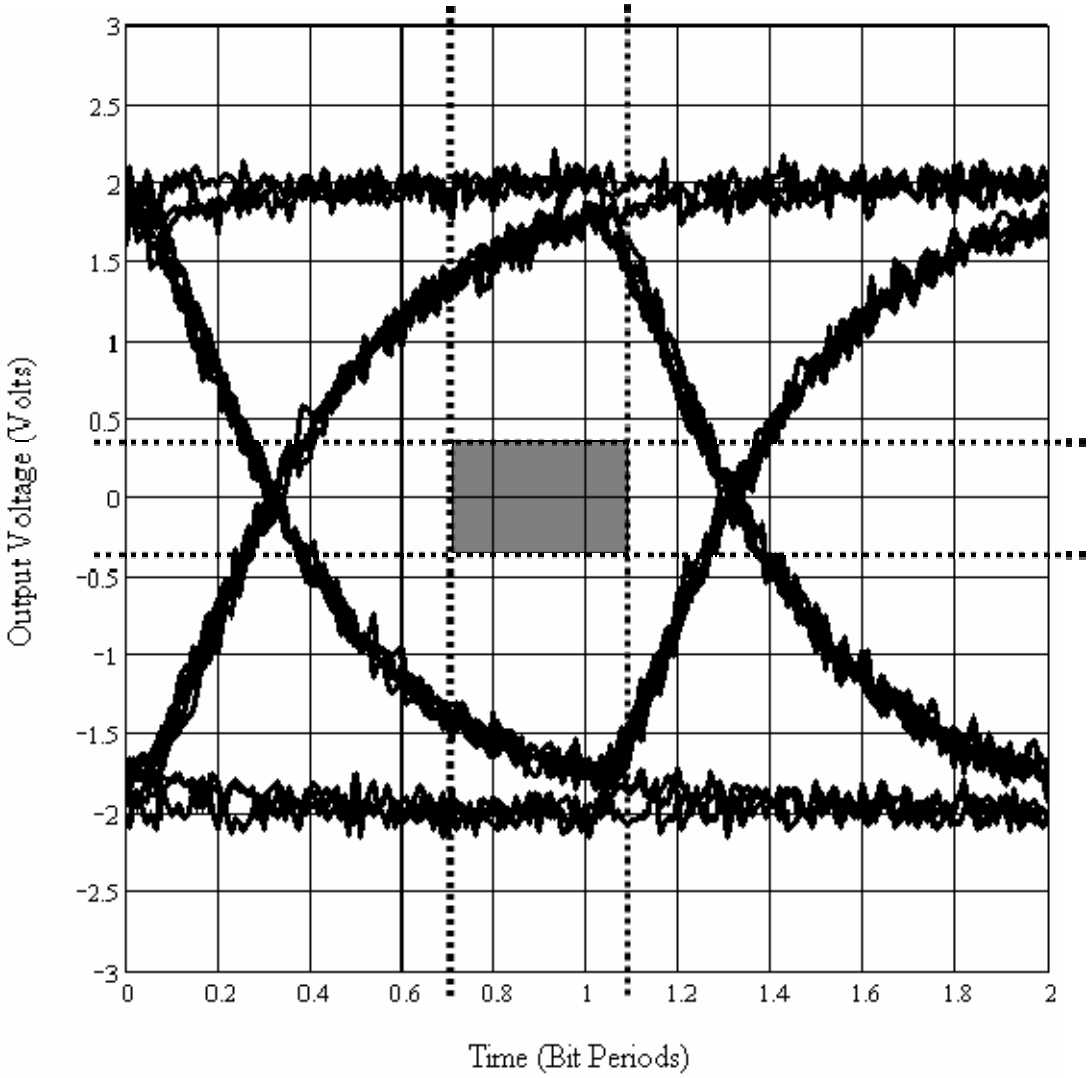

Figure 1. Eye Pattern

The designs of data network error handling and data network reliability analyses are often based on the misconception that bits traversing a data network are independent of other bits on the same network medium. However, designers of data network physical layers are familiar with a phenomenon called intersymbol interference (ISI). The definition of ISI from Federal Standard 1037C [20] is:

> "**1**. In a digital transmission system, distortion of the received signal, which distortion is manifested in the temporal spreading and consequent overlap of individual pulses to the degree that the receiver cannot reliably distinguish between changes of state, *i.e.*, between individual signal elements. *Note 1*: At a certain threshold, intersymbol interference will compromise the integrity of the received data. *Note 2*: Intersymbol interference attributable to the statistical nature of quantum mechanisms sets the fundamental limit to receiver sensitivity. *Note 3*: Intersymbol interference may be measured by eye patterns.
>
> **2.** Extraneous energy from the signal in one or more keying intervals that interferes with the reception of the signal in another keying interval.
>
> **3.** The disturbance caused by extraneous energy from the signal in one or more keying intervals that interferes with the reception of the signal in another keying interval."

It is clear that the transitions that form the sides of an eye pattern are affected by the adjacent bits. Bits much further away also can impact an eye pattern via "baseline wander" caused by accumulated effects of direct current (DC) imbalance that "charge" the capacitances and inductances in the communication path. These capacitances and inductances include the components that a signal must go through (such as transformers) and the intrinsic characteristics of the media and any other components that touch the media (e.g., each receiver and each transmitter adds parasitic capacitance to the media). This baseline wander raises or lowers the eye pattern for every bit, shifting it with respect to receivers' input threshold. This shift affects the probability that the receiver sees an input as one value or another as long as the baseline has wandered away from nominal. During its development, opponents of the 100BaseTX Ethernet design touted the fact that there existed "killer packets." These are packets containing particular data patterns that produce baseline wander bad enough to induce bit errors on their own. Most Ethernet PHY semiconductor devices have active compensation for some amount of baseline wander. When using 100BaseTX, care should be taken to provide some means to prevent killer packets from appearing on the network, or to use only those PHY devices that can compensate for killer packet levels of baseline wander.

Another adverse effect that can cause correlated bit errors is reflection due to impedance mismatches. Impedance mismatches can occur whenever the media or the electrical properties surrounding it changes. This happens whenever the media is split (e.g., for stubs or drops), when the signal passes through a connector, at receivers and transmitters, or even just by having inadequately shielded media pass near materials of different electrical characteristics. Note that these impedance concerns are true for both electrical and optical data communication.

Another phenomena with some characteristics similar to that of impedance mismatch reflections is the problem of the "Wired–Or" glitch (which can be viewed as "Wired–And" with the application of DeMorgan's theorem). Data networks that are susceptible to this problem are those that exploit a bit-dominant signaling method on the media to perform some logic function. One such function is bit-dominant bitwise priority arbitration. Examples of networks that use bit-dominant bitwise priority arbitration include the controller area network (CAN), which is commonly used in automotive applications, and the SAE AS4710 PI-bus, the military avionics standard backplane bus. The "Wired–Or" glitch occurs when two or more transmitters drive a dominant signal onto the medium and a proper subset of these transmitters stops driving the dominant signal. When this happens, the medium state near these transmitters changes to the recessive state for a time equal to the round trip delay between them and the nearest transmitter(s) still driving the medium to the dominant state. To keep a receiver from erroneously interpreting these glitches as valid changes of signal state, the receivers must be designed to tolerate these glitches or design rules must be followed that limit the duration of the glitches (usually by limiting the length of the medium as a function of the bit width) to a duration that can be tolerated.

Bit error rate testing must be done using the worst-case modulation and encoding scheme symbol patterns, the worst-case signal path (including the effects of all inductances and capacitances), and the worst-case reflections due to impedance mismatch. Design, installation, and repair rules must be established such that situations worse than those used in this testing do not occur.

Trying to tolerate failures caused by external noise sources is difficult because the external noise sources, such as crosstalk, lightning, and high-intensity radio frequency (HIRF), can cause arbitrary error patterns with unknown probabilities. The best, and most widely used, way of dealing with external noise sources is to try to make the bits immune to upset. This immunity can be produced by shielding the media from external noise sources, making the signaling scheme robust (e.g., differential drivers with large margins), and adding components to filter out noise. A number of recent developments have eroded these protections. Composite skin aircraft provide less protection against noise sources outside the aircraft. Newer, higher-speed data networks use signaling levels with smaller margins. The wider bandwidths of the higher-speed data networks make it more difficult to filter out noise. One way to counter the erosion caused by the last two trends is to minimize the speed and bandwidth required to meet the system's data throughput needs; that is, to use a network that is as efficient as possible. No matter how much effort is put into trying to make a data network immune to external noise, there can always be some noise source with a large enough magnitude to overcome these efforts. When a data network is overwhelmed by large amplitude external noise, it is important for the network to recover as soon as possible.

Some designers are now suggesting the use of wireless data networks within an aircraft, and there have even been suggestions that these wireless data networks be used for safety-critical functions. However, this appears to be a daunting design challenge given that the external noise sources (such lightning and HIRF) can cause arbitrary error patterns with unknown probabilities.

It is hard to reconcile an existing EMI requirement for a wired data network to survive 200 volts per meter of noise versus a wireless receiver's input being almost a million times more sensitive.

3.3 PROBABILITY OF ELECTRICAL COMPONENT FAILURES.

The avionics industry has a long history of evaluating the dependability of a system, at least for benign faults (i.e., faults that are inherently self-containing or obviously can be detected and contained). However, the fact that complex integrated circuits can have arbitrarily bad behavior is too often ignored. Even extremely simple analog devices can have surprising failure modes. For example, a simple MIL-STD-1553 databus transmitter was observed producing a perfect Manchester waveform output when the component had no signal input. A similar problem has been observed with a fault-free RS-485 driver transmitting a rectangular waveform when its inputs were "stuck high." When applying the evaluation criteria described in the rest of this report, one must remember that electronic circuitry can fail in a way that produces arbitrarily bad behavior, limited only by the energy provided to it (which can be considerable when stored, e.g., capacitors).

With the advent of higher-speed networks, smaller impairments to a signal can cause problems. This creates a concern for the quality (including aging effects) of connectors, media, and drivers.

As data network speeds increase, not only does the SNR decrease on the network media, it also decreases within the electronics. This reduction in SNR makes electronics more susceptible to single event upset (SEU) and metastability. The evaluation of SEU susceptibility should be done as part of the environmental evaluation.

Metastability is an electronic circuit design issue rather than an environmental issue. As clock speeds increase to create higher performance electronics, the amount of circuitry that can be driven by a single clock zone decreases. This creates more clock zones and the need for a larger number of synchronizers at the boundaries between the different clock zones. Each synchronizer has some probability of metastability failure. The metastability failure rate for a synchronizer is given by the formula

$$\alpha * f_{data} * f_{clock} * e^{-\beta t}$$

where α and β are constants unique to each synchronizer implementation,

f_{data} is the frequency of the data,

f_{clock} is the frequency of the clock,

t is the time that the synchronizer waits for its first stage flip-flop to settle to a valid value.

As data network speed increases, f_{data} and f_{clock} tend to increase proportionately and t is the inverse of f_{clock} (in the design of most synchronizers, t is one *clock* period). This means that synchronizer transient failure rates increase with system speed (S) proportional to S^2e^S. This already very steep function is exacerbated by the fact that higher clock speeds require more synchronizers. Luckily, the very characteristics that allow increased speed also tend to improve the values of α and β. However, the only way to determine accurate values of α and β is to test for them.

Synchronizer metastability error rate is tested by giving the synchronizer data that is asynchronous (and statistically independent) to the synchronizer's clock while sweeping the value of *t*. The resulting number of errors for each *t* is fitted to a semi-log line versus *t*. The intercept of this line is α and the slope is β. It is important to note that this test must be done on an actual implementation of each synchronizer design. A very widely held misconception is that α and β depend only on the design of the first stage flip-flop in the synchronizer. While the characteristics of this flip-flop have a large influence on α and β, it is not the only influence. As electronic component geometries shrink, the electrical characteristics of interconnect actually become more important than that of transistors. This increases the importance of characterizing the interconnect between the first stage flip-flop and the second stage flip-flop in each synchronizer design. Electrical characteristics of the second stage flip-flop (such as input thresholds) are also important. The difference in results from testing the actual synchronizer design versus testing just the first stage flip-flop may not be significant for applications that do not have stringent safety requirements. But, for systems that have dependability requirements in the neighborhood 10^{-9}, this difference could consume the entire dependability budget. Thus, for a synchronizer metastability error rate test to be valid, the test must be performed on the actual synchronizer design.

One factor that affects bit error rate is jitter on the input-sampling clock in the receivers. Higher data rates are more sensitive to this jitter. Often the receivers use phase-lock loops to create these clocks. The phase lock loops themselves are driven from an external clock source. The tighter jitter requirements for higher-speed data networks often require higher-quality external clocks that drive the phase lock loops. These clock quality requirements often include restrictions on short-term and long-term jitter. The same clock quality issues affect transmitters. In this case, jitter on a transmitter clock causes jitter in the data. The sum of the transmitted data jitter and received clock jitter affect the error rate of the received data.

A design factor related to the input-sampling clock jitter in receivers is the ratio of the clock's period to the smallest interval between input signal transitions. This has a large impact on the gray box in figure 1. A higher-frequency sample clock simultaneously makes the box smaller horizontally and allows the box to be placed more precisely in the center of the eye. A sample clock that is too slow can place the box too close to the edge of the eye pattern. This can be a source of bit errors in the receiver and be a source of asymmetric or so-called Byzantine faults (see section 7.8).

Jitter and frequency offsets between a transmitter's clock and a receiver's clock also can cause buffer overruns and underruns in elasticity buffers (see section 5.3.1) and can be the source of asymmetric Byzantine faults (see section 7.8).

3.4 ELECTRICAL ISOLATION PROPERTIES.

The causes of total system failure can be segregated into three main classes: exhaustion of redundancy, single point of failure, and lack of fault containment. Of these, the one that is most often seen as part of real world total system failures is the lack of fault containment. One

important aspect of fault containment is the electrical isolation between redundancies. In examining a system design for possible electrical fault propagations, the following mental process can be used, imagining that

- each redundant power supply is painted a unique color.
- each electron leaving a power supply is painted the same color as that supply.
- each component or conductor that an electron enters is painted that electron's color.
- if there is a color conflict, a possible galvanic fault propagation path has been found.

To prevent the data network from becoming a galvanic fault propagation path, these paths are usually interrupted with attenuators and resistors, fiber-optic cables, optical isolators, or transformers. Some isolation methods impose requirements on the physical-layer signaling, for example, transformers that need DC-balanced signaling, such as Manchester or 8B/10B. Some isolation methods may preclude the use of collision detection or the use of mixing dominant and recessive signals on the media to perform some logic function.

Some networks, such as USB, power over Ethernet, and IEEE 1394, transmit power on some conductors in their cables. Requiring use of this power creates a significant problem for galvanic isolation.

Many fault-tolerant architectures include the concept of receive-only nodes. The required characteristic of these nodes is that they can receive information that is transferred across the media, but are prevented from having any effect on any shared media. In these architectures, it is essential to provide assurances that these receive-only nodes cannot affect the data network.

3.5 PHYSICAL COMPOSABILITY.

As nodes are added to a data network, performance and signal quality can suffer. Some physical-layer characteristics that can be adversely affected by adding nodes to network include a decrease in signal margins, added latency and propagation delays, an increase in reflections due to impedance mismatches, and an increase in the probability of reflections constructively adding together to create higher-amplitude problems. A well-designed data network anticipates the effects of node addition and can work correctly with any number of nodes connected to the network up to an explicitly stated limit. The description of a data network may include design rules that must be followed in order for the data network to maintain sufficient physical-layer quality margin as nodes are added. These rules can include such things as topology restrictions (e.g., nodes on a bus cannot be connected any closer than a certain interval), limitations of signaling speed versus distance, or may require the setting of certain parameters within the data network's components that affect its performance (e.g., setting intermessage gap sizes or contention resolution times based on the maximum round-trip delay over a given topology installation).

4. DATA LINK LAYER.

The data link layer is the layer immediately above the physical layer in most data communication reference model stacks. It provides the functions, procedures, and protocols needed to establish,

maintain, and release data link connections between the nodes of a network. A conceptual level of data processing or control logic in the hierarchical structure of a node is responsible for maintaining control of the data link. The data link layer's functions include: bit injection into the transmitter and bit extraction at the receiver; address and control field interpretation; command and response generation, transmission, and interpretation; synchronization; error control; and flow control.

The data link layer is divided into two sublayers: the media access control (MAC) sublayer and the logical link control (LLC) sublayer. The MAC sublayer controls how a node on the network gains permission to transmit on it. MAC sublayer protocols often try to provide prioritization or fairness in granting access to the media. MAC sublayer protocols also try to maximize the use of the media and minimize the probability of starvation (not granting access to requesters). The LLC sublayer controls frame synchronization, flow control, and error checking. Conceptually, the LLC sublayer sits on top of the MAC sublayer. In this document, MAC and MAC protocol refer to the MAC sublayer.

4.1 MEDIA ACCESS CONTROL.

The MAC sublayer is a particularly important part of a data network's protocol when the network is used for real-time systems. Simple problems in the MAC can cause catastrophic loss of the services that the real-time system needs from the data network. These problems include: no access (starvation), not enough access, or wrong time access. One source of these problems is the design of the protocol itself coupled with access demands and timing of clients, including faulty clients that fail to follow the behaviors expected or required by the data network specification.

Other problems can be caused by failures (including permanent and transient failures) in the hardware that directly controls or accesses the network media. These failures may be introduced by any of the sources described in the physical-layer sections above. Of particular concern is the possible brittleness (lack of robustness) of the MAC sublayer protocol. That is, does the MAC sublayer protocol amplify the effect of small failures and errors such that they become large problems? For example, does the MAC sublayer protocol allow transient failures and errors to have an effect that persists longer than current transmissions?

4.1.1 Problems Unique to Each Type of MAC.

4.1.1.1 Master and Slave.

The simplest MAC mechanism is to designate a single node as the Master controller. This single node will have sole authority to grant access to data network's media. The most common example of this kind of MAC in avionics is the MIL-STD-1553. A centralized MAC, i.e., a single-node Master controller, has several weaknesses; the most obvious is that it is a single point of failure. That is, if the controller fails to function or functions incorrectly, the entire communication system will fail. This problem can be mitigated by adding fault tolerance, either within the controller or by having multiple controllers. However, designing such fault tolerance is difficult and no known data networks that are now used or proposed for aviation digital electronics employ such a scheme.

4.1.1.2 Bit-Dominant Arbitration.

Bit-dominant bitwise arbitration, sometimes called the Lanning protocol, is a very old MAC mechanism. It was used in telegraphy about a century ago. This mechanism uses two (or more) classes of signal that have a dominance characteristic such that if more than one signal appear on the media simultaneously, only the (most) dominant signal is perceived by receivers. Each message begins with a sequence of bits representing the message's priority, most significant bit first. As each bit is transmitted, each transmitting node checks the value on the media. If the value transmitted by a node is recessive, but the value on the media is dominant, the node recognizes that it has a lower priority than some other node that is currently transmitting. As soon as a node recognizes that it has lower priority, it stops transmitting its message. The lower priority node(s) may again try to transmit after the current message transmission completes.

This arbitration method has a number of physical-layer issues. The "Wire-Or" glitch problem was described above. Another issue is the constraint that each bit must have a duration that is longer than the worst-case round-trip delay on the media. To this constraint, one must add the effects of local clock jitter, sampling granularity error, and the signal jitter caused by the DC component of these relatively large bits.

This type of arbitration has no fairness. It is possible for one node to use all the network bandwidth and cause starvation in all other nodes. The system designer must add fairness on top of these protocols.

4.1.1.3 Carrier Sense Multiple Access and Collision Detect.

Carrier sense multiple access and collision detect is the MAC used for IEEE 802.3 (Ethernet). A node that wants to transmit first listens to the media. If the media is busy, the node waits. If the media is not busy, the node attempts to transmit. If more than one node tries to transmit at the same time, a collision is detected. When a collision is detected, the transmitting nodes stop transmitting and try again later.

Well-known problems with this arbitration scheme include:

- It is nondeterministic—e.g., miniscule changes in timing can cause changes in message order, and there is the small (but unknown) probability that collisions among transmitters can recur until they abort and then recur such that no messages ever get delivered.
- It has no fairness guarantees.
- It turns simple deaf nodes into babblers.

4.1.1.4 Time Division Multiple Access.

Time Division Multiple Access (TDMA) and its variants use a preagreed order of transmission in size of messages. These types of MAC require some form of clock synchronization, leading to these questions:

- How is the message schedule determined and agreed upon?
- How are system clocks synchronized to ensure that all nodes have the correct notion of system time?

4.1.1.5 Token Passing.

In a token passing MAC, the node that currently has access to the media must hold a token. For another node to gain access to the media, the current node must pass this token on to the other node. Some problems that can happen with token passing include:

- Corrupted tokens—which means the next node that should have gained access to the media will not know that it should have done so, and traffic will cease.
- Swallowed tokens—where the current node holding the token dies before it can send the token on. Again, traffic will cease.
- Counterfeit tokens—to solve the above problems, new tokens have to be minted. Failures in this mechanism can cause duplicate or counterfeit tokens.

4.1.1.6 Mini-Slotting.

A node using a mini-slotting MAC measures time from the end of each transmission. A node is allowed to transmit if the time it measures exceeds a threshold unique to that node and no other node has started to transmit. ARINC 629 uses a variation of this MAC. One problem with basic mini-slotting is that it has no fairness. ARINC 629 attempts to solve this problem by adding another timer that blocks a node from transmitting more than once in a period that is long enough to allow other nodes fair access to the media. However, this scheme does not prevent a node (or multiple nodes) from transmitting for a length of time that will starve other nodes.

4.1.2 The MAC Replacements.

Many data networks used in dependable real-time systems use the hardware from an existing data communication network that has an inadequate MAC and then apply a substitute MAC on top of the existing hardware. This effectively removes the MAC and turns the existing data communication network node hardware into something that is little more than a simple SERializer and DESerializer (SERDES) that just converts parallel data to serial data and back again, but requiring much more hardware (e.g., in the form of gate count) than would be needed to build just a SERDES. Many such networks are based on IEEE 802.3 (Ethernet). The system designer must consider whether the excess hardware can cause problems under unintended circumstances.

4.2 LINE-LEVEL ENCODING.

Line-level encoding is the way that logical data is physically represented on a data network. As discussed in section 3.2, bits on a network can affect each other via ISI. The characteristics of a network's line-level encoding can heavily influence ISI. In addition to affecting the data network's own signal quality, line-level encoding can also affect other equipment via radiated emissions. It is important to determine whether the spectrum radiated from the line-level encoding has components in frequencies that can adversely affect other equipment.

4.3 MESSAGE FORMATTING (FRAMING).

The message formatting or framing part of the LLC sublayer handles groups of bits sent over a link as discrete units. A message (also known as the frame or a packet) may contain control and addressing information, as well as error detection, for example cyclic redundancy check (CRC) information or forward error correction information. The size and composition of the frame varies according to the protocol. Depending on the protocol, components of a message may include preamble, start delimiter, source address, destination address, routing information, length field, flow control information, MAC information, error detection or correction information, or end delimiter.

In evaluating the dependability of a message format, the consequences of any part of that format having an error must be examined.

Preambles need to be of sufficient size to restore DC levels to the nominal value, facilitate synchronization of the bit-sampling clock to the incoming data stream, etc. Because DC levels may not have nominal values during the receipt of a preamble, there is a good probability that receiving nodes will see errors in the preamble. Some poor receiver designs assume these errors will always be at the beginning of the preamble and thus, only tolerate errors there. A more robust design would tolerate any number of failures in the preamble except for errors that make the preamble look like the next part of the message, typically a start delimiter. This is possible because preambles typically are highly redundant with no unique information residing in any one bit.

Are there parts of a message where an error could cause the loss of more than one message? This question includes not only bit errors that occur while the message transits drivers, media, and receivers, but also erroneous values that may be created by the source node or intermediate stages. An example is the corruption or counterfeiting of a token bit pattern in a token-passing MAC.

Other than redundancy bits (e.g., error detection or correction fields), is the message format efficient? Note that inefficiency leads to more bits, which leads to greater possibility of an error. Related to this concept, is the observation that some information that is transmitted in a message in one protocol (where it is vulnerable to errors) may not be transmitted in other protocol. For example, there are table-driven protocols in which all addressing, length information, etc., are held in a memory protected from errors rather than being transmitted on the network. There also are protocols that use redundant signal lines for error detection and correction instead of adding

check bits to the message. Combining these two ideas, a data network is possible (such as SAFEbus) where messages have absolutely no overhead; every message bit is a data bit.

4.4 ERROR DETECTION.

Network criteria that have significant influence on the overall safety are the error detection capabilities of the link layer, because efficient error detection directly affects the integrity of the data. A key of the underlying effectiveness of the error detection mechanism is the assumed failure model of links. It is often assumed that link failures are primarily bit flips, and the vulnerability of the link-layer error detection mechanisms to undetected errors in data are evaluated in the context of BER. Yet, the BER effectiveness evaluation is only one criterion to be evaluated. Error detection criteria should stretch to include effects such as wire crosstalk and correlated errors, such as HIRF events, unless mitigated with other means (such as shielding).

This section discusses error detection of the link layer. Link-layer errors can occur in the communication media, in its drivers and receivers, or in intermediate nodes (such as repeaters.) Section 5 addresses some error detection mechanisms that may reside in the equipment at the ends of the network or at intermediate stages within the network.

4.4.1 Protocol Violation Error Detection.

Detection of errors on the link layer should include the evaluation of the strength of a network protocol state machine to detect errors that are semantically incorrect. For example, message format fields may exhaustively use all combinations of possible values. Implementation of the protocol and protocol state machine should be able to detect such violations caused by values that are not valid. Otherwise, such erroneous messages may be interpreted in a nonintended way resulting in safety implications.

4.4.2 Parity and Frame Check Sequences.

Parity and frame check sequence evaluation criteria not including the adequate description and validation of error pattern may result in use of mechanisms that do not have adequate error detection capabilities. Typically, the validated BER can be used to for adequate error detection coverage assessment. There are many different error detection mechanisms and encodings, such as CRC, Fletcher, Adler, AND, XOR, etc., with different characteristics; however, this report focuses only on the characteristics of a few representative mechanisms.

CRCs are one of the most commonly used error detection schemes. The metric most commonly used for determining the quality of CRC error detection is Hamming distance (HD), i.e., the minimum number of independent bit flips that can result in an undetected error. Given the HD and BER for the medium, the designer can compute the probability of an undetected error. This probability should be sufficiently small for the reliability requirements of the data network. However, this coverage assumes that the bit errors are independent, an assumption that is known not to be true with the existence of ISI. With inter-bit dependencies, the calculation for error detection probability would have to be adjusted accordingly.

Another error detection metric is the ability to detect error bursts. An error burst of a particular length n is defined as sequence of n bits, the first and last of which are erroneous. The CRC can detect all error bursts of length k (where k is the degree of the generator polynomial) or smaller. While CRCs can also detect some error bursts longer than k bits, some error patterns are guaranteed to be undetectable, so the CRC should not be relied upon to detect error bursts greater than k bits in length.

Especially high error rates and correlated error probabilities may be encountered in wireless networks, where there is basically no shielding from external effects. The worst-case analysis may become a real challenge for such networks in the aviation digital electronics domain. Architectural means, such as voting of triple-redundant data channels as mitigation to inline error detection techniques, can only detect errors on the link if the channels are truly independent. Wireless network connections may be extremely vulnerable to common-mode effects on different channels due to unavailability of shielding protection.

Inline error detection may not only affect integrity (namely the probability of undetected errors), but also availability. While BER can be a useful figure to describe environmental effects, and the integrity mechanisms can be very effective in detecting errors, the detection of an error again has implications on the availability of data at the end node. Detected erroneous messages that cannot be used by the application result in decreased availability. The longer a message gets, the more likely a message may not be available due to an error. Unavailability of data can have safety effects similar to incorrect data.

4.4.3 Interactions Between Line-Level Encoding and Error Detection.

In addition to the effects of ISI causing inter-bit dependencies on the medium, line encoding can cause further dependencies among the bits due to the encoding and decoding processes. Typical line-encoding transformations include symbol encoding (e.g., Manchester and 8b/10b) that produce symbols (each of which consists of a block of bits), bit stuffing, and phase encoding. For most of these commonly used line-encoding transformations, the corresponding decode processes can cause error expansion. That is, even a fault-free decode process can create more errors on its output than it has on its input. To accurately calculate the coverage of inline error detection mechanisms, this error expansion must be taken into account. On the other hand, the line encoding itself often can detect errors on its own. Because the interactions between the line encoding mechanisms and error detection mechanisms are generally ignored in coverage calculations, published error detection coverage values for most networks are incorrect and must be recalculated.

In assessing the safety of the system, the potential impact of line encoding on the error detection capabilities must not be overlooked. Such interactions should be examined for the worst case. As the CRC (or similar inline error-encoding mechanism) is computed over the data, which is then transformed to a representation that is sent over the physical layer, the properties of the error detection change. Properties change because the encoder and decoder transform the representation. As a consequence, a single bit flip can result in multiple bit flips for the data at the link layer where the CRC is calculated. Similarly, the perceived error burst length that a CRC can tolerate may be shorter than expected due to the encoding. Figure 2 depicts a scenario of data with a frame check sequence (FCS) that is encoded using 8b/10b as transmission format.

The actual error burst is smaller than the maximum error burst tolerated by the CRC. Yet, due to the decoding of the physical data, the perceived error burst as seen at the receiver is longer than the tolerated value. If not considered, such interactions between encoding and error detection can invalidate error detection analysis.

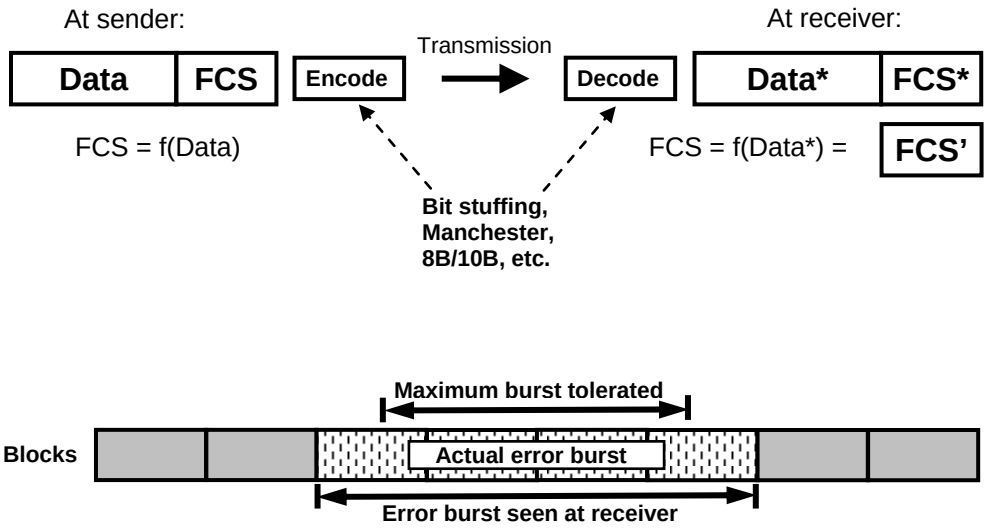

Figure 2. Error Burst Length Extension due to Encoding

A similar effect between the coding and physical layers is the multi-bit error vulnerability of protocols that employ bit stuffing to guarantee a minimum number of transitions at the line level. To properly encode and decode the bit-stuffed data, the entire message, including any CRC or other error detection field, must be bit stuffed. However, a small number of bit flips in a bit-stuffed message can result in a cascade error where data bits are interpreted as stuff bits and vice versa. In this case, the number of actual bit flips (as few as two can cause the cascade error) can result in a much larger number of bit flips in the decoded message that can exceed the error-detecting capabilities of the CRC.

When evaluating the overall error detection coverage, the error detection probability of the FCS, the error detection probability (if any) available from the coding scheme, and the possible interactions between the two must be included.

Active intermediate stages (such as network relay stations containing active logic, i.e., silicon devices) may defeat any inline error detection mechanisms (such as CRCs), because the potential failure mode of such devices may be arbitrary. The assumption of a uniform error model may not hold for such scenarios, because silicon failures may transform inline error detection codes in a way that the frame check sequence is unable to signal an error in the worst case.

5. NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

In the OSI model, the network layer provides switching and routing technologies, creating logical paths, known as virtual circuits, for transmitting data from node to node. Routing and

forwarding are functions of this layer, as well as addressing, internetworking, error handling, congestion control and packet sequencing. Above the network layer, the transport layer provides transparent transfer of data between end systems, or hosts, and is responsible for end-to-end error recovery and flow control. It ensures complete data transfer. In embedded systems, the functionality of these layers is often merged into a single layer of functionality. This section discusses the issues related to the functions of both layers together. In addition, in some newer protocols, (for example TTP/C and FlexRay) a network management layer is emerging to describe hardware or software services that facilitate message agreement, network diagnosis, and synchronization. Issues relating to these issues are also discussed within this section.

5.1 NETWORK VULNERABILITY TO ADDRESSING INFORMATION FAILURE.

If the network technology encompasses message labeling or node addressing identification information, then the failure modes of the addressing or labeling mechanisms need to be evaluated (as such mechanisms may be vulnerable to component failures or transport corruption). An example failure mode is the masquerade failure, where one network node can impersonate another node of the system. Failures of addressing or message labeling information are especially important in integrated modular aviation digital electronics systems comprising numerous aircraft functions, because failure of these mechanisms can lead to unbounded data-flow failures, which makes functional failure isolation almost impossible at the application layer.

The vulnerabilities of the data network to technology shortcomings may differ depending on the network implementation. If the network packet format includes addressing or other information that indicates message content (e.g., a message or label identification), then the network is obviously vulnerable to corruptions of these fields during transmission. For the network to be dependable, there must be mechanisms to handle any such corruptions. These mechanisms must be evaluated to establish their coverage (their ability to handle these corruptions). Note that network integrity mechanisms (e.g., frame-check sequence) may detect transmission errors; however, these mechanisms have limited coverage, as described in section 4.4. Any fault-handling mechanism must have provable coverage against all possible failure modes of the communication channel. Message routing and blocking enforcement of intermediate stages (e.g., guardians, etc.) based on addresses or labels also need to be evaluated to ensure they provide adequate coverage, as described in section 7.2.

The influence of software on network addressing information is also an issue, as discussed in section 5.8. Such software-directed access may leave a network vulnerable to failures that corrupt the addressing information.

In addition to the vulnerability from dynamic errors incurred during transmission, many network technologies require configuration tables to assist the network routing and addressing logic that may be vulnerable to static errors. The mechanisms to ensure the design correctness and run-time integrity of these configuration tables must also be evaluated and justified. These issues are discussed in sections 5.3 and 8.

In some network technologies, routing information and logical topologies may be built at run time. An example is the tree-building discovery protocol of IEEE 1394. These mechanisms must obviously be evaluated in relation to their vulnerability to component failures or data

corruption, unless failure modes or error detection can be suitably justified. The vulnerabilities that may cause the erroneous invocation of mechanisms that recreate routing information and logical topologies must also be understood and analyzed, since such invocation may seriously degrade (if not prevent) network operation. These issues are further discussed in section 5.5.

Similarly, network error-handling logic (which may be invoked by erroneous addressing information or which may impact protocol flow) needs to be analyzed to establish a bound on the influence of the invocation of the error-handling logic and its impact (i.e., degradation) on network performance. The behavior of any such logic and its associated vulnerability needs to be analyzed and justifiably bounded. This is especially true for centralized intermediate stages, as discussed in section 5.3.

5.2 NETWORK VULNERABILITY TO FLOW FAILURE.

As with network addressing failures, the network technology's flow regulation logic also needs to be evaluated. Issues relating to acknowledgement and retry logic are discussed in section 5.9. Issues relating to host-interface load balancing and buffering are discussed in section 6.1. Issues relating to intermediate stages are discussed in section 5.3.

5.3 IMPACT OF INTERMEDIATE STAGES.

If a network encompasses intermediate buffering or relay stages, then the behavior, implementation, and impact of the intermediate stages need to be established and evaluated with the network behavior. The behavior of these intermediate stages can vary considerably with network implementation. In simple form, they may be solely relaying stages. In more elaborate schemes, they can comprise store-and-forward and routing logic. Finally, in critical networks, it is common for such intermediate stages to incorporate error detection or fault-containment mechanisms. This section discusses some of the issues and network attributes related to such intermediate buffering schemes that need to be considered and evaluated.

5.3.1 Vulnerability to Intermediate-Stage Failure.

In networks that deploy intermediate stages, the influence of the intermediate-stage components may be significant. For example, in networks using stars or hubs, the intermediate-stage component impacts all data flowing through it. The availability of the intermediate-stage component must therefore be analyzed and justified to be adequate to fulfill the network availability requirements. If multiple intermediate stages are deployed, then the independence of intermediate-stage failure should be analyzed and suitably justified. If intermediate-stage-to-intermediate-stage signaling is required, then this signaling and logic needs to be analyzed for failure vulnerabilities and possible fault propagation. Similarly, any protocol common mode influence on the intermediate-stage availability must also be understood.

Integrity implications of intermediate-stage mechanisms must be carefully analyzed and evaluated. One difficulty of such an analysis is bounding the failure modes of the intermediate-stage component. Since the intermediate stage influences every bit that it is relaying, the effects of a faulty intermediate-stage component can be significant. The integrity implications of a failing intermediate stage are very much dependent on network implementation and architecture.

For example, in the ROBUS network elements of the SPIDER architecture, which votes data from three independent channels, the failure of a single intermediate-stage component can be easily detected and is effectively masked from the receiving node.

Alternatively, if a network intermediate stage is developed to have full coverage (that is, all faults are covered; for example, by using self-checking or monitoring schemes), then the failure modes of the intermediate-stage component may be suitably justified as benign (e.g., fail stop). It is imperative, however, that the coverage of the self-checking or monitoring scheme is suitably justified, as discussed in section 7.4.

It is common for networks to rely on inline integrity mechanisms, for example, CRCs checksums, parity, etc. In such cases, the failure modes of the intermediate-stage component become more significant, since the network integrity is dependent on the coverage of these codes. With complex intermediate-stage logic, it is difficult to bound failure modes of the intermediate devices and relatively simple failure modes may have significant impact on inline coverage techniques. To illustrate the impact of a relatively simple failure mechanism, consider the scenario of an intermediate-stage elasticity buffer erroneously underrunning or overrunning. If the result of such an overrun or underrun is the insertion or deletion of a single cell from a relayed Manchester stream, the resultant relayed stream may suffer a cell shift that causes data corruption for the remainder of the transmission. If such a failure is not detected by the encoding or framing scheme, this shifted data stream may easily defeat CRC coverage (as discussed in section 4.4). In such cases, the data integrity claims of the network are therefore limited to the failure rate of the relaying component. This scenario is important, as it illustrates the interdependencies of the error detection logic (i.e., the framing and encoding layer) strength and the CRC coverage. Strict enforcement and error detection mechanisms may strengthen the data integrity claims; and with that said, quantifying such behavior may be difficult. It is also important to understand where the error detection is performed. For example, if the error detection is only performed at receivers, and intermediate stages do not perform such action, the reshaping and retiming behavior of the intermediate stage may degrade the end-to-end effectiveness of such detection (i.e., the scenario of erroneous signals at the intermediate-stage input getting "cleaned-up" and reshaped by the intermediate-stage action that produces a relayed output stream with no encoding errors). The impact of reshaping and re-encoding layers of intermediate-stage logic also needs to be considered in this regard.

In addition to hard or transient logic errors, intermediate stages may also be vulnerable to out-of-specification behavior. For example, clock drift may lead to similar overrun scenarios as described above. The network vulnerability to such errors together with the potential contributors to such out-of-specification behavior need to be understood as the network is evaluated. It should be noted that there may be systematic contributions, such as long-term drift of oscillators and their performance under aging and temperature variations, etc. They may also be due to local transients, for example, acceleration and gravity forces on crystals or phase-locked loop modulations resulting from power supply instability or fluctuation. It is obviously important that the intermediate-stage elasticity buffers are sized to accommodate such variations. In addition, the intermediate-stage reaction and response to out-of-specification errors is another attribute that warrants careful consideration.

For intermediate stages that encompass store-and-forward behavior, the situation is complicated further since the behavior of the intermediate-stage component is more complex. The vulnerability of the intermediate buffer memory to transient upsets (such as SEUs) needs to be established. It is preferable if some form of protection is in place. If only error detection is in place (for example parity mechanisms, etc.), then intermediate-stage response to such errors needs to be analyzed and understood. For example, if a parity error causes a reset or machine-check exception, then the availability of the intermediate stage would be impacted as the reset procedure is initiated. The vulnerability of the intermediate stage to SEU upsets and the subsequent reinitialization time will then need to be considered when justifying network channel availability. It should be noted that a similar analysis is also required for software implemented switching schemes that use RAM-based data with parity-type schemes.

Similarly, if the intermediate buffer memory is not protected via parity or error detection and correction (EDAC) schemes, then the impact of such upsets on end-to-end integrity claims needs to be understood. Obviously, should the intermediate stage perform recalculation of integrity checksums (such as CRCs), then the impact to buffer memory upset is limited by the SEU vulnerability of the intermediate buffer RAM.

In addition to buffer memory errors, faults of the intermediate-stage configuration and routing tables also need to be analyzed in a similar manner. If these are not protected, then the impact of erroneous control flow, routing information, etc., needs to be carefully analyzed. The responses to detected errors also need to be understood in relation to their impact on intermediate-stage availability, as discussed above.

Permanent faults of the intermediate-stage control and buffering logic need to be considered. As with the simple relaying logic, these may impact data integrity claims. In addition, when buffering action is present, the vulnerabilities to erroneous message forwarding need to be analyzed and understood. Network mechanisms to detect old or out-of-order packet forwarding must therefore be analyzed and evaluated with the network performance, unless suitable benign failure modes of the intermediate stage can be justified via coverage techniques (self-checking, monitoring, etc.).

For protocols that incorporate control flow information in the transmission, for example the reset-sequence indicator proposed by ARINC 664 part 7, the erroneous behavior of a single network channel may impact higher levels of redundancy management and degrade the performance of independent redundant network channels. Such mechanisms and potential fault-propagation paths need to be considered when justifying the network availability. For example, consider the scenario of an ARINC 664 part 7 “babbling” switch that only sends frames with sequence number of 0. Such a failure could be due to a stuck address line in a switch, resulting in continuous sending of the same frame, which might happen to be a frame announcing a reset. The receipt of such a frame on either channel (of the dual network paths) causes the receiving node to reset its frame sequence, if the redundancy mechanism of ARINC 664 part 7 is used. Hence, the erroneous channel may upset the sequencing and data flow of the independent good channel. Note that due to the centralized position of the intermediate stage, failures, such as those described in the previous sections, may touch many parts of the system. Hence, the

common mode influence of such intermediate-stage behavior needs to be carefully evaluated and understood.

In addition, the intermediate-stage buffering mechanisms and protocol interactions need to be understood to mitigate issues relating to head-of-line blocking. This same problem can occur in application services, as discussed section 6.1.

5.3.2 Vulnerability of Intermediate Stage to Fault Propagation.

The vulnerability of the network intermediate stages to faults propagating from erroneous end nodes should be established. Such vulnerabilities may be related to erroneous control data, or erroneous temporal behavior. For example, consider a central guardian reintegrating onto a TDMA scheme: If the TDMA position is resolved by listening to TDMA sequence index indicators included in the TDMA traffic stream and if the integration logic of the intermediate stage is not tolerant to erroneous information, a faulty node may be able to delay or prevent the intermediate-stage recovery.

Similarly, incorrect flow management may impact the intermediate stage or switch performance. For example, babbling end nodes or other babbling intermediate stages and switches (sending syntactically continuous frames) may impact available buffer space and cause overruns unless suitable enforcement and error containment policies are in place. The buffer management policies and associated buffer sizing, etc., need to be carefully analyzed as network performance under normal and erroneous node behavior is justified.

Finally, any error handling logic that may be invoked in response to erroneous end node traffic and behavior should also be analyzed such that any associated intermediate stage and switch performance degradation or other propagated erroneous behavior can be suitably bounded.

5.4 NETWORK CONFIGURATION DATA.

Many network technologies require configuration and routing tables to be programmed to assist network operation. Therefore, design correctness of these tables is obviously important to correct network operation. Design assurance issues relating to network table correctness is discussed in section 8.4. The run-time integrity of the tables is also important. Therefore, the storage, operation, and load integrity mechanisms of the configuration data need to be evaluated with the network technology. This examination should also address run-time table placement, for example RAM protection schemes such as parity, EDAC, etc. Similar considerations to those discussed in section 5.3.1 in relation to buffer protection and recovery actions should be considered in relation to run-time configuration table placement.

In networked systems, the consistency between the copies of run-time tables in different nodes is also an important issue. Hence, protocol mechanisms to ensure table consistency should be evaluated. However, as discussed in section 5.5, the availability impact of such consistency enforcement mechanisms needs to be considered. In systems where the network tables and system or application software are tightly coupled, mechanisms to ensure software and network compatibility are needed. This is especially true if the network tables are configured separately from the application software images.

The mechanisms to load the configuration and routing tables are also important. First, the integrity of the loading mechanism needs to be established such that the configuration data does not get corrupted during the load process. Second, if the table load mechanism uses the same data path as the normal network data flow, the partitioning properties of the network path have to be established. For example, some network technologies use dedicated load protocols to facilitate the loading process. The interlock mechanisms and mode selection logic used for such load protocols need to be analyzed to ensure that the erroneous invocation of the protocols does not degrade network availability. In addition, network tables may need to be updated in a live operational mode; for example, when a network is running minimal network traffic to support the hotel functions of operating doors, lights, and basic power distribution. In the live operational mode, the mechanisms to coordinate mode change and table switching need to be carefully evaluated.

Similarly, network maintenance and query protocols sometimes are used on top of the network infrastructure, such as in the Simple Network Management Protocol (SNMP). Such network maintenance protocols also can introduce safety implications. The network vulnerability to the actions performed with such protocols need to be analyzed and considered. For example, if it is possible to invoke software exceptions via these maintenance interfaces, the impact of the exception processing on the normal application functionality needs to be bounded.

5.5 START-UP AND RECOVERY.

Network start-up and recovery mechanisms are important since, in critical environments, start-up and recovery time of the system is often a key attribute of the system performance. The behavior of network start-up performance is, therefore, another attribute that requires careful evaluation.

During start-up, the network is usually more vulnerable to faults. Unless the start-up algorithms have been designed to be fault tolerant, or the network hardware has been designed with adequate fault containment, it may not be possible to guarantee correct nor timely network start. In such cases, the availability of the network channel will need to be re-evaluated to consider the impact of potential contributors to erroneous start-up action. To illustrate such a scenario, consider a TDMA protocol where the initial frame of the protocol contains a table version identifier. This may be sent explicitly or implicitly, i.e., buried within the CRC calculation of the network frame. If the first node to send at start-up sends an incorrect table identifier and the response of the other "good" nodes receiving such a frame is to back off for a defined period of time, the erroneous node can hold off network start-up. If that node continues to send, then network start-up action may be delayed indefinitely. This is an interesting example, as it illustrates the interaction between availability and integrity mechanisms, which often occurs when integrity patches are implemented on top of networks that have been designed with a availability mindset, which is very common in commercial off-the-shelf (COTS) protocols.

Another interesting issue relating to start-up is the constraints the network implementation may place on external aircraft systems; for example, power sequencing. The network may assume that network components are powered-on within certain intervals or in a specified sequence. Such is the case when central guardian action is assumed as described in the section 7.2. In time-driven networks, the alignment of start-up behavior may significantly impact network start-up. Such issues are further discussed with the host-network interaction in section 5.8. It is, therefore,

desirable to note such constraints while the network is being evaluated. If the network is required to be safety critical, the assurance of the network-assumed behavior may drive considerable complexity and cost into these other systems. However, without such assumptions, the network justification and associated availability claims may be incomplete.

In some network technologies, for example IEEE 1394, network addressing and routing information is built when new nodes are added to the network. Such behavior should be carefully analyzed for its impact on network start-up time, which will essentially be limited by the slowest node. A faulty node, for example, a node undergoing continuous restart behavior, may also need to be considered (depending on the fault containment and coverage of the nodes implementation). If sufficient coverage cannot be justified, such a faulty node may disrupt network availability by continuously initiating network restarts. In such a scenario, all the components that may contribute to such a failure need to be analyzed while network availability is being justified. In addition, the fault tolerance of discovery-routing protocols should be established and analyzed to bound the influence of faulty logical behavior.

Authentication is another issue that is often worth considering during start-up. This is especially true in TDMA networks that may use the temporal order of messages to assist as an authentication mechanism when the network is running. At start-up, when there is not an established time base, this authentication technique is not available. Hence, the network may be more vulnerable to authentication failure. This is particularly interesting for dual-channel networks, as the lack of suitable authentication may enable a faulty node to impact multiple transmissions as it appears differently on each of the channels. If the algorithms of the associated network are only tolerant to a single failure, then such a dual-error manifestation may break the protocol assumptions and prevent correct network operation. This may even be true if guardian schemes are in place (because the guardians may lack suitable authentication capability). The strength of guardian enforcement is another attribute that requires careful consideration. This is discussed in section 7.

In general, it is important that the network technology protocol mechanism and algorithmic claims are carefully evaluated for their performance during start-up. For example, a clock synchronization algorithm may tolerate a Byzantine error when the data network is fully up and running, but may require a certain minimum number of correct nodes to be up before the fault tolerance mechanism can operate correctly. The impact of the algorithmic behavior when fewer clocks than the minimum are available should be understood. For example, what is the impact of having only two clocks available for an algorithm that requires four clocks to be Byzantine fault tolerant? Bounding such effects is required as the network performance and safety case is evaluated. In addition to the analysis of the network's start-up mechanisms, network technology re-integration mechanisms also need to be analyzed to establish their tolerance or vulnerability to erroneous protocol control flow information that may delay or prevent a node's timely integration.

Many algorithms and mechanisms are designed to work correctly only if some minimum amount of good resources are available. However, just prior to start-up, it can appear that everything has failed. A good design for start-up must be able to get past this "everything has failed" phase and

be able to bootstrap itself up to full operation. However, all too often, network designs assume that the network was "born running" and cannot tolerate failures during start-up.

5.6 GLOBAL SYNCHRONIZATION.

Data networks may have a need for clock synchronization within nodes for coordinated network access or as an application-layer service. The following paragraphs focus on clock synchronization, but a subset of the aspects considered for clock synchronization is equally applicable to synchronization of "logical clocks" (counters) used for redundancy management.

Clock synchronization algorithms consist of several steps during synchronous operation. The first step is the initial clock acquisition, in which synchronizing nodes acquire the current clock values or counter values from one or more different nodes. This can be done via messages prescheduled according to a table or can be triggered upon request. After a node has acquired synchronization with the data network, it must maintain that synchronization. It does so by periodically acquiring clock difference information from the other nodes. This can be done, as in the initial synchronization process, via messages prescheduled according to a table or can be triggered upon request. Another method is to time the arrival of normal data messages (i.e., expected arrival time of a message versus the actual arrival time) and infer the current state of the clock of the transmitter of that message versus local clock state. After preprocessing of the clock difference information (such as for the elimination of propagation delay influences), the second step at each node is the calculation of the correction value for the local clock based on the (preprocessed) clock values. This step is sometimes called the convergence function, because it should ensure convergence of the distributed node clocks towards a common clock. The next step is applying the correction value to the clock in order that all nodes have clock times that are more closely synchronized to each other than before the synchronization step was taken.

Several properties and influences, if not mitigated, may lead to an unstable or failing clock synchronization algorithm, which can lead to potentially unsafe system state.

The clock synchronization algorithms depend on the propagation delays through the network. Different propagation delays between different nodes or different data acquisition delays at nodes may lead to inconsistent or inaccurate views of the actual propagation delay, which is used to judge the clock difference between different nodes. Such differences may have effects on the stability of the algorithms. The most often cited challenge in clock synchronization is synchronization in the presence of Byzantine failures. Byzantine in this context means that different nodes have different views of the clock values from another node. It seems hard to quantify the possibility of Byzantine scenarios in the clock synchronization at first. Considering that clock synchronization algorithms often measure the difference between the expected arrival of a message and the actual arrival, any arrival time differences of sync data in the context of different or slightly varying propagation delays lead to scenarios similar to Byzantine fault scenarios. If the system does not compensate for propagation delay differences, the views of the clock values can be significantly different. Compensation of propagation delays decreases the difference but does not remove them. Such scenarios of different views of clock values at different nodes, if not considered in a stability analysis, may pose a safety thread. The stability analysis is normally captured in an analytical bound of a precision value.

The clock data values may also contain some faulty values either due to failures during propagation or due to end node failures. Considering source coverage, the correction-value calculation may have to tolerate certain failure modes to achieve a bounded precision value. One issue in clock synchronization is the combination of clock values from different communication paths that stem from the same source. In the case of triplex redundant channels, the clock values arriving on different communication channels can be voted. For dual replication, voting is not possible. In certain topologies, an intermediate active component may have effects on all values from different clock sources that travel through it; e.g., a single, short-circuited, central star component in TTP/C or FlexRay may have influence on all clock values from different sources. If the star topology is only dual, which is the case for most well-known data networks aimed at embedded real-time systems, such as TTP/C, FlexRay, and AFDX, these failures of the stars can cause failures in the end systems that cannot be mitigated by the end systems themselves because the duel inputs from the data network cannot be voted. The selection process between the different replicated communication paths determines the amount of influence a failure in one of the communications on the overall precision values and the effect, unless restricted, may have dependability implications.

The algorithm calculating or selecting the correction amount for the local clock should consider the assumed failure conditions and the number of faulty nodes it may have to tolerate in the context of the systems source coverage mechanisms to ensure a bounded precision. An analysis of the precision should consider the effects of potential masquerade failures. For example, in FlexRay, the correction value calculation function can tolerate several incorrect clock values stemming from different nodes, but may be unable to tolerate a single faulty node, if the node fails such that it masquerades as other nodes when transmitting synchronization frames.

In case of Master and Slave synchronization schemes, the switchover time from one Master to another Master used for synchronization may need to contain the time to diagnose a faulty Master. During such a diagnosis time, Slave nodes may still synchronize to their (faulty) Master node or not synchronize at all. Both scenarios can affect the precision.

When a node applies the correction of the clocks to its local clock, any task dependent on the local node clock may have to consider potential influences of this correction on its execution time. For example, the correction of clocks may have implications on the period available for the execution of tasks. Effectively, the execution time available to a node may be shortened by the precision due to correction and coordination with other nodes. This influence may have an impact on the execution time available to tasks.

The clock synchronization algorithm needs to be careful in what information it uses to do clock corrections. For example, if the correction function of the clock synchronization algorithm uses the same collected time values twice for calculation of the correction values, the algorithm can become unstable. Such a scenario may happen during start-up of the system. Such instability has been observed. Any tool verifying the stability properties of clock synchronization or person analyzing the properties should consider such effects and different configurations. Also, the compatibility and consistency of the clock synchronization configurations at different times should be checked.

Start-up of clock synchronization may have availability implications if it is dependent on the fault coverage of a single node. Similarly, during start-up, clocks may drift for a much longer time than during normal operation, because less or insufficient clock value sources are available. Such longer drift times should be included in the stability analysis and, consequently, the precision values.

5.7 FAULT DIAGNOSIS.

Some network technologies include fault diagnosis services to identify and isolate faulty member nodes. The services may run autonomously in the network hardware or comprise software application services that run on top of the network, which diagnose information provided by the network layer. Such services are strongly related to group membership and interactive consistency services, which may use fault diagnosis services to manage network state dependent application decisions and guaranteed consistent delivery of messages.

A group membership service delivers the operational status of some or all nodes of a data network to other nodes. Group membership service, or variants thereof, is a subset of an interactive consistency service. Group membership indicates the operational state of nodes (ideally consistent), while interactive consistency provides consistent agreement of nodes on any (sent) value.

Group membership information indicates the health state of a node. From such information, it can be concluded that the node is operating correctly. Yet, the information that a node sends out may not reflect the current state of a message; e.g., the operation of a node as indicated by group membership does not ensure that the message is correct (integrity violation) unless sufficient error detection coverage for the node and the communication path is assured.

Group membership is usually derived from the correct or incorrect reception of a message from a node. If these messages are correct, group membership infers that the node is correct. On the other hand, if the reception of a frame is not correct, group membership mechanisms can attribute this to a transient or permanent fault on the communication path. In aviation digital electronics systems where transient upsets may be experienced in relation to power drop-outs or massive upsets from HIRF or lightning events, the ability of the diagnosis schemes to distinguish transient external upsets and permanent node errors should also be carefully analyzed to ensure that the diagnosis algorithm meets the real-world expectations. The persistence of any indictment action that may result from the invocation of such diagnosis needs also to be understood to ensure that the loss of network availability resulting from such indictments is suitably bounded.

Requirements of group membership being consistent and effects such as inconsistent reception-status of messages at different receiving nodes and potential consequences are discussed in detail below. In general, it should be said that any diagnosis service will be nonperfect, e.g., due to transients having local effects or due to failure modes of the sending nodes (e.g., Byzantine failure modes).

Diagnosis information can be used by the network to build additional services for management of redundancy sets or simply as acknowledgement. In this context, the use of the diagnosis information needs to be in alignment with the expectations of the applications.

Such services are group membership or interactive consistency. Group membership from an application perspective intends to use the membership information for the selection of the correct data, while interactive consistency intends to provide the data consistently at all nodes.

Group membership is often based on the reception of messages from nodes and will have some temporal lag until state changes are updated. Effects of time lag should be considered in the evaluation process.

It has been proven that group membership in arbitrary fault scenarios (without source coverage) cannot guarantee correctness and consistency at the same time, where correctness means that all correct nodes are regarded nonfaulty by all other correct nodes (but faulty nodes may also be viewed as correct) and consistency means that all correct nodes are consistently seen as correct at all correct nodes. An implementation of group membership with insufficient coverage or insufficient communication rounds required (and theoretically proven) to tolerate certain failure modes may sacrifice either availability or integrity; e.g., enforcing consistency in a single-string implementation can lead to availability loss. One example of such consistency enforcement problem is the TTP/C membership algorithm leading to loss of availability for correct nodes. In certain versions of TTP/C, receiving nodes update their membership vector in accordance to their view of reception of messages from other nodes. In the case of an asymmetric view of message reception, the membership vectors at different nodes disagree. To ensure consistency, TTP/C implements a clique-resolution mechanism, which forces (possibly correct) nodes to shut down. In addition, the group membership algorithm may not identify the faulty node in TTP/C. This is an example of where a group membership algorithm implementation ensuring consistency may have implications on the availability of a communication network. Solutions to these problems, such as TTP/C's central guardian, may mitigate this particular problem in one topology (e.g., TTP/C star), but not another topology (e.g., TTP/C bus).

In CAN, any acknowledgement algorithm claiming consistency (atomic broadcast) despite an arbitrary failure mode should be analyzed in a similar manner. This is discussed in section 5.9.

Applications that use networks that provide group membership services should analyze

- the underlying assumptions,
- the consistency and correctness guarantees of group membership, and
- their effects on the application layer.

Such analysis and effects on the application should also include temporal aspects, because diagnosis information lags in time and so does membership.

During reintegration and start-up of the data network, the group membership information provided by a newly integrated or started node may include information about the system's state, which it may not have observed itself or may have obtained from other nodes. In detail,

integrating nodes may observe the network's operational state over a period of time or may integrate quickly by accepting other nodes' views of the operational state of the network. In the latter case, the use of the information provided by other nodes needs to be in alignment with application-level assumptions of the membership information; e.g., if an integrating node adopts the group membership state from other nodes and the application assumes that the membership state information includes agreement on or availability of application state information, it may also have to acquire the application state information that is associated with the semantics of a group membership bit.

5.8 CLIENT EFFECT ON NETWORK OPERATIONS.

A data network is often the glue that holds a dependable system together. A system data network tends to become either the main fault containment mechanism in itself, or is a major component of the main fault containment mechanism(s). As such, it is important for a system data network to not be adversely affected by the clients it serves, no matter how badly the clients perform.

Many data networks allow their clients to adversely affect their operation in several ways.

The first adverse interaction occurs immediately upon start-up. Many data networks allow their clients to influence the timing of network start-up by affecting the timing of their nodes. Variations in node start-up times can be caused by different host power-up sequences, different self-test mechanisms, etc., coupled with the requirement for the client to enable its node to participate in the network; e.g., in FlexRay and TTP and C, the host needs to switch on the controller. Different start-up times of node components should not be allowed to cause starvation of components (retry exhaustion). For example, after some retries of insufficient response, a FlexRay network chip starts again and the software needs to interact; if software is too slow, then there is no availability. It is possible for data network protocols to take an inordinately long time to start, or may not start at all, if the timing behavior of its nodes follow some pathological pattern during start-up.

During data network operation, some protocols allow clients to adversely affect their behavior if the clients can control addressing, routing, priorities, etc. Some systems require applications of different safety-criticality levels to share the network. When this is the case, the network must be robustly partitioned such that applications and clients of low criticality cannot adversely affect the use of the network by high-criticality application and clients.

Another possible avenue for a client to adversely affect a system data network is via unprotected test or network management paths.

5.9 ACKNOWLEDGEMENT.

For network protocols that employ acknowledgement schemes, the behaviors of this logic need to be carefully analyzed, especially with respect to inconsistent message reception (i.e., some nodes receive a message or an acknowledgement, and some do not). It cannot be assumed that any acknowledgement mechanism provides, by itself, consistent message reception (also called atomic broadcast). Also, the sender of the message may fail before resending the message; e.g., mechanisms with negative acknowledgement schemes need a way to signal such negative

acknowledgement. If signaling is not possible, inconsistencies may arise. An example for acknowledgement causing inconsistent message reception is the negative acknowledgement algorithm (sending of error flag) in CAN. If an inconsistent bit reception in the next-to-last bit occurs, some nodes will accept the message while others will not. In such a scenario, a retransmission will occur, leading to multiple message copies at some receivers and a single one at others. As a consequence, the delivery semantics have gone from "exactly once" to "at least once." Receiving nodes may not be able to distinguish the duplicate message from a legitimate second message, and message delivery to different nodes may occur at different times. In case the sender suffers a failure where it is not able to resend the message, permanent inconsistencies in message reception will arise. The implications of inconsistent message delivery, different message delivery times to the application and multiple deliveries should be analyzed with respect to the overall system and its safety.

Similarly, in a TDMA network (such as TTP or FlexRay) where acknowledgement vectors or bits are used, an inconsistent reception may cause system-level effects as described in section 6.3.1. Generally, if an acknowledgement is only based on an action of a subset of nodes, inconsistencies may occur as a consequence of the design. For example, the recessive and dominant physical-layer acknowledgement, where one receiver is sufficient to signal a dominant state, is an action of only a subset of the receivers. Also, the acknowledgement-signaling mechanism relying on the reception status of a subset of receivers, such as the reception status of the next one or two receivers, may be vulnerable to inconsistent reception.

Message retry mechanisms due to negative acknowledgement or missing positive acknowledgement may have implications on the network performance and maximum loading. To bound network loading, retry mechanisms should be analyzed for the number of retries to make sure they are bounded (or analyzed whether retries are enforced to be bounded via counters for retransmissions or bounded via timeouts).

Acknowledgement errors can affect application-level error handling or exception mechanisms, such as invocation of error routines leading into additional overhead for processors. Any safety implications of increased workload should be analyzed.

6. APPLICATION-LAYER SERVICES.

Current data network technologies comprise a number of application services that may or may not be used by an application. All services need to be analyzed in the context of a safety assessment. In its simplest form, any buffer management mechanism has associated properties that need to concur with the application assumptions. Newer generations of networks even supply voting schemes or redundancy management mechanisms. An example for such buses is ARINC 664. In this section, the criteria for data network services used by applications are examined.

6.1 HOST INTERFACE MANAGEMENT.

Buffer management should be concerned about the message access order to the network, partitioning requirements, and performance aspects of the network interface buffer as well as implications to the host.

6.1.1 Client Buffer Queue Management.

Buffer management of systems may have system-level implications. One example of system-level impact may occur if messages are associated with priority. In certain combinations of buffers and accesses, priority inversion on the system level may occur; e.g., certain implementations of CAN can have a priority inversion of messages.

These CAN implementations have a priority message queue that holds a large number of messages and an intermediate buffer that intends to contain only the highest two priority CAN messages. The buffer with the highest-priority messages is used for network arbitration and for serializing and sending on the network out of this buffer. This is called the sending buffer. One message position in the sending buffer (which is normally a dual-port memory) is intended for sending, while another position is intended for refilling from the larger message buffer with the next higher priority message while the message on the other position is sent on the network. If a higher-priority message (e.g., priority 3) arrives at the large message buffer, the lowest priority that is currently in the sending buffer (e.g., priority 4) needs to be replaced by this new message. If this replacement action coincides with arbitration on the network, another message with lower priority (e.g., priority 5) may win the arbitration, because one position of the sending buffer has just been sent and is empty and the other message (priority 4) is being replaced by a message with a priority 3 message. This is an example where a lower-priority message has won arbitration over a higher one. While this situation can be improved (i.e., not suffering from priority inversion) by supplying a sending buffer with space for three messages, similar situations may exist and the system-level implications, and safety implications should be checked due to such scenarios.

Similar phenomena can occur due to local buffer management. One example is a priority arbitration scheme where only a single first in, first out (FIFO) transmit buffer is used. If only the head-of-line message contends for the communication resource, the performance drop due to head-of-line blocking can be significant. In the worst case, a node may not get any access to the network and will not be able to send.

When evaluating networking technology for the deployment in systems, client buffer queue management mechanisms should consider effects on the access to the network such as fairness and implications to the network and the host.

6.1.2 Buffer Management Partitioning.

In a robustly partitioned system, software partitions running on a node have a strict execution budget and should adhere to their execution budget. On nodes where the data from a data network is managed by a direct memory access (DMA) controller, the DMA controller may repeatedly stall the execution time of running partitions, potentially having significant effect on the execution time of a software tasks. Unless such effects of cycle stealing are accounted for in the execution budget of software tasks or the overall node architecture, software may miss execution deadlines.

Partitioning violations due to addressing and masquerading nodes were discussed in section 5.1. Partitioning violations may also occur due to buffer management. In systems having

applications of differing safety criticalities running on one node (processor), each having common access to the communication buffer, any wrong access to the common communication system buffer can result in several undesired phenomena.

Unless the access to the common buffer is restricted or controlled for each partition, software partitions may overwrite messages of other partitions or use network resources from other partitions. A partition may even be able to send data masqueraded as another partition, unless protected. A simple (but potentially unsafe) example may be a common address area where all partitions have access but each partition is assigned a source address on the network based on its assigned range within the buffer. Any faulty access to another partition's memory area can result in faulty addressing on the network, masquerading effects, and data overwrite, just to name a few potential safety hazards.

Another area of control to the buffer is the coordination between the network side access and the software (or host processor) side of access to the buffer. The buffer management should be analyzed to ensure the mutual exclusion of buffer access (or access to certain areas). Unless the access is controlled for both the data area and potential control areas (interaction between status area potentially updated by the network, while at the same time, the software side tries to change or read the control information) interactions may result in unwanted effects. One example is the atomic write for messages sent on the network (transmit buffers). A message should not be sent until it has completely been written into intermediate buffers. If the contents of a transmit buffer are sent out onto the data network's media while software is still writing to that buffer, the resulting transmitted message could contain contents that are a mix of old contents and new contents. While such coordination may occur automatically for different processes due to scheduling of process execution on processors that is tied to a communication schedule for the data network, the dual-port memories often used for buffer management may be much more likely to be affected by such coordination errors.

Host access to receive buffer areas common to the network and to the host needs to be restricted or otherwise carefully controlled during the reception of incoming traffic from the data network. If not coordinated or controlled, data inconsistency in applications may arise. If blocked from either side, blocking effects should be considered. Blocking of message reception while the host reads the buffer (if such blocking is possible) may have system-level impacts, such as requiring the resending of messages or queuing at sender side. Blocking the host side access while the data network is updating a receive buffer may increase the execution time of software. In cases where dual buffers are used to allow incoming data network messages to be written to one buffer while the host software is reading from another buffer (ping-pong buffers), potential delays in the availability of data, and the switchover logic after message reception need to be considered.

Well-designed data network interfaces that have solved these receive-data-buffer access problems for all the corner cases on a single node may still have problems for architectures that use broadcast messages. It is possible that the receive buffer mutual exclusion mechanism on each of the receivers works correctly (the host never receives messages from the data network that have inconsistent contents due to buffer timing and access issues), but could cause the atomic property of the broadcast to be lost. That is, timing differences among the receivers may cause different receivers to see their buffers in different states, even if they all receive exactly the

same sequence of messages from the data network. In the cases where atomic broadcast must be supported, the data network may also be required to support receive-data-buffer consistency.

Similarly, network errors that can trigger host software exception loops need also be considered. This is to ensure that such exception loops do not interfere with the time budgets and partitioning mechanisms of host software.

6.1.3 Buffer Management Performance Considerations.

The performance considerations of buffer management should be considered when selecting a network. In the past, the low-speed aviation digital electronics networks (such as ARINC 429 and 629) have put less emphasis on the performance of buffer management, because memory device access times or memory bus access times was often an order of magnitude quicker than required for serving the data copying and coordination activities. With the advent of high-speed communication in avionic systems, the need for a balance on the buffer management side with respect to performance becomes more prevalent. Performance evaluations should consider the required access needed from both the network as well as the host sides, memory device and memory bus access times, and special support provided by the hardware, such as burst memory access. Interactions between performance enhancement schemes, such as burst memory access that reduce buffer access time, and interactions with access time and requirements (e.g., blocking of the memory or memory bus, may have implications to arbitration of the memory or the memory bus) should be considered in the evaluation.

For network technologies that require software functions to assist the network data flow (for example, data unpacking, data copying, etc.), the software impact of changing the network tables also needs to be considered and suitably bounded. This is especially true of network tables that are configured independently and loaded independently of software application images. Ideally, software execution margins can be suitably bounded and argued to meet the worst-case network data flow assumptions that may be run-time configured. Network technologies that support the bounding of such interactions are preferable.

6.2 SUPPORT FOR APPLICATION-LAYER REDUNDANCY.

6.2.1 Support for Active Replication.

Networks may signal the application of reception status, which may assist the application in voting or selecting a correct value. Such mechanisms should be evaluated with respect to their correctness. In case the indication status stems from the same source as the possibly faulty value, the use of such status information might be limited.

Application-layer membership is a mechanism to manage the redundancy sets at the ISO OSI application layer. Such application-layer membership algorithms should be evaluated with the same scrutiny as the node-level memberships described in section 5.7. One example that can be regarded as application-layer membership information is the network management vector in FlexRay.

Node-level and application-layer membership is often combined within some networks to incorporate message agreement and redundancy mechanisms. Such services provide a foundation on which to build active replication strategies for applications. For example, the NASA ROBUS protocol used in the SPIDER architecture presents voted message data to the network interfaces, containing the visibility and impact of erroneous data to within the network infrastructure. Other networks, for example, TTP/C, implement enforced message agreement strategies where nodes not in agreement with the majority of network nodes are forced to re-integrate. While the membership mechanisms of these two networks can be equally effective in providing a consistent view of system-wide membership, there is a difference in the amount of system resources that are adversely affected while these mechanisms sort out errors. For example, a Byzantine error in a SPIDER architecture is effectively masked with the network layer. In TTP/C, depending on the degree of Byzantine fault containment provided by the guardian, the same error may force multiple nodes to reintegrate. The side effects of these policies and their effect on applications should therefore be understood as the network technology is evaluated.

In some networks, the network host interface incorporates a life-sign mechanism to support application-membership and health diagnosis. A life-sign mechanism requires an application to perform a specific action, which is used to judge an application as correct. Based on incorrectness of the action, an application may be removed from membership. The life-sign action should be evaluated with respect to its effectiveness of detection of failures. Due to the minimal action in normal operation, the error detection coverage may be limited.

6.2.2 Support for Passive Replication.

Some networks support mechanisms for passive-redundancy strategies, i.e., the ability of multiple network nodes to share network bandwidth. These mechanisms are discussed in section 7.7. The networks' mechanisms to inform clients of the state of the passive-redundancy scheme, i.e., what application is in control and how many "spares applications" are online, should also be considered, since such information may aid the detection of latent spares exhaustion. Services to synchronize the state of spares should also be evaluated to ensure that such mechanisms do not introduce potential fault-propagation paths.

6.2.3 Support for Increased Integrity.

Some network technologies implement host interface support for self-checking pair host configurations. Self-checking pairs provide very high coverage error detection. (Multiple pairs must be used for fault tolerance.) Self-checking pair data is compared, and if it agrees, it is delivered as correct. Self-checking, pairs-based input data should be compared before computation; otherwise, the self-checking pair computation results are likely to diverge even though both halves of a pair are correct. Self-checking pairs should also be evaluated with respect to their independence from power, common memory, vulnerability common design faults, etc.

Self-checking host support is strongly influenced by the network level self-checking mechanisms discussed in section 7.4.

6.2.4 Support for Robust Partitioning.

While robust partitioning (see section 1.4.3.2) is a characteristic of an architecture, which is largely outside the control of data networking, there are certain facets of robust partitioning that may need to be supported from the network. The network first should protect itself against misbehavior of any of its clients (see Criterion 16). Then, if required, the network should protect each of its clients from its other clients. This includes protecting robustly partitioned subdivisions of each of its clients from other clients or partitioned subdivisions.

6.3 TIME SERVICE FOR TIME STAMPING AND TIME INTERRUPTS.

Application time services that may be supplied by the data network include time stamping and time interrupt. Synchronization aspects of time have been discussed in section 5.6, including a discussion of the implications of time services to the applications.

The quality of time services can be adversely affected by a data network time-service design that is not robust. Time stamping of data allows an application to determine data freshness. Sometimes all that it is needed is ordinal freshness; that is, the application only needs to know which data set is newer. In some instances, an application may use timestamps to determine the interval between two data samples, which could effect calculations that use delta time. Some applications may use data network supplied time interrupts as a replacement for a local real-time clock to do task scheduling. This has the benefits of a time source that is independent of the effects of possibly faulty software and allows for the synchronization of task scheduling among multiple processors. If these services from the data network are faulty, either from network internal faults or by propagating faults from clients, a time-stamp service could cause wrong time values to be used as inputs to calculations or, when coupled to a host's tasking clock, could cause tasks to not have enough time to execute.

7. FAULT TOLERANCE MECHANISMS.

Some network technologies incorporate fault tolerance mechanisms to mitigate the failure of network components, such as guardians, monitoring schemes, etc. Such mechanisms may be particularly advantageous in aviation digital electronics environments where high-network availability and integrity is required. These mechanisms and associated evaluation criteria are discussion the following sections.

7.1 TOPOLOGICAL FAULT TOLERANCE.

The network topology may have significant impact on the network tolerance to zonal or spatial proximity faults, for example, physical damage that affects a certain area of the vehicle.

If the network uses a bus topology, then any failure along the bus path may destroy network availability. Similarly, faults in network termination may lead to loss of availability and may also introduce other Byzantine vulnerabilities discussed in section 7.8. The bus zonal vulnerability is particularly important if multiple redundant buses are assumed to increase network availability. If all units are connected to all buses, then all buses are required to be in physical proximity at the point of their interface to the different nodes. A failure at this point of

interface may therefore damage all of the independent bus channels. Similarly, a chronic failure of a node, e.g., fire, may also damage all buses that are close to the node. Therefore, when evaluating the suitability of bus-related network technology, care should be taken to ensure that the technology or network architecture has suitably mitigated such zonal vulnerabilities, either by separating bus and or by isolating network interfaces. The secondary effects of incorporating isolation schemes should also be considered in relation to their impact on the physical-layer performance and the potential to Byzantine failure, as discussed in section 7.8.

Networks using intermediate stages may perform better in relation to zonal fault tolerance as the point-to-point relaying action of such technologies alleviates the impact of physical-layer damage. However, the placement and data path planning of such intermediate-stage schemes should also be considered as the network technology is mapped to a vehicle architecture, i.e., there is little benefit in placing two redundant central intermediate stages in the same location.

7.2 GUARDIAN SCHEMES.

Some network technologies incorporate covering functions or guardian mechanisms to contain node faults. It may be argued that such guardians increase network availability. However, the implementation and performance of the guardian function needs to be carefully evaluated to verify that suitable coverage and independence is provided.

There are several variants of guardian implementations; they may be locally (i.e., one node) implemented on-chip or placed with independent guardian chips. Alternatively, the guardian action may be supplied by network intermediate stages, for example, in centralized guardians or peer-based ring schemes. The first attribute that needs to be considered in relation to the guardian action is the amount of coverage that the guardian provides, i.e., what failure modes of the node does the guardian contain? Often due to the cost optimizations, the coverage of the guardian may be focused to cover only a subset of a node's failures. For example, in low-cost TDMA networks (e.g., FlexRay and TTP/C), local guardian schemes are often limited to time window enforcement. The extent of the protection provided is also limited to specific network modes; for example, network start-up is often left uncovered. Time window enforcement does not protect against logical protocol errors, for example, erroneous protocol signaling. Such faults must therefore be mitigated with additional guardian behavior or fault-tolerant protocol logic described in section 7.3.

Irrespective of the coverage provided by the guardian scheme, the independence of the guardian enforcement is another attribute that requires careful consideration. Often in network technology targeted for low-cost domains, the guardian function may be implemented on the same die [silicon integrated circuit (IC)] as the communications controller. The justification of independence may therefore be more difficult, as such schemes may be vulnerable to common mode failures that disable and degrade the guardian actions. For example, the use of independent clocks and partitioned dies may assist, although detailed analysis of failure modes will be needed to support independent failure claims. Another common dependence may be the power source. Network technology with truly independent physical guardian action will require less analysis and therefore may be preferred as it presents less certification risk.

In addition to the physical independence, logical guardian dependencies should also be considered. For example, if the guardian is dependent on its host controller for global time or protocol state synchronization, the coverage of the guardian may be compromised. For example, consider a TDMA time enforcement guardian that relies on its host for schedule synchronization. If the host is "deaf," i.e., simply unable to hear network traffic, it may continuously try to start. If it performs in accordance with the correct start-up activity, then—from the guardian's perspective—the faulty host may appear to operate correctly. In reality, it will be continuously disturbing protocol traffic. Such dependencies should be considered when network and guardian technology is evaluated.

To mitigate the shortcomings of simple local guardian schemes, several network technologies have evolved to incorporate intelligent central guardian schemes. The degree of intelligence in the central guardian is dependent on the network technology; varying from simple time enforcement and slightly-off-specification fault containment, to full protocol-level-policing functions, e.g., protocol semantic-state enforcement or similar message policing. Centralizing these protection mechanisms allows for more intelligent guardians to be implemented at lower costs. However, these implementations of the guardian schemes should also be evaluated to ensure that they provide adequate levels of independence and fault coverage. Protocol and node failures not covered by the guardian will need to be addressed by other means, either by fault-tolerant protocol logic (discussed in section 7.3), or additional fault detection implemented on the client nodes, such as self checking, as described in section 7.4.

The use of intermediate-stage guardians introduces additional constraints on the target system. Consider, for example, a dual-star (central guardian) network configuration. If the implementation of the central guardians lacks sufficient fault-detection coverage, then it is difficult to bound the failure modes of the guardians. The influence of a faulty guardian on protocol action must be established. For example, is it possible for the guardian to cause nonrecoverable protocol flow errors in the establishment of disjoint TMDA cliques, if the other (good) guardian is not available? If this is the case, then a system-level power-sequence may be required to ensure at least one "good" guardian before the end nodes commence communication. In addition, the vulnerability of the guardian implementation to transient errors (SEUs, etc.) will need to be bounded, as such events may take the "good" guardian off line long enough for a faulty guardian to force irrecoverable error scenarios.

As discussed in section 5.3.2, the implementation of central- and intermediate-stage guardian integration and start-up logic schemes should also be evaluated to ensure it is suitably fault tolerant to erroneous end-node faults. If guardians from different network availability channels share signals or protocol state information, then the vulnerability of such mechanisms to failures of the other channel guardian failure should also be evaluated. Similarly, the self test and scrubbing of intelligent guardian actions may be challenging (as discussed in section 7.8).

Irrespective of any guardian implementation, it is imperative that suitable tolerances for guardian enforcement action are established to provide suitable design margin. As with other critical protocol parameters, these tolerances should accommodate for worst-case aging and expected life-time degradations of all components related to the guardian. The criteria for establishing

suitable guardian parameterization would ideally be formalized and verified, as discussed in section 8.

Latent failure of guardian schemes is another consideration; this is discussed in section 7.6.

7.3 PROTOCOL LOGIC FAULT TOLERANCE.

Networking technology may also incorporate protocol-flow and algorithmic fault tolerance strategies, i.e., voting on protocol-state information or required protocol actions. Such voting may effectively contain protocol-state faults propagating from an erroneous node or other network device. The fault-tolerant, global, clock-synchronization action discussed in section 5.6 is an example of such action. Similar strategies may be applied to other protocol actions, such as start-up, reintegration, and mode change. The strength of such protocol mechanisms should be evaluated in the context of the coverage provided by the network implementation. For example, if all nodes are self-checking, then little protocol-state fault tolerance is required, as all protocol errors are contained at the source and justified to be benign. Similarly, if the guardian mechanisms contain protocol-flow errors, then less protocol-state fault tolerance is required. However, if suitable fault containment or coverage cannot be established, the protocol level, vulnerabilities to erroneous state and addressing information should be evaluated.

If fault-tolerant protocol logic is implemented, the impact of the protocol algorithms will also need to be evaluated. This means that any protocol-level mechanism needs to ensure the required agreement on protocol state for integrity and the required replication for availability. Often, in two-channel systems, there is a conflicting goal between availability and integrity. Hence, mechanisms to improve protocol integrity may reduce protocol availability; for example, logic to contain errors during start-up may render the protocol unable to start.

7.4 LOCAL TRANSMISSION MONITORING AND SELF-CHECKING SCHEMES.

Network technologies may also implement monitoring or self-checking services to improve fault detection and fault tolerance. As with the guardian action, the effectiveness of such schemes depends on the amount of independence and coverage that can be claimed by the implementation. For example, CAN incorporates an error-checking mechanism that will switch the network to a passive state if the transmissions of the controller are not suitably acknowledged. Since this is implemented within the same IC as the communications component, the action may be degraded by common-mode failures. In addition, such schemes may introduce potential fault-propagation vulnerabilities, as it is possible for a node to transition to the passive state in response to the erroneous negative acknowledgements generated from a faulty node. Such vulnerabilities should be analyzed as the network is evaluated.

Other networks may employ local wrap-back schemes where a node monitors its own transmission via local receivers. Such schemes should be analyzed for vulnerability to Byzantine faults, as a local-monitoring circuit may perceive the local wrapped-back signals as good, but receivers at the end of a loaded transmission line may see a degraded and erroneous signal. Hence, the wrap-back signal state may not be representative of the network observed state. Byzantine faults and fault tolerance strategies are discussed in more detail in section 7.8.

Some networks and protocols implement support for self-checking configurations, allowing multiple-network interface circuits to be tightly synchronized and to crosscheck each other. An example of such a network is ARINC 659. When evaluating the coverage provided by such schemes, care should be taken to examine where the cross-checking and error-containment voting is performed. In ARINC 659, checking and voting is performed at each receiver, hence full coverage of the entire transmission path is assured. Local checking in ARINC 659 is performed solely to increase network availability, with each network IC enabling and monitoring the transmissions of its other half. As with guardian functions, such cross-enabling schemes, should be analyzed to ensure that there is sufficient margin for the enabling and disabling action to ensure transmissions are not truncated to produce potentially Byzantine signals. Similarly, self-checking errors that rely on loop-back monitors may be vulnerable to Byzantine faults, as discussed above.

7.5 RECONFIGURATION AND DEGRADED OPERATION.

Network technologies may also incorporate mechanisms to implement reconfiguration or continued operation in a degraded mode. For example, some physical layers may incorporate a degraded mode of operation that allows communication to continue even if one half of a differential communications channel is faulted. If such degraded modes are to be leveraged, then the performance (e.g., bit-error rate) of the degraded operation needs to be evaluated to ensure that adequate performance is maintained. The protocol mechanism for the detection and announcement of such degraded operation should also be evaluated to verify that timely and correct diagnosis is provided.

Other protocols, such as IEEE 1394, may reroute the network path to mitigate physical or node paths. If such protocol action is to be leveraged by a system, then mechanisms used to implement such actions will need to be evaluated to ensure that the reconfiguration time is suitably bounded. As discussed in section 5.5, the issues surrounding the erroneous invocation of such logic must also be considered. The recovery mechanisms for such logic should also be investigated to ensure nodes are not permanently isolated in response to local transient errors.

7.6 LATENT FAILURE DETECTION.

Fault detection, isolation, and recovery functions used within aviation digital electronics systems are often required to be periodically tested to ensure that the detection and recovery actions remain active. Such covering functions are usually transparent to normal mode operations; hence, without a test, it is possible that such functions may fail passively and the protection will be lost. Network fault detection and covering functions are no different; therefore, network mechanisms to assist the latent fault-detection should be considered as the network technology is evaluated. To illustrate common network vulnerabilities to latent failure, consider the following scenario with a short circuit of intermediate-stage guardian function. If the network traffic can propagate through the shorted guardian without error, then the passive state of the guardian enforcement action may pass unnoticed, leaving the system vulnerable to a second uncontained failure of another network component. Similarly, consider a network component with a “stuck at good” CRC calculation circuit, i.e., all data received results in a “good” CRC unless such functions are tested. It will be difficult to detect such a state in normal protocol operation since all CRCs are nominally good.

Network mechanisms that incorporate modes and mechanisms to assist the latent fault detection of network components may be preferred. However, such mechanisms should be analyzed to ensure that they do not introduce failure vulnerabilities, as testing for latent failure may disrupt nominal network performance. Interlocks and protection mechanisms to ensure that such testing occurs only in safe system states should also be evaluated. The coverage of the network test procedures should also be evaluated to verify that all key network mechanisms are suitably verified. For complex error detection and enforcement schemes (for example, protocol semantic-correctness enforcement), the ability to achieve adequate coverage via the self-test mechanism may be challenging, since such coverage will require all decisions causal to the enforcement actions to be suitably exercised

7.7 VOTING, SELECTION, OR AGREEMENT SERVICES AND REDUNDANCY MANAGEMENT.

Networks may also incorporate redundancy management and voting mechanisms to simplify application-level fault tolerance. The self-checking configuration in section 7.4 is an example of such a scheme where increased network component redundancy is leveraged to achieve increased network integrity and availability. In self-checking configuration, a pair works and sends out messages in coordination (at the same point in time). Depending on the required availability targets, self-checking may need two or more self-checking pairs.

Another form of network redundancy is active replication in a triple modular redundancy (TMR) voting scheme. In contrast to self-checking configurations where messages are sent at the same point in time for a pair, nodes always send out the data at different points in time in a TMR scheme. Thus, TMR implements a type of temporal redundancy. In TMR schemes, end nodes need to correlate messages sent at different times before being able to vote, while in self-checking pair configurations, nodes can take the first valid message with integrity (messages that agree and stem from two halves of pair).

Network selection should consider which active replication scheme fits its needs best. Self-checking pair schemes may require special hardware support for the synchronized sending of messages but simplify voting schemes to become “pick first valid” message. On the other side, TMR-based systems may not require additional hardware, but they may require message management (storing) for the messages received at different times from different hosts before voting, as well as a voting function implemented at each end node.

Dual replication can either be targeted at ensuring availability or integrity. If replication targets integrity, the end node would perform a comparison of two values. If they agree, the integrity of the data is ensured; if they do not agree, this is a signal to the application, and integrity is not lost. Yet, the availability achieved is similar to the availability of either component (and of course the availability of the communication path). Such voting algorithms supplied by the network should be compared with the assumption of the application to avoid unsafe operation; e.g., in ARINC 664, the redundancy management layer chooses the first syntactically correct frame and is targeting availability assuming that any failure on the communication path are detected by inline integrity mechanisms (like CRCs). The first syntactically correct frame is, thus, of the integrity of the communication source. Any fault defeating the integrity mechanism leading to an undetected error in a dual-replicated, select-first-valid scheme may impact integrity

of data. Masquerading faults are faults where a faulty node mimics another node. Masquerading faults can defeat any redundancy management, because multiple inputs to any voting or selection functions may stem from the same fault zone. Network implementations and mechanism should be analyzed with respect to masquerading-fault vulnerabilities.

The network technology may also support mechanisms to implement passive replication strategies, for example the capability of redundant or replicated nodes to share the same network transmission slot. The replicated or redundant nodes take over when the first replica ceases control. In such active and Shadow schemes, consideration should be given to the time it takes to detect the failure of the components; e.g., in case of a failure during sending of a message monitor by another component, the coverage scheme may only detect the failure after it has already been (partially) sent. Thus, the receiving node may have to wait for the next message that can be sent. In addition, the network Master-Shadow mechanism should be evaluated for its ability to hand over control in a fault scenario. The effectiveness of release of control very much depends on guardians or coverage schemes deployed.

7.8 BYZANTINE FAULT TOLERANCE.

The Byzantine failure scenario, or Byzantine Generals Problem (BGP), was first presented approximately 20 years ago. Since its introduction, it has been the subject of a great many papers and scrutiny by the fault tolerance community. Numerous Byzantine-tolerant algorithms and architectures have been presented in the subsequent two decades. With the ever-increasing dependency on electronic hardware and software to perform safety-critical control functions, and the emerging trend to implement control with distributed multiprocessor systems where consensus may be a prerequisite, the practical issues relating to Byzantine behavior need to be understood. For these types of systems, existence of Byzantine fault tolerance is a litmus test for dependable systems design. However, the wide-scale industrial acceptance of the problem has yet to find maturity. Only recently has slightly-off-specification faults, a subset of the Byzantine fault class, received some widespread attention. Today, there are still many misconceptions relating to Byzantine failure, both with respect to what makes a system vulnerable, and indeed, the very nature and reality of Byzantine faults. This section describes the Byzantine problem, from a practitioner's perspective. It is the intention to provide a working appreciation of the Byzantine failure from a practical as well as a theoretical perspective. This section provides a discussion of typical circuit-centric failures and the difficulties in preventing the associated failure propagation with illustrations of real-world Byzantine failure observations.

A Byzantine fault is any fault that produces different symptoms for different observers. This can happen at any point where a signal splits, i.e., one source goes to more than one destination. Byzantine faults are a lot like metastability in that there is no way to prevent them; one can only treat the symptoms such that the faults do not become system failures.

Byzantine faults can happen in the amplitude domain. For example, assume that a digital driver gets stuck at 1 and 2. Because of manufacturing tolerances, other digital circuits using this value may assume it is a 0 or a 1. The most common fault of all (an open) into a complementary metal-oxide semiconductor (CMOS) input looks like a 1 and 2. Byzantine faults can also happen in the time domain. For example, in a synchronous redundant system, no matter how tightly you synchronize the redundant channels, there will always be some (infinitesimally small) time skew

between the channels. An input that goes to multiple channels can arrive at a clock tick and within the skew such that some channels will see the input arriving before the clock tick and some see it arriving after the clock tick. If the redundant channels vote on the input's value at the clock tick, some will use the old value and some will use the new value. Note that the voters will say some channels are faulty even though no hardware fault occurred. This is a design Byzantine fault.

A BGP is a system failure caused by a Byzantine fault. If the multiple observers do not require any mutual coordination, a BGP cannot occur. But, if the observers have to coordinate in some way or if their actions are compared (by voting or some other means) for fault tolerance, then a BGP is possible.

Byzantine fault propagation escapes most classical fault containment techniques. Solutions to the BGP are well known, but require a large amount of communication bandwidth. It has been proven that to tolerate F Byzantine faults, 3F+1 fault containment zones are needed. From this, one can deduce the surprising result that a simple triple-channel system cannot tolerate even one Byzantine fault, no matter how cleverly it is designed. The next surprising result is that to be fully tolerant to two faults, seven fault containment zones are needed.

To further illustrate the Byzantine propagation capability, one can envision a "Schrödinger's CRC," similar to the Copenhagen misinterpretation of "Schrödinger's Cat," where the CRC is simultaneously correct for any interpretation of Byzantine data. The behavior of a 1/2 bit on a CCITT-8 CRC circuit is shown in figure 3. This figure shows eight data bits followed by the eight CCITT-8 CRC bits, with one data bit to be transmitted stuck at 1/2. Because the transmitter's CRC is a linear exclusive or (XOR) combination of its data bits, each CRC bit affect by the 1/2 data bit can also be 1/2. The switching threshold voltages are shown for two receivers (*a* and *b*). These thresholds fall within the legal range of V_{IL} to V_{IH}. The resulting data received by *a* and *b* are different, but each copy has a correct CRC for its data. Thus, CRCs can provide no guarantee of protection against Byzantine fault propagation.

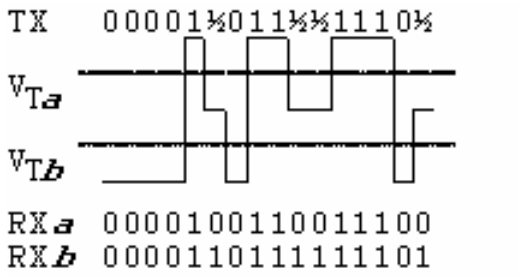

Figure 3. Schrödinger's CRC

Interactive consistency of messages is a service provided by a data network to ensure consistent message reception in the presence of Byzantine failures. Byzantine failures manifest as different nodes having a different view of the messages communicated (either no message at all or even different values).

Any voting mechanism at end nodes may have a different input set (e.g., due to one Byzantine fault, resulting in one different value or due to a node-local transient fault resulting in possible faulty but normally detected value). Any voting scheme (e.g., TMR or any selection logic) may select or vote and result in a different value (but possible a value stemming from a correct node). Such effects may need to be considered by applications for evaluation.

Even if source coverage (e.g., self-checking sources) is used, any voting or selection scheme may still result in different selection and voting outcome. Similarly, Master and Shadow implementations at a physical level may send out a frame that is corrupted either by a faulty node or transient faults. Switchover from Master to Shadow may be problematic in arbitrary fault scenarios, because some nodes may correctly receive the Master's frame and some may not. If the Shadow node does correctly receive the Master's frame, it may never be able to take over control. Such scenarios should be evaluated.

Coding techniques (such as the use of CRCs or cryptographic signatures) are sometimes proposed as a solution to the Byzantine problem. Such coding techniques do not allow justification of coverage as they are based on unsupportable assumptions.

Networks requiring Byzantine tolerance need to mitigate Byzantine failures by incorporating Byzantine filtering actions. The Byzantine filter transforms Byzantine input signals to consistently erroneous or correct signals. In such systems, the coverage of the Byzantine filter action is a critical network parameter. In addition, networks are required to incorporate classical Byzantine agreement protocols. Discussions of actual Byzantine failures and methods for coping with them can be found in reference 21 or 22. The theoretical basis for Byzantine agreement protocols can be found in reference 23.

8. DESIGN ASSURANCE.

8.1 DESIGN ASSURANCE PROCESSES.

Often, network technology forms the backbone of the system architecture. The design correctness of the network implementation is therefore of utmost importance, as the network provides a significant common mode failure vulnerability. With the increasing complexity of network technology, the design-correctness problem is increasing with every generation of silicon. Although multiple independent lanes of redundancy may be suitable to mitigate random component failure, if common network technology is used across all lanes, then the system is vulnerable to generic design defects of the network technology implementation. Dissimilar network redundancy schemes may be deployed to mitigate such issues. Because of the increased cost and unquantifiable coverage of many dissimilarity schemes, it is preferable if the network technology is designed with best-practice design procedures. Within the aviation digital electronics domain, this would correspond to DO-178B for software-related network components and DO-254 for hardware components. Network technologies that have such formal design assurance artifacts will pose less certification risk than other technologies and may be preferred for that reason. Technologies without formal design assurance processes will need to be considered on a case-by-case basis. The complexity and degree of commercial usage base of the networking technology will then need to be considered. The COTS provisions within DO-254 were designed to handle such hardware technologies that have been used in many systems that

have accumulated a huge service experience bases. While usage experience may be leveraged to assist the design assurance case, certification credit will require substantiation of the service experience data. This treatment is commonly applied to microprocessors. The applicability of such techniques to networking-related hardware will need to be considered as the network is evaluated.

8.2 AVAILABILITY OF STANDARDS AND CONFORMANCE EVIDENCE.

8.2.1 Open Specification and Standardization.

The use of open specifications and standardization might assist a certification authority in establishing the acceptability of a network. Irrespective of the formality of the design artifacts, the quality of the network technology specification is a key attribute of the network technology. It is preferable if the technology is open with a standardized and published specification, as this will enable the protocol mechanisms to be analyzed and discussed within the academic and industrial community, including the application for formal verification studies. The standardization process itself is beneficial, as the committee activity usually associated with the open standardization process may also lead to an open detailed examination of the network behaviors. However, care is required for network technology not designed specifically for use in a safety-relevant environment; the completeness of the specification will need to be carefully reviewed. Often, such standards may specify the normal mode of operation only; the protocol actions-to-erroneous behavior and the associated degraded modes of operation may not be sufficiently treated in the standard document. The evaluation of the network specification should include such completeness analysis.

Another area where specification completeness may be lacking for COTS protocols is in the area of implementation choices that have been made below the protocol specification. COTS solutions may not be not fully described because of the need to maintain competitive advantages between vendors, etc. Hence, many key implementation choices may not be visible, and this may impact assurance process where detailed understanding and analysis of the interactions of all technology layers is required. The availability of suitable design information should be considered as the network technology is evaluated.

8.2.2 Conformance and Interoperability Testing.

As with the specification, the availability of standard conformance test campaigns and specifications may also be advantageous. This is especially important for network technology that is sourced from multiple vendors, since it may assist in identifying interoperability glitches. The issues raised above in relation to specification completeness also arise in relation to the completeness of the conformance test campaigns, i.e., are all operating modes covered, and are exception and error reactions sufficiently traveled?

8.2.3 Protocol Design Correctness.

In addition to completeness, the correctness of the specification is clearly important. The use of formal methods and development of formal proof arguments for protocol algorithms show much promise here, as they can exhaustively verify the algorithmic behavior. However, when

reviewing such formal verifications, the assumptions that underpin the formal proofs need to be fully understood and evaluated against the real-world failure expectations and behavior. Similarly, the composability of the formal verifications needs to be understood to ensure interactions between different protocol algorithms (for example, membership services and clock synchronization). In some protocols, TTP/C for example, interdependencies exist that may need to be evaluated with the formal arguments. That said, formal verification of protocol algorithms can increase design-correctness confidence, and therefore, network technology that has such verification evidence may be more attractive.

Informal verifications (for example, random fault-injections campaigns) may also increase confidence in the network architecture. However, the conclusions that can be drawn from the fault-injection campaign need to be carefully scrutinized with a detailed understanding of the effects of the fault injection in relation to the technology implementation. For example, consider a heavy-ion fault injection on a communications controller with and without parity on its microsequencer memory. Without parity, this technique may provide useful insight into the performance of the system architecture. This was demonstrated during the TTP/C FIT research program that illustrated the architectural vulnerabilities to Byzantine errors (in this instance caused by bit flips in the instruction memory that resulted in slight time deviations of transmission time). However, the same campaign performed on a controller incorporating parity for all onboard RAM locations may not have been so revealing, as the parity detection mechanism may swamp the observations with parity-induced fail stops that cover up the other design weaknesses. Such a study would be less relevant in finding these other architecture and design weaknesses. The issues relating to the design and implementation visibility of COTS technologies are reiterated here, as such visibility may be required to draw any architectural inference from such studies.

8.3 DESIGN MARGIN.

The issues discussed in sections 3 and 4 also require some design assurance to ensure that adequate design and safety margins are established for the selected network technology. Such a design needs to be established and justified to be valid over the whole system lifetime, addressing parasitic and parametric shifts due to temperature effects, etc. As discussed in previous sections, this safety margin evaluation needs to be established in several domains, such as the time and value domains of signals under worst-case design parameters, bus loading, etc.

For physical-layer attributes, this means that influencing factors need to be analyzed with respect to their margin and contribution to the safety margin. Such physical-layer attributes may include an over-sampling margin that should include the transceiver skew over the whole lifetime of the product, assuming worst-case loading, aging of components (e.g., clock stability over time), temperature range of environment, etc.

8.4 CONFIGURATION TABLE CORRECTNESS AND PERFORMANCE JUSTIFICATION.

In addition to the design correctness of the network implementation, the design correctness of network configuration parameters and tables also is required. This is especially important if the table parameterization impacts protocol algorithmic-level behavior, for example, clock synchronization timing and propagation delay parameterization. In such instances, the

parameters may severely impact protocol performance. The incorrect configuration of such parameters may, therefore, invalidate any formal proofs of algorithmic correctness. Similarly, tools may be used to establish parameters for network policing policies, for example, message transmission rate limiting and maximum message jitter. In such cases, the correctness of these parameters may severely impact network performance assumptions. Therefore, when evaluating a network for suitability, consideration should be given to the rigor applied to ensuring correct network configuration parameters. Ideally, all parameters critical to network operation will have explicit formal requirements and invariants that are traceable to network functional behavior, assumptions, and requirements. Such traceability may assist the completeness checking of the guidance presented. Ideally, the guidance supplied will be suitably assured for correctness and completeness.

The network technology may also provide tooling to assist network configuration and its associated verification. Such tools are often required to handle the size and complexity of modern networking technologies and to assist with the generation of nonhuman readable binary configuration tables. If tooling is used for configuration data generation or verification, then the development pedigree of the tooling may also need to be examined as the network technology suitability is evaluated. If the tooling is inline, i.e., the tooling generates protocol configuration parameters that are not verified by subsequent process checks, then the generation tooling should be qualified in accordance with the DO-178B guidelines for development tools. Alternatively, if the tooling is simply used to verify the network configuration parameters, then they are less stringent, and DO-178B verification tool guidance should be adopted. The data flow path of inline generation and verification tooling need to be evaluated to ensure that adequate independence exists within the tool chain to prevent a common tooling failure. Ideally, the configuration inspection tools will be driven from reviewed, network-related, functional data-flow requirements and the formal network parameter constraints and invariants, as described above.

For some modern asynchronous networks, e.g., ARINC 664, the size and scale of the configuration problem is very large, and end-to-end performance (e.g., data-flow latency and jitter) is difficult to analyze and bound. The sheer complexity of the network level interactions between end nodes behaviors, switch implementations and the chosen network policing policies (e.g., message rate limiting) may greatly complicate network performance justification. However, procedures or tooling to analytically bound and justify the worst-case behavior of such networks is required to meet certification requirements. Therefore, the capability and maturity of available analysis tooling should be given careful consideration as such networks are evaluated. Similarly, network technologies that incorporate complicated MAC interactions, such as those described in section 4.1.1, may also complicate end-to-end performance calculations. Such interactions and any associated network logic (e.g., retry logic and queuing mechanisms) also need to be considered by performance calculations and associated tooling. It is possible that erroneous node behavior can drain network performance until the network has diagnosed the problem and has contained it. Networks that bound the magnitude and duration of these adverse performance influences may be preferable, as they may greatly simplify performance justification calculations.

For highly integrated, multivendor systems, there are network technologies that incorporate tooling to assist the incremental change of the network tables, allowing new functions and their associated data paths to be added to the network with minimal impact on previously analyzed functions may also be attractive, since such tooling may ease incremental certification effort.

8.5 NETWORK MONITORING AND TEST EQUIPMENT.

With the complexity of modern network technology, the ability to monitor and observe network behavior is very important to support design validation. Similarly, the ability to insert faults into the different network layers may be required to test the network redundancy management mechanisms or the fault response behavior of applications operating on top of the network infrastructure. Therefore, the availability and capability of the test equipment that exists for the network technology may also be a very important consideration. In the ideal situation, such test equipment will be able to observe all behavior of all network nodes, including network start-up and recovery actions. The portability of the test equipment should also be considered, as such equipment is often required to support flight testing.

The no-interference guarantees of the test equipment may also need to be evaluated if it is to be deployed in a flight test scenario. The ability to monitor the entire network behavior from limited test-inspection access points should also be considered. In some modern switched technologies, such access is more difficult than simpler buses. Hence, work in some newer switched technologies is being performed to develop the network-wide controllability and observability needed to test the maintenance of these new or more complicated networks, while at the same time, trying to minimize the invasiveness and logistics complexity of connecting test equipment to these networks.

9. SECURITY.

Historically, data communication security has not been an important issue in commercial aviation digital electronics. This began to change with the growing awareness and sensitivity relating to cyber security in the 1990s. Subsequent terrorist activities accelerated this awareness and sensitivity trend. At the same time, some developments in aviation digital electronics design have made aviation digital electronics systems more vulnerable. Higher levels of integration and more internetworking connectivity have increased the chances for entrance paths into critical aviation digital electronics functions. Possible gateways onto these paths include, but are not limited to radio frequency (RF) in the airplane (e.g., portable maintenance access terminals, cabin crew tablets), off-aircraft radio, external gatelink and maintenance ports (optical, RF), and passenger networks. The increasing use of COTS protocols and networking technologies (with their known weaknesses) has the potential of attracting attackers who are familiar with these weaknesses. With ever-increasing bandwidth of modern network technologies, there are also increasing pressures to fully utilize spare network capacity. Hence, the mixing of critical and noncritical end-systems and associated data on common network infrastructure is another increasing trend.

A network should protect itself against security threats (e.g., denial of service attacks) and should not allow itself to be used as a means for supporting attacks against its clients. The ability of the network to suitably secure and authenticate private transmissions between different clients on the

network should also be evaluated, if such usage scenarios are also anticipated. Network firewall schemes should also be evaluated, especially if critical and open systems share the same network infrastructure. While it may be only "security through obscurity," aviation digital electronics systems really are more secure when COTS is not used.

10. DISCUSSION.

During Phase 1 of the Evaluation Criteria for Databuses' task, draft evaluation criteria were developed and documented. A preliminary list of questions to be used as an examination of conscience was created as an aid for system designers when evaluating the use of a data network within aviation digital electronics architecture. Many issues and criteria in this list are overlooked or are underappreciated by many of today's aviation digital electronics designers. In Phase 2, the draft evaluation criteria were re-examined for completeness, applicability, and orthogonality. This involved updating the literature search. The criteria were vetted and validated by using them against a representative span of seven data networks. The criteria were simplified by compressing them into a fewer number of criteria with supporting questions. Then, the Data Network Evaluation Criteria Handbook [2] was created, which incorporates the updated criteria.

The Handbook provides a good companion to AC 20-156. The latter provides a means for manufacturers and designers to gain FAA approval of an aviation data network by showing that the data network, as designed, will perform its intended function and satisfies the applicable airworthiness requirements when installed on an aircraft or aircraft engine. It informs manufacturers and designers on what they must do, but does not include detailed explanation of how. Nor does it provide any warnings about pitfalls that may be encountered by designers who are not data network experts. For example, AC 20-156, section 4, subsection a, states that manufacturers and designers must evaluate "The maximum error rate per byte expected for data transmission." In all likelihood, a designer without a great deal of experience in data network design would use the bit-error rate as provided in the documentation supplied by the data network technology provider and do the simple mathematical calculation to convert bit-error rate to byte-error rate. However, the Handbook's Criterion 3 Probability of Bit Errors, its supporting questions, and the accompanying description of physical-layer error sources provide warnings about issues that may cause this calculation to not accurately represent the true error rate that would be experienced by the network in service. There is a mismatch in this companionship; the document and criteria structure is different between them. The Handbook's structure was designed to follow that of the typical protocol stack rather than to match AC 20-156. It remains to be seen if the structure of either of these documents and their criteria is easier to follow than the other by most readers and if one document should be changed to match the other. Merging these documents or having one subsume the other is not recommended; they each have their purpose. AC 20-156 was designed to provide assurance goals and issues. This Handbook was designed to provide a detailed technical framework for aid in providing technical data to support the assurance goals and addressing the issues published in AC 20-156. However, the Handbook was not intended to provide an acceptable means of compliance.

11. RECOMMENDATIONS.

While the Handbook and the existing AC 20-156 provide a good beginning, data networks have become so complex and important to the correct operation of avionics systems that an industry committee collaborative effort, like the one that was undertaken to create DO-178 and DO-254, should be started, and the output would be an acceptable means of compliance. The fact that the criteria and their supporting questions in the Handbook are a simplification of the original set of questions that were created, indicates the possible need for a document more substantial than that of the size and scope of a typical handbook. In addition, the high degree of interaction between a data network and the architecture it supports (which was expected but was greatly reinforced by the effort under this task) means that an evaluation of a data network cannot be done independently of the evaluation of its role in an architecture. Thus, an evaluation of a data network in its context can be seen also as an evaluation of the architecture. Given that an avionics system's architecture is often seen as a competitive differentiator in the avionics marketplace, an industry consensus document may be preferable to a handbook created by a small subset of the industry. Such a committee would create a document with a relation to AC 20-156 that is similar to the way in which DO-178 and DO-254 are used. This committee would not espouse any particular network technologies. Industry committees with charters to create avionics network technology standards would be independent of this activity. In the future, committees creating avionics network technology standards should include means of compliance in their deliberations and documentation.

12. REFERENCES.

1. CAST, *Databus Evaluation Criteria*, Positioning Paper CAST-16, Certification Authorities Software Team, February 2003. www.faa.gov/aircraft/air_cert/design_approvals/air_software/cast/cast_papers/media/cast-16.doc

2. Driscoll, K., Hall, B., Koopman, P., Ray, J., and DeWalt, M., "Data Network Evaluation Criteria Handbook," FAA report DOT/FAA/AR-09/24, June 2009.

3. Hoyme, K. and Driscoll, K., "SAFEbus," *IEEE Aerospace and Electronics Systems Magazine*, Vol. 8 (3), March 1993, pp. 34-39.

4. Aeronautical Radio, Inc. Backplane Data Bus, *ARINC Specification 659*, December 1993.

5. Aeronautical Radio, Inc. *ARINC Specification 629*, 1999.

6. Aeronautical Radio, Inc. *ARINC Specification 429*, (orig. 1977) May 2001.

7. TTTech Computertechnik GmbH, *Time-Triggered Protocol TTP/C, High-level Specification Document*, Document No. D-032-S-10-028, Austria, 2002. http://standards.computer.org/sesc/s2esc_excom/minutes/2003-12/Time%20Triggered%20Protocol%20spec.pdf

8. ISO, *Controller Area Network, ISO 11898-1*.

9. ISO, *ISO/CD11898-4: Road Vehicles—Controller Area Network (CAN)—Part 4: Time-Triggered Communication (TT-CAN)*, 2000, Geneva, Switzerland.

10. Microprocessor and Microcomputer Standards Committee of the IEEE Computer Society, *IEEE Standard 1394b-2002, IEEE Standard for a High-Performance Serial Bus—Amendment 2*, 2002.

11. Microprocessor and Microcomputer Standards Committee of the IEEE Computer Society, *IEEE Standard 1394-1995, Standard for a High Performance Serial Bus—Firewire*, 1995.

12. Mores, R., et al., "FlexRay—The Communication System for Advanced Automotive Control Systems," *Society of Automotive Engineers World Congress*, SAE International, Detroit, Michigan, March 2001.

13. Pellar, M., Berwanger, J., and Griessbach, R., *Byteflight Specifications,* http://www .byteflight.com/presentations/, BMW AG., October 1999.

14. NASA, *The Scalable Processor-Independent Design for Electromagnetic Resilience (SPIDER)*. Papers and talks available at http://shemesh.larc.nasa.gov/fm/spider/

15. SAE International, *PI-Bus*., SAE standard SAE-AS4710, May 1993.

16. Bauer, Robert, "IntelliBus Protocol: Flexible and Cost-Effective Automotive Bus," presented at *Real-Time Automotive Seminar*, 2004. Available at http://techonline.com/electronics_directory/techpaper/193103708

17. Aeronautical Radio, Inc., *Aircraft Data Network ARINC 664*, 2002.

18. Nyquist, H., Certain Topics in Telegraph Transmission Theory," *Transaction AIEE*, Vol. 47, pp. 617-644, April 1928, Reprinted as a classic paper in *Proceeding IEEE*, Vol. 90, No. 2, February 2002.

19. Shannon, C.E., "Communication in the Presence of Noise," *Proceedings of the Institute of Radio Engineers*, Vol. 37, No.1, pp. 10-21, January 1949, reprinted as a classic paper in *Proc. IEEE*, Vol. 86, No. 2, February 1998.

20. United States Government General Services Administration, *Federal Standard 1037C. Telecommunications: Glossary of Telecommunication Terms,* August 7, 1996. http://www.its.bldrdoc.gov/fs-1037/

21. Driscoll, K., Hall, B., Paulitsch, M., Zumsteg, P., and Sivencrona, H., "The Real Byzantine Generals," *Proceedings of the 23rd Digital Avionics System Conference*, October 2004.

22. Driscoll, K., Hall, B., Sivencrona, H., and Zumsteg, P., "Byzantine Fault Tolerance, From Theory to Reality," *SAFECOMP 2003*, September 2003.

23. Lamport, L., Shostak, R., and Pease, M., "The Byzantine Generals Problem," *ACM Transactions on Programming Languages and Systems*, 4, 3: 382–401, 1982.

13. GLOSSARY.

Databus and network technologies can vary considerable in their use of terminology. For this reason, a standard set of definitions for all terminology used with respect to aviation digital electronics networks does not exist. For example, the term "slot" can be used to refer to either a physical address in a cabinet or a transmitting node's temporal position within a table-driven sequence. This glossary is provided to resolve ambiguity and to keep this report self-consistent.

Anisochronous (also Aperiodic): The essential characteristic of a time scale or signal such that the time intervals between consecutive significant instants do not necessarily have the same duration or durations that are integral multiples of the shortest duration.

Asynchronous: The essential characteristic of time scales or signals such that their corresponding significant instants do not necessarily occur at the same average rate. This term often is misused to mean anisochronous.

Babbler: A node that has babbling transmissions.

Babbling: The act of transmitting a signal not in accordance with a network's protocol. Typically, this means transmitting at times not allowed by the protocol.

Backplane: A card that connects one or more cards or modules together.

Bit-dominant signaling: A bit-dominant signaling method has at least two classes of signals having the property of dominance. Signals with this dominance property have a priority such that if two or more signals appear on the media the same time, only the (most) dominant signal is perceived by receivers.

Box (also Cabinet or Rack): A mechanical enclosure that contains one or more cards or modules, which are typically connected together via a backplane.

Bridge: A client that conveys data between two or more networks.

Byzantine failure: The loss of a system service due to a Byzantine fault in systems that require consensus.

Byzantine fault: A fault presenting different symptoms to different observers.

Card: A thin, rectangular supporting member upon which electronic components are mounted. These components may be mounted on one or both sides of the card.

Client: A function that uses one or more services of the network. Note that this is a functional definition; whereas Card, Module, and Box are mechanical definitions; and Node, Device, and Drop are electrical definitions. There may be clients that have need of or are only capable of using a subset of the services provided by the network.

A client may perform any of the roles: Master, Contender, Slave, Peer, Bridge, or Monitor. A client that is capable of performing one of these roles, whether or not it is currently performing such a role, is called role-capable. For example, a client that can be a Master (even if it is not the current Master) is called Master-capable.

Consistency: All good nodes agree on diagnosis.

Correctness: In diagnosis, no good node is falsely determined to be bad.

Criteria (see Evaluation criteria)

Databus (see Data network)

Data network: The communication connection among electronic components. The terms "bus" and "databus" many times are used in a sense that is synonymous with network. However, the strict definition of "bus" is a particular network topology. Other topologies include mesh, ring, and star. The term "network" is preferred to avoid ambiguity. The term "bus" is used in this document only to denote that particular topology. This is to avoid oxymorons like "ring bus." However, "bus" and "databus" are used in this document to mean "network" when referencing other documents that use these terms in the ambiguous sense.

Device (see Node)

Drop: An electrical connection to a network. A box or module may have none, one, or multiple connections to the network.

End node: A node that is the ultimate producer or consumer of a data network's service (e.g., the transmitter or receiver of a message).

Evaluation criteria: A characteristic or feature of a data network that may have an impact on system safety. One cannot definitively say that a particular characteristic or feature would have a safety impact for any particular system because the system's architecture in which the network is used may be insensitive (not need) to the particular characteristic or feature, but other architectures would be a problem. For example, a network with a flawed retry mechanism could work just fine in a network that did not do any retries.

Frame: A term that has two distinct definitions in wide use by the aviation digital electronics communication and other communication fields. In aviation digital electronics, the term usually means one repetition of a repeating sequence of scheduled message times. In other communication fields, the term often is used synonymously with message or packet. Because of the wide use of both of these definitions, selecting one definition over the other would be foreign

to a large number of the intended audience of this report. Compounding this problem is the fact that it is often difficult to distinguish between the two definitions purely by context. Therefore, this report will try to minimize the use of this term and ensure that the correct meaning is obvious whenever it must be used.

Guardian: A device placed in the signal flow of a data network that is used to contain failures.

Head-of-line blocking: The characteristic of a FIFO buffer that causes priority inversion when the head (next item to be output) of a FIFO queue is blocked from being outputted because it has low priority, while items behind it have a high enough priority such that they could have been outputted from the FIFO if it were not for the blocked head item.

Host: Client hardware.

Inline error detection: Any error detection scheme that does not compare or vote among redundant paths.

Intermediate stage: A bridge, guardian, or other device through which data network signals must pass.

Intrinsic safety: A design technique applied to electrical equipment and wiring for hazardous locations. The technique is based on limiting energy, electrical and thermal, to a level below that required to ignite a specific hazardous atmospheric mixture.

Isochronous: The essential characteristic of a time scale or a signal such that the time intervals between consecutive significant instants either have the same duration or durations that are integral multiples of the shortest duration.

Masquerade failure: A failure that causes one node to pretend to be another.

Master: A client that has control of the assets (or a subset of the assets) of a network. Generally, there is, at most, one Master at any time. However, there may be networks that use an "oligarchy," where several Masters jointly and concurrently control a set of assets. In the cases where an oligarchy is used, the term "Master" shall mean every member of the oligarchy that can concurrently affect control of its assets. There are some sophisticated networks that allow a Master to control just a subset of its assets. In this case, there may be multiple Masters as long as the assets they each control are not also under the control of another Master. These asset subsets may be of different services; e.g., there may be a Data Transfer Master and an Interrupt Master, or the assets of a service may be partitionable; e.g., the individual links of media in a mesh topology.

Master and Shadow: A fault-tolerant scheme in which one redundant device (the Master) is in control until it fails. Upon the Master's failure, another redundant device (the Slave) takes over. This scheme may have multiple Slaves in a priority chain such that a Slave takes over whenever all higher-priority redundant devices have failed.

Message: One continuous transmission on the network.

Module: A unit of electronics that consists of one or more cards mechanically bound such that they are inserted and removed from a backplane as a single unit.

Monitor: A client that nonintrusively observes the actions of the network without being a Master, Peer, or Slave.

Network (see Data network)

Node or Device: The electronics connected to a network via a single drop.

Partitioning (see Robust partitioning)

Peer: A client that that has equal authority over the assets of a data network.

Robust partitioning: A mechanism for assuring the intended isolation of independent aircraft operational functions residing in shared computing resources in all circumstances, including hardware and programming errors. This mechanism was developed for the ARINC 650 family of characteristics. Support for this mechanism is provided particularly by ARINC 653 and 659.

SERDES: A portmanteau for "SERializer DESerializer." An electronic component that converts parallel data to serial and serial to parallel. This component usually includes a method for encoding the serial data such that a clock can be reconstructed when the data is converted from serial to parallel. This encoding may also be designed to provide of their desirable features, such as DC balance.

Signal: A variation of a physical quantity used to convey data.

Slave: A client that is responding to the control of a Master.

Slot: A predefined interval of time in which a node (or subset of a system's nodes) have exclusive access to network resources. In minislotting, the minislot interval of time defines when node may claim access to resources and then the excess is held beyond the end of minislot time interval.

Source coverage: Fault-tolerance mechanisms that contain the effects of a fault to remain within the fault's source or provide means to make all the source's faults easily detectable.

Symbol: A signal state within a defined time interval that is recognized as distinct from other symbols.

Synchronous: The essential characteristic of time scales or signals such that their corresponding significant instants occur at precisely the same average rate. Note—The timing relationship between corresponding significant instants usually varies between specified limits.

APPENDIX A—LITERATURE SURVEY ANNOTATED BIBLIOGRAPHY

This appendix lists the literature search performed in the research.

- Ademaj, A., "Achieving Fail Silence in the Time-Triggered Architecture," Vienna University of Technology.

  This paper describes application-level techniques for error detection to achieve a high degree of coverage and provide fail-silent property for the data domain (as opposed to the time domain) in the time-triggered architecture (TTA). The techniques discussed are message checksums and double-task execution.

- Ademaj, A., "Slightly-Off-Specification Failures in the Time-Triggered Architecture," Vienna University of Technology.

  This paper describes the phenomenon of where a signal that is out of the specified range (in time or value) causes some nodes to receive a correct result while others receive an incorrect result. This paper deals with temporal slightly-off-specification (SOS) failures, and describes software-implemented fault-injection techniques (SWIFI) for generating SOS failures.

- Ademaj, A., Sivencrona, H., Bauer, G., and Torin, J., "Evaluation of Fault Handling of the Time-Triggered Architecture with Bus and Star Topology," University of Vienna.

  This paper describes requirements and algorithms for star couplers, and compares the performance of a star topology to that of a bus topology, each with arbitrarily faulty nodes. The test techniques described are SWIFI and heavy-ion injection.

- Aidemark, J., Vinter, J., Folkesson, P., and Karlsson, J., "Experimental Evaluation of Time-Redundant Execution for a Brake-by-Wire Application," International Conference on Dependable Systems and Networks, 2002, Proceedings, 2002, pp. 210-215.

  This paper describes a real-time kernel that supports multiple executions (time redundancy) and voting to tolerate transient faults and can enforce fail-silent operation on the node in the event of an unrecoverable error. This application is tested in a brake-by-wire application.

- Agilent Technologies, *Measuring Jitter in Digital Systems*, Application Note 1448-1, June 2003.

  This application note explains why and how to measure jitter as it affects eye diagrams.

- Ammann, P., Ding, W., and Xu, D., "Using a Model Checker to Test Safety Properties," *Proceedings Seventh IEEE International Conference on Engineering of Complex Computer Systems*, 11-13 June 2001, pp. 212-221.

  This paper applies formal methods in the form of model checking safety predicates for an automotive cruise control example. A particular contribution is developing the notion of a "dangerous trace" to create tests for potential safety violations based on hypothesized faults in the system under test.

- Arbaugh, W.A. and Van Doorn, L. "Embedded Security: Challenges and Concerns," *IEEE Transactions on Computer*, Vol. 34, Issue 10, October 2001, pp. 40-41.

  This article has a high-level discussion of the security challenges in embedded design that are significantly different from those of enterprise security.

- ARINC, "Avionics Full Duplex Switched Ethernet (AFDX) Network, ARINC Specification 664—Part 7, Draft 4," Aeronautical Radio, Inc., February 2005.

  Specification of the full duplex-switched Ethernet, along with a list of issues.

- ARINC, "Draft 3 of Project Paper 664, 'Aircraft Data Networks' Part 5 Network Interconnection Devices,'" AEEC Letter 01-112/SAI/742, May 2001.

  This document discussed networks in the context of the aviation domain, and gives specific guidance for network design. It addresses areas such as security, quality of service, network management, and mobility services.

- ARINC, "Ethernet Physical and Data Link Layer Specification, ARINC Specification 664—Part 2, Draft 2," Aeronautical Radio, Inc., May 2005.

  This document contains a description of the ARINC specification for Ethernet Physical and Data Link-Layer Specification.

- Arlat, J., Crouzet, Y., Karlsson, J., Folkesson, P., Fuchs, E., and Leber, G.H., "Comparison of Physical and Software-Implemented Fault Injection Techniques," *IEEE Transactions on Computers*, Volume 52, Issue 9, September 2003, pp. 1115-1133.

  The authors used heavy-ion radiation, pin-level electrical fault injection, EMI, and SWIFI on the Mars high-integrity, distributed, embedded, networked system executing a real-time control task. The different techniques have different strengths and weaknesses, and should be considered in concert to get best results from a fault injection campaign. Particularly noteworthy is that SWIFI was found to be relatively ineffective unless applied with considerable forethought.

- Askerdal, O., Claesson, V., Fredricksson, L.B., Hedberg, J., Johansson, J., Larsson, H., and Sterje, C., "Error Detection and Handling," Palbus task 10.5, June 15, 2000.

  This report surveys the issues that should be considered in picking a fault model and in evaluating error detection and handling mechanisms on a critical network.

- Azadmanesh, M. and Kieckhafer, R., "Exploiting Omissive Faults in Synchronous Approximate Agreement," *IEEE Transactions on Computers*, Vol. 49, No. 10, October 2000, pp. 1031-1042.

  This paper describes a new fault model that distinguishes between omissive and transmissive faults. This proposed fault model subsumes existing fault models and allows performance of the new model to be compared to previous models. The paper summarizes existing approximate agreement algorithms and presents a new approximate agreement algorithm, which uses the proposed fault model to achieve better fault tolerance by taking advantage of locally diagnosed omissive faults.

- Baleani, M., Ferrari, A., Mangeruca, L., Sangiovanni-Vincentelli, A., "Fault-Tolerante Platforms for Automotive Safety-Critical Applications," CASES 2003, pp. 170-177.

  This paper lists architectural patterns that can be used to implement fault tolerant by-wire control systems. The emphasis is more on System-on-Chip (SoC) implementations, so it contains interconnects other than networks and low degrees of replication. But it is relevant for creating fault-tolerant nodes (e.g., 2-of-2 nodes, which the authors call dual lock-step nodes) that can be placed on a network.

- Bauer, G., Kopetz, H., and Puschner, P., "Assumption Coverage Under Different Failure Modes in the Time-Triggered Architecture," *8th IEEE International Conference on Emerging Technologies and Factory Automation*, October 2001, pp. 333-341.

  This paper considers the detection coverage of both single and multiple faults in TTA systems with bus, star, and distributed star architectures. It proposes using centralized bus guardians in star or distributed star architectures as a cost-reduction measure compared to bus guardians at each node, and claims that it provides equivalent single-fault protection.

- Bertossi, A.A., Fusiello, A., and Mancini, L.V., "Fault-Tolerant Deadline-Monotonic Algorithm for Scheduling Hard-Real-Time Tasks," *Parallel Processing Symposium, Proceedings, 11th International*, 1-5 April 1997, pp. 133-138.

  This paper describes an algorithm for scheduling redundant tasks on multiple processors more efficiently than total replication of single-processor rate monotonic analysis (RMA).

- Beveridge, M. and Koopman, P., "Jini Meets Embedded Control Networking: A Case Study in Portability Failure," *7th IEEE Workshop on Object-Oriented Real-Time Dependable Systems (WORDS 2002)*, January 2002, pp. 11–18.

  This paper discusses the shortcomings of mapping Jini, an IP-based middleware system, onto an embedded platform, such as a Controller Area Network (CAN), due to fundamentally different assumptions underlying the two networks. The lessons learned are potentially relevant to any situation in which an IP-based system is interconnected to a real-time embedded network using a non-Ethernet protocol.

- Birman, K. and Joseph, T., "Reliable Communication in the Presence of Failures," *ACM Transactions on Computer Systems*, Vol. 5, No. 1, February 1987, pp. 47-76.

  This paper describes abstract primitives for fault-tolerant group communication and maintaining consistent state among replicas. Real-time issues are not addressed.

- Blanc, S., Gil, P., Ademaj, A., et al., "Three Different Fault Injection Techniques Combined to Improve the Detection Efficiency for Time-Triggered Systems," Technical University of Valencia.

  This paper describes three fault-injection techniques: pin-level, software-implemented, and heavy-ion. The paper discusses the relative coverage of the techniques, and how they can be combined to provide better coverage than any single technique.

- Bosch, R., "CAN Specification, Version 2.0," Robert Bosch GmbH, 1991.

  This document describes various aspects of the CAN databus, including frame formats, message timing, bit timing, error handling, and fault confinement.

- Briere, D. and Traverse, P., "Airbus A320/A330/A340 Electrical Flight Controls a Family of Fault Tolerant Systems," *IEEE*, 1993, pp. 616-623.

  This paper presents an overview of the Airbus fly-by-wire system architectures. The basic building blocks of the approach are a fail-safe control channel and a monitoring channel to ensure the control channel works correctly.

- Buckwalter, L., *Avionics Databuses*, Avionics Communications, Inc.: 2nd edition, 2001.

  Surveys many existing aviation protocols including ARINC 429, ARINC 629, ARINC 659 (SAFEbus), MIL-STD-1553, AFDX, and others.

- Charzinski, J., "Performance of the Error Detection Mechanisms in CAN," *Proceedings of the 1st International CAN Conference*, Mainz, Germany, September 1994, pp. 1.20-1.29.

  This paper describes residual error probability, which is the probability that a receiving node on a CAN network will accept a message that has been corrupted after transmission. It also describes the contributions errors affecting different aspects of the CAN bus. This paper also compares the REP of CAN to SCP and VAN Manchester, two other embedded automotive protocols.

- Chau, S. and Tai, A., "A Design-Diversity Based Fault-Tolerant COTS Avionics Bus Network," California Institute of Technology.

  This paper describes a fault-tolerant avionics network that is a combination of IEEE 1394 and I2C networks. The paper outlines the benefits of COTS components and details the changes required to make the system more dependable. It sets forth the SpaceWire network as a better alternative to I2C for the design diversity portion of the fault tolerance.

- Chockler, G., Keidar, I., and Vitenburg, R., "Group Communication Specifications: A Comprehensive Survey," *ACM Computing Surveys*, Vol. 33, No. 4, December 2001, pp. 427-469.

  This offers a comprehensive survey of different approaches to group communication. Different approaches have different implementation cost and, more importantly, provide somewhat different guaranteed properties. It is important to match the properties provided to the particular application needs.

- Corno, F., Gabrielli, P., and Tosato, S., "System-Level Analysis of Fault Effects in an Automotive Environment," *DFT2003: IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, 2003, pp. 529-536.

  This paper reports the results of modeling system-level faults at the vehicle level for a distributed system using the CAN protocol. The faults are injected at a very high level (loss of functionality based on loss of packets or bursty loss of network connection). The results suggest that this approach can identify sensitivities to losses of particular types of messages in a real-time control application (i.e., that loss of some messages can affect safety more than loss of other messages even when all messages lost are involved in safety-critical, real-time control loops). Additionally, the relative sensitivity to various types of message loss depends on the operational environment and control state of the system.

- Cristian, F., "Reaching Agreement on Processor Group Membership in Synchronous Distributed Systems," *Distributed Computing*, Vol. 4, 1991, pp. 175-187.

  This paper describes achieving agreement on the correctly functioning processors in the system, and agreement on the global state of those processors. It provides a system model and discusses three protocols that achieve these agreement services with varying performance and failure-detection tradeoffs.

- Cristian, F., Aghili, H., Strong, R., and Dolev, D., "Atomic Broadcast: From Simple Message Diffusion to Byzantine Agreement," FTCS 15, Vol. 3, 1985, pp. 200-206.

  This paper presents a set of protocols for atomic broadcast that tolerate increasingly severe fault models from omission faults up to Byzantine faults. The performance (in terms of message traffic and termination time) of the various protocols is also evaluated.

- Dawson, S., Jahanian, F., Mitton, T., and Tung, T., "Testing of Fault-Tolerant and Real-Time Distributed Systems via Protocol Fault Injection," FTCS 26, 1996, pp. 404-414.

  This paper describes an architecture for fault injection and probing socket-based applications and protocols. It also describes an implementation of the architecture provides scripting support for injecting or modifying messages in real-time systems.

- Dean, A. and Upender, B., "Embedded Communication Network Pitfalls," *Embedded Systems Programming*, September 1997.

  This paper describes implementation issues with using CAN, LonTalk, and Firewire in the context of embedded applications and discusses common mistakes made in incorporating these protocols into a system architecture.

- Dolev, D., Lynch, N., Pinter, S., Stark, E., and Weihl, W., "Reaching Approximate Agreement in the Presence of Faults," *Journal of the ACM*, Vol. 22, No. 3, July 1986, pp. 499-516.

  This paper addresses the issue of attaining approximate (rather than exact) agreement in the context of the Byzantine Generals problem. This has application in fault tolerant time agreement.

- Echelon Corporation, "Introduction to the LonWorks System," 1999.

  This document describes LonWorks, a control network protocol similar to Ethernet.

- Ellerbrock, P., "Intellibus Network Protocol Specification," Intellibus Network Systems, 2001.

  This document specifies the network protocol specification of The Boeing Company's proprietary IntelliBus® system. The document describes aspects of the protocol such as communication modes, network timing, membership services, and scheduling.

- Ellims, Parker, and Zurlo, "Design and Analysis of a Robust Real-Time Engine Control Network," *IEEE Micro*, July and August 2002, pp. 20-27.

  This paper describes an engineering case study in applying the CAN protocol to a semicritical stationary internal combustion engine application. Equation 1 contains the correct formula for maximum CAN message length in contrast to other publications that have an incorrect divisor value of 5. One vulnerability identified in this approach is that sending duplicates of a message on a prioritized network can result in self-synchronization in a heavily loaded network. This means that even though two copies of a message are sent at different times to avoid them both being disrupted by a noise burst, prioritization can make one wait so long that they are actually transmitted back-to-back, defeating the intended temporal separation.

- EtherCAT Technology Group, "Technical Introduction and Overview," http://www.ethercat.org/default.asp?introduction.html, accessed March 12, 2006.

  This Ethernet-based fieldbus system claims IEC61508 SIL 4 (highest level of integrity for safety-critical systems), corresponding to the network in standard: IEC/PAS 62407 Ed. 1.0 en:2005. The claim is not substantiated in this document; also see Jansen (2004).

- FAA CAST Position Paper, CAST-16, "Databus Evaluation Criteria," February 2003. www.faa.gov/aircraft/air_cert/design_approvals/air_software/cast/cast_papers/media/cast-16.doc

  This document proposes criteria for selecting, evaluating, and implementing new networks in avionics applications.

- Feld, Joachim, "Profinet—Scalable Factory Communication for all Applications," *IEEE International Workshop on Factory Communication Systems—Proceedings, WFCS*, 2004, pp. 33-38.

  This paper gives an overview of PROFINET, which is the Industrial Ethernet Standard devised by PROFIBUS International (PI) for either modular machine and plant engineering or distributed input/output. PROFINET is designed to provide synchronized real-time operation supporting both cyclic and acyclic data transfers.

- Ferrari, P., Flammini, A., and Vitturi, S., "Response Times Evaluation of PROFINET Networks," *Proceedings of the IEEE International Symposium*, Vol. 4, June 20-23, 2005, pp. 1371-1376.

  This paper measures real-time PROFINET performance, and suggests that further improvements to the standard beyond implementations that existed at the time of publication are required to achieve hard real-time performance. Of particular importance is creating new, extremely good clock synchronization mechanisms.

- Ferreira, J., Pedreiras, P., Almeida, L., and Fonseca, J., "Achieving Fault Tolerance in FTT-CAN," *4th IEEE International Workshop on Factory Communication Systems*, 2002, pp. 125-132.

  Outlines a fault model and fault-tolerance techniques for FTT-CAN, which layers static and dynamic message segments on top of CAN. The message traffic is scheduled by a Master node. To address FT issues, the author suggests techniques like replicated buses, replicated Master nodes, and bus guardians. The system allows flexible scheduling (e.g., to dynamically allocate time for messages that were lost due to transmission errors).

- Fiorini, D., Chiani, M., Tralli, V., and Salati, C., "Can we Trust in HDLC," *ACM SIGCOMM Computer Communication Review*, Vol. 24, Iss. 5, October 1994, pp. 61-80.

  This paper revisits the conclusions of Funk (1982) and others about the unsuitability of High-Level Data Link Control (HLDC) for critical applications. It concludes that later framing mechanisms, such as point-to-point protocol (PPP), have neglected to address known problems with HDLC, and that there are no easy ways to solve the HDLC framing end escape mechanisms without building an error detection on top of HDLC. It discourages some uses of X.25 PSPDN with International Standards Organization (ISO) Transport Protocols of Classes 0 to 3 because of this continuing problem.

- Fischer, M., Lynch, N., and Paterson, M., "Impossibility of Distributed Consensus With One Faulty Process," *Journal of the ACM*, Vol. 32, Iss. 2, 1985, pp. 374-382.

  This paper establishes a proof that it is impossible to establish consensus over an asynchronous network among nodes that may be faulty because of the possibility of nontermination. (The workaround in practice involves at least partial synchrony.)

- Fletcher, J., "An Arithmetic Checksum for Serial Transmissions," *IEEE Transactions on Communications*, Vol. 30, Iss. 1, January 1982, pp. 247-252.

  This paper describes and analyzes the performance of an arithmetic checksum with lower computational cost than the Cyclic redundancy check (CRC) checksum.

- FlexRay Consortium, "FlexRay Communications System Protocol Specification, Version 2.0," http://www.flexray.com/

  This document describes specification for the FlexRay Communication Protocol. It specifies network topologies, symbol coding and frame formats, start-up procedures, and bus guardians (for fault containment).

- Forster, R., "Manchester Encoding: Opposing Definitions Resolved," *Engineering Science and Education Journal*, Vol. 9, Iss. 6, December 2000, pp. 278-280.

  This paper attempts to resolve a dispute in the literature about whether a 0-1 or 1-0 transition represents a logic 1 value in Manchester encoding (the latter being correct).

- Fredriksson, L., "CAN for Critical Embedded Automotive Networks," *IEEE Micro*, July-August 2002.

  This paper discusses layering on global clock support and TT-CAN, a time-triggered CAN protocol.

- Freeman, Mark, "Achieving Real-Time Ethernet," *Manufacturing Engineer*, Vol. 83, Iss. 3, June-July 2004, pp. 14-15.

  This is a very brief description of challenges remaining for real-time Ethernet use in a manufacturing environment. Major hurdles include making equipment robust enough for noisy and harsh manufacturing environments and precise synchronization of nodes to provide predictably high performance.

- Fuchs, E., "Validating the Fail-Silence of the MARS Architecture Dependable Computing for Critical Applications," *Proceedings Sixth IFIP International Working Conference Dependable Computing for Critical Applications: DCCA-6*, Dal Cin, M., Meadows, C., and Sanders, W.H., eds., 1998, pp. 225-247. (Accessed as Research Report 5/97 from TU Vienna.)

  This paper reports the results of using SWIFI on the Mars testbed with rolling ball and brake-by-wire applications. (It appears to be precursor work to Arlat, et al., 2003). One vulnerability found was undetected transient errors in a time-redundant configuration where tasks are executed twice and compared. The cause for this is speculated to be semipermanent latchup effects, but could not be tracked to a definitive root cause.

- Führer, T., Müller, B., Dieterle, W., et al., "Time Triggered Communication on CAN," Robert Bosch GmbH.

  This paper specifies an application-layer protocol for the CAN databus. TT-CAN provides for time-triggered access to the bus, as well as arbitration windows, where any allowed node may send a message based on the native bit arbitration of the CAN protocol. The TT-CAN protocol also provides global clock synchronization.

- Funk, G., "Message Error Detecting Properties of HDLC Protocols," *IEEE Transactions Communications*, Vol. COM-30, No. 1, January 1982, pp. 252-257.

  This paper describes a vulnerability of the HDLC protocol to small numbers of bit errors. One vulnerability is corruption of a length field that causes the 16-bit frame check sequence (FCS) to be read from the wrong location in memory, resulting in vulnerability to a single bit error that happens to occur in the length field. It also describes a vulnerability to bit errors that change stuff bits into data bits and the converse.

- Garcia-Molina, H., "Elections in a Distributed Computing System," IEEE Transactions on Computers, Vol. C.31, No. 1, January 1982.

  This paper discusses the problem of leader election, which can arise during start-up, integration, or after a failure. A set of assumptions about system behavior is stated, and algorithms for leader election are provided for the case when nodes fail and the case where they do not fail.

- Gaujal, B. and Navet, N., "Fault Confinement Mechanisms on CAN: Analysis and Improvements," *IEEE Transactions on Vehicular Technology*, Vol. 54, No. 3, May 2005.

  This study uses Markovian analysis to evaluate the risk of reaching two CAN degraded modes: busoff and error-passive. It concludes that bursts of EMI on several consecutive transmissions can result in a busoff state being reached too easily. Sampling and sliding window mechanisms are proposed to reduce the probability of the bus being taken offline due to noise bursts.

- Georges, Jean-Philippe, Divoux, Thierry, and Rondeau, Eric, "A Formal Method to Guarantee a Deterministic Behaviour of Switched Ethernet Networks for Time-Critical Applications," *Proceedings of the IEEE International Symposium on Computer-Aided Control System Design*, 2004, pp. 255-260.

  This paper uses a formal model of Ethernet to evaluate maximum end-to-end delays in a priority-enabled switched system. Experimental confirmation of the model results is said to be underway, but is not reported in this paper.

- Ginosar, R., "Fourteen Ways to Fool Your Synchronizer," *Symposium on Asynchronous Circuits and Systems*, 2002, pp. 89-96.

  This paper reviews a number of typical mistakes designers make in input synchronizers that results in metastability and other problems.

- Hall, B., Driscoll, K., Paulitsch, M., and Dajani-Brown, S., "Ringing Out Fault Tolerance: A New Ring Network for Superior Low-Cost Dependability." (To be published, DSN 2005).

This document presents a high-dependability, braided-ring network topology that provides Byzantine fault tolerance. It claims high availability and integrity levels at a low cost.

- Hedberg, J., Nilsson, G., Johansson, L., Johansson, J., Hansson, L., Jakobsson, P., Franklin, R., Bergqvist, G., Sjostrom, H., Eriksson, J., Rimen, M., and Staff, M., "Analysis and Test of Bus Systems," PALBUS Project Report, Task 10.12 and 10.13, Revision 3.0, April 3, 2001.

  This report summarizes a number of safety-related validation methods used on the PALBUS project along with practical examples on how to employ them. It has a rich variety of material.

- Hedberg, J. and Wang, Y., "Methods for Verification and Validation of Distributed Control Systems," PALBUS Project Report, Task 10.10, April 3, 2001.

  This report catalogs a variety of validation and verification methods that can be used generically to evaluate critical networks as well as techniques specific to particular safety standards.

- Herard, J., Hedberg, J., Kivipuro, M., Malm, T., Edler, H., Sjostrom, H., and Strawinski, T., "Validation of Communication in Safety-Critical Control Systems," Nordtest Report TR 543, October 2003.

  This report provides an extensive discussion of safety-critical analysis techniques applicable to by-wire critical networks. It appears to be an evolution of work started in the PALBUS Project.

- Heiner, G. and Thurner, T., "Time-Triggered Architecture for Safety-Related Distributed Real-Time Systems in Transportation Systems," *FTCS*, 1998, pp. 402-407.

  This frequently-cited paper documents the transition of drive-by-wire from university research projects into advanced automotive development activities. In particular, it describes realistic automotive safety requirements from an industrial, rather than academic, perspective.

- Hoyme, K. and Driscoll, K., "SAFEbus," *Proceedings of the IEEE/AIAA 11th Digital Avionics Systems Conference*, 1992, pp. 68-73.

  This paper describes SAFEbus, a protocol developed for the Boeing 777. It describes a hardware and software architecture for a fault-tolerant, reliable backplane bus that provides isolation between components.

- Honeywell, "RealNet."

  This document describes issues involved with the design of a RealNet controller. RealNet is a mesh-topology protocol designed to overcome previously identified shortcomings with mesh topologies.

- IEEE Std 1394, "IEEE Standard for a High Performance Serial Bus," *Microprocessor and Microcomputer Standards Committee of the IEEE Computer Society*, 1995.

  This document gives a specification for the IEEE 1394 (Firewire) protocol. It gives a detailed description of the standard requirements for various physical-layer configurations (cable and backplane), as well as describing the link-layer and transaction-layer requirements.

- Jansen, D. and Buttner, H., "Real-Time Ethernet the EtherCAT Solution," *Computing and Control Engineering Journal*, Vol. 15, Iss. 1, February-March 2004, pp. 16-21.

  The paper describes Ethernet for Control Automation Technology (EtherCAT), a Master-Slave system where hardware implemented Ethernet devices can forward specially crafted Ethernet packets at high speed while reading or writing data to relevant sections of the message. Because multiple command and requests are included in each packet, the system reduces latency data transfer by amortizing the Ethernet overhead over the multiple messages.

- Kamali, B., "Error Control Coding" *IEEE Potentials*, Vol. 14, Iss. 2, April/May 1995, pp. 15-19.

  This is a very brief introduction to the topic of error control coding. A full survey of this topic is beyond the scope of this literature survey, but this provides a general starting point on that topic.

- Kanoun, K. and Powell, D., "Dependability Evaluation of Bus and Ring Communication Topologies for the Delta-4 Distributed Fault-Tolerant Architecture," *Symposium on Reliable and Distributed Systems*, 1991, pp. 130-141.

  This paper describes a study of various communication topologies for a critical system network created with off-the-shelf components. One conclusion is that topology selection depends in detailed ways on some critical parameters. Fault coverage of network access cards was determined to be of prime importance.

- Kantz, H. and Koza, C., "The ELEKTRA Railway Signalling-System: Field Experience With an Actively Replicated System With Diversity," *Twenty-Fifth International Symposium on Fault-Tolerant Computing*, 1995, pp. 453-458.

  This paper describes a high-integrity rail-signaling system based on two-channel nodes with design diversity. The first channel performs interlock controls and the second

channel serves as a safety monitor. The authors report field experience that software failures in the asynchronous part of the systems (portions that cannot be covered by the replica determinism mechanisms) are more frequent than faults in the synchronous parts. Asynchronous parts of the system include interrupt handlers, hardware exception handling, and synchronization mechanisms.

- Kavehrad, M., Doherty, J.F., Jun-Ho Jeong, Roy, A., and Malhotra, G., "10 Gbps Transmission Over Standard Category-5 Copper Cable," *Global Telecommunications Conference*, *GLOBECOM 2003*, IEEE, Vol. 7, Iss. 1-5, December 2003, pp. 4106-4110.

  Error-correcting code that claims 10^{-10} bit error rate (BER) after decoding for proposed 10,000BaseT Ethernet. Also has a table of BER for Category-5 copper cables.

- Kerkes, J., "Real-Time Ethernet," *Embedded Systems Programming*, January 2001.

  This paper discusses Ethernet for deterministic control applications through overlay of Master-Slave protocol.

- Kirrmann, H. and Zuber, P.A., "The IEC/IEEE Train Communication Network," *IEEE Micro*, Vol. 21, Iss. 2, March-April 2001, pp. 81-92.

  This article describes the Train Control Network (TCN), IEEE standard 1473-1999. The article describes the architecture and physical interconnections of the network, as well as message timings and node architectures.

- Koopman, P., "32-Bit Cyclic Redundancy Codes for Internet Applications," *Proceedings of the International Conference on Dependable Systems and Networks*, 2002, pp. 459-468.

  This paper publishes 32-bit CRC polynomials that outperform existing standard CRC polynomials (such as those used in Ethernet). The paper also defines a methodology for finding CRC polynomials with specific properties desired by system designers.

- Koopman, P., "Embedded System Security," IEEE Computer, July 2004.

  This article identifies the fact that embedded systems designers face security challenges beyond those normally addressed in the context of enterprise system security.

- Koopman, P., "Lost Messages and System Failures," *Embedded Systems Programming*, Vol. 9, No. 11, October 1996, pp. 38-52.

  This article discusses the sources of message loss or corruption on a network (Ethernet and LonTalk), and the effect lost messages can have on system reliability. It also discusses ways to analyze the network and reduce message loss.

- Koopman, P. and Chakravarty, T., "Analysis of Train Communication Network Protocol Error Detection Capabilities," http://www.tsd.org/papers/

  This paper contains a discussion of the different ways errors can occur and analysis of the effectiveness of the TCN-bit encoding and error detection schemes at handling these sources of error. It includes a novel analysis of Manchester bit encoding error detection using a semi-bit model.

- Koopman, P. and Chakravarty, T., "Cyclic Redundancy Code (CRC) Polynomial Selection for Embedded Networks," *Proceedings of the 2004 International Conference on Dependable Systems and Networks*, pp. 145.

  This paper discusses considerations for error detection (Hamming Distance, data word length, etc.) and methods for selecting optimal CRC polynomials. It publishes table of optimum CRCs for up to 16-bit CRC.

- Koopman, P., Morris, J., and Narasimhan, P., "Challenges in Deeply Networked System Survivability," *NATO Advanced Research Workshop on Security and Embedded Systems*, August 2005.

  This paper describes survivability challenges for coupled enterprise plus embedded systems that go beyond those traditionally considered in enterprise and desktop systems. In particular, manipulation of timing of messages or bypassing low-pass filters on periodically generated messages has the potential to create attacks in either direction across an embedded-to-enterprise gateway node.

- Kopetz, H., "Fault Containment and Error Detection in the Time-Triggered Architecture," *Sixth International Symposium on Autonomous Decentralized Systems*, April 2003, pp. 139-146.

  This paper introduces several critical failure modes for safety-critical, time-triggered systems, including babbling-idiot failures, masquerading failures, slightly-off-specification failures, crash and omission failures, and massive transient disturbances. TTP/C and FlexRay are analyzed in the context of their ability to handle these failures.

- Kopetz, H., "Real-Time Systems: Design Principles for Distributed Embedded Applications," Kluwer Academic Publishers, 1997.

  This book is one of the few text books in the area of design, and it addresses many of the major areas in the field.

- Kopetz, H., "On the Fault Hypothesis for a Safety-Critical Real-Time System," Draft, November 23, 2003, Accessed at http://aswsd.ucsd.edu/2004/pdfs/0401Kopetz.pdf on March 12, 2006.

  This paper describes the importance of having a precise fault hypothesis for ultradependable systems, and catalogs the various issues that must be addressed by a fault hypothesis for it to be considered complete. A precise articulation of the TTA fault hypothesis is presented comprising 14 distinct elements.

- Kopetz, H., Ademaj, A., and Hanzlikm A., "Integration of Internal and External Clock Synchronization by the Combination of Clock State and Clock Rate Correction in Fault-Tolerant Distributed Systems," *Proceedings of the 25th IEEE International Real-Time Systems Symposium*, 2004, pp. 415-425.

  This paper proposes adding an external rate synchronization to existing distributed, fault-tolerant clock synchronization algorithm. This allows a system to maintain internal clock state and maintain consistency with other clusters of nodes and an external time reference.

- Kopetz, H. and Grünsteidl, G., "TTP—A Protocol for Fault Tolerant Real-Time Systems," IEEE Computer, January 1994.

  This is an overview of TTP, including clock synch, group membership, mode changes, blackout handling, and replication.

- Koptez, H. and Nossal, R., "Temporal Firewalls in Large Distributed Real-Time Systems," *Proceedings of the 6th IEEE Computer Society Workshop on Future Trends of Distributed Computing Systems*, October 1997, pp. 310–315.

  This paper introduces the concept of a temporal firewall that is, partitioning the system according into self-contained subsystems where control data does not pass between the subsystems. This can simplify initial design and limit the impact future changes have on the system as a whole.

- Kopetz, H., Ademaj, A., Grillinger, P., and Steinhammer, K., "The Time-Triggered Ethernet (TTE) Design," *Eighth IEEE International Symposium of Object-Oriented Real-Time Distributed Computing*, 18-20 May 2005, pp. 22-33.

  This paper proposes a time-triggered Ethernet protocol that combines standard Ethernet and principles from TTP/C.

- Lala, J.H. and Harper, R.E., "Architectural Principles for Safety-Critical Real-Time Applications," *Proceedings of the IEEE*, Vol. 82, Iss. 1, January 1994, pp. 25-40.

  This paper describes architectural issues in creating critical distributed aviation systems, including issues such as attaining agreement in systems having redundant sensors, which in general do not provide exactly identical values.

- Lamport, L., Shostak, R., and Pease, M., "The Byzantine Generals Problem," *ACM Transactions on Programming Languages and Systems*, Vol. 4, No. 3, July 1982, pp. 382-401.

  This paper is the classic articulation of the Byzantine Generals problem, and the application of this to agreement with arbitrarily faulty nodes.

- Lampson, B., Abadi, M., Burrows, M., and Wobber, E., "Authentication in Distributed Systems: Theory and Practice," *ACM Transactions on Computer Systems*, Vol. 10, No. 4, November 1992, pp. 265-310.

  This paper defines a set of rules and a formal notation for describing authentication. These rules are built around the "speaks for" operator defined therein, and are applied to distributed computing scenarios. These rules are used to demonstrate security and information flow in various system activities like login and delegation in the context of various encryption methods and trust models.

- Larses, O., "Modern Automotive Electronics From a Dependable Systems Perspective," Technical Report TRITA-MMK 2003:38 ISSN 1400-1179, ISRN/KTH/MMK/R-03/38-SE, Mechatronics Lab., Royal Institute of Technology, 2003, Stockholm, Sweden.

  This report provides an architectural perspective on building high-dependability distributed systems from an automotive perspective. It includes particular emphasis on the topics of modeling, reliability, maintainability, and cost reduction.

- Latronico, E., "Reliability Validation of Group Membership Services for X-by-Wire Protocols," [thesis] Carnegie Mellon University, Pittsburgh, Pennsylvania.

  This work presents the importance of testing the fault model used to prove guarantees, such as group membership and clock synchronization, against expected physical fault rates. It presents a methodology for doing a design time analysis of the reliability of the maximum fault assumption of a proof. It also contains a detailed analysis of physical faults in the aviation domain, giving representative error rates and literature survey results. It discusses and compares the agreement guarantees and conviction policies of TTP/C and SPIDER. It also discusses type of faults, ranging from benign to arbitrary (Byzantine).

- Latronico, E. and Koopman, P., "Design Time Reliability Analysis of Distributed Fault Tolerance Algorithms," *Dependable Systems and Networks Conference*, June 2005, pp. 486-495.

  This paper discusses the importance of mapping hybrid fault models used in protocol guarantee proofs to actual physical faults and making realistic assumptions about physical fault rates. This allows system designers to assess the reliability of the maximum fault assumption for various guarantees at design time. This paper also discusses important ideas about group membership and conviction policies in fault-tolerant systems.

- Latronico, E., Miner, P., and Koopman, P., "Quantifying the Reliability of Proven SPIDER Group Membership Guarantees," *Conference on Dependable Networks and Systems (DSN)*, June 2004.

  This paper examines how often the assumptions behind a group membership guarantee fail to hold for a safety-critical aerospace distributed system. That same question can be asked of any property "proven" for a network: how often are the proofs invalid because assumptions made by the proof do not hold?

- Lee, Y.H., Rachlin, E., and Scandura, P.A., "Safety and Certification Approaches for Ethernet-Based Aviation Databuses," FAA report DOT/FAA/AR-05/52, December 2005.

  Design recommendations for Ethernet as an avionics network.

- Livani, M. and Kaiser, J., "EDF Consensus on CAN Bus Access for Dynamic Real-Time Applications," University of Ulm.

  This paper describes the implementation of distributed dynamic-scheduling approach to hard real-time on a CAN network.

- Maier, R., Bauer, G., and Stöger, G., "Time Triggered Architecture: A Consistent Computing Platform," IEEE Micro, July-August 2002.

  This document is an overview of TTA and TTP/C.

- Manzone, A., Pincetti, A., and DeCostantini, D., "Fault Tolerant Automotive Systems: An Overview," On-Line Testing Workshop, *Proceedings of the Seventh International*, 9-11 July 2001, pp. 117-121.

  This paper summarizes requirements for automotive drive-by-wire systems.

- McLaughlin, R.T., "The Immunity to RF Interference of a CAN System," *IEEE Colloquium on Integrity of Automotive Electronic Systems*, March 1993, pp. 4/1-4/8.

  This paper describes tests performed on a CAN network wired with various wire types and subjected to EM noise at various frequencies and intensities. The tests indicate the effect of wire type on bus errors. The CAN network was found to be robust to RF interference.

- Merlin, P. and Randell, B., "State Restoration in Distributed Systems," *FTCS 8*, pp. 129-134, 1978.

  This paper describes "backward error recovery" in the context of distributed systems as a means of returning the system to a correct state after a failure occurs. It uses an occurrence model to model system dependencies and relationships between actions performed by independent nodes. Distributed recovery is also addressed. Author notes that backward recovery will not be sufficient in all cases.

- Miller, J., "DC Free Encoding for Data Transmission System," U.S. Patent #4027335, 1977.

  This is the so-called "Miller 2" patent, which attempts to rectify the DC imbalance in the original Miller-encoding scheme.

- Miller, J., "DC Free Encoding for Data Transmission," U.S. Patent #4234897, 1978.

  This paper describes further improvements to Miller encoding to eliminate DC bias.

- Miller, A., "Recording and/or Reproducing System," U.S. Patent #3108261, 1963.

  This is the original patent describing Miller encoding.

- Morris, J. and Koopman, P., "Critical Message Integrity Over a Shared Network," *5th IFAC International Conference on Fieldbus Systems and Their Applications, FeT*, 2003.

  This paper suggests the use of lightweight digital signatures, based on CRCs, to isolate safety-critical messages on a network shared with nonsafety-critical messages and to prevent masquerade failures.

- Morris, J., Kroening, D., and Koopman, P., "Fault Tolerance Tradeoffs in Moving From Decentralized to Centralized Embedded Systems," *Dependable Systems and Networks*, 2004, pp. 377-386.

  This paper discusses the impact of introducing centralized components into distributed architectures in the context of the time-triggered protocol (TTP) star coupler as a centralized network guardian. The system is modeled using a symbolic model verification (SMV) model checker, and the effects of faults in the star coupler are

analyzed. The results indicate that the failures the star couplers were intended to mitigate can actually be introduced by failures in the star couplers.

- Obenza, Ray, "Guaranteeing Real-Time Performance Using RMA," SEI, Carnegie Mellon University.

  This paper contains a detailed discussion of RMA for periodic and a periodic tasks.

- Partridge, C., Hughes, J., and Stone, J., "Performance of Checksums and CRCs Over Real Data," *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pp. 68-76.

  This paper contains a comparison of the performance of CRC and checksum error detection over real data (UNIX file system data) versus performance over random data. One noteworthy result is that checksum performance is significantly degraded for nonrandom data.

- Paulitcsh, M. and Steiner, W., "Fault-Tolerant Clock Synchronization for Embedded Distributed Multi-Cluster Systems," *Euromicro Conference on Real-Time Systems*, 2003, pp. 249-256.

  This paper describes issues that arise in performing clock synchronization on multicluster systems, which are systems with paths having more than two message "hops," but which still have a hub-and-spoke topology. The application discussed in the paper is oriented to TTA systems.

- Paulitsch, M., Morris, J., Hall, B., Driscoll, K., Koopman, P., and Latronico, E., "On the Effectiveness of Error Detection Coverage of Cyclic Redundancy Codes in Ultra-Dependable Systems," *Proceedings of the International Conference on Dependable Systems and Networks*, June 28-July 1, 2005.

  This paper discusses scenarios where the effectiveness of using CRCs to detect data corruption is reduced because of other system design factors. It makes design recommendations to avoid these pitfalls.

- Pease, M., Shostak, R., and Lamport, L., "Reaching Agreement in the Presence of Faults," *Journal of the Association for Computing Machinery 27*, Vol. 2, April 1980.

  This seminal paper frames the Byzantine Generals question, in which a node in a distributed network can provide differing data to different nodes within the system.

- Pimentel, J. and Kaniarz, J., "A CAN-Based Application Level Error Detection and Fault Containment Protocol," IFAC (International Federation of Automatic Control), 2004.

  This paper proposes an approach for error detection and fault containment for CAN-based systems using a top-down approach (from the perspective of OSI layers) rather

than the more common approach of providing additional protocol mechanisms. Initial results on testing with a steer-by-wire application provided encouraging results.

- Poledna, S., "Fault Tolerance in Safety Critical Automotive Applications: Cost of Agreement as a Limiting Factor," Robert Bosch AG.

  This paper contains a discussion of application specific agreement measures (based on known variance in sensor inputs) to achieve agreement with lower cost than systematic agreement protocol and replication strategies.

- Powell, D., "Failure Mode Assumptions and Assumption Coverage," *Fault Tolerant Computing Symposium*, 1992, pp. 386-395.

  This paper proposes a method for formal analysis of failure mode assumptions when those assumptions are used in creating a safety or dependability case. A key point arising from this paper is the notion that even though a system is "proven" correct, that proof is only valid with respect to assumptions; and, those assumptions in aggregate are violated at least some of the time. Thus, for example, a formal proof of correctness shifts that dependability evaluation into the assumptions of that proof.

- Powell, D., "Distributed Fault Tolerance Lessons Learnt From Delta-4," *LAAS-CNRS Research Report 93192,* 1993.

  This paper documents lessons learned from the Delta-4 high-integrity computing project, which attempted to employ off-the-shelf components. A key result is the extreme importance of having a fail-silent network attachment controller.

- Ringler, T., Steiner, J., Belschner, R., and Hedenetz, B., "Increasing System Safety for by-Wire Applications in Vehicles by Using a Time Triggered Architecture," *Proceedings of the 17th International Conference on Computer Safety, Reliability and Security*, 1998, pp. 243-253.

  This paper includes a discussion of scheduling the TDMA cycle for a TTA system and the tools that support that scheduling.

- Rufino, J., Verissimo, P., Arroz, G., Almeida, C., and Rodrigues, L., "Fault-Tolerant Broadcasts in CAN," *Proceedings of the 28th International Symposium on Fault-Tolerant Computing Systems*, Munich, Germany, June 1998, pp. 150-159.

  This paper identifies a failure mode that can cause inconsistent or duplicate message delivery on a CAN network (violating any atomic broadcast assumptions that may be made about CAN). It proposes protocol suites that can be layered on top of CAN to provide unordered or totally ordered reliable broadcast.

- Rushby, J., "A Comparison of Bus Architectures for Safety-Critical Embedded Systems," SRI International, CSL Technical Report, September 2001.

  This paper contains a comparison of SAFEbus, SPIDER, TTP, and FlexRay over fault hypothesis and fault containment units, clock synchronization, bus guardians, start-up and restart, services, flexibility, and assurance.

- Rushby, J., "Critical System Properties: Survey and Taxonomy," *Reliability and System Safety*, Vol. 43, No. 2, 1994, pp. 189-219.

  This report identifies four major critical system properties: dependability, safety, security, and real-time, each of which has its own history of development and unique (and sometimes conflicting) requirements. It suggests component interaction and tightness of coupling as primary factors for considering the interaction of traditional design approaches from the various areas, and provides a taxonomy of design methods.

- Rushby, J., "An Overview of Formal Verification for the Time-Triggered Architecture," Presented at *7th International Symposium Formal Technique in Real-Time and Fault-Tolerant Systems 2002*, Oldenburg, Germany, Appears in Springer Verlag LNCS 2469, September 2002, pp. 83-105.

  This paper documents formal verification efforts for the TTA system.

- SAE Int, "PI-Bus," SAE standard SAE-AS4710, May 1993.

- Seale, E., "The Evolution of a SPIDER Fault Protection, Incremental Development, and the Mars Reconnaissance Orbiter Mission," *IEEE Aerospace Conference #1098*, 2003.

  This paper gives a high-level overview of SPIDER, a system architecture and network protocol developed for NASA. It describes the system architecture, network layers, and fault protection at various levels of the system. One interesting note is that many system properties are emergent from carefully specified local behaviors. The paper also describes the reuse and improvement of the SPIDER architecture for the Mars Reconnaissance Orbiter.

- Scheidler, C., Puschner, P., Boutin, S., Fuchs, E., Gruensteidle, G., Papadopoulos, Y., Pisecky, M., Rennhack, J., and Virnich, U., "Systems Engineering of Time-Triggered Architectures—The Setta Approach," *Proceedings of the 16th IFAC Workshop on Distributed Computer Control Systems*, 2000.

  This paper describes the SETTA approach to engineering safety-critical distributed real-time systems. This includes a three-phase "V" process model, and several scheduling and safety-analysis techniques. Additional domain-specific validators are used for automotive, aerospace, and rail applications.

- Shin, K. and Ramanathan, P., "Diagnosis of Processors With Byzantine Faults in a Distributed Computing System," *Fault Tolerant Computing Symposium*, 1987, pp. 55-60.

  This paper addresses the problem of identifying which processor(s) are faulty given that a Byzantine disagreement has occurred.

- Schill, J., "An Overview of the CAN Protocol," *Embedded Systems Programming*, September 1997.

  This paper contains a high-level description of the CAN protocol and application-level protocols like SDS and Devicenet.

- Sivencrona, H., Heberg, J., and Rocklinger, H., "Comparative Analysis of Dependability Properties of Communication Protocols in Distributed Control Systems," Palbus Report, Task 10.2, April 1, 2001.

  This report discusses safety-critical protocol properties, comparing: CAN, TTP/C, and Sercos. Profibus DP, FlexRay, and TTCAN are discussed in less detail.

- Sivencrona, H., Heberg, J., and Bridal, O., "Design Principles for Dependable Time-Triggered Control Systems," Palbus Report, Task 10.7, December 13, 2000.

  This report gives an extensive list of design suggestions for creating a dependable networked system using the TTP/C protocol.

- Slember, J. and Narasimhan, P., "Using Program Analysis to Identify and Compensate for Nondeterminism in Fault-Tolerant, Replicated Systems," *IEEE Symposium on Reliable Distributed Systems*, Florianopolis, Brazil, October 2004, pp. 251-263.

  Achieving replica determinism is extremely difficult, because computers (especially ones that execute operating systems) have myriad sources of inherent nondeterminism. This paper describes compiler-based tools to identify and compensate for nondeterminism, and builds on the second author's earlier work on fault tolerant middleware (see especially reference 17 from SRDS 1999).

- Steiner, W. and Paulitsch, M., "The Transition From Asynchronous to Synchronous System Operation: An Approach for Distributed Fault-Tolerant Systems," *Conference on Distributed Computing Systems*, 2002, pp. 329-336.

  This paper addresses the issue of moving from an initial asynchronous system at power-up to a synchronized system after power-up is at least partially completed. A particular fault type considered at some length is the issue of nodes masquerading at start-up.

- Steiner, W., Rushby, J., Sorea, M., and Pfeifer, H., "Model Checking a Fault-Tolerant Start-Up Algorithm: From Design Exploration to Exhaustive Fault Simulation," *Conference on Dependable Systems and Networks*, July 2004.

  This paper builds on Steiner, et al. (2002) by describing how model checking was applied to the start-up algorithm for TTA.

- Stock, M., "CAN Aerospace—Interface Specification for Airborne CAN Applications, V 1.7," Michael Stock Flight Systems, 2004.

  This document describes an application-layer protocol for CAN that is specifically designed for aeronautics applications. It provides a time-triggered schedule, clock synchronization, and domain-specific data formats.

- Stott, D., Ries, G., Hsueh, M., and Iyer, R., "Dependability Analysis of a High-Speed Network Using Software-Implemented Fault Injection and Simulated Fault Injection," *IEEE Transactions on Computers*, Vol. 47 (1) 108-119, January 1998.

  This paper compares two different fault injection techniques, simulated fault injection and SWIFI, by analyzing the effect of transient faults at the application layer. Faults are injected on Ethernet-based systems. The goal of these experiments is to compare the fault injection techniques and to determine what application-layer communication failures can result from real communications faults.

- Temple, C., "Avoiding the Babbling-Idiot Failure in a Time-Triggered Communication System," *28th Annual International Symposium on Fault-Tolerant Computing*, June 1998, pp. 218-227.

  This paper presents the concept of a bus guardian, an independent device with a priori knowledge of the bus schedule that only allows a node to transmit at the correct time. The paper also specifies criteria required for a node to be considered "fail-silent." An architecture for nodes incorporating the bus guardian is presented; this architecture addresses issues such as clock synchronization and schedule enforcement.

- Tindell, K., Burns, A., and Wellings, A., "Calculating Controller Area Network (CAN) Message Response Times," University of New York, Department of Computer Science.

  This paper establishes a method for computing the worst-case bound on message latency on a CAN network, in spite of the prioritization. The analysis includes error frames and remote transmission requests. Unfortunately, the formula for Cm in section 3 is incorrect—the divisor should be 4 instead of 5. See Ellims, et al. (2002) for the correct formula.

- Torres-Pomales, W., Malekpour, M., and Miner, P., "ROBUS-2: A Fault-Tolerant Broadcast Communication System," NASA TM-2005-213540, 2005.

  This document describes ROBUS-2, the communication system employed in NASA's SPIDER protocol. The document describes the behavior and structure of nodes, distributed coordination and redundancy management, operating modes, and start-up and restart issues.

- Tran, E., "Multi-Bit Error Vulnerabilities in the Controller Area Network Protocol," (thesis) Carnegie Mellon University, Pittsburgh, Pennsylvania.

  This thesis describes how transmission encoding can adversely affect error detection mechanisms. In this case, interaction between bit stuffing and the CRC checksum in a CAN network can lead to an unexpectedly high probability of undetected corruption.

- TTA Group, "Time-Triggered Protocol TTP/C High Level Specification Document, Protocol Version 1.1." http://www.ttagroup.org/

  This document details an abstract description of the services and mechanisms of the Time-Triggered Protocol. The preface notes that some aspects of the TTP/C implementation are covered by granted or pending patents. It includes an architecture description that covers network layers, network topologies, message structure and states, services, and fault hypotheses in detail.

- TTTech, "Comparison CAN/TTCAN—Byteflight—FlexRay—TTP," Austria 2004.

  This paper compares four protocols on the basis of various design criteria like bus speed, message formats, integrity mechanisms, protocol services, and fault-tolerance strategies.

- Unruh, J., Mathony, H., and Kaiser, K., "Error Detection Analysis of Automotive Communications Protocols," Robert Bosch GmbH.

  This paper identifies bit error scenarios in a CAN network that can result in an unexpectedly high probability of undetectable error through conversion of stuff bits into data bits or corruptions that result in a shortened message. It indicates that CAN networks can provide residual error rates that meet automotive safety requirements.

- Wang, Y., Sjostrom, H., Fredricksson, L.B., Nilsson, G., Bergqvist, G., and Jakobsson, P., "Design and Implementation of Dependable CAN-Based Distributed Systems," Palbus Task 10.6, March 29, 2001.

  This report gives extensive advice on designing dependable CAN-based network systems, based on the CAN Kingdom approach created by Kvaser.

- Wargo, C.A. and Chas, C., "Security Considerations for the E-Enabled Aircraft," *Proceedings of the IEEE Aerospace Conference*, Vol. 4, March 2003.

  This paper discusses the security issues that arise when shared IP networks are shared between safety-critical and nonsafety-critical applications in an aviation setting. It describes possible threats and identifies current security mechanisms that may be used to mitigate these threats.

- Widder, J., "Booting Clock Synchronization in Partially Synchronous Systems," *Distributed Computing*, Springer Lecture Notes in Computer Science, Vol. 2848, 2003, pp. 121-135.

  This paper addresses the issues that arise in booting a network when (before the network is fully booted), not enough processes are available to ensure a Byzantine quorum during the boot process.

- Yeh, Y.C., "Design Considerations in Boeing 777 Fly-by-Wire Computers," IEEE, 1995.

  This paper gives an overview of the electrical systems used for fly-by-wire in the Boeing 777 aircraft. It describes the use of triple-triple redundancy (triple redundant processors replicated across triple redundant communication links), which allows the system to tolerate faults while simultaneously improving availability. The paper also argues for a single, rigorously tested version of software, rather than multiple versions, which it claims cannot be truly independent due to interaction in the design assurance process.

APPENDIX B—SURVEY RESPONSES

RESPONSE #1.

1) Please list and classify the networking technologies used in your products according to the table below:

Network Number	Network Technology or Standard Name [1]	Is the network intra-box or inter-box or both?	Approved Under Which FAA FAR part(s) (e.g., 23, 25)	Design Assurance Levels for Hardware and Software	Highest Hazard Category Supported
1	C-5 Modernization	both	N/A (Military Cert)	A	CAT
2					
3					
4					
5					
6					

For networks listed above that are not compliant to a publicly available standard, please attach documents that describe the networks or say how to obtain such documents if possible.

If an answer to any of the following questions is specific to one of the networks listed in this table, please precede the answer with the Network Number in a circle or parentheses. An answer without a Network Number *is assumed to apply to all of the networks listed above. If the answer to one of the following questions is clearly stated in a publicly available standard or attached documentation, just enter "*see doc*". If proprietary restrictions prevent answering a question, please enter "proprietary".*

2) Is it required or desired that third parties or other companies produce components that connect to the same network? If yes, how are these components assured to be compatible and not cause problems under all circumstances?
   a) We require some of more complex and safety-critical systems, especially software, suppliers to conduct software safety analyses per IEEE STD 1228-1994 Software Safety Plans. We also require extensive testing per DO-178B Level A or equivalent alternative integrity methods.

[1] If only parts of the standard are relevant, please list which parts that are used by the system. For standards that have several variants or versions, please list the specific variant or version.

3) What mechanisms prevent or detect network malfunction (e.g., loss or corruption of data, corrupted addresses, loss of sync, blocked media access)? Are these mechanisms part of the network itself or the applications connected by the network? (Typical mechanisms include voters, guardians, redundant paths, CRCs, etc.)
   a) We require software safety features and engineered safety features in both hardware and software, such as miscompares (duplex monitoring), voting (tri-plex monitoring), validity checks, wrap checks, out of range checking, exception handling, staleness checks, and continuous monitoring of all safety critical functions and alert the crew.

4) If a network must support different hazard categories (e.g., catastrophic, hazardous) and associated design assurance levels, what are the highest and lowest levels? What mechanisms, if any, are used to prevent a lower level from affecting a higher level?
   a) Highest HW DAL I, SW Level A; lowest HW DAL IV, SW Level E. Some programs use a hazard risk index, software evidence assurance level. Redundancy and standby backups are most common ways of lowering levels. Also safety margin and operator input in addition to software are mitigation. We check all data coming in from lower level to higher level. Segregation and isolation of higher level software is sometimes used.

5) Does the network need to support system architectures to achieve fault-tolerance (e.g., self-checking pairs, triple modular redundancy, hot and cold standby nodes etc? If so, what network support is needed (e.g., synchronization, membership agreement, atomic broadcast, node shadowing, Byzantine Agreement, fault containment, independent redundant paths)?
   a) All of the above and more. Fault containment is popular. "Checkers checking the checkers" for validity of data and to prevent corruption hazardously misleading information. Certain built in test and reasonableness checks and layers of cross checking safety-critical functions are used as technical integrity.

6) Does the system or network require coordinated action or consensus between different networked components for time synchronization, mode changes, fault diagnosis, etc?
   a) yes

7) What is done to deal with network-related common-mode faults, both internal and external (e.g., power)?
   a) We do everything possible to prevent common mode software faults through up front focus on system safety and safety-critical systems design safeguards and safety features. Dissimilar software will not prevent bad requirements. So strategy is requirement based and functional based system safety programs to prevent common mode faults. Comprehensive testing rigor, failure modes effects testing, fault insertion testing and putting through the integrity testing ringer is the key.

8) What is done to mitigate spatial-proximity faults, i.e., multiple failures or fault propagation due to components being in close physical proximity (e.g., due to local structural failure)?
   a) We don't really think physical separation is possible nor applicable with modular day modular or open architecture avionics. Prevent structural failure, prevent fire propagation.

9) Does the network provide mechanisms to contain host software failures (i.e., prevent software failures in a host from affecting network operation)? If so, what network mechanisms ensure this containment?
   a) Yes, but these vary. When failures are detected we declare that leg as a fail and go to backup, but alert the crew they have lost it.

10) What type of interface is used between the network and its clients? (e.g., pseudo or true dual port memory? DMA? Is data-flow and timing controlled by the clients or the network?)
   a) We use dual and independent red and green MIL-STD-1553 data buses, ARINC-429, other such as fibre optic, Firewire. They have different conventions and schemes.

11) Does the network use multiple modes of operation? If so, please describe the mode change mechanisms and any interlocks that are present between the network and the application(s) that it supports.
   a) Yes, there are several schemes used. We have a timeline and allocated space for each function. Time and space partitioning is used and we check all functions for integrity per requirements. We prevent calls to non operational modes, residual code and non valid locations.

12) Is the network used to download software, constants, or databases? If so, is the network required to continue communicating essential data while these downloads are being performed or is the downloading done only at times when it is deemed to be safe to do so? If the later, what mechanisms are used to ensure that downloading is not done when it is not safe to do so?
    a) Data loaders are done prior to flight. CRC checks used to ensure integrity. Procedures are used to download TOLD, etc during preflight. Things like FMS, waypoints, etc are entered in flight. Not sure of automatic downloads.

13) What is the approximate network design complexity (gates or lines of software), including any shared components such as switches as well as client interfaces (e.g., BIU)?
    a) Level A

14) Have the network components been designed in accordance with recommended design assurance guidelines, e.g., DO-178B (what level), DO-254 (complex, simple, COTS)? Was formal verification included?
    a) Yes, absolutely for airworthiness and cert

15) How are these physical-layer properties (including design margin) validated and verified during design, installation, normal operation, and any later modification?
    a) Some analyzed, all thoroughly tested

16) Do your systems provide scrubbing methods for finding latent faults or degraded components?
    a) Not sure, probably not since system is programmed to do exactly what one instructs. Some logic, criteria, intelligence may be needed.

17) What tools and techniques were used to assess system throughput, latency, jitter and other performance?
    a) This would take too long to explain. Lots of analyses, synthesis , testing observation, best judgment. Latency times to annunciate special alerts, warnings and cautions did require human factors to establish acceptable times.

18) Does the network interface require electrical fault isolation? If so, how is this achieved (e.g., opto-coupler, transformer, and resistors)?
    a) Not sure, but it was built into the design by EEs, I'm a software type

19) Do your systems have current or future security requirements that have to be supported by the network, e.g., is an entity authenticated by the mere fact that it exists at a certain location within the network; do multiple levels of security (even if it is just the two levels of secure versus non-secure) have to be kept separate? Does the use of standard protocols such as Ethernet and IP increase the perceived risk?
    a) Yes all sorts of security since we are military airlifters. I can't even discuss since it is that secure and we simple don't know and can't share.

20) From your experience, what have been the largest problems associated with including data networks in avionics?
    a) Lack of the recognition by uninformed management that we need comprehensive system and software safety analysis of the safety-critical application software and functions on the bus.

21) What have been the most unusual problems associated with including data networks in avionics?
    a) Anticipating all of the many failures and software anomalies during testing because of wrong decomposing of system level requirements in software and designing it as intended. Too many reworked under spiraling approach. Do it right by decomposing requirements in software domains.

RESPONSE #2.

1) Please list and classify the networking technologies used in your products according to the table below:

Network Number	Network Technology or Standard Name [2]	Is the network intra-box or inter-box or both?	Approved Under Which FAA FAR part(s) (e.g., 23, 25)	Design Assurance Levels for Hardware and Software	Highest Hazard Category Supported
1	ARINC 429	Inter-box	14 CFR 23 and 14 CFR 25	Level A and Level C	Major
2	CAN	Inter-box	14 CFR 25	Level A	Catastrophic
3	ASCB	Inter-box	14 CFR23 and 14 CFR 25	Level A	Catastrophic
4	RS-422	Inter-box	14 CFR 23	Level A	Catastrophic
5					
6					

For networks listed above that are not compliant to a publicly available standard, please attach documents that describe the networks or say how to obtain such documents if possible.

If an answer to any of the following questions is specific to one of the networks listed in this table, please precede the answer with the Network Number in a circle or parentheses. An answer without a Network Number *is assumed to apply to all of the networks listed above. If the answer to one of the following questions is clearly stated in a publicly available standard or attached documentation, just enter "*see doc*". If proprietary restrictions prevent answering a question, please enter "proprietary".*

2) Is it required or desired that third parties or other companies produce components that connect to the same network? If yes, how are these components assured to be compatible and not cause problems under all circumstances?
(1) yes. Single transmitter on each line. Components assured to be compatible by complying with system requirements and completing system integration tests on the aircraft and hot bench.
(2-4) no.

3) What mechanisms prevent or detect network malfunction (e.g., loss or corruption of data, corrupted addresses, loss of sync, blocked media access)? Are these mechanisms part of the network itself or the applications connected by the network? (Typical mechanisms include voters, guardians, redundant paths, CRCs, etc.)

(1) odd sign parity bit per data word. (2) CRC (3) CRC, redundant path for critical data (4) CRC, voters.

[2] If only parts of the standard are relevant, please list which parts that are used by the system. For standards that have several variants or versions, please list the specific variant or version.

4) If a network must support different hazard categories (e.g., catastrophic, hazardous) and associated design assurance levels, what are the highest and lowest levels? What mechanisms, if any, are used to prevent a lower level from affecting a higher level?
(1) does not have to support different hazard categories in our application. Uses a single transmitter per bus to completely manage all traffic. Does not deal with contention. (2) does not have to support different hazard categories in our application. Uses priority of message label to allow messages of higher criticality to have access to the bus when the need it. (3) supports message traffic from multiple level D to level A. Uses predetermined message traffic to ensure time available for all traffic. Uses CSMA techniques. (4) supports message traffic from multiple level D to level A. Uses CSMA and CD techniques.

5) Does the network need to support system architectures to achieve fault-tolerance (e.g., self-checking pairs, triple modular redundancy, hot and cold standby nodes etc? If so, what network support is needed (e.g., synchronization, membership agreement, atomic broadcast, node shadowing, Byzantine Agreement, fault containment, independent redundant paths)?

no

6) Does the system or network require coordinated action or consensus between different networked components for time synchronization, mode changes, fault diagnosis, etc?
(1) no (2) no (3) yes—changes between air mode and ground mode based on agreement between two designated components but defaults to air mode. (4) no

7) What is done to deal with network-related common-mode faults, both internal and external (e.g., power)?
(1) network can power through dead or un-powered receivers. Survives lightning and HIRF events. (2) network can power through dead or un-powered receivers. Survives lightning and HIRF events. Survives single line of the pair shorts to common ground or power with reduced speed capability. (3) network can power through dead or unpowered receivers. Survives lightning and HIRF events. (4) network can power through dead or unpowered receivers. Survives lightning and HIRF events. Survives single line of the pair shorts to common ground or power.

8) What is done to mitigate spatial-proximity faults, i.e., multiple failures or fault propagation due to components being in close physical proximity (e.g., due to local structural failure)?
Not usually a consideration in a business jet. Everything is ‘close’.

9) Does the network provide mechanisms to contain host software failures (i.e., prevent software failures in a host from affecting network operation)? If so, what network mechanisms ensure this containment?
(1) no. (2) contains babbling transmitter using hardware message priority structure. (3) not sure. (4) not sure.

10) What type of interface is used between the network and its clients? (e.g., pseudo or true dual port memory? DMA? Is data-flow and timing controlled by the clients or the network?)
(1) interrupt driven service routines tightly coupled in the OS. Data flow and timing controlled by the network transmitter. (2) dual port memory using RX and TX FIFOs. Data flow of the receiver controlled by the network, data flow of the transmitter controlled by the client. Timing controlled by the network. (3) dual port memory using RX and TX FIFOs. Data flow of the receiver and transmitter controlled by the network. Timing controlled by the network. (4) dual port memory using RX and TX FIFOs. Data flow of the receiver and transmitter controlled by the network. Timing controlled by the network.

11) Does the network use multiple modes of operation? If so, please describe the mode change mechanisms and any interlocks that are present between the network and the application(s) that it supports.
(1) no. (2) no. (3) yes, ground and air mode.—changes between air mode and ground mode based on agreement between two designated components but defaults to air mode. (4) no.

12) Is the network used to download software, constants, or databases? If so, is the network required to continue communicating essential data while these downloads are being performed or is the downloading done only at times when it is deemed to be safe to do so? If the later, what mechanisms are used to ensure that downloading is not done when it is not safe to do so?
Different mechanisms are used to load software than the communication network.

13) What is the approximate network design complexity (gates or lines of software), including any shared components such as switches as well as client interfaces (e.g., BIU)?

(1) 50 to 100 lines of software. One dual opto-coupler per receiver. One dual op-amp per transmitter. (2) 25 to 50 lines of software. One UART per TX /RX. (3) 500 to 1000 lines of software. One UART per TX/RX. (4) 100 to 500 lines of software. One UART per TX/RX.

14) Have the network components been designed in accordance with recommended design assurance guidelines, e.g., DO-178B (what level), DO-254 (complex, simple, COTS)? Was formal verification included?
DO-178B commensurate level for the criticality, no CPLD involved.

15) How are these physical-layer properties (including design margin) validated and verified during design, installation, normal operation, and any later modification?
(1) Observation of waveforms using oscilloscopes and waveform analyzers on the aircraft. (2) CANALYZER tools as well as waveform analysis. (3) validated by supplier and supplier reps on the aircraft. (4) validated by supplier and supplier reps on the aircraft.

16) Do your systems provide scrubbing methods for finding latent faults or degraded components?
(1) timing constraints to fail network and toggle to redundant source of information. (2) multi-path routing and timing measurements to identify best data path. (3)don't know (4) don't know.

17) What tools and techniques were used to assess system throughput, latency, jitter and other performance?
(1) analysis of transmission rate vs. update requirements. Dead bus recovery analysis, weak transmitter forced.
(2) competing transmitter, babbling transmitter, test software that only timed and recorded communications.
(3) unknown. (4) unknown.

18) Does the network interface require electrical fault isolation? If so, how is this achieved (e.g., opto-coupler, transformer, and resistors)?
(1) Opto-isolator, resistor. (2) resistor. (3) transformer. (4) Opto-coupler.

19) Do your systems have current or future security requirements that have to be supported by the network, e.g., is an entity authenticated by the mere fact that it exists at a certain location within the network; do multiple levels of security (even if it is just the two levels of secure versus non-secure) have to be kept separate? Does the use of standard protocols such as Ethernet and IP increase the perceived risk?

We do not allow the communication network of the electrical or avionic system to be the mechanism that is involved in loading of software. We are concerned about network security when software is to be loaded, and especially if the Internet is planned to be used to provide that software.

20) From your experience, what have been the largest problems associated with including data networks in avionics?
NO CLEAR DEFINITION OF "DETERMINISTIC" THAT IS COMMONLY UNDERSTOOD BY REGULATORS AND DEVELOPERS. PLEASE DEFINE THIS TERM SO THAT PROPOSED BUS TECHNOLOGIES CAN BE EVALUATED BEFORE THEY ARE SHOT DOWN ARBITRARILY.

21) What have been the most unusual problems associated with including data networks in avionics?
(a) Regulators or regulations that require direct point to point wiring for certain data being exchanged instead of allowing the data network to provide the same exact information. This has contributed to much additional wiring and weight in the aircraft. (b) Providing convincing evidence that a zone in the aircraft has an agreeable and quantifiable exposure to HIRF fields and lightning. For example, if an F-15 was ever to light up your aircraft with target acquisition radar, you would be in much greater danger from the inbound heat-seeking missile than you would ever suffer from radar produced electronic upsets.

SURVEY RESPONSE #3.

1) Please list and classify the networking technologies used in your products according to the table below:

Network Number	Network Technology or Standard Name [3]	Is the network intra-box or inter-box or both?	Approved Under Which FAA FAR part(s) (e.g., 23, 25)	Design Assurance Levels for Hardware and Software	Highest Hazard Category Supported
1	HSDB	Intra-box	FAR part 23	Hazardous	Hazardous
2					
3					
4					
5					
6					

For networks listed above that are not compliant to a publicly available standard, please attach documents that describe the networks or say how to obtain such documents if possible.

HSDB is a GARMIN proprietary data bus that currently utilizes IEEE 802.3 full duplex 10BASE-T point-to-point communication scheme. There are no publicly available documents that describe this network.

If an answer to any of the following questions is specific to one of the networks listed in this table, please precede the answer with the Network Number in a circle or parentheses. An answer without a Network Number *is assumed to apply to all of the networks listed above. If the answer to one of the following questions is clearly stated in a publicly available standard or attached documentation, just enter "*see doc*". If proprietary restrictions prevent answering a question, please enter "proprietary".*

2) Is it required or desired that third parties or other companies produce components that connect to the same network? If yes, how are these components assured to be compatible and not cause problems under all circumstances?

Not at this time.

[3] If only parts of the standard are relevant, please list which parts that are used by the system. For standards that have several variants or versions, please list the specific variant or version.

3) What mechanisms prevent or detect network malfunction (e.g., loss or corruption of data, corrupted addresses, loss of sync, blocked media access)? Are these mechanisms part of the network itself or the applications connected by the network? (Typical mechanisms include voters, guardians, redundant paths, CRCs, etc.)

CRC's and internal communication checks are used to detect network malfunctions. These are mechanisms of the applications connected to the network.

4) If a network must support different hazard categories (e.g., catastrophic, hazardous) and associated design assurance levels, what are the highest and lowest levels? What mechanisms, if any, are used to prevent a lower level from affecting a higher level?

HSDB currently supports minor, major and hazardous data and a hazardous design assurance levels. The OS is responsible for preventing a lower level from affecting higher levels data?

5) Does the network need to support system architectures to achieve fault-tolerance (e.g., self-checking pairs, triple modular redundancy, hot and cold standby nodes etc? If so, what network support is needed (e.g., synchronization, membership agreement, atomic broadcast, node shadowing, Byzantine Agreement, fault containment, independent redundant paths)?

No.

6) Does the system or network require coordinated action or consensus between different networked components for time synchronization, mode changes, fault diagnosis, etc?

No.

7) What is done to deal with network-related common-mode faults, both internal and external (e.g., power)?

Since the network is a point-to-point network, and the LRUs that make up the network have different designs, the network is not as susceptible to common-mode faults as a network that would utilize a common router topology.

8) What is done to mitigate spatial-proximity faults, i.e., multiple failures or fault propagation due to components being in close physical proximity (e.g., due to local structural failure)?

Nothing.

9) Does the network provide mechanisms to contain host software failures (i.e., prevent software failures in a host from affecting network operation)? If so, what network mechanisms ensure this containment?

The point-to-point topology allows the LRU to contain host software failures to only affect the two connected LRUs. The software in the LRU will prevent failures from being propagated on the network.

10) What type of interface is used between the network and its clients? (e.g., pseudo or true dual port memory? DMA? Is data-flow and timing controlled by the clients or the network?)

The interface between LRUs is a dedicated point-to-point interface. Each LRU controls the data-flow and timing controls of the network.

11) Does the network use multiple modes of operation? If so, please describe the mode change mechanisms and any interlocks that are present between the network and the application(s) that it supports.

No.

12) Is the network used to download software, constants, or databases? If so, is the network required to continue communicating essential data while these downloads are being performed or is the downloading done only at times when it is deemed to be safe to do so? If the later, what mechanisms are used to ensure that downloading is not done when it is not safe to do so?

Yes. Downloading of software is done while the system is in a configuration mode, not during normal operation. The mode of the system is controlled by how the PFD is powered up.

13) What is the approximate network design complexity (gates or lines of software), including any shared components such as switches as well as client interfaces (e.g., BIU)?

This is a very simple network. It utilizes some shared source code to make sure all LRUs communicating on the network utilizes the same protocol. This network could be replaced with a simple dedicated serial receive and transmit lines.

14) Have the network components been designed in accordance with recommended design assurance guidelines, e.g., DO-178B (what level), DO-254 (complex, simple, COTS)? Was formal verification included?

Yes. The network has been designed to DO-178B level B. Formal verification was performed on the software. The hardware utilizes IEEE 802.3 full duplex 10BASE-T COTS components.

15) How are these physical-layer properties (including design margin) validated and verified during design, installation, normal operation, and any later modification?

The network has extensive diagnostics that utilize both hardware and software monitoring of the health of the network. These diagnostics have been utilized during design, normal operation and during stress testing to make sure it complies with the design margins specified.

16) Do your systems provide scrubbing methods for finding latent faults or degraded components?

The network has extensive diagnostics that utilize both hardware and software monitoring of the health of the network to make sure the network is performing as desired.

17) What tools and techniques were used to assess system throughput, latency, jitter and other performance?

We have utilized separate hardware (network sniffer) that has the capability to monitor all network traffic to make sure that the throughput, latency, jitter and other performance related aspects of the network meet our design requirement.

18) Does the network interface require electrical fault isolation? If so, how is this achieved (e.g., opto-coupler, transformer, and resistors)?

No.

19) Do your systems have current or future security requirements that have to be supported by the network, e.g., is an entity authenticated by the mere fact that it exists at a certain location within the network; do multiple levels of security (even if it is just the two levels of secure versus non-secure) have to be kept separate? Does the use of standard protocols such as Ethernet and IP increase the perceived risk?

No, this network does not have any current of future security support. We do not feel that the use of standard protocols increases perceived risk on our closed network.

20) From your experience, what have been the largest problems associated with including data networks in avionics?

We feel that since the network is so simple, we have not run into many of the common network related issues such as collisions and retries. Most of our challenges have been with trying to find a prioritization scheme that would prioritize data sources without creating too much redundant data in the system.

21) What have been the most unusual problems associated with including data networks in avionics?

Probably the most unusual issue we have run into has been related to quickly and efficiently determining if a LRU is on line.

APPENDIX C—ORIGINAL EVALUATION CRITERIA

This appendix comprises a series of questions that forms a list of the draft evaluation criteria that a system designer should consider when selecting or creating a data network for aviation digital electronics systems. This is the original list developed during Phase 1.

These questions are extracted from this report's narrative description of the evaluation criteria and follow the same document structure. If the answer to any of these questions is "Unsure" or "Unsatisfactory," the designer must determine if that criterion is applicable to the anticipated system design. If it is applicable, then the designer must do one of the following:

- Select a different data network
- Correct the shortcoming(s) in the data network
- Change the system design to accommodate the shortcoming(s)

It is preferable for the dependability features of a network to be standard, or at least be widely used. The rationale for this is similar to the argument for the use of commercial off-the-shelf (COTS) components: the wider the use, the higher the likelihood that design errors will be found and corrected. The alternative is to create ad hoc do-it-yourself dependability mechanisms.

Note that the questions below should not be answered out of context. The preceding narrative of this report must be read to correctly interpret the full meaning of these questions.

C.1 PHYSICAL LAYER.

C.1.1 ENVIRONMENT.

Does the network meet the aerospace environment requirements of DO-160 or other imposed environmental requirements?

C.2 PROBABILITY OF BIT ERRORS.

Has an upper bound on anticipated bit error rate been established via testing using the worst-case anticipated signal path characteristics, environment, local clock characteristics, sampling rate, modulation and encoding schemes?

If the data network exploits the mixing of dominant and recessive signals on its media to perform some logic function, have the receivers been designed to tolerate Wired–Or glitches, or have suitable design rules been created that limit the duration of the glitches to a duration that can be tolerated?

Does the network have a high symbol-encoding efficiency?

C.3 PROBABILITY OF ELECTRICAL COMPONENT FAILURES.

Do the data network's electronic components have established hardware failure rates and characteristics so that aviation digital electronics designers can do the required failure modes and effects analysis (FMEA) and fault tree calculations?

Are signal margins robust enough to handle aging effects on connectors, media, and drivers? That is, could aging cause a network to not meet the aerospace environment requirements of DO-160 or other imposed environmental requirements at some point in the aviation digital electronics system's lifetime?

Has a valid synchronizer metastability error rate test been done for every synchronizer design in the data network's electronics and has an acceptable ceiling on the error rate been established?

Have the clock requirements for the data network's electronic circuitry been clearly and completely stated? Are these requirements achievable in an aviation digital electronics environment?

C.3.4 ELECTRICAL ISOLATION PROPERTIES.

Does the data network's physical layer allow for electrical (galvanic) isolation among redundancies?

Does the data network's physical layer allow for receive-only connections to its media? Can high assurance be given that these receive-only connections cannot affect the data network?

C.3.5 PHYSICAL COMPOSABILITY.

Does the data network have characteristics or design rules that will guarantee that it will reliably work with any number of nodes up to some explicitly given maximum number of nodes?

For data networks that have the freedom to assume a number of different topologies or topology variations, does the network have characteristics or design rules that will guarantee that it will reliably work with all possible variations (that are not precluded by its design rules)?

C.4 DATA LINK LAYER.

C.4.1 MEDIA ACCESS CONTROL.

Does the media access control (MAC) sublayer protocol provide guaranteed bounds on media access performance (latency and jitter) regardless of the behavior of its clients?

Does the MAC sublayer protocol amplify small failures and errors into loss of MAC services? In particular, does the MAC sublayer protocol allow transient failures and errors to have an effect that persists longer than current transmissions?

Does the MAC sublayer protocol have a single point of failure?

Is the MAC sublayer protocol efficient (how much of the data network's bandwidth can be used in the worst case)?

C.4.2 LINE-LEVEL ENCODING.

Does the spectrum radiated from the line encoding have components in frequencies that can adversely affect other equipment?

C.4.3 MESSAGE FORMATTING (FRAMING).

Is the data network's framing structure brittle (i.e., do simple errors cause a node to lose more than one message per error)?

Do receivers tolerate any preamble bit(s) being erroneous? Do receivers tolerate any preamble bit(s) other than the one(s) that would make the preamble look like a start delimiter? Is the Hamming distance between the start delimiter bit pattern and any shift of the preamble bit pattern always greater than one?

Are there parts of a message where an error could cause the loss of more than one message?

Other than redundancy bits (e.g., error detection or correction fields), is the message format efficient?

C.4.4 ERROR DETECTION.

Does the network meet the required integrity values (undetected error probabilities) for the worst-case error pattern requirements (error bursts, maximum bit error rate)?

Does the network meet the availability numbers for the worst-case error probabilities and distributions?

Have the potential effects due to encoding of the data on the physical layer and its implications to the coverage of error detection been quantified?

C.5 NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

C.5.1 NETWORK VULNERABILITY TO ADDRESSING INFORMATION FAILURE.

Does the network technology use message addressing or message identification fields?

Does the network technology implement mechanisms to detect or mitigate the corruption of message address or message identification fields?

Has the fault coverage of these detection and mechanism been established?

Are the message addressing or message identification fields vulnerable to host software corruption?

Does the network technology use tables to assist with message addressing and routing?

Does the network technology implement adequate checking mechanisms to ensure the run-time integrity of the routing tables?

Does the network technology build routing information at run time?

Are the algorithms and associated mechanisms used to build run-time routing tables vulnerable to corruption or run-time errors?

What network action causes the network routing tables to be rebuilt? Can erroneous node software or electronic and electric hardware invoke incorrect invocation the table building activity?

Is the network routing discovery time suitably bounded?

C.5.2 NETWORK VULNERABILITY TO FLOW FAILURE.

What is the network addressing scheme's impact on the flow regulation?

C.5.3 IMPACT OF INTERMEDIATE STAGES.

Does the network require intermediate stages?

Is the availability of the intermediate stage sufficient to fulfill network channel availability requirements?

Are network intermediate stages for different network channels independent?

Is intermediate-stage-to-intermediate-stage signaling required between independent network channels?

Does the intermediate-stage-to-intermediate-stage signal path introduce any fault propagation or common mode influence?

Do network intermediate stages incorporate sufficient fault detection and coverage? Can fail-stop, intermediate-stage behavior be justified?

Does the network technology rely on inline integrity checking mechanisms?

Can the network intermediate-stage action introduce failure modes that will defeat network frame- or integrity-checking logic?

What is the intermediate-stage response to erroneous signals? Do intermediate stages ignore erroneous framing and clean and reshape erroneous data streams?

What is the intermediate-stage response to out-of-specification errors, i.e., elasticity exhaustion?

Does the network perform store-and-forward action?

Does the intermediate stage perform recalculation of integrity check sequences?

Is the intermediate-stage buffer memory suitably protected from transient upsets?

Does the protection mechanism simply detect or does it detect and absorb transient upsets?

Does the intermediate-stage response to transient upsets lower the availability of the intermediate-stage? That is, do transient errors force intermediate-stage resets and re-integration?

What mechanisms exist to detect erroneous message forwarding? That is, the forwarding of old messages or messages to incorrect addresses?

Can intermediate-stage errors affect higher-order redundancy management mechanisms to reduce overall network availability?

Can network intermediate-stage action introduce head-of-line blocking scenarios?

What network mechanisms exist to mitigate these effects?

Can blabbing or other erroneous node action impact intermediate-stage performance, for example, result in buffer exhaustion?

Can intermediate-stage start-up and re-integration time be bounded with a faulty node present?

Can intermediate-stage start-up or re-integration be impeded by the erroneous action of intermediate stages on other channels?

C.5.4 NETWORK CONFIGURATION DATA.

Are network configuration data tables stored with sufficient integrity?

How are network configuration tables loaded?

What network mechanisms are used to ensure network configuration tables are not corrupted during loading?

Does network configuration loading use the same network data paths as normal traffic?

Are specialized load protocols used for performing network loading?
Does the network incorporate sufficient interlocks to prevent the inadvertent invocation of such download protocols?

Are network tables updated during live network operation?

How are network modes and network table versions agreed at run time?

Does the network incorporate maintenance or query protocols operating on top of the live network operation?

Does the network incorporate sufficient interlocks to prevent the inadvertent invocation of such protocols?

C.5.5 START-UP AND RECOVERY.

Is network start-up time bounded?

Are the network start-up and integration mechanism fault tolerant?

Does the network start-up require coordinated power sequencing?

Is the required power sequencing action assured to the required network availability?

C.5.6 GLOBAL SYNCHRONIZATION.

Has the stability of the algorithm been analyzed under different environmental and expected fault conditions including stability after power-up of nodes and under all expected network configurations? Similarly, has the precision been analyzed and is it bounded under expected fault scenarios and operational conditions?

Have the effects of single dependencies of synchronization reference data been considered (faulty or no synchronization data)? Are such dependencies adequately mitigated for the required safety levels?

Have the effects of different synchronization data view (due to different propagation delay, data acquisition delays, etc.) been considered in the stability analysis?

Have the effects of merging of data from different network paths and potential differences due to different paths and propagation delays been considered in the algorithm stability analysis?

In algorithms using multiple clock sources, has the use or election of the source been considered under the assumed failure conditions and considering source coverage mechanisms?

Are mechanisms in place that adequately verify or support that the data used for synchronization does indeed stem from the assumed or elected source? That is, the synchronization algorithm is not vulnerable to masquerade faults?

Have effects of the clock correction been analyzed (such as task-time dependence on the clock synchronization, and the influence of the correction on the available (potentially decreased) execution time to tasks)?

C.5.7 FAULT DIAGNOSIS.

What is the source of the diagnosis information relied upon? What is the information's source integrity value and how is in agreement with the assumptions?

What is the influence of faulty relaying components, lightning, high-intensity radio frequency (HIRF), and other external effects on the diagnosis data and diagnosis algorithms? Have such influences been analyzed and quantified to have acceptable effects?

Does the diagnosis assume certain failure modes? Are the consistency and correctness guarantees of the diagnosis information (such as group membership) in agreement with the application's use? Are the assumptions and properties quantified to the level required?

Does the application use diagnosis information? Is the application's use of diagnosis information in compliance with guarantees and assumptions of the diagnosis information? Are there effects on the application layer?

Are the semantics of the group membership information (e.g., nodes operational or node not operational) in alignment with the application-level assumptions of its use; e.g., does the application assume correctness of the message even though the membership only indicates operation? Has the safe use of membership semantics been analyzed in "corner" cases like start-up and integration of systems?

Have temporal lags in error detection and diagnosis been quantified and found suitable?

Have start-up and integration been included in the analysis of diagnosis and group membership?

C.5.8 CLIENT EFFECT ON NETWORK OPERATIONS.

Is there an effect of network clients on the network operations? What is the assumed faulty behavior? In case a restricted failure model of clients is assumed, what substantiates the restricted faulty behavior?

Does the analysis of clients include start-up and integration scenarios and are they considered temporal effects of clients (such as potential long software response times)?

Does the network assume partitioning guarantees? What are the effects of clients with respect to partitioning? What has been done to substantiate any partitioning claims?

C.5.9 ACKNOWLEDGEMENT.

Has the acknowledgement scheme been analyzed assuming adequate failures modes (e.g., inconsistent message reception)? What are the effects of such failures? Have adverse effects been suitably compensated by other means?

Have effects of negative acknowledgements and message retries been analyzed during the performance analysis? Is the number of retries bounded and is this bounded number determined as a consequence the impact of retries on the performance?

What are the effects of acknowledgement errors on the application? Are effects suitably bounded (e.g., with respect to processor overhead)?

C.6 APPLICATION SERVICES.

C.6.1 HOST INTERFACE MANAGEMENT.

Has the buffer management been analyzed with respect to effects and system-level implications (e.g., combined priority and buffer management causing head-of-line blocking or priority inversions)?

In mixed criticality systems, are communications buffers reserved per partition, and are they assured not to be accessible for other software partitions in cases where there are partitioning requirements for software? Have potential masquerading effects due to buffer management been considered (e.g., software partitions masquerading as other partitions).

How is the access to the buffers controlled (control between different software partitions and control of access between network and host software access)? Has this been analyzed with respect to safety (effects of different criticality partitions)? Are system-level effects of blocking of buffer access been analyzed (e.g., increased software execution time, increased buffer size needs)?

Has the performance of the buffer been analyzed with respect to the needs of the communication (i.e., are the network speeds and the buffer-management speeds balanced)?

Are there relationships between network configuration flow and buffer management configuration? Is there an effect of changing network tables on the buffer management tables or mechanisms? Have such relationships been considered during the design, and what is done to ensure compatibility?

C.6.2 SUPPORT FOR APPLICATION-LAYER REDUNDANCY.

Has the application group membership service been analyzed with respect to its performance, availability, and integrity targets under adequate fault scenarios? How are failures in membership or similar services covered? What assumptions were made in determining this coverage? How are these assumptions verified?

Are the assumptions of correctness and completeness of the application-level, group membership service in agreement with the system-level assumptions? Has the effect of the lag in update of the application been analyzed in the system context?

Do the services provided conform to the requirements from the application?

How is the synchronization ensured between different replicants (that are to be voted or agreed)?

Any voting or selection scheme may lead to different results at different observers for certain failure modes (Byzantine or local nonreception of a single message). Has the impact of different results been taken into account in the safety analysis?

Are there other mechanisms that may influence the selection logic? Are such mechanisms analyzed with respect to potential unwanted interaction?

Is the network implementation vulnerable to masquerading faults resulting in potential defeat of redundancy?

Are passive replication strategies analyzed with respect to control handover in failure cases? How is the state of faulty replica signaled to the application?

Do the two halves of a self-checking pair communication interface compare their two copies of received data before it is allowed to be used for computation? Is this comparison designed to be immune to Byzantine faults? Are the halves of a self-checking pair suitably independent (e.g., independent power, separate memory, or memory sections)? Is the design of self-checking pairs backed by an assurance process?

C.6.3 TIME SERVICE FOR TIME STAMPING AND TIME INTERRUPT.

Is the time service robust with respect to potential implications such as effect of clock differences? Is the use of time stamping and time-interrupt services adequately mitigated or included in a robustness analysis of application data algorithms?

C.7 FAULT-TOLERANCE MECHANISMS.

C.7.1 TOPOLOGICAL FAULT-TOLERANCE.

Is the network vulnerable to spatial proximity faults? How is the tolerance to spatial impact faults guaranteed? Is the network availability assured even in case of spatial proximity faults?

Is adequate communication path availability ensured despite faulty end equipment (babbling devices or short circuits)?

Have common network resources (such as switches) been placed at adequate distances from each other so as not to be vulnerable, but to be independent?

C.7.2 GUARDIAN SCHEMES.

Does the network deploy guardians? What is the intent of the guardians? What coverage is the guardian assumed to provide (what failure modes can the guardian detect or contain)? What is done to substantiate the coverage claims? Does the claimed coverage of failure modes consider boundary system conditions such as start-up or integration?

What is done to assure independence of the guardian (power, time, logical dependence, physical dependence)?

Have potential side effects of the guardian behavior been considered (especially central guardians)? Is there an effect on inline coverage due to guardians?

Have tolerance margins (different oscillators) due to difference between the guardian and the guarded device been established and quantified?

C.7.3 PROTOCOL LOGIC FAULT-TOLERANCE.

Does the network protocol logic rely on information from single sources? Or, is protocol logic dependent on agreeing data from different sources? Does protocol dependence rely on different sources to increase the robustness of algorithms to potential node failures?

Are the integrity and availability levels of the protocol logic met by the network?

C.7.4 LOCAL TRANSMISSION MONITORING AND SELF-CHECKING SCHEMES.

Does the network technology incorporate local health monitoring schemes to detect node health?

Are the network monitoring schemes suitably independent?

Are network monitoring schemes vulnerable to faulty status sent by erroneous nodes?

Does the network technology use local transmission wrap back?

Has the transmission wrap back been analyzed for Byzantine failure vulnerability?

Does the network incorporate self-checking pair configurations?

Has the coverage and independence of the self-checking configuration been justified?

C.7.5 RECONFIGURATION AND DEGRADED OPERATION.

Does the provided network technology degrade modes of operation?

Is the network performance under degraded mode sufficient to meet network functional requirements?

Is network degraded-mode operation suitably annunciated to network clients?

Does the network perform dynamic reconfiguration of routing to mitigate bad network paths or nodes?

Is the reconfiguration time suitably bounded?

C.7.6 LATENT FAILURE DETECTION.

Does the network technology support mechanism for latent fault detection?

Is the coverage of the network latent fault detection suitable to establish the health of all critical network protection functions?

Are suitable test activation interlocks incorporated into the network technology to inadvertent test mode activation?

C.7.7 VOTING, SELECTION, OR AGREEMENT SERVICES AND REDUNDANCY MANAGEMENT.

Does the network support selection and agreement services?

Are such agreement services targeted at integrity or availability?

Is the overhead for agreement function evaluated and justified to be acceptable? Have temporal effects been evaluated in the selection?

C.7.8 BYZANTINE FAULT-TOLERANCE.

Has the network been analyzed with respect to Byzantine fault tolerance?

Does the network claim to be Byzantine fault tolerant?

What are the effects of Byzantine fault tolerance on integrity and availability?

Has the Byzantine fault-tolerant algorithm been suitable analyzed and coverage been justified (Byzantine fault-containment properties)?

Has the Byzantine filtering coverage been justified?

C.8 DESIGN ASSURANCE.

C.8.1 DESIGN ASSURANCE PROCESSES.

Has the network technology been developed to be in compliance with aviation digital electronics design assurance guidelines, i.e., DO-178B and or DO-254?

Does the network technology's commercial-usage volume support COTS classification?

Is the network technology behavior simple enough to be fully testable?

C.8.2 AVAILABILITY OF STANDARDS AND CONFORMANCE EVIDENCE.

Is the network technology supported by an open specification?

Is the specification that is being standardized available to open-industrial committees?

Is the network specification complete? Does the network technology address all operational modes of the network, including erroneous node action and associated fault recovery actions?

Is the network specification sufficiently detailed to address all required protocol action?

Does the networking technology have published conformance test criteria and campaigns?

Do the conformance test criteria cover all network protocol behaviors, including fault detection and recovery actions?

Have the critical protocol mechanisms and algorithms of the network technology been formally verified?

Have the assumptions underpinning the formal verifications been reviewed to ensure that they are consistent with real-world targeted environment?

Are protocol mechanisms and associated formal proofs composable? That is, do protocol mechanisms and associated proofs stand by themselves or are they interrelated?

Has the network technology been subjected to other validation activities?

Did the fault injection campaign include suitably sufficient visibility to observe the key behaviors of all important network mechanisms?

Have anomalies and fault observations from such activities been adequately mitigated?

C.8.3 DESIGN MARGIN.

Has the network design margin been established for worst-case component behaviors?

Have all contributions to network design margins been identified?

C.8.4 CONFIGURATION TABLE CORRECTNESS AND PERFORMANCE JUSTIFICATION.

Has the criteria for correct network parameterization and configuration been established?

Are the network configuration criteria traceable to network function behavior or top-level requirements and specification?

Does tooling assist network configuration and verification?

Is the tooling qualified in accordance with the tool guidance established in DO-178B?

Have procedures and criteria to bound the worst-case performance of the network been established?

Do the worst-case performance criteria address detailed MAC sublayer interactions?

Do the worst-case performance criteria address worst-case fault-detection and reconfiguration actions?

Does the network technology provide automated tools to assist worst-case performance calculation?

Are intermediate-stage buffers and end-node queues adequately sized?

Are tools that provide performance bounding qualified in accordance with the tooling guidelines of DO-178B?

Do the network technology and associated tooling accommodate incremental change management?

C.8.5 NETWORK MONITORING AND TEST EQUIPMENT.

Is test equipment available for the network technology?

Does the test equipment facilitate the observation of all network operational modes?

Does the network test equipment facilitate sufficient fault injection to exercise sufficient network fault-detection mechanisms?

Does the network test equipment support observation modes with sufficient noninterference guarantees to support flight testing?

How many network test access points are required to monitor the entire holistic network behavior?

C.9 SECURITY.

Does the network technology use open COTS protocols?

Does the network technology have known security issues?

Does the network technology support sufficient secure services for user and application authentication?

Does the network technology support secured data transmission mechanism?

Does the network technology require specialized security augmentations, e.g., firewalls?

Does the network support multilevel security? How many levels or security domains does the network technology support?

Is network configuration data protected and secured during deployment and during load?

APPENDIX D—CONDENSED EVALUATION CRITERIA

D.1 PHYSICAL LAYER.

Criterion 1 Environment: Do the network specification(s) and available components allow for the creation of a network that meets the applicable aerospace environment requirements of DO-160 or other imposed environmental requirements?

Criterion 2 Line-Level Encoding: Was the electromagnetic compatibility testing (e.g., DO-160 Sections 19-22) done with the actual data line encoding (e.g., Manchester, 8B/10B), worst-case network data, worst-case message sizes, and worst-case message repetition rate(s)?

Criterion 3 Probability of Bit Errors: Was the bit error rate determined under worst-case conditions?

Has an upper bound on anticipated bit error rate been established via testing using the worst-case anticipated signal path characteristics (e.g., impedances and impedance discontinuities), environment (e.g., DO-160), local clock and clock recovery characteristics (e.g., drift, jitter), sampling margin (worst-case eye pattern), encoding schemes, and data patterns? If the network uses signal regeneration without elasticity buffers, the worst-case accumulated jitter must be included in this determination. If the data network exploits the mixing of dominant and recessive signals on its media to perform some logic function, have the receivers been designed to tolerate Wired–Or glitches or have suitable design rules been created that limit the duration of the glitches to a duration that can be tolerated?

Criterion 4 Probability of Electrical Component Failures: Do the data network's electronic components have established hardware failure rates (permanent and transient) and characteristics so that avionics designers can do the required failure modes and effects analysis (FMEA) and fault tree calculations?

Are signal margins robust enough to handle aging effects on connectors, media, and drivers? That is, could aging cause a network to not meet the aerospace environment requirements of DO-160 or other imposed environmental requirements at some point in the avionics system's lifetime? Has a valid synchronizer metastability error rate test been done for every synchronizer design in the data network's electronics and has an acceptable ceiling on the error rate been established?

Criterion 5 Electrical Isolation Properties: Is there sufficient protection against electrical fault propagation?

Does the data network's physical layer allow for electrical (galvanic) isolation among redundancies? If a fault in a node causes the highest voltage in that node to appear on one side of this isolation barrier, can the isolation prevent damage on the other side of the barrier? If an application of the network requires receive-only connections to the media in order to prevent

fault propagation, can it be shown (with sufficient assurance) that these receive-only connections prevent fault propagation to the media?

Criterion 6 Physical Composability: Does the data network have characteristics or design rules that will guarantee that it will reliably work with any number of nodes up to some explicitly given maximum number of nodes?

For a data network that has the freedom to assume a number of different topologies or topology variations, does the network have characteristics or design rules that will guarantee that it will reliably work with all possible variations that are not precluded by its design rules (including sufficient design margin)?

D.2 DATA LINK LAYER.

Criterion 7 Media Access Control (MAC): The Media Access Control sublayer protocol must provide appropriately small bounds on message delivery times regardless of likely faults.

Can the behavior of one or more network clients increase latency and jitter beyond the desired bound for other network clients? If collision is used as a protocol element, are there bounds on delays incurred due to a collision? Can misbehavior of one network client disrupt more than one or two other network clients? Does the MAC sublayer protocol amplify small failures and errors into loss of MAC services? In particular, does the MAC sublayer protocol allow transient failures and errors to have an effect that persists longer than current transmissions? Does the MAC sublayer protocol specify a single physical point of failure, such as a dedicated protocol Master node? Does the MAC sublayer protocol have a single logical point of failure such as the current token holder crashing in a token-based protocol, duplicated tokens, or a babbling-idiot node asserting it has high-priority traffic to send? Does the MAC sublayer protocol have a bounded worst-case bandwidth at maximum network loading? If a mixed or hybrid MAC is used (e.g., Master and Slave polling on top of Ethernet hardware), are there conflicting properties within the portions of the hybrid MAC that could cause vulnerabilities?

Criterion 8 Message Formatting (Framing): Message framing must ensure that only complete, properly synchronized messages are accepted at clients, and that improper synchronization is recovered from in-bounded time.

Are there distinctive preamble or postamble bit patterns, including break characters, used to delimit messages? Is there sufficient Hamming Distance and bit-slip tolerance present to prevent an ordinary data pattern from being interpreted as a message preamble under expected likely error conditions? Do receivers tolerate any preamble bit(s) being erroneous? Do receivers tolerate any preamble bit(s) other than the one(s) that would make the preamble look like a start delimiter? Is the Hamming distance between the start delimiter bit pattern and any shift of the preamble bit pattern always greater than one? Is the data network's framing structure brittle (i.e., do simple errors cause a node to lose more than one message per error)? Are there parts of a message where an error could cause the loss of more than one message? Are there two

independent checks on whether expected and actual frame length match (e.g., both a length field and an unambiguous end delimiter; distinctive start and end delimiters)?

Criterion 9 Error Detection: The data link layer must provide sufficient guarantees of error-free message delivery.

Does the network meet the required integrity values (undetected error probabilities, Hamming Distance) for the worst-case error pattern requirements (error bursts, maximum bit error rate, temporary blackouts)? Does the network deliver valid messages within bounded latency with sufficient probability despite expected error rates? Does the network meet availability requirements for the worst-case error probabilities and distributions? Have the potential effects due to encoding of the data on the physical layer and its implications to the coverage of error detection been quantified? For example, have corrupted bit-stuffing formats been accounted for in error analysis? Are messages vulnerable to corruption of length fields that cause receiving clients to use the incorrect frame location for frame check sequence (FCS) fields? If it is a required service of the network, can receivers of data be certain of the sender's identity even in the presence of faults?

D.3 NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

Criterion 10 Network Vulnerability to Addressing Information Failure: Mechanisms should ensure correct forwarding, routing, or conversion failures despite likely failure scenarios of network components.

Does the network technology use message addressing or message identification fields? Does the network technology implement mechanisms to detect or mitigate the corruption of message address or message identification fields? Has the fault coverage of these detection and mechanism been established? Are the message addressing or message identification fields vulnerable to host software corruption? Does the network technology use tables to assist with message addressing and routing? Does the network technology implement adequate checking mechanisms to ensure the run-time integrity of the routing tables? Does the network technology build routing information at run time? Are the algorithms and associated mechanisms used to build run-time routing tables vulnerable to corruption or run-time errors? What network action causes the network routing tables to be rebuilt? Can erroneous node software or electronic and electric hardware invoke incorrect invocation the table building activity? Is the network routing discovery time suitably bounded?

Criterion 11 Impact of Intermediate Stages: Intermediate stages (e.g., repeaters, gateways, routers, and switches) must guarantee sufficient availability and integrity requirements, as well as sufficient logical and physical independence from other replicated intermediate stages, to ensure correct operation in adequate failure scenarios.

If the network uses intermediate stages, is the availability of the intermediate stage sufficient to fulfill network channel availability requirements? Are network intermediate stages for different network channels independent? Is intermediate-stage-to-intermediate-stage signaling required between independent network channels? Does the intermediate-stage-to-intermediate-stage

signal path introduce any fault propagation or common mode influence? Do network intermediate stages incorporate sufficient fault detection and coverage? Can fail-stop, intermediate-stage behavior be justified? Does the network technology rely on inline integrity checking mechanisms? Can the network intermediate-stage action introduce failure modes that will defeat network frame- or integrity-checking logic? What is the intermediate-stage response to erroneous signals? Do intermediate stages ignore erroneous framing and clean up and reshape erroneous data streams? What is the intermediate-stage response to out-of-specification errors, i.e., elasticity exhaustion? Does the network perform store-and-forward action? Does the intermediate stage perform recalculation of integrity check sequences? Is the intermediate-stage buffer memory suitably protected from transient upsets? Does the protection mechanism simply detect or does it detect and absorb transient upsets? Does the intermediate-stage response to transient upsets lower the availability of the intermediate stage? That is, do transient errors force intermediate-stage resets and reintegration? What mechanisms exist to detect erroneous message forwarding, i.e., the forwarding of old messages or messages to incorrect addresses? Can intermediate-stage errors affect higher-order redundancy management mechanisms to reduce overall network availability? Can network intermediate-stage action introduce head-of-line blocking? What network mechanisms exist to mitigate these effects? Can blabbing or other erroneous node action impact intermediate-stage performance, for example, result in buffer exhaustion? Can intermediate-stage start-up and reintegration time be bounded with a faulty node present? Can intermediate-stage start-up or reintegration time be impeded by the erroneous action of intermediate stages on other channels?

Criterion 12 Network Configuration Data: Network topology and component configuration data must be correct with respect to the applications' fault tolerance and performance requirements considering table production, loading, errors during operation, run-time environmental effects (e.g., impact of radiation), and maintenance actions.

Are network configuration data tables stored with sufficient integrity? How are network configuration tables loaded? What network mechanisms are used to ensure network configuration tables are not corrupted during loading? Does network configuration loading use the same network data paths as normal traffic? Are specialized load protocols used for performing network loading? Does the network incorporate sufficient interlocks to prevent the inadvertent invocation of such download protocols? Are network tables updated during live network operation? How are network modes and network table versions agreed at run time? Does the network incorporate maintenance or query protocols operating on top of the live network operation? Does the network incorporate sufficient interlocks to prevent the inadvertent invocation of such protocols?

Criterion 13 Start-up and Recovery: Component and network integration, start-up, and recovery must be performed in bounded time considering dependability requirements, environmental and deployment constraints, interactions with different systems and applications (such as application-level timing impact and power architecture influence).

Is network start-up time bounded even in case of fault scenarios? Are the network start-up and integration mechanisms fault tolerant? Are the assumed faults consistent with the coverage and FMEA declarations? If network start-up requires coordinated power sequencing, is the required

power-sequencing action assured to the required network availability? Is the start-up dependent on single components? What are the effects of such dependence considering failure scenarios?

Criterion 14 Global Synchronization: The synchronization and interaction constraints of systems must have timing bounds and dependencies for reasonable failure scenarios as well as the effects resulting from synchronization, such as perceptions.

Has the stability of the algorithm been analyzed under different environmental and expected fault conditions including stability after power-up of nodes and under all expected network configurations? Similarly, has the precision been analyzed and is it bounded under expected fault scenarios and operational conditions? Have the effects of single dependencies of synchronization reference data been considered (faulty or no synchronization data)? Are such dependencies adequately mitigated for the required safety levels? Have the effects of different synchronization data view (e.g., due to different propagation delay, data acquisition delays) been considered in the stability analysis? Have the effects of merging of data from different network paths and potential differences due to different paths and propagation delays been considered in the algorithm stability analysis? In algorithms using multiple clock sources, has the use or election of the source been considered under the assumed failure conditions and considering source coverage mechanisms? Are mechanisms in place that adequately verify or support that the data used for synchronization does indeed stem from the assumed or elected source? That is, is the synchronization algorithm invulnerable to masquerade faults? Have effects of the clock correction been analyzed such as task-time dependence on the clock synchronization and the influence of the correction on the available (potentially decreased) execution time to tasks?

Criterion 15 Fault Diagnosis: Any diagnosis, detection, or system-level agreement mechanisms must justify fault assumptions and consider effects of diagnosis action.

What is the source of the diagnosis information relied upon? What is the information's source integrity value and how is in agreement with the assumptions? What is the influence of faulty relaying components, lightning, high-intensity radio frequency, and other external effects on the diagnosis data and diagnosis algorithms? Have such influences been analyzed and quantified to have acceptable effects? Does the diagnosis assume certain failure modes? Are the consistency and correctness guarantees of the diagnosis information (such as group membership) in agreement with the application's use? Are the assumptions and properties quantified to the level required? Does the application use diagnosis information? Is the application's use of diagnosis information in compliance with guarantees and assumptions of the diagnosis information? Are there effects on the application layer? Are the semantics of the group membership information (e.g., nodes operational or node not operational) in alignment with the application-level assumptions of its use; e.g., does the application assume correctness of the message even though the membership only indicates operation? Has the coverage of the mechanisms used to establish a nodes health with respect to group membership signaling been evaluated against requirements? Has the safe use of membership semantics been analyzed in corner cases like start-up and integration of systems? Have temporal lags in error detection and diagnosis been quantified and found suitable? Have start-up and integration been included in the analysis of diagnosis and group membership?

Criterion 16 Client Effect on Network Operations: Can a network client adversely effect network operations?

What is the assumed faulty behavior? In case a restricted failure model of clients is assumed, what substantiates the restricted faulty behavior? Does the analysis of clients include start-up and integration scenarios and are they considered temporal effects of clients (such as potential long software response times)? Do client-side actions or data impact network addressing (i.e., message identifiers and labels and addresses are written under client control)? Does the network allow client impact of protocol-level control flow, such as mode changes? What network mechanisms are in place to ensure that such actions or data do not endanger the operation of the network?

Criterion 17 Acknowledgement: Does the acknowledgement mechanism work adequately under all fault scenarios?

If the network provides acknowledgement service(s), what are the guarantees of this service? What are the impacts of the acknowledgement mechanisms? Are they fault tolerant? Has the acknowledgement scheme been analyzed assuming adequate failures modes (e.g., inconsistent or Byzantine message reception)? What are the effects of such failures? Have adverse effects been suitably compensated by other means? Have effects of negative acknowledgements and message retries been analyzed during the performance analysis? Is the number of retries bounded, and is this bounded number determined as a consequence of the impact of retries on the performance? What are the effects of acknowledgement errors on the application? Are effects suitably bounded (e.g., with respect to processor overhead)?

D.4 APPLICATION-LAYER SERVICES.

Criterion 18 Host Interface Management: The protocol's interface to the host application must provide promised prioritization, latency, and loss prevention of messages for supported categories of service.

Has the buffer management been analyzed with respect to effects and system-level implications (e.g., combined priority and buffer management causing head-of-line blocking or priority inversions)? In mixed criticality systems, are communications buffers reserved per partition and are they assured to be inaccessible for other software partitions in cases where there are partitioning requirements for software? Have potential masquerading effects due to buffer management been considered (e.g., software partitions masquerading as other partitions)? How is the access to the buffers controlled (i.e., control between different software partitions and control of access between the network and host software access)? Has this access been analyzed with respect to safety (effects of different criticality partitions)? Are system-level effects of blocking of buffer access been analyzed (e.g., increased software execution time, increased buffer size needs)? Has the performance of the buffer been analyzed with respect to the needs of the communication (i.e., is the network speed and the buffer-management speed balanced)? Are there relationships between network configuration flow and buffer management configuration? Is there an effect of changing network tables on the buffer management tables or mechanisms?

Have such relationships been considered during the design, and what is done to ensure compatibility?

Criterion 19 Support for Application-Layer Redundancy: Any support for application-layer redundancy must fully support stated redundancy management mechanisms for a specified fault model.

Has the application-layer, group-membership service been analyzed with respect to its performance, availability, and integrity targets under adequate fault scenarios? How are failures in membership or similar services covered? What assumptions were made in determining this coverage? How are these assumptions verified? Are the assumptions of correctness and completeness of the application-layer, group membership service in agreement with the system-level assumptions? Has the effect of the lag in update of the application been analyzed in the system context? Do the services provided conform to the requirements from the application? How is the synchronization ensured between different replicants (that are to be voted or agreed)? Any voting or selection scheme may lead to different results at different observers for certain failure modes (Byzantine or local nonreception of a single message); has the impact of different results been taken into account in the safety analysis? Are there other mechanisms that may influence the selection logic? Are such mechanisms analyzed with respect to potential unwanted interaction? Is the network implementation vulnerable to masquerading faults resulting in potential defeat of redundancy? Are passive replication strategies analyzed with respect to control handover in failure cases? How is the state of faulty replica signaled to the application? Do the two halves of a self-checking pair communication interface compare their two copies of received data before it is allowed to be used for computation? Is this comparison designed to be immune to Byzantine faults? Are the halves of a self-checking pair suitably independent (e.g., independent power, separate memory or memory sections)? Is the design of self-checking pairs backed by an assurance process?

Criterion 20 Robust Partitioning (ARINC 651): If the network is required to provide robust-partitioning guarantees, what has been done to substantiate any partitioning claims?

If required, does the network enforce robust partitioning among its clients, even if multiple clients share the same node? If required, does the network enforce robust partitioning among its partitions within its clients? Does the partitioning against software faults include vulnerabilities resulting from timing faults?

Criterion 21 Time Service for Time Stamping and Time Interrupt: If time stamping and time-interrupt services are provided, do they provide dependable and correct time services?

Is the time service robust with respect to potential implications such as effect of clock differences? Is the use of time stamping and time-interrupt services adequately mitigated or included in a robustness analysis of application data algorithms? If the protocol is based on a centralized time service, is failover for the time Master supported? If it is based on a distributed time service, is timekeeping maintained even for Byzantine clock faults?

D.5 FAULT-TOLERANCE MECHANISMS.

Criterion 22 Topological Fault Tolerance: Redundant network components must be physically separated and isolated to prevent correlated outages due to physical equipment damage and credible electrical faults.

Is the network vulnerable to spatial proximity faults, such as physical damage that is within a specified distance limit? If a single piece of equipment is faulty, can its faults propagate to take down the entire network (e.g., can a single fault cause babbling behavior simultaneously on redundant network paths)? Is adequate communication path availability ensured despite faulty end equipment (babbling devices or short circuits)? Are common network resources (such as switches) been placed at adequate distances from each other so as not to be vulnerable, but to be independent? Are redundant resources attached to different power sources within the system?

Criterion 23 Guardian Schemes: Some technique, such as network guardians, must be used to ensure that a single-point client failure will not take down the network.

Does the network deploy guardians? What is the intent of the guardians? What coverage is the guardian assumed to provide (e.g., what failure modes can the guardian detect or contain)? What is done to substantiate the coverage claims? Does the claimed coverage of failure modes consider boundary system conditions such as start-up or integration? What is done to assure independence of the guardian (power, time, logical dependence, physical dependence)? Have potential side effects of the guardian behavior been considered (especially central guardians)? Is there an effect on inline coverage due to guardians? Have tolerance margins (different oscillators) due to difference between the guardian and the guarded device been established and quantified?

Criterion 24 Protocol Logic Fault Tolerance: The protocol must ensure that errors in protocol logic and protocol state do not result in unacceptable reduction of integrity and availability.

Can mismatches in protocol logic and state occur during protocol start-up? If such mismatches occur, is there a bound on the time until they are resolved? Does the network protocol logic rely on information from single sources? Or, is protocol logic dependent on agreeing data from different sources? Does protocol dependence rely on different sources to increase the robustness of algorithms to potential node failures? Are the integrity and availability levels of the protocol logic met by the network?

Criterion 25 Protocol Transmission Monitoring and Self-Checking Schemes: The protocol must reliably detect transient and permanent faults in both nodes and message transmissions.

Does the network incorporate local health monitoring schemes to detect node health? Are the network monitoring schemes suitably independent? Are network monitoring schemes vulnerable to faulty status sent by erroneous nodes? Does the network technology use local transmission wrap-back? Have transmission wrap-back been analyzed for Byzantine failure vulnerability?

Does the network incorporate self-checking pair configurations? Has the coverage and independence of the self-checking configuration been justified?

Criterion 26 Reconfiguration and Degraded Operation: Capabilities provided in the presence of a specified number and type of faults must be sufficient to meet operational requirements.

Can a faulty node cause good nodes to be evicted from system configuration or otherwise cause degradation into a mode that worsens the effects of the fault? Does the network technology provide degraded modes of operation? Is the network performance under degraded mode sufficient to meet network functional requirements? Is network degraded mode operation suitably annunciated to network clients? Does the network perform dynamic reconfiguration of routing to mitigate bad network paths or nodes? Is the reconfiguration time suitably bounded? If online reintegration is supported, what mechanisms are in place to cope with intermittent failures and ensure the health of nodes to be reintegrated?

Criterion 27 Latent Failure Detection: Latent faults must not result in network failure.

Can an accumulation of latent faults overwhelm a network's ability to tolerate faults? Can a latent fault in a failure detection, isolation, and recovery (FDIR) mechanism (e.g., a stuck-at-good fault detector) lead to network failure due to lack of coverage? Does the network technology support a mechanism for latent fault detection, especially in mismatched protocol state among network clients? If state variables are maintained by the protocol at each client, can multiple unrelated state variables be corrupted before the first corruption is detected? Is the coverage of the network latent fault detection suitable to establish the health of all critical network protection functions? Are suitable test activation interlocks incorporated into the network technology to inadvertent test mode activation?

Criterion 28 Voting, Selection, or Agreement Services and Redundancy Management: The network protocol must support the ability to determine which nodes are part of the active network quorum.

Does the network support selection and agreement services directly? Does the network provide adequate mechanisms for application software to implement selection and agreement services? Are such agreement services targeted at integrity or availability? Is the overhead for agreement function evaluated and justified to be acceptable? Have temporal effects been evaluated in the selection? Are the time constants of tracking changes in membership fast enough for the application? Are assumptions made by the membership and agreement service justifiable?

Criterion 29 Fault Model: The protocol must be evaluated with respect to a precisely and completely stated fault model, and the fault model must be compatible with that of the system architecture.

Does the network claim to be Byzantine fault tolerant? Has the network been analyzed with respect to Byzantine fault tolerance? What are the effects of Byzantine fault tolerance on integrity and availability? If a hybrid fault model is used, is there justification for the relative

rates of occurrence of different classes of faults (e.g., Byzantine, strictly omissive, symmetric) for different failure scenarios. Has the Byzantine fault-tolerant algorithm been suitably analyzed and coverage been justified (Byzantine fault-containment properties)? Has a suitable Byzantine filtering coverage bound been verified? Has the fault model been analyzed with respect to influences to all services even the ones that are possibly not used (this should exclude any effects of services not used)?

D.6 DESIGN ASSURANCE.

Criterion 30 Design Assurance Processes: Have appropriate design assurance processes been followed for design and deployment of the network? (The following is a suggested alternative wording for this criterion. But it has the potential for becoming a circular dependency.) **The hardware and software design should be demonstrated to satisfy published regulatory guidelines.**

Has the network technology been developed to be in compliance with avionics design assurance guidelines, i.e., DO-178B and/or DO-254? Does the network technology's commercial-usage volume support commercial off-the-shelf (COTS) classification? If COTS classification is used for classification, has the COTS product been deployed in similar applications to justify suitable coverage for correctness claims? Is the network technology behavior simple enough to be fully testable? Do diverse approaches (e.g., design diversity) provide adequate and quantifiable coverage if credit is taken for such diversity?

Criterion 31 Availability of Standards and Conformance Evidence: The technology and protocol of the network should be clearly specified and be analyzable.

The use of open specifications and standardization might assist a certification authority in establishing the acceptability of a network. Credit for analysis of specification properties and interoperability between different networks should be supported by conformance and interoperability testing. The use of formal methods to demonstrate protocol design correctness should be proposed and accepted by the certification authorities. Is the network technology supported by an open specification? Is the specification that is being standardized available to open-industrial committees? Is the network specification complete? Does the network technology address all operational modes of the network including erroneous node action and associated fault recovery actions? Is the network specification sufficiently detailed to address all required protocol action? Does the networking technology have published conformance test criteria and campaigns? Do the conformance test criteria cover all network protocol behaviors, including fault detection and recovery actions? Have the critical protocol mechanisms and algorithms of the network technology been formally verified? Have the assumptions underpinning the formal verifications been reviewed to ensure that they are consistent with real-world targeted environment? Are protocol mechanisms and associated formal proofs composable? That is, do protocol mechanisms and associated proofs stand by themselves or are they interrelated? Has the network technology been subjected to other validation activities? Did the fault-injection campaign include suitably sufficient visibility to observe the key behaviors of all important network mechanisms? Have anomalies and fault observations from such activities been adequately mitigated?

Criterion 32 Design Margin: Required design margins should be supported by reviewable evidence.

Has the network design margin been established for worst-case component behaviors? Have all contributions to network design margins been identified?

Criterion 33 Configuration Table Correctness and Performance Justification: Justification for configuration table correctness and performance justification should be provided.

Has the criteria for correct network parameterization and configuration been established? Are the network configuration criteria traceable to network function behavior or top-level requirements and specification? Does tooling assist network configuration and verification? Is the tooling qualified in accordance with the tool guidance established in DO-178B? Have procedures and criteria to bound the worst-case performance of the network been established? Do the worst-case performance criteria address detailed MAC sublayer interactions? Do the worst-case performance criteria address worst-case fault detection and reconfiguration actions? Does the network technology provide automated tools to assist worst-case performance calculation? Are intermediate-stage buffers and end-node queues adequately sized? Are tools providing performance bounding qualified in accordance with the tooling guidelines of DO-178B? Does the network technology and associated tooling accommodate incremental change management?

Criterion 34 Network Monitoring and Test Equipment: Do adequate test equipment, network access points, mechanisms, and procedures exist to ensure that the network is configured correctly and operating correctly (including meeting its specified behavior in the presence of any faults)?

The use of network monitoring and test equipment to establish certification credit should not invalidate the data being observed and should be demonstrated to perform in accordance with the operational requirements for the features being used. Is test equipment available for the network technology? Does the test equipment facilitate the observation of all network operational modes? Does the network test equipment facilitate sufficient fault injection to exercise sufficient network fault-detection mechanisms? Does the network test equipment support observation modes with sufficient noninterference guarantees to support flight testing? How many network test access points are required to monitor the entire holistic network behavior? Is this feasible for achieving flight test requirements? Is monitoring and test equipment assured to be noninterfering during operation? Is it guaranteed that the network behavior does not change if monitoring is not performed and correct behavior is inferred from monitoring or testing?

D.7 SECURITY.

Criterion 35: Can security weaknesses adversely affect network dependability (safety)?

Does the network technology use open COTS protocols? Does the network technology have known security issues? Does the network technology require specialized security augmentations, e.g., firewalls?

Criterion 36: If needed, does the network deal adequately with the security issues of privacy (also known as confidentiality or secrecy), integrity, authentication, or authorization?

Does the network technology support sufficient secure services for user and application authentication? Does the network technology support secured-data transmission mechanism? Does the network support multilevel security? How many levels or security domains does the network technology support? Is network configuration data protected and secured during deployment and during load?

APPENDIX E—UPDATED EVALUATION CRITERIA

This appendix contains the condensed and updated version of the evaluation criteria. This version was developed after condensing the original list and then testing the condensed list by applying it against a number of networks. The lessons learned from that exercise resulted in this list, which has far fewer main criteria. Many of the independent questions from the original list were kept as subordinate questions in this list to be used as thought provokers as one thinks about how a particular network stacks up against each of the criteria in this list.

E.1 PHYSICAL LAYER.

Criterion 1 Environment: Do the network specification(s) and available components allow for the creation of a network that meets the applicable aerospace environment requirements of the most recent version of DO-160 or other imposed environmental requirements?

Criterion 2 Line-Level Encoding: Was the electromagnetic compatibility testing (e.g., DO-160 Sections 19-22) done with the actual data line encoding (e.g., Manchester, 8b/10b), worst-case network data, worst-case message sizes, worst-case bit rate, worst-case pulse widths, and worst-case message repetition rate(s)?

Does the worst-case message size take into account worst-case variable-length encoding, as encountered in bit-stuffed encoding?

Criterion 3 Probability of Bit Errors: Was the bit (or symbol) error rate determined under worst-case conditions expected to be encountered in the application environment?

Has an upper bound on anticipated bit (or symbol) error rate been established via testing using the worst-case anticipated signal path characteristics (e.g., impedances and impedance discontinuities), environment (e.g., DO-160), local clock and clock recovery characteristics (e.g., drift, jitter), sampling margin (worst-case eye pattern), encoding schemes, and data patterns? If the network uses signal regeneration without elasticity buffers, the worst-case accumulated jitter must be included in this determination. If the data network exploits the mixing of dominant and recessive signals on its media to perform some logic function, have the receivers been designed to tolerate Wired–Or glitches, or have suitable design rules been created that limit the duration of the glitches to a duration that can be tolerated?

Criterion 4 Probability of Electrical Component Failures: Do the data network's electronic components have established hardware failure rates (permanent and transient) and characteristics so that avionics designers can do the required failure modes and effects analysis (FMEA) and fault tree calculations and is the protocol defined well enough that one can determine the effects on the protocol by any and all component faulty behaviors?

Are signal margins robust enough to handle aging effects on connectors, media, and drivers? That is, could aging cause a network to not meet the aerospace environment requirements of DO-160 or other imposed environmental requirements at some point in the avionics system's

lifetime? Has a valid synchronizer metastability error rate test been done for every synchronizer design in the data network's electronics and has an acceptable ceiling on the error rate been established?

Criterion 5 Electrical Isolation Properties: Is there sufficient protection against electrical fault propagation?

Does the data network's physical layer allow for electrical (galvanic) isolation among redundancies? If a fault in a node causes the highest voltage in that node to appear on one side of this isolation barrier, can the isolation prevent damage on the other side of the barrier? If an application of the network requires receive-only connections to the media to prevent fault propagation, can it be shown (with sufficient assurance) that these receive-only connections prevent fault propagation to the media?

Criterion 6 Logical and Physical Composability: Does the data network have characteristics or design rules that will guarantee that it will reliably work with any size network, up to some explicitly given maximum size?

For a data network that has the freedom to assume a number of different topologies or topology variations, does the network have characteristics or design rules that will guarantee that it will reliably work with all possible variations that are not precluded by its design rules (including sufficient design margin)? What is the certification affect of changing the size of the network (e.g., number of members in a clock sync algorithm)? Do these design rules and characteristics include the effects of hubs, repeaters, or other devices that extend network propagation delay or electrical loading? For data networks in which bit rate is limited by network distance, are there design rules to ensure that a particular network distance supports a given bit rate under worst-case conditions?

E.2 DATA LINK LAYER.

Criterion 7 Media Access Control (MAC): The Media Access Control protocol must provide appropriately small bounds on message delivery times regardless of likely faults.

Can the behavior of one or more network clients increase latency and jitter beyond the desired bound for other network clients? If collision is used as a protocol element, are there bounds on delays incurred due to a collision? Can misbehavior of one network client disrupt more than one or two other network clients? Does the MAC sublayer protocol amplify small failures and errors into loss of MAC services? In particular, does the MAC sublayer protocol allow transient failures and errors to have an effect that persists longer than current transmissions? Does the MAC sublayer protocol specify a single physical point of failure, such as a dedicated protocol Master node? Does the MAC sublayer protocol have a single logical point of failure, such as the current token holder crashing in a token-based protocol, duplicated tokens, or a "babbling idiot" node asserting it has high-priority traffic to send? Does the MAC sublayer protocol have a bounded worst-case bandwidth at maximum network loading? If a mixed or hybrid MAC is used (e.g., Master and Slave polling on top of Ethernet hardware), are there conflicting properties

within the portions of the hybrid MAC that could cause vulnerabilities? Does the MAC affect messaging delivery ordering?

Criterion 8 Message Formatting (Framing): Message framing must ensure that only complete, properly synchronized messages are accepted at clients, and that improper synchronization is recovered from in bounded time.

Are there distinctive preamble or postamble bit patterns, including "break" characters, used to delimit messages? Is there sufficient Hamming distance and bit slip tolerance present to prevent an ordinary data pattern from being interpreted as a message preamble under expected likely error conditions? Do receivers tolerate any preamble bit(s) being erroneous? Do receivers tolerate any preamble bit(s) other than the one(s) that would make the preamble look like a start delimiter? Is the Hamming distance between the start delimiter bit pattern and any shift of the preamble bit pattern always greater than one? Is the data network's framing structure brittle (i.e., do simple errors cause a node to lose more than one message per error)? Are there parts of a message where an error could cause the loss of more than one message? Are there two independent checks on whether expected and actual frame length match (e.g., both a length field and an unambiguous end delimiter; distinctive start and end delimiters)? If this is an implicit token protocol; e.g., reservation carrier sense multiple access (CSMA), or mini-slot system, or time-sliced protocol; e.g., time division multiple access (TDMA), is there sufficient information in the message to validate that the time position of the message is interpreted correctly, avoiding incorrect interpretation of the message due to timing inaccuracies?

Criterion 9 Error Detection: The Data Link Layer must provide sufficient guarantees of message delivery free of undetected errors, using error detection mechanisms with coverages that have been calculated correctly.

Does the network meet the required integrity values (undetected error probabilities, Hamming distance) for the worst-case error pattern requirements (error bursts, maximum bit error rate, and temporary blackouts)? Does the network deliver valid messages within bounded latency with sufficient probability despite expected error rates? Does the network meet availability requirements for the worst-case error probabilities and distributions? Have the potential effects due to encoding of the data on the physical layer and its implications to the coverage of error detection been quantified? For example, have corrupted bit stuffing formats been accounted for in error analysis? Are messages vulnerable to corruption of length fields which cause receiving clients to use the incorrect frame location for frame check sequence (FCS) fields? If it is a required service of the network, can receivers of data be certain of the sender's identity, even in the presence of faults? If messages use FCS hidden data (data that is included in the FCS calculation but not transmitted as part of the message), has there been an accounting for the loss of FCS coverage?

E.3 NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

Criterion 10 Network Vulnerability to Addressing Information Failure: Mechanisms should ensure correct forwarding, routing, or conversion failures despite likely failure scenarios of network components.

Does the network technology use message addressing or message identification fields? Does the network technology implement mechanisms to detect or mitigate the corruption of message address or message identification fields? Has the fault coverage of these detections and mechanisms been established? Are the message addressing or message identification fields vulnerable to host software corruption? Does the network technology use tables to assist with message addressing and routing? Does the network technology implement adequate checking mechanisms to ensure the run-time integrity of the routing tables? Does the network technology build routing information at run time? Are the algorithms and associated mechanisms used to build run-time routing tables vulnerable to corruption or run-time errors? What network action causes the network routing tables to be rebuilt? Can erroneous node software or electronic and electric hardware invoke incorrect invocation the table building activity? Is the network routing discovery time suitably bounded? Can addressing or routing errors cause lost packets or fragments to circulate on the media and inordinately consume needed bandwidth? If multiple layers of addressing are used, such as MAC and internet protocol (IP), then errors and masquerading faults in all levels of addressing must be considered?

Criterion 11 Impact of Intermediate Stages: Intermediate stages (e.g., repeaters, gateways, routers, and switches) must guarantee sufficient availability and integrity, as well as sufficient logical and physical independence from other replicated intermediate stages to ensure correct operation.

If the network uses intermediate stages, is the availability of the intermediate stage sufficient to fulfill network channel availability requirements? Are network intermediate stages for different network channels independent? Is intermediate-stage-to-intermediate-stage signaling required between independent network channels? Does the intermediate-stage-to-intermediate-stage signal path introduce any fault propagation or common mode influence? Do network intermediate stages incorporate sufficient fault detection and coverage? Can fail-stop intermediate-stage behavior be justified? Does the network technology rely on inline integrity checking mechanisms? Can the network intermediate-stage action introduce failure modes that will defeat network frame- or integrity-checking logic? What is the intermediate-stage response to erroneous signals? Do intermediate stages ignore erroneous framing and clean up and reshape erroneous data streams? What is the intermediate-stage response to out-of-specification errors, i.e., elasticity exhaustion? Does the network perform store-and-forward action? Does the intermediate stage perform recalculation of integrity check sequences? Is the intermediate-stage buffer memory suitably protected from transient upsets? Does the protection mechanism simply detect, or does it detect and absorb transient upsets? Does the intermediate-stage response to transient upsets lower the availability of the intermediate stage? That is, do transient errors force intermediate-stage resets and reintegration? What mechanisms exist to detect erroneous message forwarding, i.e., the forwarding of old messages or messages to incorrect addresses? Can intermediate-stage errors affect higher-order redundancy management mechanisms to reduce

overall network availability? Can network intermediate-stage action introduce head-of-line blocking? What network mechanisms exist to mitigate these effects? Can babbling or other erroneous node action impact intermediate-stage performance, for example, result in buffer exhaustion? Can intermediate-stage start-up and reintegration time be bounded with a faulty node present? Can intermediate-stage start-up or reintegration be impeded by the erroneous action of intermediate stages on other channels? Are the failure assumptions of the inter-stage justified by a complete FMEA? Is it possible for a faulty guardian to cause irrecoverable network failures?

Criterion 12 Network Configuration Data: Network topology and component configuration data must be commensurate with the applications' fault tolerance and performance requirements—considering table production, loading, errors during operation, run-time environmental effects (e.g., impact of radiation), and maintenance actions.

Are network configuration data tables stored with sufficient integrity? How are network configuration tables loaded? What network mechanisms are used to ensure network configuration tables are not corrupted during loading? Does network configuration loading use the same network data paths as normal traffic? Are specialized load protocols used for performing network loading? Does the network incorporate sufficient interlocks to prevent the inadvertent invocation of such download protocols? Are network tables updated during live network operation? How are network modes and network table versions agreed at run time? Does the network incorporate maintenance or query protocols operating on top of the live network operation? Does the network incorporate sufficient interlocks to prevent the inadvertent invocation of such protocols?

Criterion 13 Start-up and Recovery: Component and network integration, start-up, and recovery must be performed in bounded time considering dependability requirements, environmental and deployment constraints, interactions with different systems and applications (such as application-level timing impact and power architecture influence).

Is network start-up time bounded even in case of fault scenarios? Are the network start-up and integration mechanism fault tolerant? Are the assumed faults consistent with the coverage and FMEA declarations? If network start-up requires coordinated power sequencing, is the required power sequencing action assured to the required network availability? Is the start-up dependent on single components? What are the effects of such dependence considering failure scenarios? If host nodes are required to participate in network integration, start-up, or recovery, are those host node behaviors considered in the analysis? Is there a host or other node behavior within the protocol fault model that can cause repeated integration, start-up, or recovery events and thus lead to unbounded time to achieve normal network operation even if each integration, start-up, or recovery attempt completes individually within bounded time? If there is more than one statically designated network Master, is the leader election process guaranteed to converge within bounded time under worst-case assumptions and all faults within the specified fault model? If system start-up is inhibited in the presence of hardware failures (e.g., start-up is precluded with a network fault on one redundant bus), is the risk of system unavailability after an in-flight restart mitigated?

Criterion 14 Global Synchronization: If synchronization is required, have the synchronization mechanism(s) been shown to work correctly under all defined scenarios, including faults?

Has the stability of the algorithm been analyzed under different environmental and expected fault conditions including stability after power-up of nodes and under all expected network configurations? Similarly, has the precision been analyzed and is it bounded under expected fault scenarios and operational conditions? Have the effects of single dependencies of synchronization reference data been considered (faulty or no synchronization data)? Are such dependencies adequately mitigated for the required safety levels? Have the effects of different synchronization data view (e.g., due to different propagation delay, data acquisition delays, etc.) been considered in the stability analysis? Have the effects of merging of data from different network paths and potential differences due to different paths and propagation delays been considered in the algorithm stability analysis? In algorithms using multiple clock sources, has the use or election of the source been considered under the assumed failure conditions and considering source coverage mechanisms? Are mechanisms in place that adequately verify or support that the data used for synchronization does indeed stem from the assumed or elected source, i.e., the synchronization algorithm is not vulnerable to masquerade faults? Have effects of the clock correction been analyzed (such as task time dependence on the clock synchronization and the influence of the correction on the available (potentially decreased) execution time to tasks)? For asynchronous interfaces between two or more isochronous clock domains running at the same frequency, is there a mechanism to prevent pathological metastability (the metastability condition persists indefinitely due to the clocks not having any relative drift)?

Criterion 15 Fault Diagnosis: Any diagnosis, detection, or system-level agreement mechanisms must justify fault assumptions and consider effects of diagnosis action.

What is the source of the diagnosis information? What is the information's source integrity value and how is it in agreement with the assumptions? What is the influence of faulty relaying components, lightning, high-intensity radio frequency (HIRF), and other external effects on the diagnosis data and diagnosis algorithms? Have such influences been analyzed and quantified to have acceptable effects? Does the diagnosis assume certain failure modes? Are the consistency and correctness guarantees of the diagnosis information (such as group membership) in agreement with the application's use? Are the assumptions and properties quantified to the level required? Does the application use diagnosis information? Is the application's use of diagnosis information in compliance with guarantees and assumptions of the diagnosis information? Are there effects on the application level? Are the semantics of the group membership information (e.g., nodes operational or node not operational) in alignment with the application-level assumptions of its use? That is, does the application assume correctness of the message even though the membership only indicates operation? Has the coverage of the mechanisms used to establish node health with respect to group membership signaling been evaluated against requirements? Has the safe use of membership semantics been analyzed in "corner" cases like start-up and integration of systems? Have temporal lags in error detection and diagnosis been quantified and found suitable? Have start-up and integration been included in the analysis of diagnosis and group membership?

Criterion 16 Client Effect on Network Operations: Can a network client adversely effect network operations?

What is the assumed faulty behavior? In case a restricted failure model of clients is assumed, what substantiates the restricted faulty behavior? Does the analysis of clients include start-up and integration scenarios and are they considered temporal effects of clients (such as potential long software response times)? Do client-side actions or data impact network addressing (i.e., message identifiers (labels or addresses) are written under client control)? Does the network allow client impact of protocol-level control flow, such as mode changes? What network mechanisms are in place to ensure that such actions or data do not endanger the operation of the network? Do any protocol operations, such as start-up, require behaviors or constraints on behaviors of the network clients (and if so, are such behavioral requirements or constraints ensured in the system design)?

Criterion 17 Acknowledgement: Does the acknowledgement mechanism work adequately under all fault scenarios?

If the network provides acknowledgement service(s): What are the guarantees of this service? What are the impacts of the acknowledgement mechanisms? Are they fault tolerant? Has the acknowledgement scheme been analyzed assuming adequate failures modes (e.g., inconsistent or Byzantine message reception)? What are the effects of such failures? Have adverse effects been suitably compensated by other means? Have effects of negative acknowledgements and message retries been analyzed during the performance analysis? Is the number of retries bounded and is this bounded number determined as a consequence the impact of retries on the performance? What are the effects of acknowledgement errors on the application? Are effects suitably bounded (e.g., with respect to processor overhead)? If acknowledgements are not supported, does the protocol support "aging" or "staleness" indications for periodic messages that have not been received for more than a period?

E.4 APPLICATION SERVICES.

Criterion 18 Host Interface Management: The protocol's interface to the host application (including gateways) must provide promised prioritization, latency, and loss prevention of messages for supported categories of service.

Has the buffer management been analyzed with respect to effects and system-level implications (e.g., combined priority and buffer management causing head-of-line blocking or priority inversions)? In mixed criticality systems, are communications buffers reserved per partition and ensured to be inaccessible for other software partitions in cases where there are partitioning requirements for software? Have potential masquerading effects due to buffer management been considered (e.g., software partitions masquerading as other partitions)? How the access to the buffers is controlled (control between different software partitions and control of access between network and host software access)? Has this been analyzed with respect to safety (effects of different criticality partitions)? Are system-level effects of blocking of buffer access been analyzed (e.g., increased software execution time, increased buffer size needs)? Has the performance of the buffer been analyzed with respect to the needs of the communication (i.e., is

the network speed and the buffer management speed balanced)? Are there relationships between network configuration flow and buffer management configuration? Is there an effect of changing network tables on the buffer management tables or mechanisms? Have such relationships been considered during the design and what is done to ensure compatibility? Is there a way to ensure that received messages are not overwritten in the buffers before being retrieved or, that such overwriting is detectable and handled appropriately by the system an error condition?

Criterion 19 Support for Application-Layer Redundancy: Any support for application-layer redundancy must fully meet stated redundancy management mechanism requirements.

Has the application group membership service been analyzed with respect to its performance, availability, and integrity targets under adequate fault scenarios? How are failures in membership or similar services covered? What assumptions were made in determining this coverage? How are these assumptions verified? Are the assumptions of correctness and completeness of the application-level, group membership service in agreement with the system-level assumptions? Has the effect of the lag in update of the application been analyzed in the system context? Do the services provided conform to the requirements from the application? How is the synchronization ensured between different replicants (that are to be voted or agreed)? Any voting or selection scheme may lead to different results at different observers for certain failure modes (Byzantine or local nonreception of a single message); has the impact of different results been taken into account in the safety analysis? Are there other mechanisms that may influence the selection logic? Are such mechanisms analyzed with respect to potential unwanted interaction? Is the network implementation vulnerable to masquerading faults resulting in potential defeat of redundancy? Are passive replication strategies analyzed with respect to control handover in failure cases? How is the state of faulty replica signaled to the application? Do the two halves of a self-checking pair communication interface compare their two copies of received data before it is allowed to be used for computation? Is this comparison designed to be immune to Byzantine faults? Are the halves of a self-checking pair suitably independent (e.g., independent power, separate memory, or memory sections)? If the same message information is sent over redundant buses, are those redundant copies crosschecked, or is the first seemingly valid message used? If the same message information is sent over redundant buses, are those copies sent at the same time or staggered to mitigate against correlated network disturbances?

Criterion 20 Robust Partitioning (ARINC 651): If the network is required to provide robust partitioning guarantees, what has been done to substantiate any partitioning claims?

If required, does the network enforce robust partitioning amongst its clients, even if multiple clients share the same node? If required, does the network enforce robust partitioning amongst its partitions within its clients? Does the partitioning against software faults include vulnerabilities resulting from timing faults?

Criterion 21 Time Service for Time Stamping and Time Interrupt: If time stamping and time-interrupt services are provided, are they sufficiently dependable and robust?

Is the time service robust with respect to potential implications, such as effect of clock differences? Is the use of time stamping and time-interrupt services adequately mitigated or included in a robustness analysis of application data algorithms? If the protocol is based on a centralized time service, is failover for the time master supported? If it is based on a distributed time service, is timekeeping maintained even for Byzantine clock faults?

E.5 FAULT-TOLERANCE MECHANISMS.

Criterion 22 Topological Fault Tolerance: Redundant network components must be physically separated and isolated to prevent correlated outages due to physical equipment damage, loss of electrical power, and credible media faults.

Is the network vulnerable to spatial proximity faults, such as physical damage, that are within a specified distance limit? If a single piece of equipment is faulty, can its faults propagate to take down the entire network (e.g., can a single fault cause babbling behavior simultaneously on redundant network paths)? Is adequate communication path availability ensured despite faulty end equipment (babbling devices or short circuits)? Are common network resources (such as switches) placed at adequate distances from each other so as not to be vulnerable, but to be independent of physical damage or spatial proximity faults? Are redundant resources attached to different power sources within the system?

Criterion 23 Guardian Schemes: Some technique, such as network guardians, must be used to ensure that a single-point client failure will not take down the network.

Does the network deploy guardians? What is the intent of the guardians? What coverage is the guardian assumed to provide (what failure modes can the guardian detect or contain)? What is done to substantiate the coverage claims? Does the claimed coverage of failure modes consider boundary system conditions such as start-up or integration? What is done to assure independence of the guardian (power, time, logical dependence, physical dependence)? Have potential side effects of the guardian behavior been considered (especially central guardians)? Is there an effect on inline coverage due to guardians? Have tolerance margins (different oscillators) due to difference between the guardian and the guarded device been established and quantified?

Criterion 24 Protocol Logic Fault Tolerance: The protocol must ensure that errors in protocol logic and protocol state do not result in unacceptable reduction of integrity and availability.

Can mismatches in protocol logic and state occur during protocol start-up? If such mismatches occur, is there a bound on the time until they are resolved? Does the network protocol logic rely on information from single sources? Or, is protocol logic dependent on agreeing data from different sources? Does protocol dependence rely on different sources to increase the robustness

of algorithms to potential node failures? Are the integrity and availability levels of the protocol logic met by the network?

Criterion 25 Protocol Transmission Monitoring and Self-Checking Schemes: The protocol must reliably detect transient and permanent faults in both nodes and message transmissions.

Does the network incorporate local health monitoring schemes to detect node health? Are the network-monitoring schemes suitably independent? Are network-monitoring schemes vulnerable to faulty status sent by erroneous nodes? Does the network technology use local transmission wrap-back? Have transmission wrap-back been analyzed for Byzantine failure vulnerability? Does the network incorporate self-checking pair configurations? Has the coverage and independence of the self-checking configuration been justified?

Criterion 26 Reconfiguration and Degraded Operation: Capabilities provided in the presence of a specified number and type of faults must be sufficient to meet operational requirements.

Can a faulty node cause good nodes to be evicted from system configuration or otherwise cause degradation into a mode that worsens the effects of the fault? Does the network technology provide degraded modes of operation? Is the network performance under degraded mode sufficient to meet network functional requirements? Is network degraded mode operation suitably annunciated to network clients? Does the network perform dynamic reconfiguration of routing to mitigate bad network paths or nodes? Is the reconfiguration time suitably bounded? If online reintegration is supported, what mechanisms are in place to cope with intermittent failures and ensure the health of nodes to be reintegrated?

Criterion 27 Latent Failure Detection: Latent faults must not accumulate to where they threaten network failure (availability or integrity).

Can an accumulation of latent faults overwhelm a network's ability to tolerate faults? Can a latent fault in a failure detection, isolation, and recovery (FDIR) mechanism (e.g., a "stuck-at-good" fault detector) lead to network failure due to lack of coverage? Does the network technology support a mechanism for latent fault detection, especially in mismatched protocol state among network clients? If state variables are maintained by the protocol at each client, can multiple unrelated state variables be corrupted before the first corruption is detected? Is the coverage of the network latent fault detection suitable to establish the health of all critical network protection functions? Are suitable test activation interlocks incorporated into the network technology to inadvertent test mode activation?

Criterion 28 Voting, Selection, or Agreement Services and Redundancy Management: The network protocol must support the ability to determine which nodes are part of the active network quorum.

Does the network support selection and agreement services directly? Does the network provide adequate mechanisms for application software to implement selection and agreement services?

Are such agreement services targeted at integrity or availability? Is the overhead for agreement function evaluated and justified to be acceptable? Have temporal effects been evaluated in the selection? Are the time constants of tracking changes in membership fast enough for the application? Are assumptions made by the membership/agreement service justifiable? What vulnerabilities exist while the quorum is inconsistent?

Criterion 29 Fault Model: The protocol must be evaluated with respect to a precisely and completely stated fault model, and the fault model must be compatible with that of the system architecture.

Does the network claim to be Byzantine fault tolerant? Has the network been analyzed with respect to Byzantine fault tolerance? What are the effects of Byzantine fault tolerance on integrity and availability? If a hybrid fault model is used, is there justification for the relative rates of occurrence of different classes of faults (e.g., Byzantine, strictly omissive, symmetric) for different failure scenarios? Has the Byzantine fault-tolerant algorithm been suitable analyzed and coverage been justified (Byzantine fault-containment properties)? Has the Byzantine filtering coverage been justified? Has the fault model been analyzed with respect to influences to all services even the ones that are possibly not used (this should exclude any effects of services not used)?

E.6 DESIGN ASSURANCE.

Criterion 30 Design Assurance Processes: Have appropriate design assurance processes been followed for design and deployment of the network?

Has the network technology been developed to be in compliance with avionics design assurance guidelines, i.e., DO-178B and or DO-254? Does the network technology's commercial-use volume support COTS classification (including intellectual property block used within new integrated circuits as well as manufactured devices)? If COTS classification is used for classification, has the COTS product been deployed in similar applications to justify suitable coverage for correctness claims? Is the network technology behavior simple enough to be fully testable? Do diverse approaches (e.g., design diversity) provide adequate and quantifiable coverage if credit is taken for such diversity? Is there a plan for hardware aspects of certification (PHAC) or a plan for software aspects of certification (PSAC) for the network infrastructure?

Criterion 31 Availability of Standards and Conformance Evidence: The technology and protocol of the network should be clearly specified and be analyzable.

The use of open specifications and standardization might assist a certification authority in establishing the acceptability of a network. Credit for analysis of specification properties and interoperability between different networks should be supported by conformance and interoperability testing. The use of formal methods to demonstrate protocol design correctness should be proposed and accepted by the certification authorities. Is the network technology supported by an open specification? Is the specification that is being standardized available to open-industrial committees? Is the network specification complete? Does the network technology address all operational modes of the network, including erroneous node action and

associated fault recovery actions? Is the network specification sufficiently detailed to address all required protocol action? Does the networking technology have published conformance test criteria and campaigns? Do the conformance test criteria cover all network protocol behaviors, including fault detection and recovery actions? Have the critical protocol mechanisms and algorithms of the network technology been formally verified? Have the assumptions underpinning the formal verifications been reviewed to ensure that they are consistent with real-world targeted environment? Are protocol mechanisms and associated formal proofs composable? That is, do protocol mechanisms and associated proofs stand by themselves or are they interrelated? Has the network technology been subjected to other validation activities? Did the fault-injection campaign include suitably sufficient visibility to observe the key behaviors of all important network mechanisms? Have anomalies and fault observations from such activities been adequately mitigated?

Criterion 32 Design Margin: Required design margins should be supported by reviewable evidence.

Has the network design margin been established for worst-case component behaviors? Have all contributions to network design margins been identified?

Criterion 33 Configuration Table Correctness and Performance Justification: Justification for configuration table correctness and performance justification should be provided.

Has the criteria for correct network parameterization and configuration been established? Are the network configuration criteria traceable to network function behavior or top-level requirements or specification? Does tooling assist network configuration and verification? Is the tooling qualified in accordance with the tool guidance established in DO-178B? Have procedures and criteria to bound the worst-case performance of the network been established? Do the worst-case performance criteria address detailed MAC layer interactions? Do the worst-case performance criteria address worst-case fault-detection and reconfiguration actions? Does the network technology provide automated tools to assist worst-case performance calculation? Are intermediate-stage buffers and end-node queues adequately sized? Are tools providing performance bounding qualified in accordance with the tooling guidelines of DO-178B? Does the network technology and associated tooling accommodate incremental change management?

Criterion 34 Network Monitoring and Test Equipment: Does there exist adequate test equipment, network access points, mechanisms, and procedures to ensure that the network is configured correctly and operating correctly (including meeting its specified behavior in the presence of any faults)?

The use of network monitoring and test equipment to establish certification credit should not invalidate the data being observed and should be demonstrated to perform in accordance with the operational requirements for the features being used. Is test equipment available for the network technology? Does the test equipment facilitate the observation of all network operational modes? Does the network test equipment facilitate sufficient fault injection to exercise sufficient network fault detection mechanisms? Does the network test equipment support observation modes with sufficient noninterference guarantees to support flight testing? How many network

test access points are required to monitor the entire holistic network behavior? Is this feasible for achieving flight test requirements? Is monitoring and test equipment assured to be noninterfering during operation? Is it guaranteed that the network behavior does not change if monitoring is not performed and correct behavior is inferred from monitoring or testing?

E.7 SECURITY.

Criterion 35: Can security weaknesses adversely affect network dependability (safety)?

Does the network technology have security issues that can adversely affect the ability of the network to supply the services needed to support system safety? In particular, is the network susceptible to denial-of-service attacks (e.g., 100BaseTX "killer packets", ping of death)? Does the network technology use open COTS protocols that are well known enough to be targets of security threats? Does the network technology require specialized security augmentations; e.g., firewalls?

Criterion 36: If needed, does the network deal adequately with the security issues of privacy (also known as confidentiality or secrecy), integrity, authentication, or authorization?

Does the network technology support sufficient secure services for user and application authentication? Does the network technology support secured data transmission mechanism? Does the network support multilevel security? How many levels or security domains does the network technology support? Is network configuration data protected and secured during deployment and during load?

APPENDIX F—EVALUATION CRITERIA TEST RESULTS

F.1 ARINC 629.

F.1.1 PHYSICAL LAYER.

Criterion 1 Environment: Does the network specification(s) and available components allow for the creation of a network that meets the applicable aerospace environment requirements of DO-160 or other imposed environmental requirements?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 2 Line-Level Encoding: Was the electromagnetic compatibility testing (e.g., DO-160 Sections 19-22) done with the actual data line encoding (e.g., Manchester, 8B/10B), worst-case network data, worst-case message sizes, and worst-case message repetition rate(s)?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 3 Probability of Bit Errors: Was the bit error rate determined under worst-case conditions?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 4 Probability of Electrical Component Failures: Do the data network's electronic components have established hardware failure rates (permanent and transient) and characteristics so that avionics designers can do the required failure modes and effects analysis (FMEA) and fault tree calculations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 5 Electrical Isolation Properties: Is there sufficient protection against electrical fault propagation?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 6 Physical Composability: Does the data network have characteristics or design rules that will guarantee that it will reliably work with any number of nodes up to some explicitly given maximum number of nodes?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.1.2 DATA LINK LAYER.

Criterion 7 Media Access Control (MAC): The MAC sublayer protocol must provide appropriately small bounds on message delivery times regardless of likely faults.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 8 Message Formatting (Framing): Message framing must ensure that only complete, properly synchronized messages are accepted at clients, and that improper synchronization is recovered from in bounded time.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 9 Error Detection: The data link layer must provide sufficient guarantees of error-free message delivery.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.1.3 NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

Criterion 10 Network Vulnerability to Addressing Information Failure: Mechanisms should ensure correct forwarding, routing, or conversion failures despite likely failure scenarios of network components.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 11 Impact of Intermediate Stages: Intermediate stages (e.g., repeaters, gateways, routers, and switches) must guarantee sufficient availability and integrity requirements, as well as sufficient logical and physical independence from other replicated intermediate stages, to ensure correct operation in adequate failure scenarios.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 12 Network Configuration Data: Network topology and component configuration data must be correct with respect to the applications' fault tolerance and performance requirements considering table production, loading, errors during operation, run-time environmental effects (e.g., impact of radiation), and maintenance actions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 13 Start-up and Recovery: Component and network integration, start-up, and recovery must be performed in bounded time considering dependability requirements, environmental and deployment constraints, interactions with different systems and applications (such as application-level timing impact and power architecture influence).

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 14 Global Synchronization: The synchronization and interaction constraints of systems must have timing bounds and dependencies for reasonable failure scenarios as well as the effects resulting from synchronization such as perceptions.

- Understandable: Poorly worded. Needs a major rewrite.
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 15 Fault Diagnosis: Any diagnosis, detection, or system-level agreement mechanisms must justify fault assumptions and consider effects of diagnosis action.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 16 Client Effect on Network Operations: Can a network client adversely effect network operations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 17 Acknowledgement: Does the acknowledgement mechanism work adequately under all fault scenarios?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.1.4 APPLICATION-LAYER SERVICES.

Criterion 18 Host Interface Management: The protocol's interface to the host application must provide promised prioritization, latency, and loss prevention of messages for supported categories of service.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 19 Support for Application-Layer Redundancy: Any support for application-layer redundancy must fully support stated redundancy management mechanisms for a specified fault model.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 20 Time Service for Time Stamping and Time Interrupt: If time stamping and time interrupt services are provided, do they provide dependable and correct time services?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 21 Robust Partitioning (ARINC 651): If the network is required to provide robust partitioning guarantees, what has been done to substantiate any partitioning claims?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.1.5 FAULT-TOLERANCE MECHANISMS.

Criterion 22 Topological Fault Tolerance: Redundant network components must be physically separated and isolated to prevent correlated outages due to physical equipment damage and credible electrical faults.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 23 Guardian Schemes: Some technique, such as network guardians, must be used to ensure that a single point client failure will not take down the network.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 24 Protocol Logic Fault-Tolerance: The protocol must ensure that errors in protocol logic and protocol state do not result in unacceptable reduction of integrity and availability.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 25 Protocol Transmission Monitoring and Self-Checking Schemes: The protocol must reliably detect transient and permanent faults in both nodes and message transmissions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 26 Reconfiguration and Degraded Operation: Capabilities provided in the presence of a specified number and type of faults must be sufficient to meet operational requirements.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 27 Latent Failure Detection: Latent faults must not result in network failure.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 28 Voting, Selection, or Agreement Services and Redundancy Management: The network protocol must support the ability to determine which nodes are part of the active network quorum.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 29 Fault Model: The protocol must be evaluated with respect to a precisely and completely stated fault model, and the fault model must be compatible with that of the system architecture.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.1.6 DESIGN ASSURANCE.

Criterion 30 Design Assurance Processes: Have appropriate design assurance processes been followed for design and deployment of the network? (or) The hardware and software design should be demonstrated to satisfy published regulatory guidelines.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 31 Availability of Standards and Conformance Evidence: The technology and protocol of the network should be clearly specified and be analyzable.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 32 Design Margin: Required design margins should be supported by reviewable evidence.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 33 Configuration Table Correctness and Performance Justification: Justification for configuration table correctness and performance justification should be provided.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 34 Network Monitoring and Test Equipment: Do adequate test equipment, network access points, mechanisms, and procedures exist to ensure that the network is healthy and configured correctly?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.1.7 SECURITY.

Criterion 35: Can security weaknesses adversely affect network dependability (safety)?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 36: If needed, does the network deal adequately with the security issues of privacy (also known as confidentiality or secrecy), integrity, authentication, or authorization?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.2 ARINC 659.

F.2.1 PHYSICAL LAYER.

Criterion 1 Environment: Does the network specification(s) and available components allow for the creation of a network that meets the applicable aerospace environment requirements of DO-160 or other imposed environmental requirements?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 2 Line-Level Encoding: Was the electromagnetic compatibility testing (e.g., DO-160 Sections 19-22) done with the actual data line encoding (e.g., Manchester, 8B/10B), worst-case network data, worst-case message sizes, and worst-case message repetition rate(s)?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 3 Probability of Bit Errors: Was the bit error rate determined under worst-case conditions?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 4 Probability of Electrical Component Failures: Do the data network's electronic components have established hardware failure rates (permanent and transient) and characteristics so that avionics designers can do the required failure modes and effects analysis (FMEA) and fault tree calculations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 5 Electrical Isolation Properties: Is there sufficient protection against electrical fault propagation?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 6 Physical Composability: Does the data network have characteristics or design rules that will guarantee that it will reliably work with any number of nodes up to some explicitly given maximum number of nodes?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.2.2 DATA LINK LAYER.

Criterion 7 MAC: The MAC sublayer protocol must provide appropriately small bounds on message delivery times regardless of likely faults.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Add: "Does the MAC affect messaging delivery ordering?"
- Orthogonal: OK
- Bins: OK

Criterion 8 Message Formatting (Framing): Message framing must ensure that only complete, properly synchronized messages are accepted at clients, and that improper synchronization is recovered from in bounded time.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 9 Error Detection: The data link layer must provide sufficient guarantees of error-free message delivery.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.2.3 NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

Criterion 10 Network Vulnerability to Addressing Information Failure: Mechanisms should ensure correct forwarding, routing, or conversion failures despite likely failure scenarios of network components.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 11 Impact of Intermediate Stages: Intermediate stages (e.g., repeaters, gateways, routers, and switches) must guarantee sufficient availability and integrity requirements as well as sufficient logical and physical independence from other replicated intermediate stages to ensure correct operation in adequate failure scenarios.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 12 Network Configuration Data: Network topology and component configuration data must be correct with respect to the applications' fault tolerance and performance requirements considering table production, loading, errors during operation, run-time environmental effects (e.g., impact of radiation), and maintenance actions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 13 Start-up and Recovery: Component and network integration, start-up, and recovery must be performed in bounded time considering dependability requirements, environmental and deployment constraints, interactions with different systems and applications (such as application-level timing impact and power architecture influence).

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 14 Global Synchronization: The synchronization and interaction constraints of systems must have timing bounds and dependencies for reasonable failure scenarios, as well as the effects resulting from synchronization such as perceptions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Add: "For asynchronous interfaces between two or more isochronous clock domains running at the same frequency, is there a mechanism to prevent pathological metastability (the metastability condition persists indefinitely due to the clocks not having any relative drift)?"
- Orthogonal: OK
- Bins: OK

Criterion 15 Fault Diagnosis: Any diagnosis, detection, or system-level agreement mechanisms must justify fault assumptions and consider effects of diagnosis action.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 16 Client Effect on Network Operations: Can a network client adversely effect network operations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 17 Acknowledgement: Does the acknowledgement mechanism work adequately under all fault scenarios?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.2.4 APPLICATION-LAYER SERVICES.

Criterion 18 Host Interface Management: The protocol's interface to the host application must provide promised prioritization, latency, and loss prevention of messages for supported categories of service.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 19 Support for Application-Layer Redundancy: Any support for application-layer redundancy must fully support stated redundancy management mechanisms for a specified fault model.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 20 Time Service for Time Stamping and Time Interrupt: If time stamping and time-interrupt services are provided, do they provide dependable and correct time services?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 21 Robust Partitioning (ARINC 651): If the network is required to provide robust partitioning guarantees, what has been done to substantiate any partitioning claims?

- Understandable: Need to add some support to clarify this criterion.
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.2.5 FAULT-TOLERANCE MECHANISMS.

Criterion 22 Topological Fault Tolerance: Redundant network components must be physically separated and isolated to prevent correlated outages due to physical equipment damage and credible electrical faults.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 23 Guardian Schemes: Some techniques, such as network guardians, must be used to ensure that a single point client failure will not take down the network.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 24 Protocol Logic Fault Tolerance: The protocol must ensure that errors in protocol logic and protocol state do not result in unacceptable reduction of integrity and availability.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 25 Protocol Transmission Monitoring and Self-Checking Schemes: The protocol must reliably detect transient and permanent faults in both nodes and message transmissions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 26 Reconfiguration and Degraded Operation: Capabilities provided in the presence of a specified number and type of faults must be sufficient to meet operational requirements.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 27 Latent Failure Detection: Latent faults must not result in network failure.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 28 Voting, Selection, or Agreement Services and Redundancy Management: The network protocol must support the ability to determine which nodes are part of the active network quorum.

- Understandable: This criterion is simultaneously overreaching and incomplete (with respect to its supporting questions). There is no need for a network (particularly one based on masking rather than reconfiguration) to maintain quorum data. The supporting questions probe areas beyond just maintaining an active network quorum.

- Measurable: OK

- Loopholes: OK

- Supporting: OK

- Orthogonal: OK

- Bins: OK

Criterion 29 Fault Model: The protocol must be evaluated with respect to a precisely and completely stated fault model, and the fault model must be compatible with that of the system architecture.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.2.6 DESIGN ASSURANCE.

Criterion 30 Design Assurance Processes: Have appropriate design assurance processes been followed for design and deployment of the network? (or) The hardware and software design should be demonstrated to satisfy published regulatory guidelines.

- Understandable: The second form of this criterion could lead to a circular requirement if the handbook were used to create regulatory guidelines.
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 31 Availability of Standards and Conformance Evidence: The technology and protocol of the network should be clearly specified and be analyzable.

- Understandable: Do the documents that support this have to be in the public domain? Are they free or available at a reasonable cost? Should questions that are economic rather than technical be part of the handbook?
- Measurable: see immediately above
- Loopholes: see immediately above
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 32 Design Margin: Required design margins should be supported by reviewable evidence.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 33 Configuration Table Correctness and Performance Justification: Justification for configuration table correctness and performance justification should be provided.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 34 Network Monitoring and Test Equipment: Do adequate test equipment, network access points, mechanisms, and procedures exist to ensure that the network is healthy and configured correctly?

- Understandable: Cost of test equipment? Again, should questions that are economic rather than technical be part of the handbook? For example, if a certain piece of test equipment is absolutely required to establish a safety case but was prohibitively expensive, should the equipment be deemed to be not available?

- Measurable: OK

- Loopholes: OK

- Supporting: OK

- Orthogonal: OK

- Bins: OK

F.2.7 SECURITY.

Criterion 35: Can security weaknesses adversely affect network dependability (safety)?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 36: If needed, does the network deal adequately with the security issues of privacy (also known as confidentiality or secrecy), integrity, authentication, or authorization?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.3 ARINC 664 PART 7 (AFDX).

F.3.1 PHYSICAL LAYER.

Criterion 1 Environment: Does the network specification(s) and available components allow for the creation of a network that meets the applicable aerospace environment requirements of DO-160 or other imposed environmental requirements?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 2 Line-Level Encoding: Was the electromagnetic compatibility testing (e.g., DO-160 Sections 19-22) done with the actual data line encoding (e.g., Manchester, 8B/10B), worst-case network data, worst-case message sizes, and worst-case message repetition rate(s)?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 3 Probability of Bit Errors: Was the bit error rate determined under worst-case conditions?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 4 Probability of Electrical Component Failures: Do the data network's electronic components have established hardware failure rates (permanent and transient) and characteristics so that avionics designers can do the required FMEA and fault tree calculations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 5 Electrical Isolation Properties: Is there sufficient protection against electrical fault propagation?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 6 Physical Composability: Does the data network have characteristics or design rules that will guarantee that it will reliably work with any number of nodes up to some explicitly given maximum number of nodes?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Add "Also logical composability—What is the certification effect of changing the size of the network (e.g., number of members in a clock sync algorithm)?"
- Orthogonal: OK
- Bins: OK

F.3.2 DATA LINK LAYER.

Criterion 7 MAC: The MAC sublayer protocol must provide appropriately small bounds on message delivery times regardless of likely faults.
Understandable: OK

- Measurable: OK

- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 8 Message Formatting (Framing): Message framing must ensure that only complete, properly synchronized messages are accepted at clients, and that improper synchronization is recovered from in bounded time.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 9 Error Detection: The data link layer must provide sufficient guarantees of error-free message delivery.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.3.3 NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

Criterion 10 Network Vulnerability to Addressing Information Failure: Mechanisms should ensure correct forwarding, routing, or conversion failures, despite likely failure scenarios of network components.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Add: If multiple layers of addressing are used, such as MAC and internet protocol (IP), then errors and masquerading faults in all levels of addressing must be considered.
- Orthogonal: OK
- Bins: OK

Criterion 11 Impact of Intermediate Stages: Intermediate stages (e.g., repeaters, gateways, routers, and switches) must guarantee sufficient availability and integrity requirements as well as sufficient logical and physical independence from other replicated intermediate stages to ensure correct operation in adequate failure scenarios.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Are the failure assumptions of the interstage justified by a complete failure modes and effects analysis (FMEA)? Is it possible for a faulty guardian to cause irrecoverable network failures?
- Orthogonal: OK
- Bins: OK

Criterion 12 Network Configuration Data: Network topology and component configuration data must be correct with respect to the applications' fault tolerance and performance requirements considering table production, loading, errors during operation, run-time environmental effects (e.g., impact of radiation), and maintenance actions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 13 Start-up and Recovery: Component and network integration, start-up, and recovery must be performed in bounded time considering dependability requirements, environmental and deployment constraints, interactions with different systems and applications (such as application-level timing impact and power architecture influence).

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 14 Global Synchronization: The synchronization and interaction constraints of systems must have timing bounds and dependencies for reasonable failure scenarios as well as for the effects resulting from synchronization, such as perceptions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 15 Fault Diagnosis: Any diagnosis, detection, or system-level agreement mechanisms must justify fault assumptions and consider effects of diagnosis action.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 16 Client Effect on Network Operations: Can a network client adversely effect network operations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 17 Acknowledgement: Does the acknowledgement mechanism work adequately under all fault scenarios?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.3.4 APPLICATION-LAYER SERVICES.

Criterion 18 Host Interface Management: The protocol's interface to the host application must provide promised prioritization, latency, and loss prevention of messages for supported categories of service.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 19 Support for Application-Layer Redundancy: Any support for application-layer redundancy must fully support stated redundancy management mechanisms for a specified fault model.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 20 Time Service for Time Stamping and Time Interrupt: If time stamping and time-interrupt services are provided, do they provide dependable and correct time services?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 21 Robust Partitioning (ARINC 651): If the network is required to provide robust partitioning guarantees, what has been done to substantiate any partitioning claims?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.3.5 FAULT-TOLERANCE MECHANISMS.

Criterion 22 Topological Fault Tolerance: Redundant network components must be physically separated and isolated to prevent correlated outages due to physical equipment damage and credible electrical faults.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 23 Guardian Schemes: Some technique, such as network guardians, must be used to ensure that a single point client failure will not take down the network.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 24 Protocol Logic Fault-Tolerance: The protocol must ensure that errors in protocol logic and protocol state do not result in unacceptable reduction of integrity and availability.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 25 Protocol Transmission Monitoring and Self-Checking Schemes: The protocol must reliably detect transient and permanent faults in both nodes and message transmissions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 26 Reconfiguration and Degraded Operation: Capabilities provided in the presence of a specified number and type of faults must be sufficient to meet operational requirements.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 27 Latent Failure Detection: Latent faults must not result in network failure.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 28 Voting, Selection, or Agreement Services and Redundancy Management: The network protocol must support the ability to determine which nodes are part of the active network quorum.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 29 Fault Model: The protocol must be evaluated with respect to a precisely and completely stated fault model, and the fault model must be compatible with that of the system architecture.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.3.6 DESIGN ASSURANCE.

Criterion 30 Design Assurance Processes: Have appropriate design assurance processes been followed for design and deployment of the network? (or) The hardware and software design should be demonstrated to satisfy published regulatory guidelines.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 31 Availability of Standards and Conformance Evidence: The technology and protocol of the network should be clearly specified and be analyzable.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 32 Design Margin: Required design margins should be supported by reviewable evidence.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 33 Configuration Table Correctness and Performance Justification: Justification for configuration table correctness and performance justification should be provided.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 34 Network Monitoring and Test Equipment: Do adequate test equipment, network access points, mechanisms, and procedures exist to ensure that the network is healthy and configured correctly?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.3.7 SECURITY.

Criterion 35: Can security weaknesses adversely affect network dependability (safety)?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 36: If needed, does the network deal adequately with the security issues of privacy (also known as confidentiality or secrecy), integrity, authentication, or authorization?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.4 CONTROLLER AREA NETWORK.

F.4.1 PHYSICAL LAYER.

Criterion 1 Environment: Does the network specification(s) and available components allow for the creation of a network that meets the applicable aerospace environment requirements of DO-160 or other imposed environmental requirements?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 2 Line-Level Encoding: Was the electromagnetic compatibility testing (e.g., DO-160 Sections 19-22) done with the actual data line encoding (e.g., Manchester, 8B/10B), worst-case network data, worst-case message sizes, and worst-case message repetition rate(s)?

- Understandable: OK
- Measurable: OK. Obtaining worst-case message sizes is complicated in Controller Area Networks (CAN) due to bit stuffing, but is practicable.
- Loopholes: OK
- Supporting: Consider adding the following question: Does the worst-case message size take into account worst-case variable-length encoding, as encountered in bit-stuffed encoding? Should also add worst-case bit rate and pulse widths for networks that allow variations in these.
- Orthogonal: OK
- Bins: OK

Criterion 3 Probability of Bit Errors: Was the bit error rate determined under worst-case conditions?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 4 Probability of Electrical Component Failures: Do the data network's electronic components have established hardware failure rates (permanent and transient) and characteristics so that avionics designers can do the required FMEA and fault tree calculations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 5 Electrical Isolation Properties: Is there sufficient protection against electrical fault propagation?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: See Criterion 22
- Bins: OK

Criterion 6 Physical Composability: Does the data network have characteristics or design rules that will guarantee that it will reliably work with any number of nodes up to some explicitly given maximum number of nodes?

- Understandable: OK

- Measurable: OK. These characteristics are not part of the specification per se, but it should be practicable to create design rules for using CAN that provide the desired properties based on specific implementation technology.

- Supporting: Consider adding this question:

- For data networks in which bit rate is limited by network distance, are there design rules to ensure that a particular network distance supports a given bit rate under worst-case conditions?

- Orthogonal: OK

- Bins: OK

F.4.2 DATA LINK LAYER.

Criterion 7 MAC: The MAC sublayer protocol must provide appropriately small bounds on message delivery times regardless of likely faults.

- Understandable: OK
- Measurable: OK. CAN might have an issue due to message prioritization.
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 8 Message Formatting (Framing): Message framing must ensure that only complete, properly synchronized messages are accepted at clients, and that improper synchronization is recovered from in bounded time.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 9 Error Detection: The data link layer must provide sufficient guarantees of error-free message delivery.

- Understandable: OK
- Measurable: OK. CAN has the potential for bit stuffing to undermine cyclic redundancy code (CRC) effectiveness due to a cascade effect of a pair of inverted bits (the first inversion due to a stuff bit).
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.4.3 NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

Criterion 10 Network Vulnerability to Addressing Information Failure: Mechanisms should ensure correct forwarding, routing, or conversion failures despite likely failure scenarios of network components.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK
- Typographical error: "message investigation fields" should be "message identification fields" (two occurrences)

Criterion 11 Impact of Intermediate Stages: Intermediate stages (e.g., repeaters, gateways, routers, and switches) must guarantee sufficient availability and integrity requirements as well as sufficient logical and physical independence from other replicated intermediate stages to ensure correct operation in adequate failure scenarios.

- Untested

Criterion 12 Network Configuration Data: Network topology and component configuration data must be correct with respect to the applications' fault tolerance and performance requirements considering table production, loading, errors during operation, run-time environmental effects (e.g., impact of radiation), and maintenance actions.

- Untested

Criterion 13 Start-up and Recovery: Component and network integration, start-up, and recovery must be performed in bounded time considering dependability requirements, environmental and deployment constraints, interactions with different systems and applications (such as application-level timing impact and power architecture influence).

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 14 Global Synchronization: The synchronization and interaction constraints of systems must have timing bounds and dependencies for reasonable failure scenarios as well as for the effects resulting from synchronization such as perceptions.

- Untested

Criterion 15 Fault Diagnosis: Any diagnosis, detection, or system-level agreement mechanisms must justify fault assumptions and consider effects of diagnosis action.

- Untested

Criterion 16 Client Effect on Network Operations: Can a network client adversely effect network operations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK

- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 17 Acknowledgement: Does the acknowledgement mechanism work adequately under all fault scenarios?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.4.4 APPLICATION-LAYER SERVICES.

Criterion 18 Host Interface Management: The protocol's interface to the host application must provide promised prioritization, latency, and loss prevention of messages for supported categories of service.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Consider adding this CAN-relevant question:
- Is there a way to ensure that received messages are not overwritten in the buffers before being retrieved (or, that such overwriting is detectable and handled appropriately by the system an error condition?)
- Orthogonal: OK
- Bins: OK

Criterion 19 Support for Application-Layer Redundancy: Any support for application-layer redundancy must fully support stated redundancy management mechanisms for a specified fault model.

- Untested

Criterion 20 Time Service for Time Stamping and Time Interrupt: If time stamping and time-interrupt services are provided, do they provide dependable and correct time services?

- Untested

Criterion 21 Robust Partitioning (ARINC 651): If the network is required to provide robust partitioning guarantees, what has been done to substantiate any partitioning claims?

- Untested

F.4.5 FAULT-TOLERANCE MECHANISMS.

Criterion 22 Topological Fault Tolerance: Redundant network components must be physically separated and isolated to prevent correlated outages due to physical equipment damage and credible electrical faults.

- Orthogonal: See criterion 5
- Untested (partial)

Criterion 23 Guardian Schemes: Some technique, such as network guardians, must be used to ensure that a single point client failure will not take down the network.

- Loopholes: Network guardians are sometimes used in CAN, but are not part of the standard. Is it important to distinguish these two cases?

- Untested (partial)

Criterion 24 Protocol Logic Fault-Tolerance: The protocol must ensure that errors in protocol logic and protocol state do not result in unacceptable reduction of integrity and availability.

- Understandable: OK

- Measurable: Can generally be applied to CAN, but might require an assumption such as protocol logic correctly detects intermessage period. Possibly this criterion is not worded to apply to CAN, but consideration should be given to binary countdown protocols and how this should be interpreted for that case.

- Loopholes: OK

- Supporting: OK

- Orthogonal: OK

- Bins: OK

Criterion 25 Protocol Transmission Monitoring and Self-Checking Schemes: The protocol must reliably detect transient and permanent faults in both nodes and message transmissions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 26 Reconfiguration and Degraded Operation: Capabilities provided in the presence of a specified number and type of faults must be sufficient to meet operational requirements.

- Untested

Criterion 27 Latent Failure Detection: Latent faults must not result in network failure.

- Untested

Criterion 28 Voting, Selection, or Agreement Services and Redundancy Management: The network protocol must support the ability to determine which nodes are part of the active network quorum.

- Untested

Criterion 29 Fault Model: The protocol must be evaluated with respect to a precisely and completely stated fault model, and the fault model must be compatible with that of the system architecture.

- Understandable: OK
- Measurable: Unclear (see loopholes)
- Loopholes: How should this be approached if the network specification does not state a fault model? Should the system architect propose a fault model and then judge the network protocol against it?
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.4.6 DESIGN ASSURANCE.

Criterion 30 Design Assurance Processes: Have appropriate design assurance processes been followed for design and deployment of the network? (or) The hardware and software design should be demonstrated to satisfy published regulatory guidelines.

- Understandable: OK
- Measurable: OK
- Loopholes: The software and firmware inside the protocol chips has to be considered, as well as application software and hardware.
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 31 Availability of Standards and Conformance Evidence: The technology and protocol of the network should be clearly specified and be analyzable.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 32 Design Margin: Required design margins should be supported by reviewable evidence.

- Understandable: It is unclear what design margins mean in this context. Does this include every electrical parameter on the hardware data sheet? Are there other items?
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 33 Configuration Table Correctness and Performance Justification: Justification for configuration table correctness and performance justification should be provided.

- Untested

Criterion 34 Network Monitoring and Test Equipment: Are there adequate test equipment, network access points, mechanisms, and procedures to ensure that the network is healthy and configured correctly?

- Understandable: Some of this seems to go beyond the protocol specification and into the implementation and supporting equipment situation, but is probably appropriate.
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.4.7 SECURITY.

Criterion 35: Can security weaknesses adversely affect network dependability (safety)?

- Understandable: It is unclear what "open COTS protocols" means. Does it mean "open COTS security protocols"?
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 36: If needed, does the network deal adequately with the security issues of privacy (also known as confidentiality or secrecy), integrity, authentication, or authorization?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.5 FLEXRAY.

F.5.1 PHYSICAL LAYER.

Criterion 1 Environment: Does the network specification(s) and available components allow for the creation of a network that meets the applicable aerospace environment requirements of DO-160 or other imposed environmental requirements?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 2 Line-Level Encoding: Was the electromagnetic compatibility testing (e.g., DO-160 Sections 19-22) done with the actual data line encoding (e.g., Manchester, 8B/10B), worst-case network data, worst-case message sizes, and worst-case message repetition rate(s)?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 3 Probability of Bit Errors: Was the bit error rate determined under worst-case conditions?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 4 Probability of Electrical Component Failures: Do the data network's electronic components have established hardware failure rates (permanent and transient) and characteristics so that avionics designers can do the required FMEA and fault tree calculations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK

- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 5 Electrical Isolation Properties: Is there sufficient protection against electrical fault propagation?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Consider adding here (or elsewhere):

  If the network isolation strategy requires direct current (DC) balancing, does the bit encoding provide exact DC balancing with worst-case data patterns over all ranges of relevant time periods? Probably covered under Criterion 3. The effect of unbalanced DC should only show up in the error rate. However, Criterion 3 may have to be reworded so that error rates other than just for bits are included.

- Orthogonal: OK
- Bins: OK

Criterion 6 Physical Composability: Does the data network have characteristics or design rules that will guarantee that it will reliably work with any number of nodes up to some explicitly given maximum number of nodes?

- Understandable: OK
- Measurable: OK
- Loopholes: As stated, this does not directly address repeaters and hubs; see additional supporting question below.
- Supporting: Consider adding this question:

  Do these design rules and characteristics include the effects of hubs, repeaters, or other devices that extend network propagation delay or electrical loading?

- Orthogonal: OK
- Bins: OK

F.5.2 DATA LINK LAYER.

Criterion 7 MAC: The MAC sublayer protocol must provide appropriately small bounds on message delivery times regardless of likely faults.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: There is some overlap with Criterion 17 in terms of faults that occur during start-up, but OK to leave this criterion as is; see Criterion 17 for comments. One should consider the fault propagation due to babbling faults (e.g., via an active star). Where would this feature be evaluated? It is similar to the fault isolation Honeywell was planning to use for different bus segments.
- Bins: OK

Criterion 8 Message Formatting (Framing): Message framing must ensure that only complete, properly synchronized messages are accepted at clients, and that improper synchronization is recovered from in bounded time.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Consider adding the following. FlexRay attempts to mitigate this potential risk by putting the frame identification (ID) in each message even though in a fault-free system the frame ID can be inferred from timing information:
- If this is an implicit token protocol (e.g., reservation carrier sense multiple access (CSMA), or minislot system) or time-sliced protocol (e.g., time division multiple access (TDMA)), is there sufficient information in the message to validate that the time position of the message is interpreted correctly, avoiding incorrect interpretation of the message due to timing inaccuracies?
- Orthogonal: Adding the above supporting question introduces a little overlap with Criterion 10, but this is probably an overall improvement in the methodology.
- Bins: OK

Criterion 9 Error Detection: The data link layer must provide sufficient guarantees of error-free message delivery.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.5.3 NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

Criterion 10 Network Vulnerability to Addressing Information Failure: Mechanisms should ensure correct forwarding, routing, or conversion failures despite likely failure scenarios of network components.

- Understandable: OK
- Measurable: OK
- Loopholes: For protocols that have more than one addressing mode and mechanism, all the modes and mechanisms that are used must be considered.
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 11 Impact of intermediate stages: Intermediate stages (e.g., repeaters, gateways, routers, and switches) must guarantee sufficient availability and integrity requirements as well as sufficient logical and physical independence from other replicated intermediate stages to ensure correct operation in adequate failure scenarios.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK
- Typographical error: "blabbing" should probably be "babbling" above

Criterion 12 Network Configuration Data: Network topology and component configuration data must be correct with respect to the applications' fault tolerance and performance requirements considering table production, loading, errors during operation, run-time environmental effects (e.g., impact of radiation), and maintenance actions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Add: "Are inline checks precomputed and stored in configuration tables or computed on the fly by transmitting nodes for static information such as message identifiers and configuration tables?"
- Orthogonal: OK
- Bins: OK

Criterion 13 Start-up and Recovery: Component and network integration, start-up, and recovery must be performed in bounded time considering dependability requirements, environmental and deployment constraints, and interactions with different systems and applications (such as application-level timing impact and power architecture influence).

- Understandable: Should probably add: "faults within the specified fault model during start-up" to this criterion.
- Measurable: OK
- Loopholes: Sometimes start-up behavior depends on host system operation in addition to protocol implementation per se. A supporting question is suggested below to address this.
- Supporting: Consider adding the below questions:
  - If host nodes are required to participate in network integration, start-up, or recovery, are those host node behaviors considered in the analysis?
  - Is there a host or other node behavior within the protocol fault model that can cause repeated integration, start-up, or recovery events, and thus lead to unbounded time to achieve normal network operation even if each integration, start-up, or recovery attempt completes individually within bounded time?
  - If there is more than one statically designated network Master, is the leader election process guaranteed to converge within bounded time under worst-case assumptions and all faults within the specified fault model?

- If system start-up is inhibited in the presence of hardware failures (e.g., start-up is precluded with a network fault on one redundant network), is the risk of system unavailability after an in-flight restart mitigated?

- Orthogonal: OK

- Bins: OK

Criterion 14 Global Synchronization: The synchronization and interaction constraints of systems must have timing bounds and dependencies for reasonable failure scenarios as well as the effects resulting from synchronization such as perceptions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 15 Fault Diagnosis: Any diagnosis, detection, or system-level agreement mechanisms must justify fault assumptions and consider effects of diagnosis action.

- Untested

Criterion 16 Client Effect on Network Operations: Can a network client adversely effect network operations?

- Understandable: OK

- Measurable: OK

- Loopholes: OK

- Supporting: Consider adding the following question:

  Do any protocol operations, such as start-up, require behaviors or constraints on behaviors of the network clients (and if so, are such behavioral requirements or constraints ensured in the system design)?

- Orthogonal: OK

- Bins: OK

Criterion 17 Acknowledgement: Does the acknowledgement mechanism work adequately under all fault scenarios?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Consider adding this question:

  If acknowledgements are not supported, does the protocol support aging or staleness indications for periodic messages that have not been received for more than a period?

- Orthogonal: OK
- Bins: OK

F.5.4 APPLICATION-LAYER SERVICES.

Criterion 18 Host Interface Management: The protocol's interface to the host application must provide promised prioritization, latency, and loss prevention of messages for supported categories of service.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 19 Support for Application-Layer Redundancy: Any support for application-layer redundancy must fully support stated redundancy management mechanisms for a specified fault model.

- Understandable: OK
- Measurable: OK
- Loopholes: This does not appear to address handling of identical messages sent over redundant buses.
- Supporting: Consider adding:
  - If the same message information is sent over redundant buses, are those redundant copies crosschecked, or is the first seemingly valid message used?

- If the same message information is sent over redundant buses, are those copies sent at the same time or staggered to mitigate against correlated network disturbances?

- Orthogonal: OK

- Bins: OK

Criterion 20 Time Service for Time Stamping and Time Interrupt: If time stamping and time-interrupt services are provided, do they provide dependable and correct time services?

- Untested

Criterion 21 Robust Partitioning (ARINC 651): If the network is required to provide robust partitioning guarantees, what has been done to substantiate any partitioning claims?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.5.5 FAULT-TOLERANCE MECHANISMS.

Criterion 22 Topological Fault Tolerance: Redundant network components must be physically separated and isolated to prevent correlated outages due to physical equipment damage and credible electrical faults.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 23 Guardian Schemes: Some technique, such as network guardians, must be used to ensure that a single point client failure will not take down the network.

- Understandable: OK

- Measurable: OK

- Loopholes: OK

- Supporting: Add: If the network supports mixed topologies with guardians on only a subset of the topologies, do the guardians provide the required protection?

- Orthogonal: OK

- Bins: OK

Criterion 24 Protocol Logic Fault Tolerance: The protocol must ensure that errors in protocol logic and protocol state do not result in unacceptable reduction of integrity and availability.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 25 Protocol Transmission Monitoring and Self-Checking Schemes: The protocol must reliably detect transient and permanent faults in both nodes and message transmissions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 26 Reconfiguration and Degraded Operation: Capabilities provided in the presence of a specified number and type of faults must be sufficient to meet operational requirements.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 27 Latent Failure Detection: Latent faults must not result in network failure.

- Understandable: OK

- Measurable: OK

- Loopholes: OK

- Supporting: FlexRay's guardian is quite complicated. Should failure detection, isolation, and recovery (FDIR) mechanisms, which need to be detectable in reasonable time and with sufficient coverage, be addressed?

- Orthogonal: OK

- Bins: OK

Criterion 28 Voting, Selection, or Agreement Services and Redundancy Management: The network protocol must support the ability to determine which nodes are part of the active network quorum.

- Untested

Criterion 29 Fault Model: The protocol must be evaluated with respect to a precisely and completely stated fault model, and the fault model must be compatible with that of the system architecture.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.5.6 DESIGN ASSURANCE.

Criterion 30 Design Assurance Processes: Have appropriate design assurance processes been followed for design and deployment of the network? (or) The hardware and software design should be demonstrated to satisfy published regulatory guidelines.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 31 Availability of Standards and Conformance Evidence: The technology and protocol of the network should be clearly specified and be analyzable.

- Understandable: OK
- Measurable: OK
- Loopholes: OK

- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 32 Design Margin: Required design margins should be supported by reviewable evidence.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 33 Configuration Table Correctness and Performance Justification: Justification for configuration table correctness and performance justification should be provided.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 34 Network Monitoring and Test Equipment: Are there adequate test equipment, network access points, mechanisms, and procedures to ensure that the network is healthy and configured correctly?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.5.7 SECURITY.

Criterion 35: Can security weaknesses adversely affect network dependability (safety)?

- Understandable: It is unclear if the traditional security-by-obscurity argument will be detected by this question—some claim that the use of a non-Ethernet protocol makes their system secure, which is of dubious value.

- Measurable: OK

- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 36: If needed, does the network deal adequately with the security issues of privacy (also known as confidentiality or secrecy), integrity, authentication, or authorization?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.6 SAE AS5643/IEEE 1394B (FIREWIRE).

F.6.1 PHYSICAL LAYER.

Criterion 1 Environment: Does the network specification(s) and available components allow for the creation of a network that meets the applicable aerospace environment requirements of DO-160 or other imposed environmental requirements?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 2 Line-Level Encoding: Was the electromagnetic compatibility testing (e.g., DO-160 Sections 19-22) done with the actual data line encoding (e.g., Manchester, 8B/10B), worst-case network data, worst-case message sizes, and worst-case message repetition rate(s)?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 3 Probability of Bit Errors: Was the bit error rate determined under worst-case conditions?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 4 Probability of Electrical Component Failures: Do the data network's electronic components have established hardware failure rates (permanent and transient) and characteristics so that avionics designers can do the required FMEA and fault tree calculations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 5 Electrical Isolation Properties: Is there sufficient protection against electrical fault propagation?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 6 Physical Composability: Does the data network have characteristics or design rules that will guarantee that it will reliably work with any number of nodes up to some explicitly given maximum number of nodes?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.6.2 DATA LINK LAYER.

Criterion 7 MAC: The MAC sublayer protocol must provide appropriately small bounds on message delivery times regardless of likely faults.

- Understandable: Should add a clarification for the reconfiguration time due to a nontransient failure. Can a system temporarily exceed the small bounds on message delivery times during such reconfiguration? If so, then the duration of this excursion must be limited.
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 8 Message Formatting (Framing): Message framing must ensure that only complete, properly synchronized messages are accepted at clients, and that improper synchronization is recovered from in bounded time.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 9 Error Detection: The data link layer must provide sufficient guarantees of error-free message delivery.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.6.3 NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

Criterion 10 Network Vulnerability to Addressing Information Failure: Mechanisms should ensure correct forwarding, routing, or conversion failures despite likely failure scenarios of network components.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: This is so esoteric, but not improbable for some networks, that it should be added as a supporting question: Can addressing or routing errors cause lost packets or fragments to circulate on the media and inordinately consume needed bandwidth?
- Orthogonal: OK
- Bins: OK

Criterion 11 Impact of Intermediate Stages: Intermediate stages (e.g., repeaters, gateways, routers, and switches) must guarantee sufficient availability and integrity requirements as well as sufficient logical and physical independence from other replicated intermediate stages to ensure correct operation in adequate failure scenarios.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 12 Network Configuration Data: Network topology and component configuration data must be correct with respect to the applications' fault tolerance and performance requirements considering table production, loading, errors during operation, run-time environmental effects (e.g., impact of radiation), and maintenance actions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 13 Start-up and Recovery: Component and network integration, start-up, and recovery must be performed in bounded time considering dependability requirements, environmental and deployment constraints, and interactions with different systems and applications (such as application-level timing impact and power architecture influence).

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 14 Global Synchronization: The synchronization and interaction constraints of systems must have timing bounds and dependencies for reasonable failure scenarios as well as the effects resulting from synchronization, such as perceptions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 15 Fault Diagnosis: Any diagnosis, detection, or system-level agreement mechanisms must justify fault assumptions and consider effects of diagnosis action.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 16 Client Effect on Network Operations: Can a network client adversely effect network operations?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 17 Acknowledgement: Does the acknowledgement mechanism work adequately under all fault scenarios?

- Understandable: Should add a qualifier? Acknowledgements need to work only if there is an acknowledgement service. Or, is the "not applicable to this network in general" result sufficient?
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.6.4 APPLICATION-LAYER SERVICES.

Criterion 18 Host Interface Management: The protocol's interface to the host application must provide promised prioritization, latency, and loss prevention of messages for supported categories of service.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 19 Support for Application-Layer Redundancy: Any support for application-layer redundancy must fully support stated redundancy management mechanisms for a specified fault model.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 20 Time Service for Time Stamping and Time Interrupt: If time stamping and time-interrupt services are provided, do they provide dependable and correct time services?

- Understandable: OK

- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 21 Robust Partitioning (ARINC 651): If the network is required to provide robust partitioning guarantees, what has been done to substantiate any partitioning claims?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.6.5 FAULT-TOLERANCE MECHANISMS.

Criterion 22 Topological Fault Tolerance: Redundant network components must be physically separated and isolated to prevent correlated outages due to physical equipment damage and credible electrical faults.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 23 Guardian Schemes: Some techniques, such as network guardians, must be used to ensure that a single point client failure will not take down the network.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 24 Protocol Logic Fault Tolerance: The protocol must ensure that errors in protocol logic and protocol state do not result in unacceptable reduction of integrity and availability.

- Understandable: OK

- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 25 Protocol Transmission Monitoring and Self-Checking Schemes: The protocol must reliably detect transient and permanent faults in both nodes and message transmissions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 26 Reconfiguration and Degraded Operation: Capabilities provided in the presence of a specified number and type of faults must be sufficient to meet operational requirements.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 27 Latent Failure Detection: Latent faults must not result in network failure.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 28 Voting, Selection, or Agreement Services and Redundancy Management: The network protocol must support the ability to determine which nodes are part of the active network quorum.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK

- Orthogonal: OK
- Bins: OK

Criterion 29 Fault Model: The protocol must be evaluated with respect to a precisely and completely stated fault model, and the fault model must be compatible with that of the system architecture.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.6.6 DESIGN ASSURANCE.

Criterion 30 Design Assurance Processes: Have appropriate design assurance processes been followed for design and deployment of the network? (or) The hardware and software design should be demonstrated to satisfy published regulatory guidelines.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 31 Availability of Standards and Conformance Evidence: The technology and protocol of the network should be clearly specified and be analyzable.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 32 Design Margin: Required design margins should be supported by reviewable evidence.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK

- Bins: OK

Criterion 33 Configuration Table Correctness and Performance Justification: Justification for configuration table correctness and performance justification should be provided.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 34 Network Monitoring and Test Equipment: Does there exist adequate test equipment, network access points, mechanisms, and procedures to ensure that the network is healthy and configured correctly?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.6.7 SECURITY.

Criterion 35: Can security weaknesses adversely affect network dependability (safety)?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 36: If needed, does the network deal adequately with the security issues of privacy (also known as confidentiality or secrecy), integrity, authentication, or authorization?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.7 TTP/C.

F.7.1 PHYSICAL LAYER.

Criterion 1 Environment: Does the network specification(s) and available components allow for the creation of a network that meets the applicable aerospace environment requirements of DO-160 or other imposed environmental requirements?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 2 Line-Level Encoding: Was the electromagnetic compatibility testing (e.g., DO-160 Sections 19-22) done with the actual data line encoding (e.g., Manchester, 8B/10B), worst-case network data, worst-case message sizes, and worst-case message repetition rate(s)?

- Understandable: Section 19-22 is understandable, but spelling out the issues addressed would help. Also, which version of DO-160?
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 3 Probability of Bit Errors: Was the bit error rate determined under worst-case conditions?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 4 Probability of Electrical Component Failures: Do the data network's electronic components have established hardware failure rates (permanent and transient) and characteristics so that avionics designers can do the required FMEA and fault tree calculations?

- Understandable: The supporting questions seem not to be related to the criteria. However, the reason for these questions are to stretch the conventional wisdom (which is already embodied in the criteria itself) to include aging and metastability effects that are often overlooked.
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 5 Electrical Isolation Properties: Is there sufficient protection against electrical fault propagation?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 6 Physical Composability: Does the data network have characteristics or design rules that will guarantee that it will reliably work with any number of nodes up to some explicitly given maximum number of nodes?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.7.2 DATA LINK LAYER.

Criterion 7 MAC: The MAC sublayer protocol must provide appropriately small bounds on message delivery times regardless of likely faults.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Should a question or comment be included that asked for the fault hypothesis or typical failures and failure rates that are expected? This requires an applicant or user to specify the likely faults and also allows a quantifiable evaluation of the results provided.
- Orthogonal: OK
- Bins: OK

Criterion 8 Message Formatting (Framing): Message framing must ensure that only complete, properly synchronized messages are accepted at clients, and that improper synchronization is recovered from in bounded time.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 9 Error Detection: The data link layer must provide sufficient guarantees of error-free message delivery.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Add: If messages use frame check sequence (FCS) hidden data (data that is included in the FCS calculation but not transmitted as part of the message), has there been an accounting for the loss of FCS coverage?
- Orthogonal: OK
- Bins: OK

F.7.3 NETWORK LAYER, TRANSPORT LAYER, AND NETWORK MANAGEMENT.

Criterion 10 Network Vulnerability to Addressing Information Failure: Mechanisms should ensure correct forwarding, routing, or conversion failures despite likely failure scenarios of network components.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 11 Impact of Intermediate Stages: Intermediate stages (e.g., repeaters, gateways, routers, and switches) must guarantee sufficient availability and integrity requirements as well as sufficient logical and physical independence from other replicated intermediate stages to ensure correct operation in adequate failure scenarios.

- Understandable: Too many questions?
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 12 Network Configuration Data: Network topology and component configuration data must be correct with respect to the applications' fault tolerance and performance requirements considering table production, loading, errors during operation, run-time environmental effects (e.g., impact of radiation), and maintenance actions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 13 Start-up and Recovery: Component and network integration, start-up, and recovery must be performed in bounded time considering dependability requirements, environmental and deployment constraints, interactions with different systems and applications (such as application-level timing impact and power architecture influence).

- Understandable: OK

- Measurable: OK

- Loopholes: OK
- Supporting: OK
- Orthogonal: Start-up and clock synchronization are very intertwined on most synchronous networks.
- Bins: OK

Criterion 14 Global Synchronization: The synchronization and interaction constraints of systems must have timing bounds and dependencies for reasonable failure scenarios, as well as for the effects resulting from synchronization such as perceptions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 15 Fault Diagnosis: Any diagnosis, detection, or system-level agreement mechanisms must justify fault assumptions and consider effects of diagnosis action.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 16 Client Effect on Network Operations: Can a network client adversely effect network operations?

- Understandable: OK
- Measurable: OK
- Loopholes: Should we include start-up availability?
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 17 Acknowledgement: Does the acknowledgement mechanism work adequately under all fault scenarios?

- Understandable: OK
- Measurable: OK

- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.7.4 APPLICATION-LAYER SERVICES.

Criterion 18 Host Interface Management: The protocol's interface to the host application must provide promised prioritization, latency, and loss prevention of messages for supported categories of service.

- Understandable: The definition of host interface should be broadened to include network gateways as well.
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 19 Support for Application-Layer Redundancy: Any support for application-layer redundancy must fully support stated redundancy management mechanisms for a specified fault model.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Should this question be deleted from this criterion: "Is the design of self-checking pairs backed by an assurance process?" Seems to belong to chapter 8.
- Orthogonal: See immediately above
- Bins: OK

Criterion 20 Time Service for Time Stamping and Time Interrupt: If time stamping and time-interrupt services are provided, do they provide dependable and correct time services?

- Understandable: OK
- Measurable: OK

- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 21 Robust Partitioning (ARINC 651): If the network is required to provide robust partitioning guarantees, what has been done to substantiate any partitioning claims?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Need questions that elaborate how software or hardware faults are addressed with partitioning. For example, does the network adequately support partitioning against hardware component failures? Does the network adequately ensure partitioning against software faults. Does the partitioning against software faults include vulnerabilities resulting from timing faults? Does the partitioning against software faults include network-related software (such as drivers or software support of start-up, integration, or voting)?
- Orthogonal: OK
- Bins: OK

F.7.5 FAULT-TOLERANCE MECHANISMS.

Criterion 22 Topological Fault Tolerance: Redundant network components must be physically separated and isolated to prevent correlated outages due to physical equipment damage and credible electrical faults.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Can common network resources (such as switches) been placed at adequate distances from each other so as not to be vulnerable, but to be independent?" May want to include physical damage or spatial proximity fault in question.
- Orthogonal: OK
- Bins: OK

Criterion 23 Guardian Schemes: Some technique, such as network guardians, must be used to ensure that a single point client failure will not take down the network.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: Include question on whether guardian supports all operational modes (such as start-up).
- Orthogonal: What makes a guardian different from any other coverage scheme such as command and monitor? Where are the general coverage questions? Is this criterion orthogonal to other coverage schemes.
- Bins: OK

Criterion 24 Protocol Logic Fault Tolerance: The protocol must ensure that errors in protocol logic and protocol state do not result in unacceptable reduction of integrity and availability.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 25 Protocol Transmission Monitoring and Self-Checking Schemes: The protocol must reliably detect transient and permanent faults in both nodes and message transmissions.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: How is this different from criterion addressing guardians?
- Bins: OK

Criterion 26 Reconfiguration and Degraded Operation: Capabilities provided in the presence of a specified number and type of faults must be sufficient to meet operational requirements.

- Understandable: OK
- Measurable: OK

- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 27 Latent Failure Detection: Latent faults must not result in network failure.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 28 Voting, Selection, or Agreement Services and Redundancy Management: The network protocol must support the ability to determine which nodes are part of the active network quorum.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: "Have temporal effects been evaluated in the selection?" Should be completely rewritten and further explained. For example, was the allowable lag in getting consensus on the net or quorum? What vulnerabilities exist while the quorum is inconsistent?
- Orthogonal: OK
- Bins: OK

Criterion 29 Fault Model: The protocol must be evaluated with respect to a precisely and completely stated fault model, and the fault model must be compatible with that of the system architecture.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.7.6 DESIGN ASSURANCE.

Criterion 30 Design Assurance Processes: Have appropriate design assurance processes been followed for design and deployment of the network? (or) The hardware and software design should be demonstrated to satisfy published regulatory guidelines.

- Understandable: Suggest replacing the question with the declarative form for this criterion.
- Measurable: OK
- Loopholes: Clarifications of commercial off-the-shelf software (COTS) classifications may be useful. This may be required at the component and subcomponent level; i.e., how should we treat intellectual property (IP) blocks such as random access memories (RAMs), phase-locked loops (PLL) and reset blocks?
- Supporting: Add: Is there a plan for hardware aspects for certification (PHAC) and a plan for software aspects for certification (PSAC) for the network infrastructure?
- Orthogonal: OK
- Bins: Mixed designs such as COTS complicate this assignment.

Criterion 31 Availability of Standards and Conformance Evidence: The technology and protocol of the network should be clearly specified and be analyzable.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 32 Design Margin: Required design margins should be supported by reviewable evidence.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 33 Configuration Table Correctness and Performance Justification: Justification for configuration table correctness and performance justification should be provided.

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 34 Network Monitoring and Test Equipment: Does there exist adequate test equipment, network access points, mechanisms, and procedures to ensure that the network is healthy and configured correctly?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

F.7.7 SECURITY.

Criterion 35: Can security weaknesses adversely affect network dependability (safety)?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK

Criterion 36: If needed, does the network deal adequately with the security issues of privacy (also known as confidentiality or secrecy), integrity, authentication, or authorization?

- Understandable: OK
- Measurable: OK
- Loopholes: OK
- Supporting: OK
- Orthogonal: OK
- Bins: OK