SAE Aerospace
An SAE International Group

AEROSPACE INFORMATION REPORT

SAE AIR5532

Issued 2003-06

Generic Aircraft-Store Interface Framework (GASIF)

FOREWORD

The Generic Aircraft-Store Interface Framework (GASIF) is a common framework for modeling and defining aircraft-store logical interfaces. Its purpose is to assist and harmonize aircraft-store interface standardization and to provide a common reference point for the development and comparison of aircraft-store interface control documents (ICDs). The GASIF framework is backward compatible with the Aircraft-store Electrical Interconnection System (AEIS) Interface Standard, MIL-STD-1760. The primary goal of GASIF, however, is to address the system environment pertaining to aircraft-store logical interoperability.

GASIF complies with the OSI Basic Reference Model (ITU-T Rec. X.200 | ISO/IEC 7498-1) in that it describes operations and mechanisms which are assignable to layers as specified in the OSI Basic Reference Model. GASIF is also compatible with the SAE Generic Open Architecture (GOA) Framework (AS4893) and with SAE prior work relating to the modeling and specification of Real Time Communication Protocols (RTCP).

This document was prepared by the AS-1B3 Task Group, Aircraft-Store System Integration, under the jurisdiction of the AS-1B Subcommittee, Aircraft-Store Integration, of the SAE AS-1 Committee, Aircraft Systems and System Integration.

TO PLACE A DOCUMENT ORDER: **Tel: 877-606-7323 (inside USA and Canada)**
Tel: 724-776-4970 (outside USA)
Fax: 724-776-0790
Email: custsvc@sae.org
SAE WEB ADDRESS: http://www.sae.org

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

1. SCOPE:

This SAE Aerospace Information Report (AIR) defines a Generic Aircraft-Store Interface Framework (GASIF). This is a common framework for modeling and specifying aircraft-store logical interfaces. GASIF complies with the OSI Basic Reference Model (ITU-T Rec. X.200 | ISO/IEC 7498-1) in that it describes operations and mechanisms which are assignable to layers as specified in the OSI Basic Reference Model.

This AIR provides a mapping of the Interface Standard for Aircraft-store Electrical Interconnection System (AEIS), MIL-STD-1760, in Appendix C.

1.1 Purpose:

The purpose of GASIF is to foster greater interoperability between aircraft and stores through standards coordination and common approaches to aircraft-store interface control documents (ICD).

1.2 Field of Application:

The field of application for GASIF is military aircraft/store avionics, where a ‘store’ is a device (such as an air-launched weapon) that when loaded on an aircraft allows it to perform a specific mission.

2. REFERENCES:

2.1 Applicable Documents:

The following publications form a part of this document to the extent specified herein. The latest issue of SAE publications shall apply. The applicable issue of other publications shall be the issue in effect on the date of the purchase order. In the event of a conflict between the text of this document and references cited herein, the text of this document takes precedence. Nothing in this document, however, supersedes applicable laws and regulations unless a specific exemption has been obtained.

2.1.1 SAE Publications: Available from SAE, 400 Commonwealth Drive, Warrendale, PA 155096-0001.

AS4893 Generic Open Architecture (GOA) Framework
AS15531 Digital Time Division Command/Response Multiplex Data Bus

2.1.2 ANSI Publications: Available from ANSI, 25 West 43rd Street, New York, NY 10036-8002.

ISO/IEC 7498-1 Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model
ISO 8648 Information processing systems – Open Systems Interconnection – Internal Organization of the Network Layer

2.1.3 U.S. Government Publications: Available from DODSSP, Subscription Services Desk, Building 4D, 700 Robbins Avenue, Philadelphia, PA 19111-5094.

MIL-STD-1760 Interface Standard for Aircraft-store Electrical Interconnection System (AEIS)

2.2 Terminology:

(N)-LAYER: A subdivision of the OSI architecture, as defined in ISO/IEC 7498-1, constituted by subsystems of the same rank (N) [derived from ISO/IEC 7498-1].

The (N)-, (N+1)- and (N-1)- notation is used to identify and relate adjacent layers. The term (N)-layer applies to any specific layer; the term (N+1)-layer is the next higher layer; the term (N-1)-layer is the next lower layer.

2.3 Definitions:

ACKNOWLEDGEMENT: A function of the (N)-layer that allows a receiving (N)-entity to inform a sending (N)-entity of the receipt a (N)-protocol-data-unit [derived from ISO/IEC 7498-1].

APPLICATION-PROCESS: An element within a real open system that performs the information processing for a particular application [derived from ISO/IEC 7498-1].

CARRIAGE STORE: A type of store that meets the requirements of a MIL-STD-1760 Carriage Store.

(N)-CONFIRMATION-PRIMITIVE: (N)-service-primitive issued by a (N)-service to confirm completion of a service.

(N)-CONNECTION: An association requested by an (N+1)-entity for the transfer of data between two or more (N+1)-entities. The association is established by the (N)-layer and provides explicit identification of a set of (N)-data-transmissions and agreement concerning the (N)-data-transmission services to be provided for the set. A (N)-association is a cooperative relationship among invocations of (N)-entities [derived from ISO/IEC 7498-1].

(N)-CONNECTION-MODE-TRANSMISSION: The conveyance of (N)-service-data-units from one (N+1)-entity to one or more (N+1)-entities in the context of a (N)-connection [derived from ISO/IEC 7498-1].

(N)-CONNECTIONLESS-MODE-TRANSMISSION: The conveyance of (N)-service-data-units from one (N+1)-entity to one or more (N+1)-entities not in the context of a (N)-connection, and not required to maintain any logical relationship between (N)-service-data-units [derived from ISO/IEC 7498-1].

CORRESPONDENT (N)-SERVICE-USER: The (N)-service-user that receives a (N)-indication-primitive as a result of a peer (N)-service-user issuing a (N)-request-primitive.

2.3 (Continued):

DISPENSER: A type of store that meets the requirements of a MIL-STD-1760 Dispenser.

END SYSTEM: A system that is the source or sink of user-data and does not provide a relay function [derived from ISO/IEC 7498-1].

(N)-ENTITY: An active element within a system embodying a set of capabilities defined for the (N)-layer [derived from ISO/IEC 7498-1].

EXPEDITED (N)-SERVICE-DATA-UNIT: A small (N)-service-data-unit whose transfer is expedited. The (N)-layer ensures that an expedited-data-unit will not be delivered after any subsequent (N)-service-data-unit or expedited unit sent on that connection [derived from ISO/IEC 7498-1].

FLOW CONTROL: A function that controls the flow of data within a layer or between adjacent layers [derived from ISO/IEC 7498-1].

(N)-INDICATION-PRIMITIVE: (N)-service-primitive issued by the (N)-service to indicate that a particular service has been requested by the (N)-service-user or to indicate than an event has occurred within the (N)-service.

(N)-INSTRUCTION-PRIMITIVE: (N)-service-primitive issued by the (N)-service-user to specify to the (N)-service what to do in case of particular events.

LATENCY DELAY: The time that can elapse between the (N)-service receiving a (N)-request-primitive from the (N)-service-user and receiving the logically related (N)-response-primitive from the correspondent (N)-service-user.

LAXITY: The maximum allowable time that can elapse between the start and completion of a confirmed (N)-service-procedure.

MISSION STORE: A type of store that meets the requirements of a MIL-STD-1760 Mission Store.

NETWORK ENVIRONMENT, NE: An abstract representation of a set of concepts, elements, functions, services, protocols, etc., as defined by the OSI Reference Model, ISO/IEC 7498, and the derived specific standards which, when applied, enable a Transport Service among open systems

OPEN SYSTEM: The representation of those aspects of a real open system that are pertinent to OSI [derived from ISO/IEC 7498-1].

OPEN SYSTEM INTERCONNECTION ENVIRONMENT (OSIE): An abstract representation of a set of concepts, elements, functions, services, protocols, etc., as defined by the OSI Reference Model, ISO/IEC 7498, and the derived specific standards which, when applied, enable communications among open systems [derived from ISO/IEC 7498-1].

PEER (N)-ENTITIES: Entities within the same (N)-layer [derived from ISO/IEC 7498-1].

2.3 (Continued):

(N)-PROTOCOL: A set of rules and formats (semantic and syntactic) that determines the communication behavior of (N)-entities [derived from ISO/IEC 7498-1].

(N)-PROTOCOL-CONTROL-INFORMATION: Information exchanged between (N)-entities to co-ordinate their joint operation [derived from ISO/IEC 7498-1].

(N)-PROTOCOL-DATA-UNIT: A unit of data specified in a (N)-protocol and consisting of (N)-protocol-control-information and possible (N)-user-data [derived from ISO/IEC 7498-1].

REAL SYSTEM: A set of one or more systems, the associated software, physical resources, physical processes, human operators, etc., that forms an autonomous whole capable of performing information processing and/or information transfer [derived from ISO/IEC 7498-1].

REAL SYSTEM ENVIRONMENT, RSE: An abstract representation of the external interface behavior of a real system (or real open system).

REAL OPEN SYSTEM: A real system that complies with the requirements of OSI standards in its communication with other real systems [derived from ISO/IEC 7498-1].

RE-ASSEMBLING: A function performed by a (N)-entity to map multiple (N)-protocol-data-units into one (N)-service-data-unit. It is the reverse function of segmenting [derived from ISO/IEC 7498-1].

(N)-RELAY: A (N)-function by means of which a (N)-entity forwards data received from one peer-(N)-entity to another peer-(N)-entity [derived from ISO/IEC 7498-1].

(N)-REQUEST-PRIMITIVE: (N)-service-primitive issued by the (N)-service-user to the (N)-service in order to request a particular service.

(N)-RESPONSE-PRIMITIVE: (N)-service-primitive issued by the correspondent (N)-service-user to the (N)-service in response to a (N)-indication-primitive issued by the (N)-service.

RESPONSE TIME: The time between the (N)-service issuing a (N)-indication-primitive and the (N)-service-user issuing the logically related (N)-response-primitive.

ROUTING: A function within a layer that translates the title of an entity, or the service-access-point-address to which the entity is attached, into a path by which the entity can be reached. The (N)-entity-title is a name that is used to identify unambiguously a (N)-entity. The (N)-SAP-address is a (N)-address that is used to identify a single (N)-SAP [derived from ISO/IEC 7498-1].

SEGMENTING: A function performed by a (N)-entity to map one (N)-service-data-unit into multiple (N)-protocol-data-units [derived from ISO/IEC 7498-1].

(N)-SERVICE: A capability of the (N)-layer and the layers beneath it, which is provided to (N+1)-entities at the boundary between the (N)-layer and the (N+1)-layer [derived from ISO/IEC 7498-1].

2.3 (Continued):

(N)-SERVICE-ACCESS-POINT, (N)-SAP: The point at which (N)-services are provided by an (N)-entity to an (N+1)-entity [derived from ISO/IEC 7498-1].

(N)-SERVICE-DATA-UNIT: An amount of information whose identity is preserved when transferred between peer-(N+1)-entities and which is not interpreted by the supporting (N)-entities [derived from ISO/IEC 7498-1].

SERVICE PARAMETER: An implementation parameter associated with a particular (N)-service-primitive.

(N)-SERVICE-PRIMITIVE: A single interaction between the (N)-service and the (N)-service-user. (N)-service-primitives are conceptual in nature and are independent of any kind of interface control implementation. One of more service parameters may be associated with an N-service-primitive.

(N)-SERVICE-PROCEDURE: A sequence of one or more (N)-service primitives that are logically related.

(N)-SERVICE-USER: (N+1)-entity that uses a facility of the (N)-service.

STORE: A device that is loaded on a military aircraft on a non-permanent basis to allow it to perform a specific mission. Stores may be either separable from the aircraft in flight (deployed or jettisoned) or non-separable from the aircraft in flight.

SUBLAYER: A subdivision of a layer [derived from ISO/IEC 7498-1].

SUBNETWORK: An abstraction of a collection of equipment and physical media that forms an autonomous whole and which can be used to interconnect real systems for the purpose of data transfer. A subnetwork may itself be an open system, although this is not necessarily always the case. See Internal Organization of the Network Layer, ISO 8648 [derived from ISO/IEC 7498-1].

SUBNETWORK ACCESS PROTOCOL, SNAcP: A Network Sublayer that provides (with the layers underneath) a subnetwork service that does not necessary comply with OSI requirements [derived from ISO 8648].

SUBNETWORK-DEPENDENT CONVERGENCE PROTOCOL, SNDCP: A Network Sublayer that uses the SNAcP Service to provide a subnetwork service that does comply with OSI requirements [derived from ISO 8648].

SUBNETWORK-INDEPENDENT CONVERGENCE PROTOCOL: A sublayer of the Network Layer that uses the SNDCP Service to provide a Network Service in accordance with OSI requirements [derived from ISO 8648].

2.3 (Continued):

TRANSIT DELAY: The time that can elapse between the (N)-service receiving a (N)-request-primitive and issuing the logically related (N)-indication-primitive or between the (N)-service receiving a (N)-response-primitive and issuing the logically related (N)-confirmation-primitive.

(N)-USER-DATA: The data transferred between (N)-entities on behalf of the (N+1)-entities for which the (N)-entities are providing services [derived from ISO/IEC 7498-1].

2.4 Symbols and Abbreviations:

AEIS: MIL-STD-1760 Aircraft-store Electrical Interconnection System
CL-NS: Connectionless-mode Network Service
CL-SS: Connectionless-mode Session Service
CL-TS: Connectionless-mode Transport Service
CO-NS: Connection-mode Network Service
CO-TS: Connection-mode Transport Service
CO-SS: Connection-mode Session Service
GASIF: SAE Generic Aircraft-Store Interface Framework
GOA: SAE Generic Open Architecture
ICD: Interface Control Document
NE: Network Environment
(N)-PCI: (N)-layer Protocol Control Information
(N)-PDU: (N)-layer Protocol Data Unit
(N)-SAP: (N)-Service Access Point
(N)-SDU: (N)-Service Data Unit
OSI: Open System Interconnection
OSIE: OSI Environment
RSE: Real System Environment
RTCP: Real Time Communication Protocol
SNAcP: Subnetwork Access Protocol (also SNAc Layer)
SNDCP: Subnetwork-dependent Convergence Protocol (also SNDC Layer)
SNICP: Subnetwork-independent Convergence Protocol (also SNIC Layer)

3. OVERVIEW:

The Generic Aircraft-Store Interface Framework (GASIF) is a common framework for modeling and defining aircraft-store logical interfaces. The systems of interest are the aircraft interface, the store interface and any intervening interconnection system. GASIF is only concerned with the external behavior of these systems and not with the internal implementation. GASIF is therefore a framework for a functionally equivalent abstract model of the aircraft-store interface and is not representative of the system architecture.

A suitable framework for the internal architecture of each system is defined by the Generic Open Architecture (GOA) Framework, AS4893. There is no intrinsic mapping between the GOA Framework and GASIF, however a convergence of the two frameworks is suggested in Appendix A.

3. (Continued):

GASIF organizes the aircraft-store logical interface into three nested environments (see Figure 1). The Real System Environment (RSE) encompasses the complete logical interface. Within the RSE, the aircraft and store application processes exchange data using the services of the Open System Interconnection Environment (OSIE). Within the OSIE, application-oriented protocols exchange protocol-data-units using the services of the Network Environment (NE). The GASIF Open System Interconnection Environment complies with the OSI Basic Reference Model, ISO/IEC 7498-1, in that it describes operations and mechanisms that are assignable to layers as specified in the OSI Basic Reference Model. The Network Environment is further layered in accordance with the OSI Internal Organization of the Network Layer, ISO 8648.

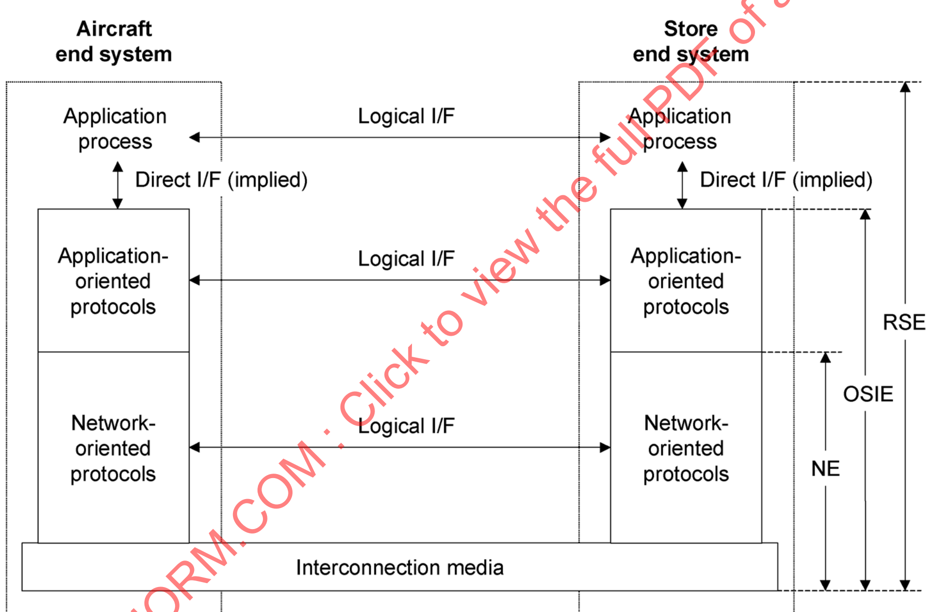

FIGURE 1 - Layered Aircraft-Store Interface

GASIF identifies two types of interface:

a. LOGICAL INTERFACE: An abstract data flow between systems. In a layered framework, logical interfaces exist between peer entities in the same layer. Within the GASIF OSIE, logical interfaces comprise protocol-data-units, such as those defined in MIL-STD-1760 Appendix B.

b. DIRECT INTERFACE: A service/service-user interface between adjacent layers within the same system. Unlike the GOA Framework, GASIF is only concerned with direct interfaces that are implied from the external behavior of the system. Within the GASIF OSIE, direct interfaces are couched in terms of service primitives (and service procedures) which are conceptual in nature and are independent of any kind of internal system architecture or interface implementation.

3. (Continued):

The aircraft-store interface is a real-time application, and therefore the GASIF OSIE must be compatible with a Real-time Communication Protocol (RTCP). The requirements of an RTCP are characterized in Appendix B. To support RTCP, GASIF uses five types of service primitive: (1) request, (2) indication, (3) response, (4) confirmation, (5) instruction. The first four types have already been defined for the standard protocols associated with the OSI Basic Reference Model. The fifth (instruction) has been introduced specifically for RTCP facilities such as remote interrupts and time distribution.

An important requirement of GASIF is to support the modeling of aircraft-store interface standards such as MIL-STD-1760. A high level mapping of the logical element of MIL-STD-1760 to GASIF is provided in Appendix C. Various interconnection scenarios between aircraft and MIL-STD-1760 Mission Stores are presented in Appendix E.

A full modeling of MIL-STD-1760 protocols is beyond the scope of this document.

4. INTERFACE ENVIRONMENTS:

The following terms are introduced in this section, as defined in 2.3: Real System Environment, Open System Interconnection Environment, Network Environment, application processes, end system, relay system.

The aircraft-store logical interface is divided into three nested environments (see Figure 1). The widest environment, covering all aspects of the functional interface, is termed the Real System Environment (RSE). Within the RSE, the aircraft and store application processes communicate data and control information using the services of the Open System Interconnection Environment (OSIE). Within the OSIE, application-oriented protocols are defined, which are shared between (possibly dissimilar) end systems. Data and protocol-control-information created by an application-oriented protocol are transferred between end systems using the services of the Network Environment (NE). The NE is concerned with low-level communication protocols.

The concept of the layered aircraft-store interface when applied to the aircraft carriage of Stores using a Dispenser is shown in Figure 2. This figure shows a particular instance of communication where the end systems are the aircraft and the Dispenser-carried Store, and the Dispenser is a relay system. It is noted that in other instances of communication the Dispenser may be one of the end systems rather than a relay system.

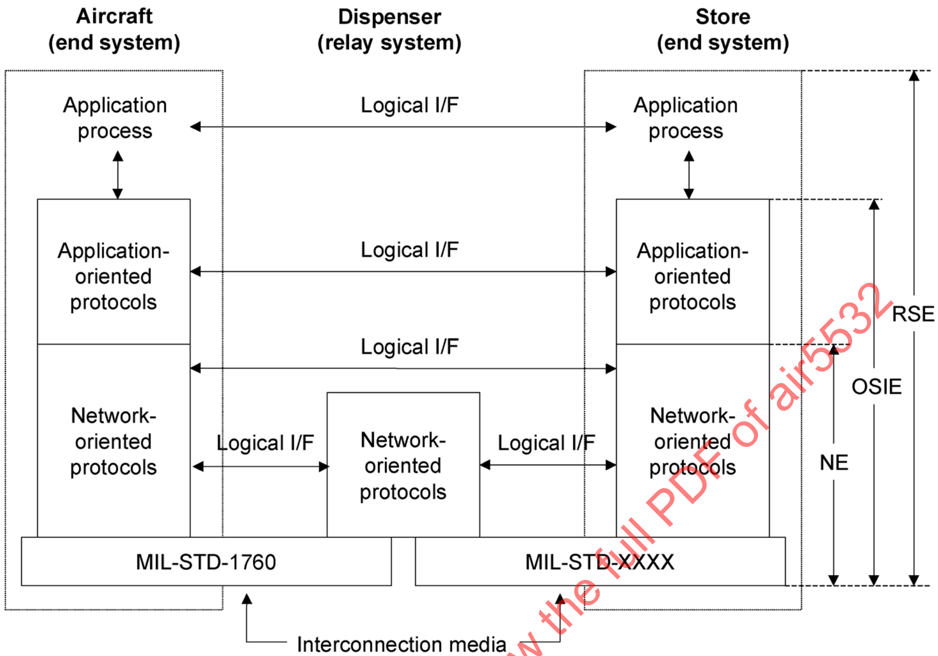

FIGURE 2 - Layered Aircraft-Store ICD for Instance of Communication Between Aircraft and Dispenser-Carried Store

4. (Continued):

The scope of GASIF is the Real System Environment, and therefore it includes the aircraft and store application processes. An application process may include software, pilot activity and system inputs/outputs. Application processes work on data entities (store control parameters), and may be required to exchange these data entities across the aircraft-store interface. The definition of each data entity is within the scope of the Real System Environment, but is above the scope of the OSI Environment.

The OSIE is concerned with the communication of application process data between the aircraft and store. The abstract syntax used to represent data entities in messages or files is defined within the OSIE.

5. GASIF MODELS AND CONCEPTS:

5.1 Open System Models:

The following terms are introduced in this section, as defined in 2.3: real system, real open system, open system.

GASIF is concerned with the logical interoperability of 'real systems': the real systems of interest being aircraft interfaces and store interfaces. Real systems that meet open standards for interoperability are called 'real open systems'. GASIF is intended to be a model for an 'open system', and not a model for a 'real open system'. The difference between an open system model and a real open system model is important, and is addressed in ISO/IEC 7498-1. According to that definition, a real open system model provides a particular view of the real open system implementation (or design). The GOA Framework is an example of a real open system model.

Like MIL-STD-1760, GASIF is only concerned with the external behavior of real open systems, and is not concerned with the implementation. To specify the external behavior of interconnected real open systems, a functionally equivalent abstract model called an open system replaces each real open system. The open system model may bare no relation to the actual design of the real open system, provided the abstract model exhibits the real open system's external behavior.

5.2 Open System Protocol Models:

The following terms are introduced in this section, as defined in 2.3: (N)-layer, (N)-entity, (N)-service, (N)-service-access-point, (N)-protocol-data-unit, (N)-service-data-unit, (N)-service-primitive, (N)-protocol-control-information.

In an open system, the application-oriented protocols shown in Figures 1 (and 2) may be treated either as an indivisible protocol entity or subdivided into a number of protocol layers. The protocol entity in each layer would offer a defined service to the layer above and require services from the layer below. Such subdivision is only necessary if it is of assistance to the open system model and need not reflect any real open system implementation. It may subsequently be useful to further subdivide a specific protocol layer to define its functions more clearly. The same layering principle applies to the network-oriented protocols shown in Figures 1 (and 2), but with certain constraints.

Figure 3 is a generic model for a single protocol layer: the (N)-layer. The (N)-entity, using the services of the (N-1)-layer, provides a specific communication service to the (N+1)-entity. This (N)-service is requested by the (N+1)-entity via a (N)-service-access-point, or (N)-SAP. The purpose of the (N)-layer is to allow the logical communication (N+1)-protocol-data-units, or (N+1)-PDUs, between peer (N+1)-entities in the (N+1)-layer. The meaning of the information contained in the (N+1)-PDUs is of no interest to the (N)-layer or lower layers.

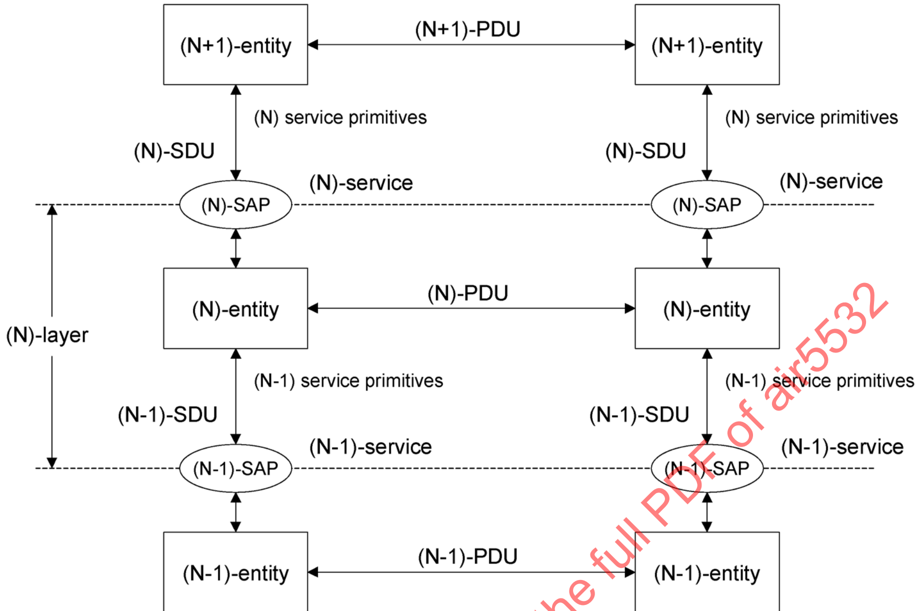

FIGURE 3 - Generic Model of Single (N)-Protocol Layer

5.2 (Continued):

The logical exchange of (N+1)-PDUs is achieved by the (implied) direct exchange of (N)-service-data-units, or (N)-SDUs, via the (N)-SAPs. One (N+1)-PDU equates exactly with one (N)-SDU. The (N)-SDUs are exchanged, together with other parameters, using a defined set of service primitives, which will be discussed later.

The (N)-SDUs are logically exchanged between peer (N)-entities in the (N)-layer through the logical exchange of (N)-PDUs. A (N)-PDU may comprise (N)-protocol-control-information, or (N)-PCI, a (N)-SDU, or both. The logical exchange of (N)-PDUs is achieved by the (implied) direct exchange of (N-1)-SDUs via (N-1)-SAPs using service primitives (in general).

Figure 4 is an open system model showing the application-oriented protocols in Figure 1 subdivided into three protocol layers: arbitrarily denoted (N), (N-1) and (N-2). The network-oriented protocols could be similarly subdivided into a number of protocol layers. It can be seen that the logical interface grows in complexity at the lower layers. The logical interface between application processes is the intended 'information flow' of Data Entities (syntax independent). In Figure 4, this information flow is achieved by the (N)-service. The protocol between the peer (N)-entities is represented by the (N)-PCI. The protocol between peer (N-1)-entities is similarly represented by the (N-1)-PCI and so on. In Figure 4, the complete protocol provided by all the application-oriented protocols is therefore represented by (N)-PCI + (N-1)-PCI + (N-2)-PCI.

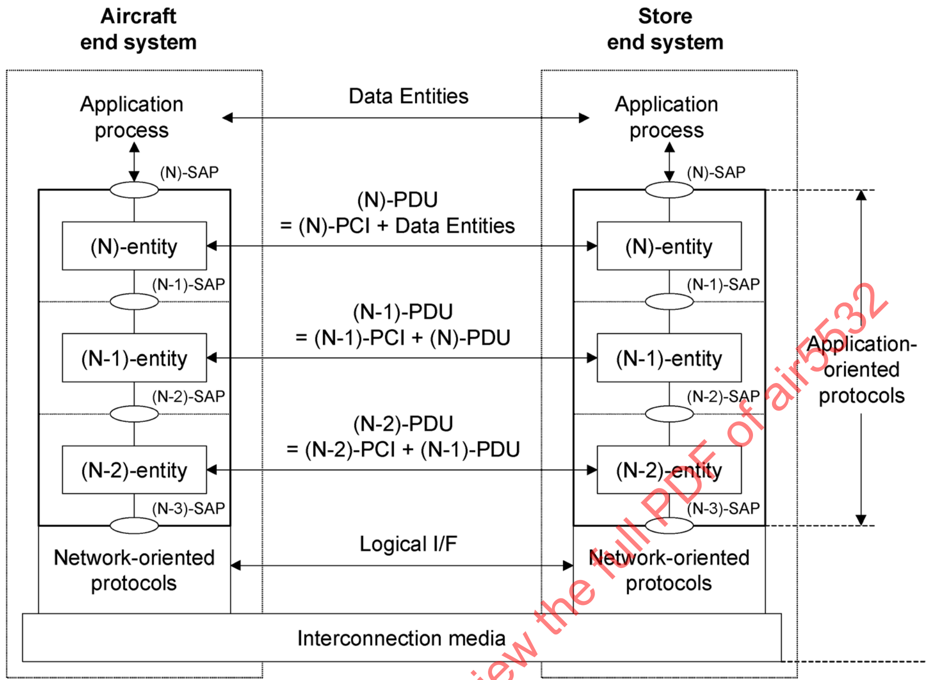

FIGURE 4 - Notional Subdivision of Application-Oriented Protocols

5.3 Principles of Protocol Layering:

The GASIF Open System Interconnection Environment is initially divided into seven layers in accordance with the Basic Reference Model for OSI, ISO/IEC 7498-1. However, the modeling of specific aircraft-store protocols (such as MIL-STD-1760) may justify further subdivisions or the merging of layers. The principles used to determine the optimum layering for a specific protocol should follow these guidelines:

a. Do not create so many layers as to unnecessarily complicate standards or store ICDs.

b. Create a boundary at a point where the description of services can be small and the number of interactions across the boundary are minimized.

c. Create layers to handle functions that are manifestly different in the process performed or the technology involved, or require a different level of abstraction in the handling of data. Conversely, collect similar functions into the same layer.

5.3 (Continued):

d. Create layers that correspond with any off-the-shelf standard that is employed in the logical interface, or past experience has demonstrated to be successful.

e. Create a layer of easily localized functions so that the layer protocol could be totally redefined to take advantage of new advances in architectural, hardware or software technology without changing the services expected from and provided to the adjacent layers.

5.4 Modes of Communication:

The following terms are introduced in this section, as defined in 2.3: (N)-connection, (N)-connection-mode-transmission, (N)-connectionless-mode-transmission, (N)-peer-entities.

GASIF will support two modes of communication: connection mode and connectionless mode, as defined in ISO/IEC 7498-1. Each protocol layer may provide connection mode services, connectionless mode services or both.

5.4.1 Connection Mode: A connection is an association established for the transfer of data between two or more peer protocol entities. This association logically binds the peer protocol entities together using the services in the next lower layer. For example, the Network Environment may support a connection between application-oriented protocol entities. The use of a connection-mode service by peer protocol entities generally proceeds through three phases: connection establishment, data transfer, and connection release. However there are instances in GASIF when it is convenient to model some data transmission services as connection-mode, but the connection is established at system initialization and is not released.

5.4.2 Connectionless Mode: Connectionless-mode transmission is the transmission of a single unit of data between two or more peer protocol entities without establishing a connection. A connectionless-mode service requires the protocol entity initiating the transmission to provide all the information required to deliver the unit of data together with the unit of data to be transmitted.

5.5 (N)-Service Description:

The following terms are introduced in this section, as defined in 2.3: (N)-service-user, correspondent (N)-service-user, (N)-service-procedure, (N)-service-primitive, service parameter, (N)-request-primitive, (N)-indication-primitive, (N)-response-primitive, (N)-confirmation-primitive, (N)-instruction primitive, transit delay, response time, laxity, latency delay.

The following provides a general interface model between the (N)-service and the (N)-service-user. This model is implied from the external behavior of the system and does not necessarily represent a real system implementation. Usually the (N)-service-user is the (N+1)-entity, except when the (N)-service-user is an application process.

5.5.1 Service Procedure: The interaction at the (N)-service-access-point between the (N)-service-user and the (N)-service may be modeled using service primitives, time sequence diagrams and service parameters. Service primitives provide the signaling mechanism for the interaction between the (N)-service-user and the (N)-service. Different types of service primitive are defined depending on the character of the specific interaction. Time sequence diagrams define the discipline and procedure for using the service primitives in the interaction, and service parameters further define specific implementation details.

5.5.2 Service Primitives: Service primitives represent the interaction between the (N)-service-user and the (N)-service. They are conceptual in nature and are independent of any kind of interface control implementation. A service primitive is presented in the form of an event, which is logically instantaneous and indivisible. It indicates the propagation direction, which may be either from the (N)-service-user to the (N)-service or from the (N)-service to the (N)-service-user. One or more service parameters may be associated with a service primitive. For real-time applications, five types of service primitive are defined. The first four types (request, indication, response and confirmation) have already been defined for the standard protocols associated with the Basic Reference Model for OSI. The fifth (instruction) has been introduced here specifically for the real-time store control application.

a. REQUEST: The request primitive is issued by the (N)-service-user to request a particular instance of communication with a correspondent (N)-service-user.

b. INDICATION: The indication primitive is issued by the (N)-service to indicate to the (N)-service-user that either an instance of communication has been requested by a corresponded (N)-service-user, or to indicate that an event has occurred in the network.

c. RESPONSE: The response primitive is issued by the (N)-service-user in response to an indication primitive issued by the (N)-service.

d. CONFIRMATION: The confirmation primitive is issued by the (N)-service to the (N)-service-user to confirm completion of a requested or instructed (N)-service activity.

e. INSTRUCTION: The instruction primitive is used by the (N)-service-user to specify to the (N)-service what to do in case of particular events. Not all instruction primitives in real systems may be visible in open system models.

Figure 5 shows an example of an instance of communication between (N+1)-entities using the (N)-service. Each (N+1)-entity is a (N)-service-user from the perspective of the (N)-service. The term ‘correspondent (N)-service-user’ is relative to the flow of service primitives. In this example, the correspondent (N)-service-user has issued an instruction primitive to the (N)-service that is relevant to the exchange of (N+1)-PDU. This instruction primitive, for instance, might indicate the willingness of the correspondent (N)-service-user to accept an indication primitive.

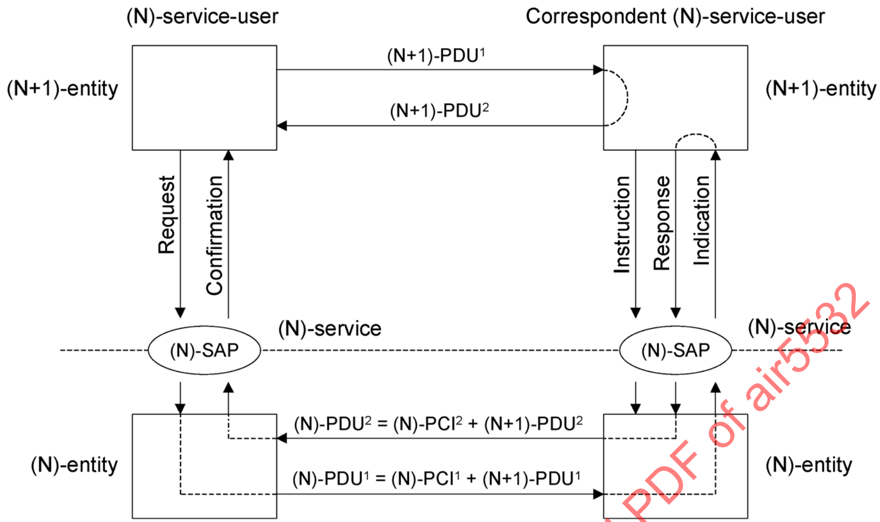

FIGURE 5 - Example of Service Primitives Using Model of Protocol Layer

5.5.2 (Continued):

In this example, at some undefined time later, the (N)-service-user logically transfers (N+1)-PDU^1 to the corespondent (N)-service-user. This is achieved by issuing a request primitive to the (N)-service with an (N)-SDU as one of its parameters. It will be remembered that the (N)-SDU is a direct representation of the logical (N+1)-PDU. The (N)-SDU is conveyed to the correspondent (N)-service-user as a parameter of an indication primitive that is issued by the (N)-service. In some scenarios, this may complete the communication. However, in this example the correspondent (N)-service-user responds with (N+1)-PDU^2. This protocol data unit may be an acknowledgement, i.e. (N+1)-PCI only, or it may contain requested data from the correspondent (N+1)-service-user, i.e. (N+1)-PCI + (N+2)-PDU.

This response is achieved by the correspondent (N)-service-user issuing a response primitive with a second (N)-SDU as a parameter. This (N)-SDU is exchanged with the requesting (N)-service-user by the (N)-service issuing a confirmation primitive. This completes the instance of communication in this example. There are many other scenarios involving service primitives.

5.5.3 Time Sequence Diagrams: Time sequence diagrams are used to illustrate how sequences of service primitives are related in time. A time sequence diagram is partitioned by vertical lines into areas representing (N)-service-users or the (N)-service. Each vertical line represents an (N)-service-access-point. Arrows that cross the (N)-service-access-point indicate the propagation direction of the specific service primitive, i.e., to or from the (N)-service-user. The (N)-service area does not give any indication of the internal mechanisms, which are invisible to all (N)-service-users.

5.5.3 (Continued):

Time sequence diagrams indicate the allowed sequence of events at each (N)-service-access-point, and, where appropriate, the allowed sequence of events at all distributed (N)-SAPs that are solicited to provide the requested (N)-service. At each (N)-SAP, the sequence of events is positioned along lines representing the elapse of time evolving downwards. A causal relationship between service primitives at the same (N)-SAP is symbolized by a dotted loop. A causal relationship between service primitives at different (N)-SAPs symbolized by a dotted line. This line does not presume in any way the protocol implemented in the (N)-layer.

An example of a time sequence diagram is shown in Figure 6. This provides another view of the sequence of service primitives shown in Figure 5. On the example time sequence diagram, the following timing parameters are also defined:

a. TRANSIT DELAY: The time that can elapse between a request primitive being issued at one (N)-SAP and the resulting indication primitive being issued at another (N)-SAP.

b. RESPONSE TIME: The time between an indication primitive being issued to an (N)-service-user and the resulting response primitive being issued by the same (N)-service-user.

c. LAXITY: Defines the maximum allowable time that can elapse between a requested or instructed (N)-service activity and its completion denoted by the (N)-service issuing a confirmation primitive.

d. LATENCY DELAY: The time that can elapse between the request primitive being issued by the (N)-service user and the resulting response primitive being issued by the correspondent (N)-service-user. The latency delay is the transit delay plus the response time.

5.5.4 Service Parameters: Service parameters define the details for the implementation of the action associated with a specific service primitive. Depending on the specific service primitive, different parameters are used. Not all service parameters may be visible to the external interface. Typically, service parameters may specify service topology, quality of service and results reporting.

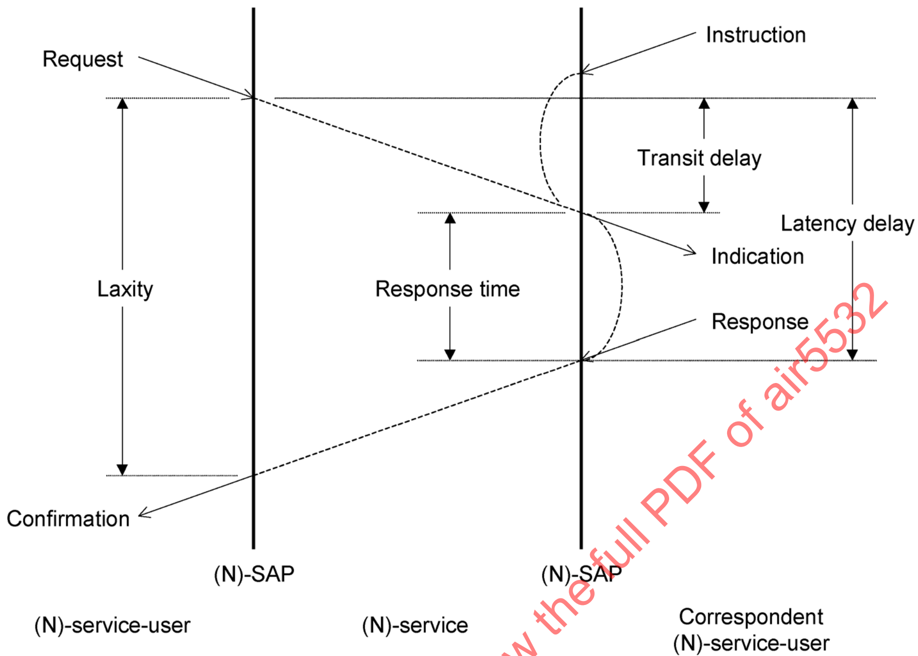

FIGURE 6 - Example of a Time Sequence Diagram

5.6 Acknowledgement Scenarios:

The following term is introduced in this section, as defined in 2.3: acknowledgement.

Different types of acknowledgement can be associated with connection establishment, data transfer and connection release. Generally, an acknowledgement maps to a confirmation primitive being issued by some protocol layer.

The lowest form of acknowledgement occurs entirely within the (N)-service. That is, the (N-1)-service issues a confirmation primitive, but the (N)-service does not issue confirmation primitive to the (N)-service-user. This scenario is depicted in Figures 7 and 8. An initial unsuccessful (N)-PDU transfer between (N)-entities may result in automatic ‘retries’ within the (N)-service. If this retry strategy is unsuccessful then the (N)-service-user may be informed of (N)-service transmission error. Despite error notification, this is considered an unacknowledged (N+1)-PDU transfer however.

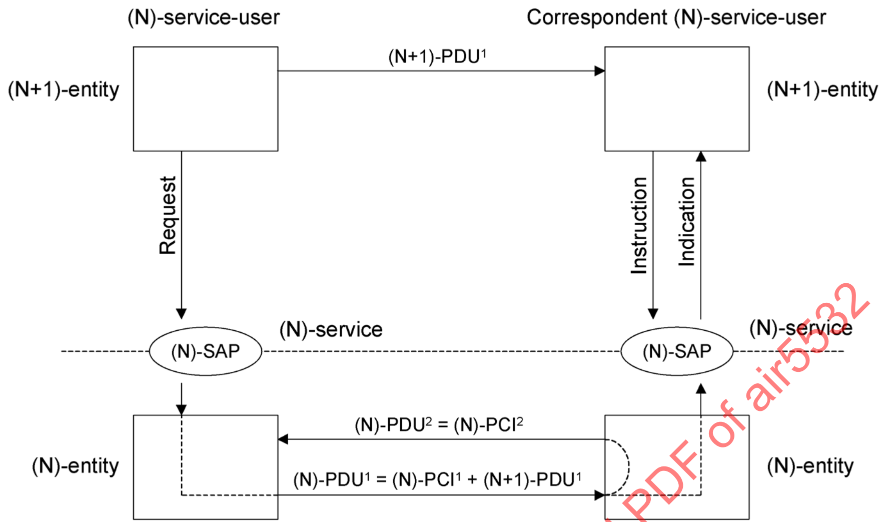

FIGURE 7 - Acknowledgement Within (N)-Service

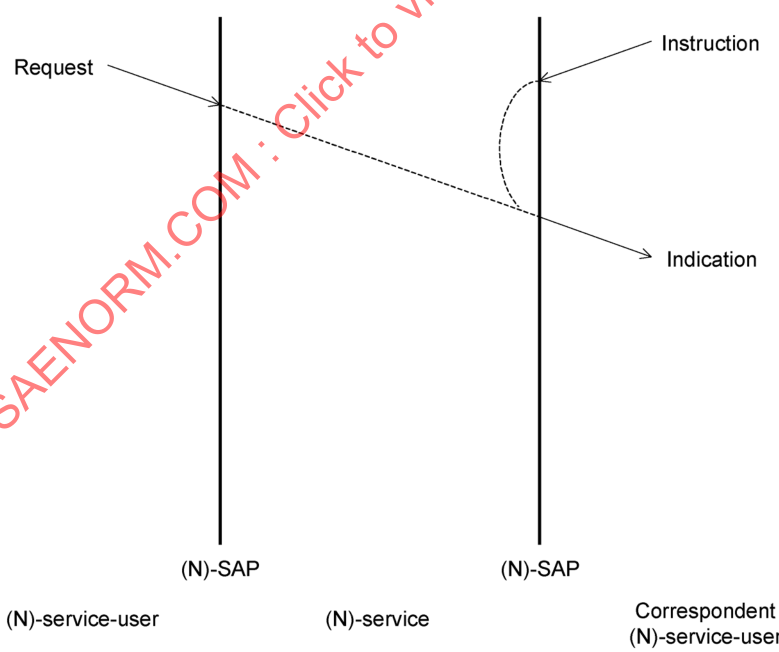

FIGURE 8 - Time Sequence Diagram of Figure 7

5.6 (Continued):

The second form of acknowledgement is shown in Figures 9 and 10. In this scenario, the (N)-service-user receives a confirmation primitive to indicate that the applicable (N)-PDU has reached the correspondent (N)-entity, which has been issued with an indication primitive. However, there is no acknowledgement that the correspondent (N)-service-user has successfully received and acted upon the (N+1)-PDU.

The final form of acknowledgement can be represented by the earlier Figures 5 and 6. Here the transfer of the (N+1)-PDU^1 is confirmed with a return (N+1)-PDU^2, comprising (N+1)-PCI only.

It will be noted that if the (N)-service does not provide acknowledgement to the (N)-service-user, it may still support an (N+1)-service that does provide an acknowledgement facility to the (N+1)-service-user. For example, file transfer might be acknowledged after the entire file has been transmitted, and not for the transmission of each block within the file.

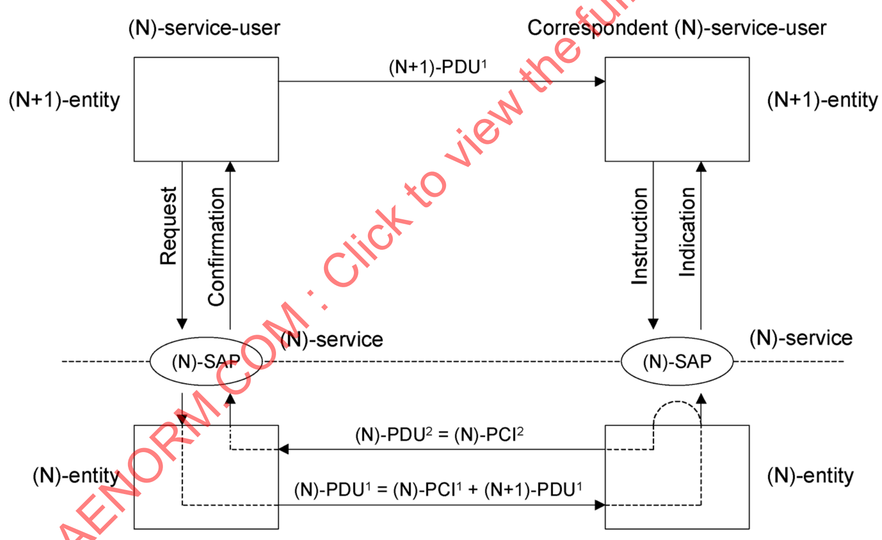

FIGURE 9 - Acknowledgement of (N)-Service Activity

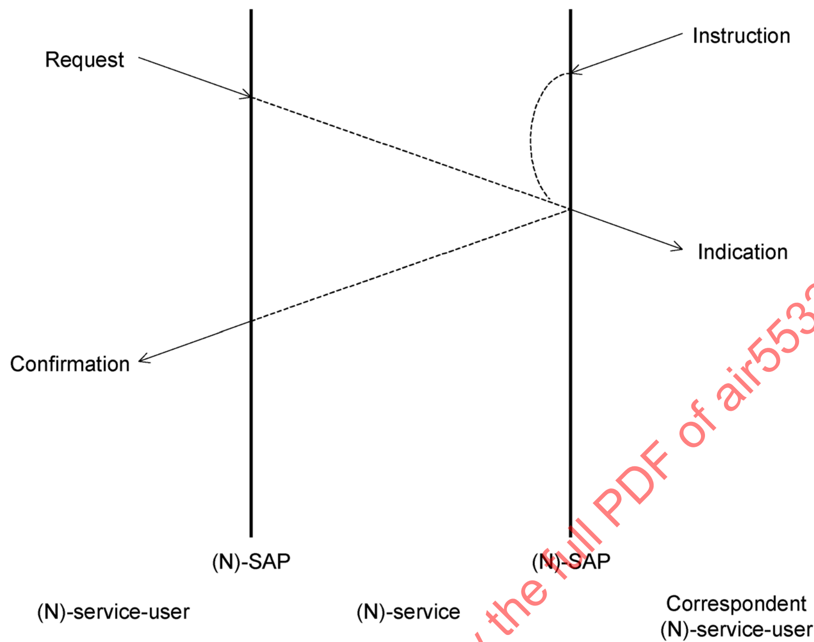

FIGURE 10 - Time Sequence Diagram of Figure 9

5.7 Synchronization Scenarios:

The following term is introduced in this section, as defined in 2.3: expedited (N)-protocol-data-unit.

The aircraft-store interface is a real-time application, as characterized in Appendix B, and therefore GASIF must be capable of modeling synchronization operations. Synchronization operations in the network environment may take two forms: remote interrupts and time distribution.

5.7.1 Remote Interrupts: The term 'remote interrupt' has system synchronization significance and does not necessarily imply a hardware interrupt to a processor. A remote interrupt may be carried out in a number of ways.

The time sequence diagram of Figure 11 shows the scenario when a request is made by the (N)-service-user to provide an interrupt to another (N)-service-user (or users) for the purposes of synchronization. Such a data transfer may be 'normal' or 'expedited'. Many protocols support expedited data transfer as a distinct facility in connection-mode. Where such a facility is provided, the (N)-service ensures that a small expedited (N)-protocol-data-unit can be delivered with priority over normal data, with minimum transit delay. The expedited (N)-PDU will arrive before other data and it is assumed that it will be processed as a priority by the receiving (N+1)-entity.

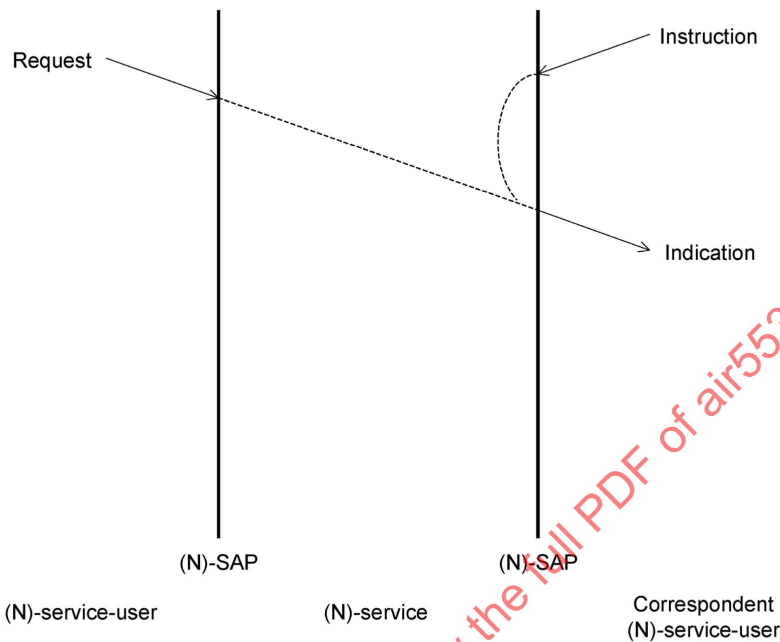

FIGURE 11 - Remote Interrupt Requested by (N)-Service-User

5.7.1 (Continued):

Figure 12 shows another scenario where an instruction is made by the (N)-service-user for the (N)-service to provide an interrupt to another (N)-service-user (or users) after the occurrence of a particular event within the (N)-service. Such an event could be related to a global time reference held within the (N)-service. Figure 13 shows the case where the (N)-service-user requests an interrupt to a remote (N)-entity, resulting in the occurrence of a particular event. This might put the remote (N)-entity into a particular mode, such as shutdown or override shutdown. Finally, Figure 14 shows the scenario when the (N)-service-user instructs the (N)-service to provide an automatic remote interrupt invocation service, based on the cyclical occurrence of an event within the (N)-service.

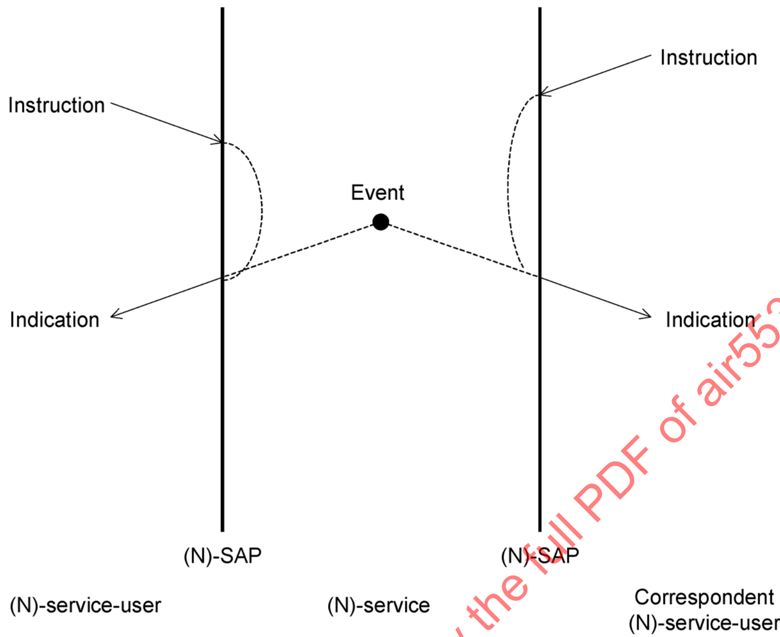

FIGURE 12 - Event Remote Interrupt by (N)-Service

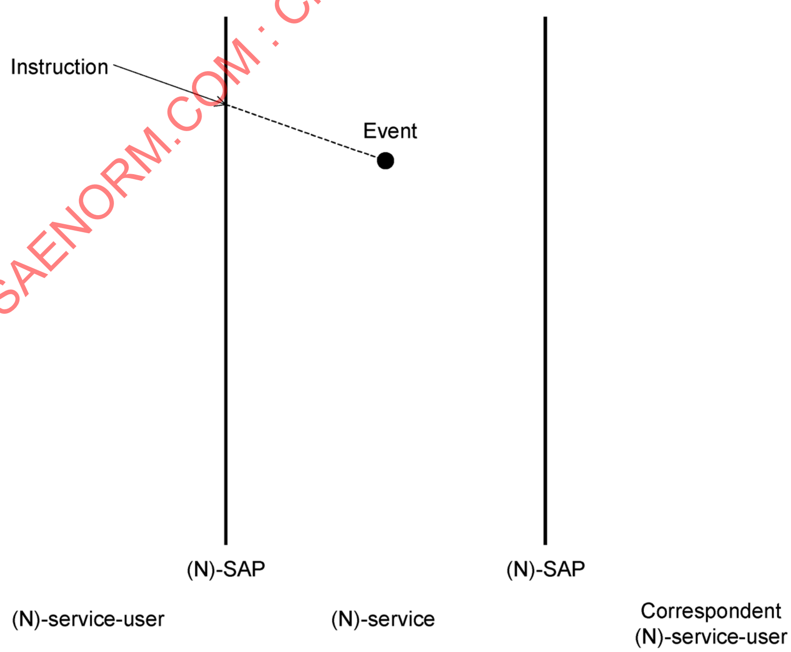

FIGURE 13 - Requested (N)-Service Event

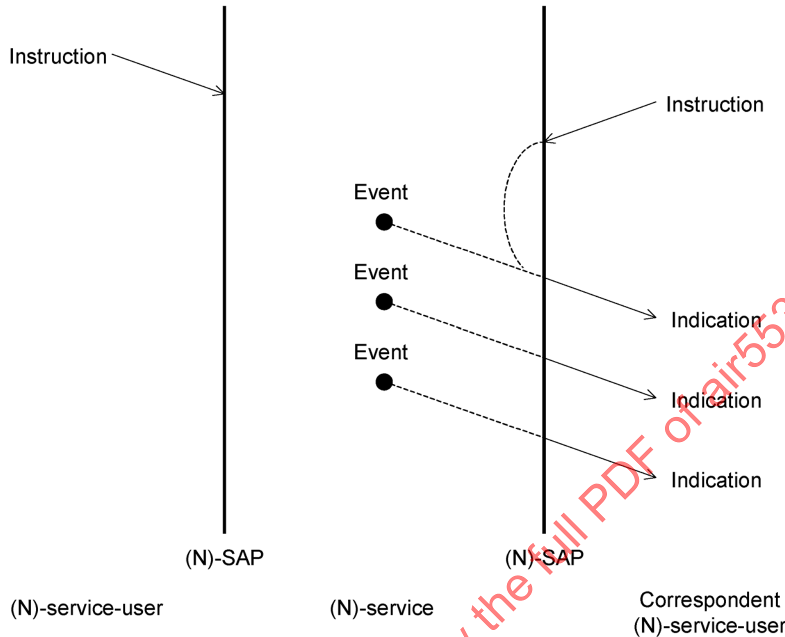

FIGURE 14 - Automatic Remote Interrupt Invocation

5.7.2 Time Distribution: Time distribution within the (N)-service may take two forms: immediate return of (N)-service current time to the (N)-service-user when requested, and deferred notification to the (N)-service-user of the occurrence of specified (N)-service time events (or epochs). These two scenarios are shown in Figures 15 and 16.

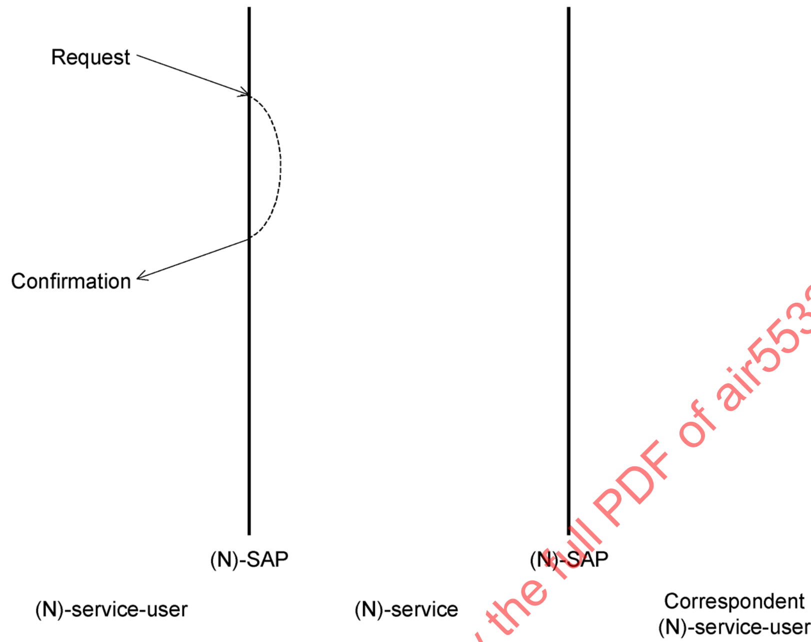

FIGURE 15 - Current Time Operation

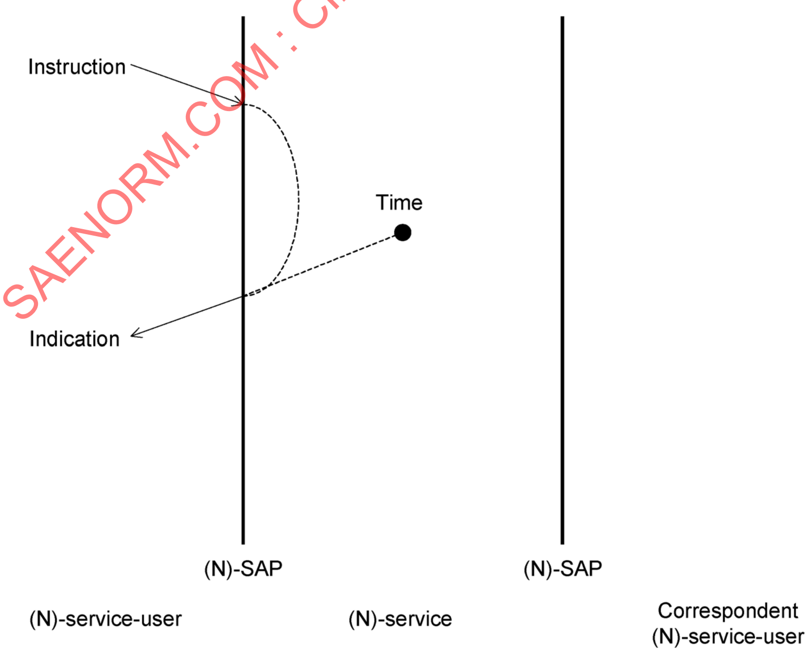

FIGURE 16 - Deferred Time Notification

6. OSI LAYERS:

The GASIF Open System Interconnection Environment (OSIE) complies with the OSI Basic Reference Model (ITU-T Rec. X.200 | ISO/IEC 7498-1) in that it will describe operations and mechanisms which are assignable to layers as specified in the OSI Basic Reference Model.

The OSI Basic Reference Model specifies seven protocol layers, as depicted in Figure 17. A summary of the service provided by each layer from an aircraft-store interconnection point-of-view is provided here. The Application Layer, Presentation Layer and Session Layer comprise the application-oriented protocols depicted in Figure 1 (and 2). The Transport Layer, Network Layer, Data-Link Layer and Physical Layer comprise the network-oriented protocols (Network Environment). A complete description of each of these layers is provided in ISO/IEC 7498-1.

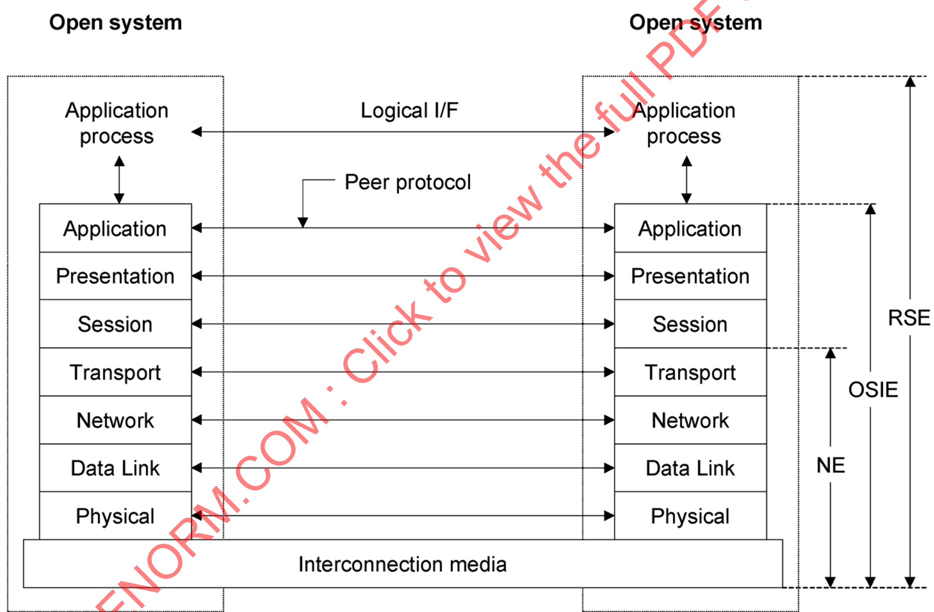

FIGURE 17 - Seven Layer Reference Model for OSI

NOTE: The MIL-STD-1760 protocol is mapped to the protocol layers in the GASIF Open System Interconnection Environment in Appendix C of this AIR.

6.1 Application Layer:

The Application Layer is the highest layer within the OSI Basic Reference Model and provides the sole means for the application process to access the OSIE. In real systems, the application-entities will perform various communications management functions that may not be visible to the external logical interface. The service provided by the Application Layer is normally unique to the particular application process.

6.2 Presentation Layer:

The Presentation Layer defines the abstract syntax of data entities that are exchanged between aircraft and store application processes (as collated into messages or files).

6.3 Session Layer:

The Session Layer is the lowest layer of the application-oriented protocols. For connectionless-mode communication, the Session Layer only provides a 'straight-through' function. For connection-mode communication, the Session Layer provides the means necessary for co-operating presentation-entities to organize and to synchronize their dialogue and to manage their data exchange. The following connection-mode services are generally provided:

a. Session-connection establishment and release
b. Normal and expedited data transfer
c. Token management
d. ession-connection synchronization and resynchronization
e. Exception reporting
f. Activity management

For the transfer of files between presentation-entities it is necessary to have a connection-mode session-service. When providing a connectionless-mode service, the Session Layer is required to provide a one-to-one mapping of session-connectionless-mode-transmissions onto transport-connectionless-mode-transmissions).

6.4 Transport Layer:

The following terms are introduced in this section, as defined in 2.3: routing, segmenting, flow control.

The Transport Layer is the highest layer within the Network Environment. The transport-service therefore provides for the transparent transfer of data between application-oriented protocols (strictly, session-entities) and relieves them of any concern with the detailed way in which this transfer of data is achieved. All protocols defined in the Transport Layer have end-to-end significance, and therefore the Transport Layer is relieved of any concern with routing and relaying considerations. This particular point is illustrated in Figure 18, which shows communication involving a relay open system.

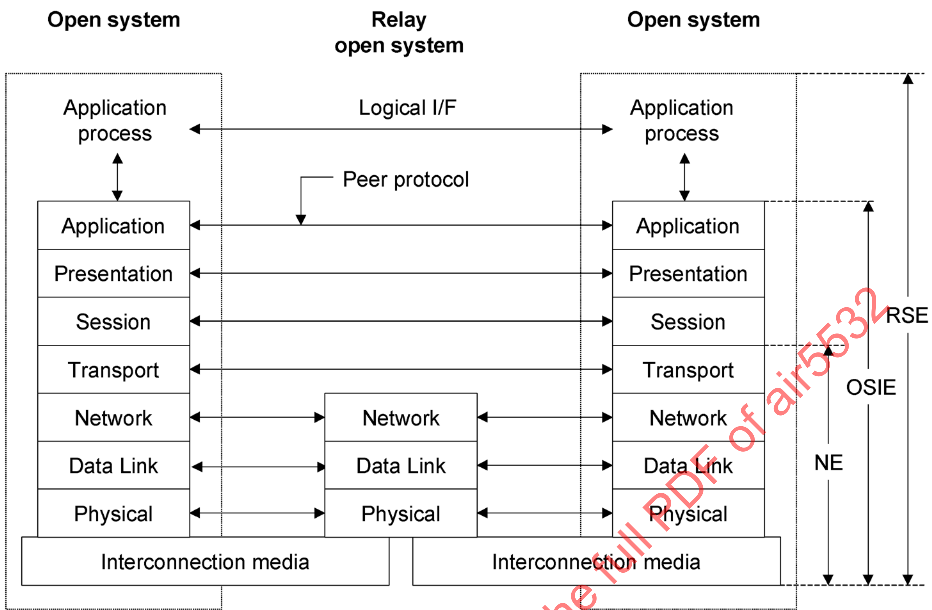

FIGURE 18 - ISO Communication Involving a Relay Open System

6.4 (Continued):

In connectionless-mode, the Transport Layer generally maps end-to-end transport-connectionless-mode transmissions on to network-connectionless-mode transmissions. The Transport Layer may also achieve end-to-end error detection.

In connection-mode, the Transport Layer generally:

a. Establishes an end-to-end transport-connection between peer application-oriented protocols, having determined the optimum size of transport-protocol-data-unit.

b. Transfers normal data, perhaps using segmenting, sequence control and flow control operations, as well as possible error detection and recovery (other functions are also possible, as documented in the OSI Basic Reference Model).

c. Transfers expedited data (i.e. with priority over normal data) for system functions such as remote interrupts.

d. Releases transport-connection.

6.4 (Continued):

For file transfer, a connection-mode-transport-service is required. In MIL-STD-1760, this service is provided by the exchange of MDT Transfer Data messages, which use record/block numbering to achieve file segmenting and sequence control. Above the Transport Layer, data is exchanged between end systems in the form of files or messages; but below the Transport Layer, data is exchanged over single or multiple subnetworks in the form of messages only (often called ‘packets’).

6.5 Network Layer:

The basic service of the Network Layer is to provide transparent data transfer between transport-entities. The network-service provides independence of the underlying communications media in all things other than the ‘quality of service’, which, for real-time communication protocols would normally be guaranteed by the system design. Quality of service parameters may include packet size, transmission rate and transit delay, plus service availability.

The Network Layer may provide a connection-mode service, a connectionless-mode service or both. However a connectionless-mode transport service is generally achieved using a network service that is also connectionless. Figure 19 summarizes the valid combinations of service-mode for the presentation, session, transport and network layers, where ‘CL-SS’ denotes a connectionless-mode session-service, ‘CO-TS’ denotes a connection-mode-transport-service, and so on.

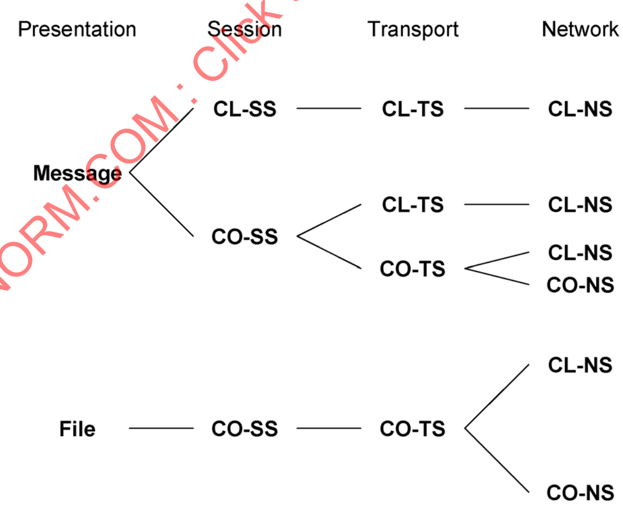

FIGURE 19 - Valid Transmission Modes for P-PDUs

For aircraft-store interface protocols (such as MIL-STD-1760), the Network Layer can be complex, and is divided into sublayers as described in Section 7.

6.6 Data Link Layer:

In the Data Link Layer, protocol-data-units are normally termed ‘frames’. Frames generally provide the lowest level protocol control information such as synchronization patterns, data link address, word or frame parity, frame size, acknowledgement, and so on.

6.7 Physical Layer:

The Physical Layer is responsible for converting the logical data bits and unique sync patterns of frames created in the Data Link Layer into physical waveforms than can be passed (with loss and distortion) over the physical interconnection media and then recovered back into frames.

7. SUBLAYERING OF NETWORK LAYER:

The following terms are introduced in this section, as defined in 2.3: Subnetwork Access Protocol, Subnetwork Dependant Convergence Protocol, Subnetwork Independent Convergence Protocol.

For aircraft-store interface protocols such as MIL-STD-1760, the GASIF Network Layer is subdivided into three sublayers. The model broadly follows the OSI Internal Organization of the Network Layer, ISO 8648, however that standard addresses real open systems and GASIF is only concerned with open systems. Therefore the models in ISO 8648 have been adapted where necessary to achieve consistency with the GASIF scope.

The sub-layering of the Network Layer is represented in Figure 20, which nominally shows an instance of communication involving an intervening open relay system. Three Network Layer protocols are identified to describe how multiple protocols may be employed to construct the network-service. These protocols are:

a. Subnetwork Access Protocol (SNAcP)
b. Subnetwork Dependent Convergence Protocol (SNDCP)
c. Subnetwork Independent Convergence Protocol (SNICP)

However, the use of a separate, discrete protocol-entity fulfilling each individual protocol in Figure 20 is not a general requirement.

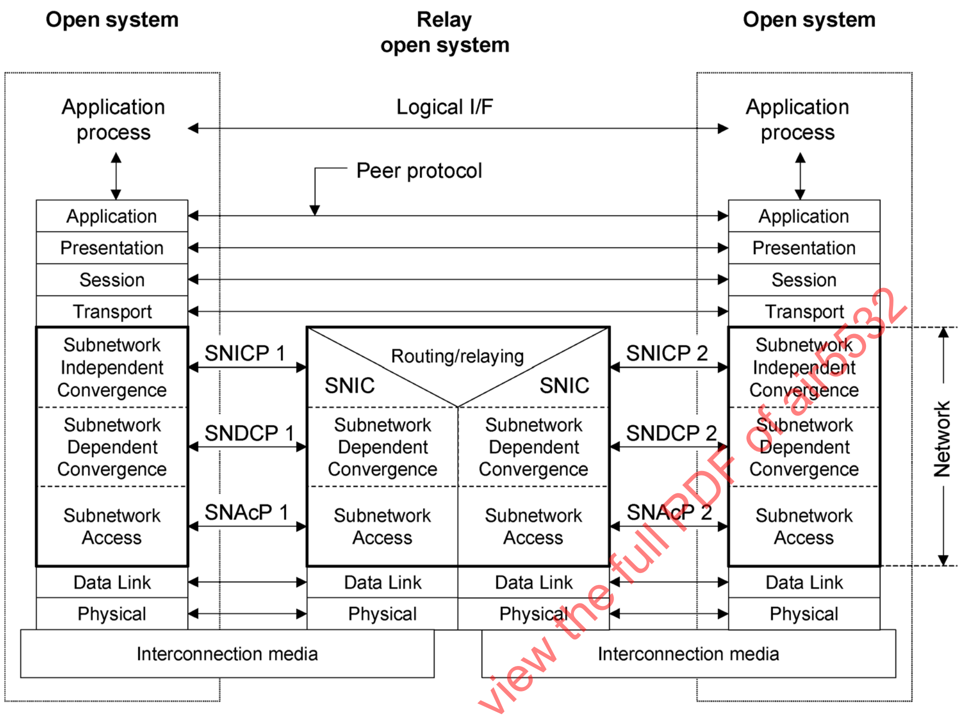

FIGURE 20 - Network Sublayers

7.1 Subnetwork Access Protocol:

The SNAcP is an optional layer that describes the Network Layer facilities associated with many subnetwork protocols that do not provide a complete or regularized OSI network service. The service provided by the SNAcP to the layer above is left undefined. The SNAcP, Data Link Layer and Physical Layer may be merged to suit the specific subnetwork protocol. One such example is the MIL-STD-1760 Digital Time Division Command/Response Multiplex Data Bus, AS15531.

7.2 Subnetwork Dependant Convergence Protocol:

The SNDCP operates over the SNAcP and is used to provide either the capabilities assumed by the SNICP or the complete GASIF network-service when a separate SNICP is not required (that is, it provides a direct service to the GASIF Transport Layer).

The SNDCP permits the assumptions made by the layer above to be de-coupled from the detailed operation of the particular SNAcP. The provision of underlying capabilities required by the layer above using a particular subnetwork-service may require the operation of an explicit protocol (with PCI exchange between peer SNDCP-entities). However there may also be cases where the SNDCP consists simply of a set of rules for manipulation of the subnetwork-service. This set of rules would not involve the exchange of PCI.

7.3 Subnetwork Independent Convergence Protocol:

The SNICP will provide a network-service in situations where an open relay system, such as a Carriage Store or Miniature Store Carriage System, intervenes between end systems, perhaps introducing multiple hops of subnetwork-protocols. The SNICP therefore, supports routing and relaying functions. Relaying functions enable a network-entity to forward information received from one network-entity to another. Routing functions determine an appropriate route between network addresses.

8. NOTES:

8.1 Key Words:

Aircraft, AS15531, avionics, communications, data-links, dispenser, GASIF, GOA, ISO/IEC 7498, layer, MIL-STD-1553, MIL-STD-1760, model, network, open system, Open System Interconnection, OSI, protocol, store

PREPARED UNDER THE JURISDICTION OF
SAE SUBCOMMITTEE AS-1B, AIRCRAFT-STORE INTEGRATION, OF
SAE COMMITTEE AS-1, AIRCRAFT SYSTEMS AND SYSTEM INTEGRATION

APPENDIX A
SUGGESTED GASIF-GOA RELATIONSHIP

A.1 OVERVIEW:

There is no intrinsic mapping between GASIF and the GOA Framework, AS4893. GASIF is an open system model, as defined herein, and is only interested in the external behavior of the system. Direct interfaces between adjacent layers in the same system are abstract in nature and are only considered if they are implied by the system's external behavior. GASIF is independent of the internal architecture of the system. On the other hand, the GOA Framework is a real open system model, as defined herein. Real open system models are distinct from open system models in that the internal (albeit abstract) implementation of the system is also of interest.

Figure A1 suggests a GOA view of the GASIF stack, assuming the GOA System Services have no cognizance of the application-specific details of aircraft-store interconnection. An alternative GOA view is to map some application-specific aircraft-store interconnection services into an extended operating system service (XOS) within GOA Layer 3. This alternate view might map the GASIF transport service (and possible higher GASIF layers) to a GOA Layer 3 service.

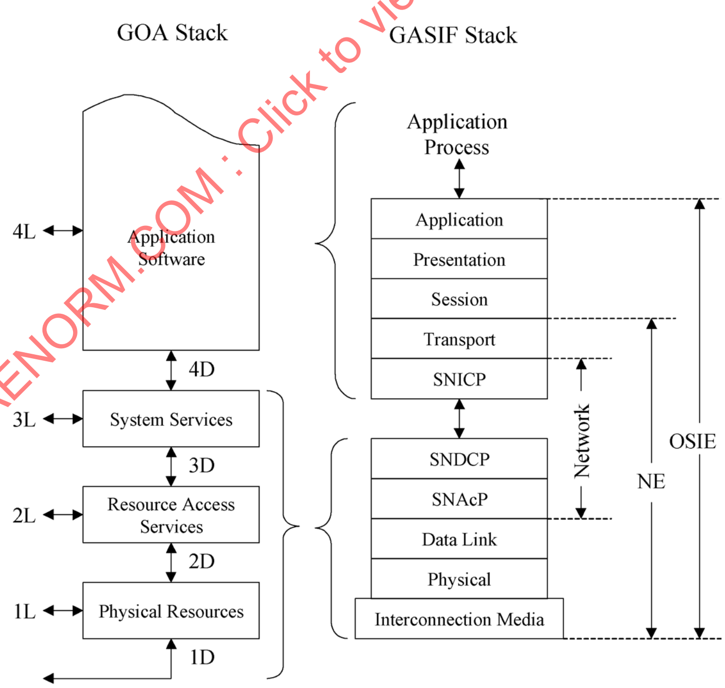

FIGURE A1 - Mapping of GASIF Stack to GOA Stack

A.2 NETWORK ENVIRONMENT:

The transport service provides for the transparent transfer of data between peer session-entities and provides routing, segmenting/multiplexing and flow control functions. In many real systems, this transport service is mapped to a GOA Layer 3 service. For aircraft-store interfacing, however, Transport Layer functions require cognizance of specific protocols such as MIL-STD-1760. That is, the GASIF transport service provides an end-to-end communication service that might be beyond the scope of the generalized transport service provided by the GOA Layer 3 service.

In the context of aircraft-store end-to-end communication, the generalized transport service of the GOA Layer 3 service might be relegated to an OSI subnetwork service from the GASIF point of view. Subsequent hops might be required using a MIL-STD-1760 subnetwork and a Dispenser subnetwork to achieve end-to-end communication. The GASIF Subnetwork Independent Convergence Protocol and Transport Layer may therefore be considered part of the GOA application software (GOA Layer 4).

However, GOA allows application software that provides a service to be optionally represented as an extended operating system (XOS) service. An alternative GOA viewpoint, therefore, is to map the GASIF Transport Layer and SNICP to an extending operating system (XOS) service.

A.3 OPEN SYSTEM INTERCONNECTION ENVIRONMENT:

It is suggested that the GASIF application-oriented protocols (session, presentation and application) are mapped to GOA Layer 4 (application software).

A.4 REAL SYSTEM ENVIROMENT:

The term, ‘application’, requires careful handling when dealing with GOA and GASIF concepts together. The GOA Layer 4 comprises application software, where the application software forms a part (or perhaps whole) of the GOA application. The GOA application is the capability (service/ function) provided by a system specific to the satisfaction of a set of user requirements.

The GASIF application process (as defined in 2.3) is the element within a real system that provides the information processing for a particular application (aircraft-store interfacing). The application process may include software, system inputs and outputs, human activity, physical processes etc. The real system boundary of interest may differ from GOA and GASIF viewpoints.

The aircraft-store interface is a real-time application. Real-time applications, as defined here, are characterized by the presence of hard deadlines where failure to meet a deadline must be considered a system fault. In common with other real-time applications, while the aircraft-store interface can be ‘open’ in the sense that it can use published standards to ensure interoperability, it may be ‘closed’ in the sense that information flows (and deadlines) are predetermined during system design for each implemented aircraft-store configuration.

The Generic Aircraft-Store Interface Framework therefore accommodates the following characteristics, which are typical of real-time applications.

APPENDIX B
FEATURES OF AIRCRAFT-STORE INTERFACE

The aircraft-store interface is a real-time application. Real-time applications, as defined here, are characterized by the presence of hard deadlines where failure to meet a deadline must be considered a system fault. In common with other real-time applications, while the aircraft-store interface can be 'open' in the sense that it can use published standards to ensure interoperability, it may be 'closed' in the sense that information flows (and deadlines) are predetermined during system design for each implemented aircraft-store configuration.

The Generic Aircraft-Store Interface Framework, therefore, accommodates the following characteristics, which are typical of real-time applications.

B.1 DISTRIBUTED:

Store control may be a distributed application. Applications and other processing functions may be distributed between an aircraft and a carriage store or dispenser. This results in the need to distribute possibly data, communications functions, control, data-storage, and system management functions.

Characteristics of the control being distributed may include:

a. Application control, i.e. controlling the execution of distributed tasks in carriage stores, dispensers and so on.

b. Provisions for the distribution of a global time reference, for synchronization purposes, and to allow time stamps on data.

c. Provisions for remote interrupts (in the system context).

The distribution of system management functions may require:

a. Configuration management, including stores inventory management (throughout mission), store energization, and so on.

b. Monitoring of system status, health and faults (the latter is either hard or intermittent).

c. System state support, for instance, normal or exception states with degraded modes and changing system priorities and objectives.

B.2 UNINTERUPTED OPERATION:

The aircraft-store interface, in common with other real-time applications, is characterized by requiring uninterrupted operational performance. This implies the tolerance of the occurrence of both scheduled and unscheduled events. Scheduled events may be executed either off or on line. The unscheduled events may be either planned (such as remote interrupts or asynchronous events) or unplanned. Unscheduled unplanned events include faults and defects.

B.3 HIGHLY AUTOMATED:

The aircraft-store interface may be characterized by many functions being automated; that is, predefined. One example of an automated function might be a store release sequence carried out by a dispenser on behalf of the aircraft.

B.4 CONTROL:

The aircraft-store interface may be required to execute control functions. This may imply closed loops with cyclic data, and fail-safe operation with high integrity.

B.5 PREDEFINED SYSTEM CONFIGURATION:

The aircraft-store interface will have system configurations that are predefined or planned in advance during system design. Therefore, for a given store carriage configuration, all system parameters such as processor or communication network loads can be calculated and resources allocated. This includes the system dependability as well as the event of changing system states with anomalies resulting in the exception handling of, for instance, data, applications/processing, system management, and initialization/re-initialization.

This appendix provides a mapping of the MIL-STD-1760 protocol to the GASIF framework. The applicable version of MIL-STD-1760 is Revision C, Notice 1.

APPENDIX C
MIL-STD-1760 MAPPING TO GASIF

This appendix provides a mapping of the MIL-STD-1760 protocol to the GASIF framework. The applicable version of MIL-STD-1760 is Revision C, Notice 1.

C.1 APPLICATION PROCESS:

The Data Entity List in Table XXVI maps to the Application Processes, with the exception of the data entity syntax, which maps to the Presentation Layer.

C.2 APPLICATION LAYER:

The service provided by the Application Layer is normally unique to the particular application process. No MIL-STD-1760 protocol maps to the Application Layer.

C.3 PRESENTATION LAYER:

The following protocols in MIL-STD-1760 relate to the Presentation Layer:

a. Store Control, Store Monitor and Store Description message formats (data words only) in Tables B-XI, B-XII and B-XIII respectively, and referenced tables

b. The data entity syntax tables referenced by the Data Entity List in Table XXVI. The Data Entity List itself is part of the application process logical interface

C.4 SESSION LAYER:

C.4.1 Connection-mode Service:

For the transfer of files between presentation-entities it is necessary to have a connection-mode session-service. In MIL-STD-1760, connection-mode session-services are provided by the exchange of Mass Data Transfer (MDT) Transfer Control and Transfer Monitor messages (albeit that PCI in some layers below the Session Layer is also represented in this protocol).

C.4.2 Connectionless-mode Service:

When providing a connectionless-mode service, the Session Layer is required to provide a one-to-one mapping of session-connectionless-mode-transmissions onto transport-connectionless-mode-transmissions).

C.5 TRANSPORT LAYER:

For file transfer, a connection-mode-transport-service is required. In MIL-STD-1760, this service is provided by the exchange of MDT Transfer Data messages, which use record/block numbering to achieve file segmenting and sequence control.

Above the Transport Layer, MIL-STD-1760 data is exchanged between end systems in the form of files or messages; but below the Transport Layer, data is exchanged over single or multiple subnetworks (such as MIL-STD-1553B) in the form of messages only (often called ‘packets’).

C.6 NETWORK LAYER:

In MIL-STD-1760, there is little cognizance of a network-service other than those elements of MIL-STD-1553B that implicitly map to the Network Layer together with MIL-STD-1760 rules for mode code usage and so on. For the configuration of one mission store directly connected to one aircraft station, no other protocol is required in the Network Layer. However, the potential complexity of the Network Layer increases when the following developments are taken into consideration:

a. There is a desire to standardize a routing/relaying protocol for the transfer of data between an aircraft and a mission store via an intervening carriage store when the carriage store is configured as a relay open system. The network packet would be required to take two MIL-STD-1553B ‘hops’: the first hop from the aircraft to the carriage store, the second hop from the carriage store to the mission store (or in reverse).

b. There is a similar requirement to develop a protocol for routing/relaying data through a miniature store carriage system to support miniature mission stores. Here, there is the extra complexity of the interface being different on either side of the miniature store carriage system. The miniature store carriage system may be directly connected to the aircraft, or it may be connected via an intervening carriage store (such as a rotary launcher in a bay). In the latter case, the network packet will be required to make three hops with a different ‘quality of service’ offered by one of the hops.

c. The MIL-STD-1760 AEIS has provision for a future interface to be employed as a high-speed alternative or complement to MIL-STD-1553B. This new protocol would further complicate the Network Layer, particularly for multi-hop data transfers employing vastly different transmission media and technology in each hop.

d. Miniature mission store interface also includes reserved interconnection media for future high speed data communication, perhaps to exploit the improved ‘quality of service’ that may soon be available in the MIL-STD-1760 AEIS.

To handle this complexity, it is necessary to subdivide the Network Layer into sublayers, with subnetwork access protocols in the lowest sublayer. An example of a subnetwork service is MIL-STD-1553B. More information on network sublayers is provided later.

C.6 (Continued):

The Network Layer may provide a connection-mode service, a connectionless-mode service or both. However a connectionless-mode transport service is generally achieved using a connectionless-mode network service. For store-carriage configurations employing an open relay system, these communication modes will require a routing/relaying protocol.

MIL-STD-1760 has provisioned for future connectionless-mode and connection-mode routing/relaying protocols, albeit that this provision is oriented to MIL-STD-1553B. Specifically, Subaddress 7 is reserved for ‘peeling’. In a future connection-mode-network-service, Subaddress 7 may be used for connection establishment, data transfer and connection release. In a future connectionless-mode-network-service, Subaddress 7 may be used for data transfer, where the address of the destination transport-entity is provided simultaneously with the data being transferred. The number of mission store data words is restricted in MIL-STD-1760 to 30 or less (out of a possible 32) to allow the transport-address to be provided in the other two words.

An alternate routing/relaying technique, suitable for scheduled periodic messages only, is to download a transaction table into the carriage store or miniature store carriage system during initialization. However, such a technique must be supplemented by a provision for unscheduled messages (both planned and unplanned). The transaction table technique provides a connection-mode network service with a priori connection establishment and no connection release.

Figure 19 shows the valid Presentation Layer, Session Layer and Transport Layer transmission modes for messages and files depending on the transmission mode of the network service.

C.7 DATA LINK LAYER:

The MIL-STD-1553B protocol merges the lower part of the Network Layer, the Data Link Layer and the Physical Layer together and does not suggest any subdivision of functions between the Network Layer and the Data Link Layer. As the MIL-STD-1553B data-link-service is not accessible in real systems (it is implemented inside protocol devices), it may not be a particularly useful exercise to model MIL-STD-1760 Data Link Layer as a distinct service.

C.8 PHYSICAL LAYER:

In MIL-STD-1760, the MIL-STD-1553B BC and RT frames are represented by Manchester bi-phase waveforms with defined characteristics at the source and receiving end. MIL-STD-1760, has specific interconnection media and fault-tolerance requirements, and requires a more stringent waveform specification at the transmitting-end and tolerates more distortion and loss than MIL-STD-1553B at the receiving-end.