Naval Research Laboratory

Washington, DC 20375-5320

NRL/FR/5533--92-9525

Interface Protocol Requirements for Shipboard Damage Control Systems

DAVID L. TATE

Human Computer Interaction Laboratory
Information Technology Division

September 30, 1992

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave Blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 30, 1992	Final 1 Oct. 91 - 30 Sept. 92

4. TITLE AND SUBTITLE

Interface Protocol Requirements for Shipboard Damage Control Systems

5. FUNDING NUMBERS

PE - 62121N
PR - RH21S22

6. AUTHOR(S)

David L. Tate

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
Washington, DC 20375-5320

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL/FR/5533—92-9525

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

Damage control systems are evolving rapidly into large, complex systems having the potential for collecting vast quantities of detailed information. To handle this information effectively, damage control systems will be transformed from their traditional centralized architectures to more efficient and more survivable distributed architectures. A coherent method of transferring information across distributed networks is required in the form of an interface protocol that can be used by damage control systems and personnel to quickly assess damage situations and initiate corrective actions. This report addresses the requirements of such a protocol and discusses related network and computer architecture issues, and their impact on the overall system design.

14. SUBJECT TERMS

Damage control
Computerized control systems
Computer communication protocols

15. NUMBER OF PAGES

19

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

CONTENTS

INTERFACE PROTOCOL REQUIREMENTS FOR SHIPBOARD DAMAGE CONTROL SYSTEMS

INTRODUCTION

One of the goals of current research in the field of shipboard damage control is to have the ability to detect, transmit, analyze, and display damage control (DC) information quickly and effectively in order to expedite corrective actions. An important part of this research is the development of advanced sensors that will provide more precise, detailed, and complex sensor information, and have the ability to modify or reconfigure the sensor's operational characteristics. Shipboard sensors for smoke, heat, flame, fire classification, flooding, hull damage, and chemical, biological, and radiological (CBR) agents are under development that will provide detailed information pertaining to the specific characteristics of the threat [1]. This influx of information must be managed effectively both at the operator's console and among the circuits connecting the various devices.

As the number and sophistication of advanced sensors increase, significant changes will be made to the methods of displaying the information, and to the methods of connecting sensors to control and display devices. The use of computer workstations with graphical displays has become commonplace in office environments, and work has been performed on developing these same graphical techniques for shipboard damage control [2]. An effective means of managing information, displaying data, and controlling equipment is the use of graphical user interface techniques. This approach is very effective in reducing the information-processing burden of the operator; it allows the operator to retrieve the information when it is needed.

Changes to the methods of connecting sensors to control/display devices, and the mechanisms by which sensor information is transferred will also be required. Network architectures for system and device interconnect is the current trend, and this trend is expected to continue for the foreseeable future. To use networking effectively, a coherent method of transferring DC information across these networks is needed. This method of information transfer is the interface protocol, and it comprises not only the information being transferred, but also the rules by which transactions between devices are conducted. This report presents the requirements and recommendations for a baseline implementation of this protocol.

DAMAGE CONTROL SYSTEM ARCHITECTURE

An important consideration in the design of any system that has remote sensing capabilities is the system architecture, or method of connecting the various devices to each other. Most damage control sensors are connected to alarm systems that operate locally or through point-to-point connections to Damage Control Central (DCC) or one of the ship's Repair Stations. This centralized architecture is satisfactory for small systems, but it becomes unduly complicated for widely dispersed systems that have a large number of

Manuscript approved September 25, 1992.

sensors. For these systems, an architecture where sensors and equipment are distributed along a shared network provides a more flexible and expandable system.

Centralized Architecture

In a centralized DC architecture, the equipment and sensors for damage control and hull, mechanical and electrical (HM&E) systems are connected to a central processing system (Fig. 1). The processing can be performed by a device as simple as a lighted pushbutton control panel or by a more complex system such as a state-of-the-art computer. In either case, all of the sensors have a direct interface to the processor. This architecture has been used in the past, where sensors are connected directly to control panels in DCC or one of the ship's Repair Stations.

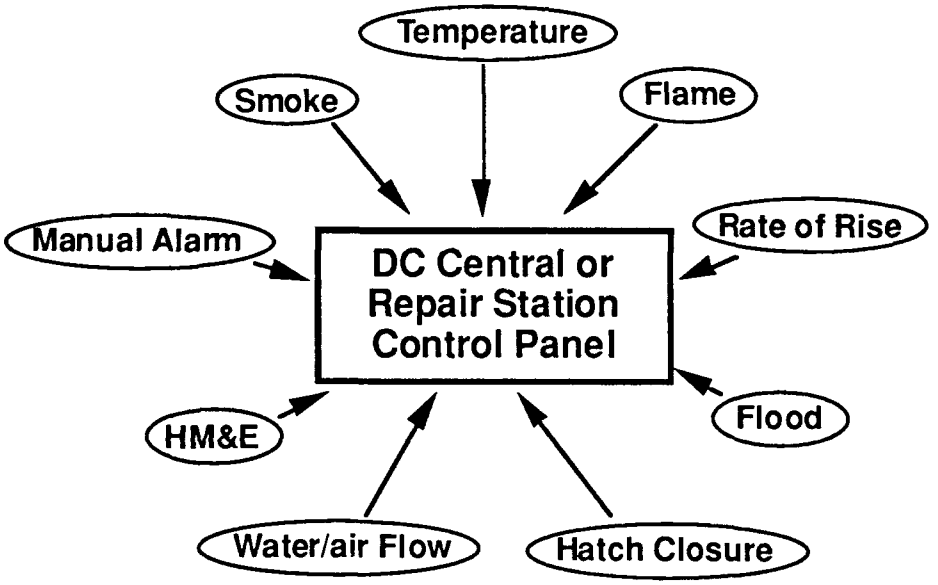

Fig. 1 — Centralized architectures are characterized by a central processing station to which all sensors are directly connected. This architecture has several drawbacks which make it unsuitable for future damage control systems.

A centralized architecture has several drawbacks. One is the long, elaborate wiring systems required to connect each sensor to the control panel. Sensors are located throughout the ship, and each one requires a separate, dedicated circuit to the control panel. With a large number of sensors on board, and multiple sensors located in many spaces, redundant cabling is commonplace between DCC and vital areas. Complicated wire runs across and between decks are often used to reduce the length of these cables. The weight of the wiring alone for large systems can be substantial, even prohibitive.

Another drawback of the centralized architecture is that several single points of failure exist. A system failure resulting from failure of the central processor is an obvious deficiency. If the control panel in DCC fails, all the sensors that are connected to it are useless. The sensors may be functioning perfectly, but without a means of displaying their status, they provide no help to DC personnel. The cabling that provides the circuit between the sensor and DCC is another point of failure. Because some of these cables are necessarily long, damage to a part of the cable that is not located near the sensor nor near DCC could cause the loss of sensor information. In situations where multiple cables share a common conduit or raceway, the loss of use of many detectors can be caused by damage occurring nowhere near the sensors or the control panel.

This architecture also has limited flexibility. Relocating the control panel is virtually impossible, since it would require rerouting all the cabling from the sensors. Since the control panel cannot be moved, the compartment where it is located must be manned at all times. Although damage control sensors are monitored at all times anyway, other architectures provide ways to perform monitoring from alternate locations. With a centralized architecture, if DCC must be evacuated, the sensors and displays may continue to operate perfectly, but DC personnel will have no access to the information.

Expandability of a centralized architecture is also a problem. The control panel interfaces must be designed to accommodate all current capabilities and any anticipated expansion. Multiplexing techniques can reduce the effect of having to add new components to the system, but new cabling to new sensors is still required. Running a single cable from DCC to a new, remotely located sensor may be a difficult task in itself.

Centralized architectures are usually sufficient for small systems, but new Navy ships have far too may sensors to be able to use this architecture in the future.

Distributed Architecture

To meet the needs of future systems, an architecture that supports sensors and equipment that are distributed along a shared network is required. This distributed architecture typically uses a local area network (LAN) to which sensors, alarms, control panels, or operator workstations can be connected (Fig. 2). The specific attributes of the LAN can be chosen from a variety of media, topologies, transmission speeds, and network management schemes. Through appropriate LAN interfaces, devices can be accessed by using standard LAN access techniques that do not require a specific layout or interconnection scheme.

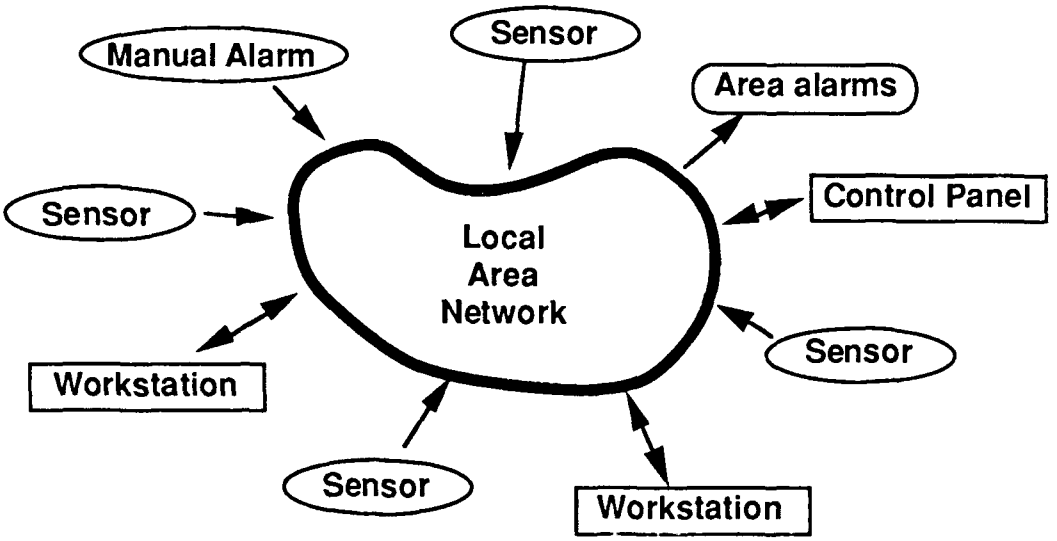

Fig. 2 — Distributed architectures support sensors and equipment that are distributed along a shared local area network (LAN). This architecture provides the flexibility and expandability needed for future damage control systems.

The distributed architecture provides much more flexibility than the centralized architecture. A typical installation would have the LAN installed throughout the ship, with access available through relatively short "drops" to various devices. Because of the distributed nature of the LAN, devices can be located anywhere along the network and can be relocated, added, or removed with little effort. The addition of a new sensor would require laying only the "drop" cable from the sensor to the LAN. The diagram in Fig. 2 assumes that sensors have their own built-in LAN interfaces, which will not be the case where new DC systems are being backfitted onto older ships with stand-alone sensors. To accommodate these older designs, LAN

multiplexer/interface units can provide the necessary connection to the LAN (Fig. 3). These units can provide sensor monitoring functions and LAN access for multiple sensors of various types.

The distributed architecture is easily expanded by simply providing LAN interfaces to the new devices. Access to shipboard database systems via the LAN would be especially helpful to DC personnel. Through these databases, information such as ship's diagrams, maintenance schedules, equipment inventories, duty rosters, and numerous other types of information would be available. Even access to other ship systems located on other LANs can be attained by using network gateways. These devices perform the traffic management and translation of protocols to allow devices on different networks to share resources.

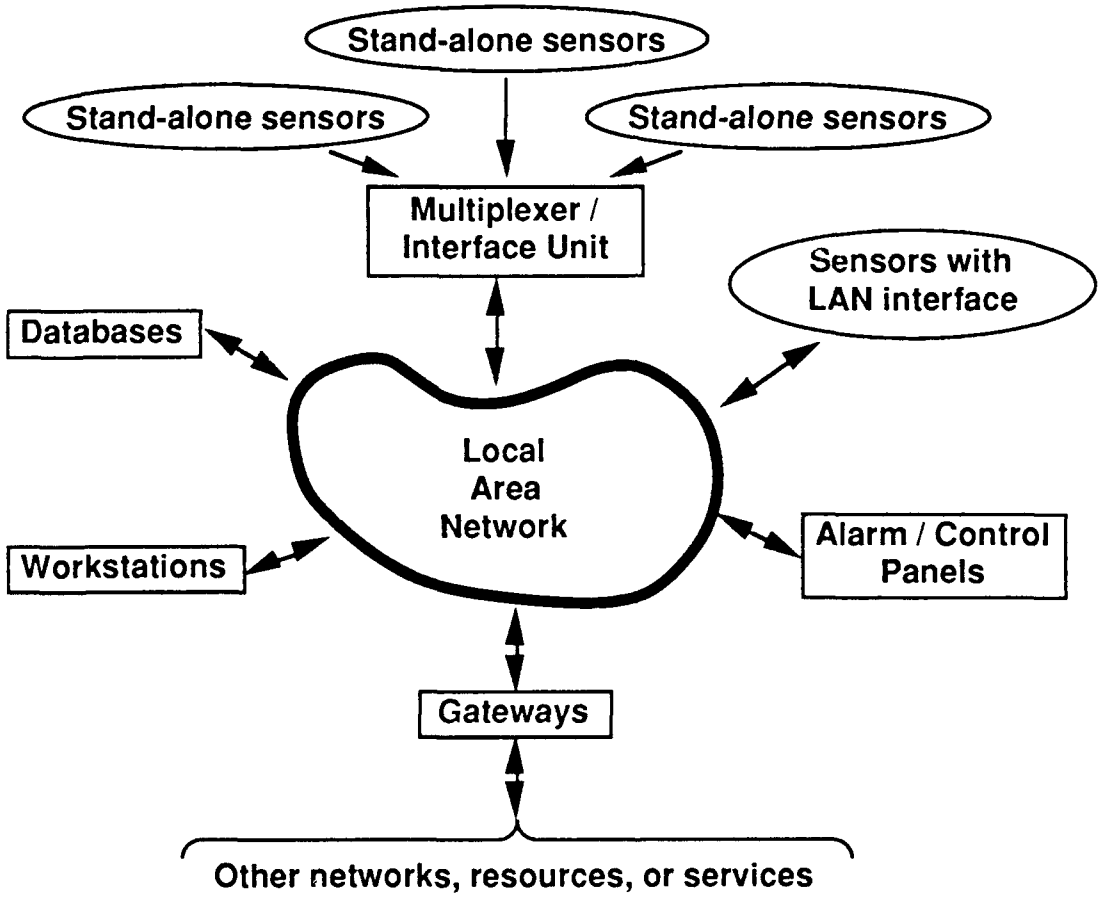

Fig. 3 — Distributed architectures can provide increased capabilities through access to networked databases, computer workstations, and other networks, resources, and services. LAN multiplexer/interface units can provide the necessary interfaces to existing sensors to meet backfit requirements.

The distributed architecture also reduces the susceptibility to a single point of failure. Obviously a failure of the LAN would have catastrophic implications, but techniques are available to ensure its survivability. Redundant, survivable networks can be designed so that damage to any part of the LAN can be compensated for by alternate routing paths. Network management algorithms keep track of network connectivity and routes between devices and dynamically adapt for problems. This is standard practice on mission-critical systems that use this type of architecture. The five-bus Data Multiplexing System used on the DDG-51 class destroyers [3] exemplifies the use of redundant, survivable networks for damage control.

Distributed network architectures are common in various computer-based systems and will continue to be used for the foreseeable future. New systems designed for damage control, including the Integrated

Survivability Management System (ISMS) [4] will use this architecture to meet the requirements of future DC systems.

Information Processing Structure

Our DC system architecture uses an information-processing structure that can be divided into three separate areas—sensor development, transfer protocol development, and console/display development (Fig. 4). Sensor development addresses the research of new methods for sensing and measuring damage control parameters, providing the necessary mechanisms for reading the sensor data, and providing ' 'nterfaces for monitoring and controlling sensor performance. Transfer protocol development addresses the formulation of techniques for transferring sensor data from sensors to consoles, and the development of procedures and rules for managing data transfer transactions. Console/display development pertains to the development of methods of displaying information graphically, providing operator interaction through graphical input techniques, and filtering raw sensor data to provide more concise information to the operator. Figure 4 shows the functional areas of the information processing structure, which may differ from the physical construction. For example, the multiplexer interface unit described above may include both the sensor interface and LAN interface functions.

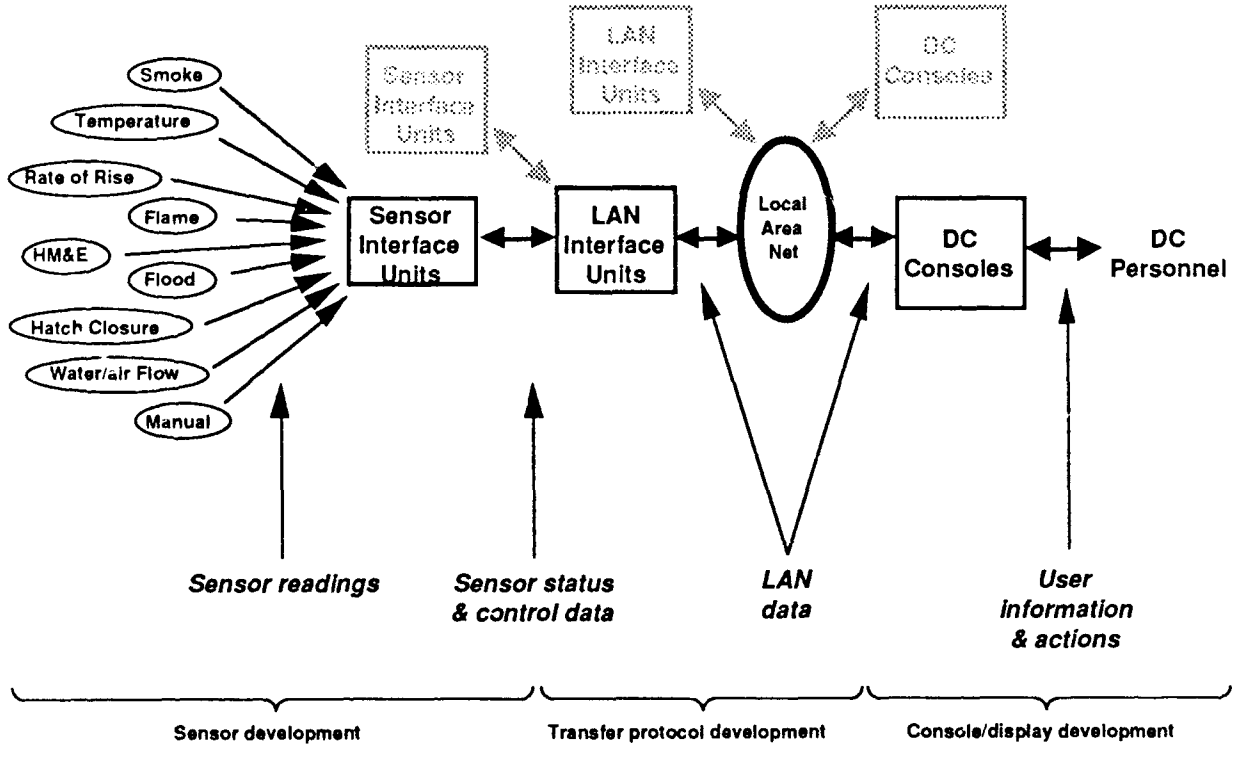

Fig. 4 — Information processing structure for damage control workstations

NETWORK DESIGN ISSUES

Distributed architectures can be implemented by using various types of LAN designs. Equivalent capabilities can be acquired from various media, topologies, and management techniques. The particular characteristics of the network must be transparent to the damage control systems so that they can operate in different environments, and even between different types of networks, if needed.

Media Types

A wide variety of transmission media are available for data communications and networking [5]. The media types available for damage control systems include twisted-pair cables, coaxial and twin coaxial cables, CATV circuits, and optical fiber circuits. Each medium has advantages and disadvantages, and shipboard systems may comprise a combination of media. Mixed media is common in some laboratory environments where networks may use twisted-pair cables (e.g., LocalTalk, RS-422, or Ethernet 10BaseT), coaxial cables (e.g., Ethernet 10Base2 and 10Base5), optical fibers (e.g., FDDI, SAFENET II, or SONET). CATV systems (also referred to as broadband systems) can provide multiple separate networks through a common cable instead of duplicate cabling to achieve network isolation. CATV techniques would also permit standard video signals to be used on some channels to allow shipboard cameras to share the same cable as the data communications systems. The NRL Integrated Communications Environment Network (NICENET) [6] is a good example of a communications network that connects various types of media, data sources, and video sources into an integrated broadband system.

Because of the tendency to mix transmission media types within a network environment, and because more advanced technology media are always under development, care must be taken when designing DC systems so that the capabilities of the system are not dependent on any particular medium. This generally is not a problem if standard network access schemes are followed. Methods for transmitting and receiving data on the network medium are generally hidden from the application program by providing higher level routines for supplying or accepting data. This allows the programs to transfer data across the network without having to know the details of the network design. Damage control application programs must not bypass these routines to gain direct access to the net, since such techniques may cause the program to fail if the network medium is changed.

Network Topologies

There are many different possibilities for the topological structure of a network—ranging in complexity from small, simple structures to mixed, complex hierarchical designs. Figure 5 shows examples of some of the topology types. The star topology is a centralized configuration where network control is administered by the star controller to which all network devices (or nodes) are connected. All data are transmitted to and received from the star controller, and the controller handles all network management functions. Devices have uncontested access to the medium, but they compete for processing time in the controller. In a single-controller configuration, the network becomes a centralized architecture having the inherent problems described above.

The bus topology uses a common shared medium that is distributed to all locations requiring access to the network. Traffic management is not centrally controlled but is distributed among all the devices through well-defined network-access procedures. Contention for use of the medium can be a problem here, but uncontested transmissions are immediate and direct. To handle contention problems, each device must have the ability to determine whether or not the network is busy, and whether or not its use of the net was successful. Carrier-sensed multiple access with collision detection (CSMA/CD) that is used on Ethernet is an example of one technique to determine medium availability (through sensing the carrier) and transmission success (through collision detection) in networks using a bus topology.

In the ring topology, devices are connected only to their immediate neighbor in the network. Data being transmitted to distant devices are passed from device to device until they reach their destination. The total amount of cabling can be smaller than with other topologies, but additional delay may be created if data must pass through a large number of intermediate nodes. The Survivable Adaptable Fiber Optic Embedded Network II (SAFENET II) [7] is an example of a topology that uses dual Fiber Distributed Data Interface (FDDI) token ring LANs.

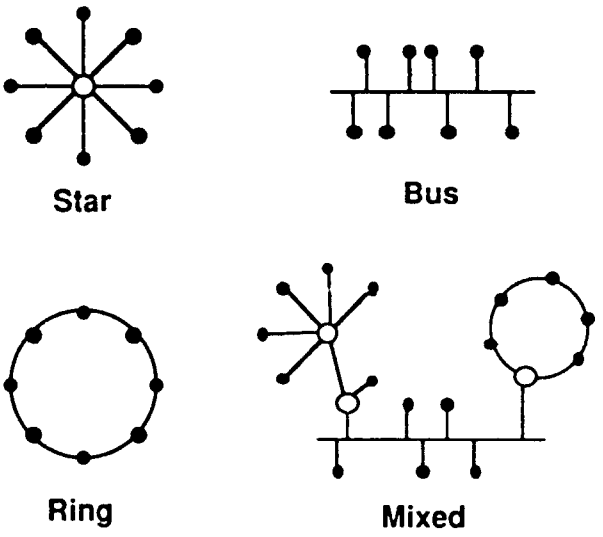

Fig. 5 — Example topological layouts for networks

Mixed topologies are often used when networks of different topologies are connected to form an internet. This connection is accomplished through a gateway that performs the necessary media and protocol conversions to allow the networks to interact with each other.

Data Transmission Speed

The need for faster transmission speeds to support the transfer of large amounts of data has led to a progression from twisted-pair wiring to coaxial cables to optical fibers for high-speed networks. The speed at which a network can transfer data is determined in part by the type of medium used, but delay and throughput affect speed regardless of the medium. Delay can be defined as the time between the transmission of the first bit of information from the source to the time of its delivery to the destination. Delay is usually measured as average response time and is affected by circuit data rate, processing delay times, queuing delay times, and system load. Because of the mission-critical nature of damage control sensor information, it is imperative that the network through which this data is transferred have a very low delay time. Data networks that are designed for large data transfers with little concern for long delay times are not suitable for damage control use. Use of shared data networks with low delay times could be used if DC information could preempt other data through prioritization. Gateways that block unnecessary traffic from the damage control net may be required in configurations where internet connections are permitted.

Throughput is defined as the number of bits sent divided by the time between transmission of the first bit and delivery of the last bit. Throughput is primarily determined by the effective bandwidth of the processing equipment and transmission medium. High throughput is most often required in data networks for functions such as file transfer, but in shipboard damage control systems the high peak loads caused by casualty situations can create large amounts of data that must be handled quickly. High throughput is required to prevent system overload resulting from a large number of alarm conditions, as would occur in a major conflagration.

Network Management

For distributed damage control systems to work effectively, the underlying network management must be handled transparently with no active participation by the damage control applications. The application programs must be designed to pass information via data packets to the network management modules,

which in turn perform all the functions necessary to assure reliable transfer of information between devices. The network management functions include reliable data delivery, packet routing, assured network connectivity, and communications security.

Routing

Devices on a distributed network usually have multiple routes that they can use to transfer data between sources and destinations. The route that any particular data packet traverses must be unknown to the application program but known by the network management software. This separation of function permits changes to be made to the network with no modifications necessary to the application program. Large blocks of data may actually be divided into smaller packets, and separate routing for these packets may be used to provide faster throughput. Routing algorithms can select the fastest route to use between source and destination, and can perform dynamic adjustments to routing schemes based on system load, subnet transmission speed, queue size, and link integrity. The application program must not rely on any particular routing scheme for reliable operation, and it must not circumvent network management functions in an attempt to achieve higher performance.

Reliability

Reliability of the network is required in two distinct areas. Data delivery reliability is the assurance by the network management software that information from the source device is received without error at the destination device. Error detection and correction techniques of various types are used to assure delivery, and they may be used at various stages of the transfer process. Application programs can assume that completed data transmission means error-free delivery, but they must not assume that any specific technique or transmission speed is used to assure delivery. Even in very high-speed networks, delays can be incurred during peak loads that affect transmission speed but not transmission reliability. Care must be taken to design application programs that do not require immediate reply from any particular device on the network, since some data delivery schemes cannot provide a guaranteed transmission time.

Another aspect of network reliability that is required by damage control systems is the assurance of network integrity. Because DC systems are designed to combat damage, the network itself must also be protected from damage. Redundant networking can provide a secondary network as a backup when the primary network fails. Alternate routing can be used when a portion of the network is lost but alternate routes still exist. Regardless of the method used to assure network integrity, the application program must not be affected by any modifications to interfaces, routing techniques, or other mechanisms made by the network management software. Likewise, the application program must not rely on specific network integrity characteristics such as specific functionality or use of the primary (or secondary) network.

Communications Security

Communications security (COMSEC) is a serious concern in networks handling classified data and having public access. COMSEC is not a problem, however, in a shipboard environment because of the physical security provided by isolation from other networks. Damage control sensor information is generally not of a classified nature and therefore requires no security measures, but as DC information systems become more computerized, some level of security will be required. When integrated with other ship systems that handle both classified and unclassified information, multilevel security becomes a problem. Multilevel security, and COMSEC in general, can be considered a network management function, and as such must not require or permit any interaction with DC application programs. Other security-related functions such as system and application log-in and the display of security classification bars on the operators console as described in the Department of Defense Human Computer Interface Style Guide [8] must be

transparent to the underlying application. Although operator interfacing may be required for password entry or operator identification, these functions must be performed without affecting the application program.

COMPUTER ARCHITECTURE ISSUES

New types of DC equipment will invariably be implemented on a variety of computer systems with different processor architectures. Because different processors use different byte-ordering schemes for multibyte quantities, it is important to define the scheme to be used when interprocessor communication is being performed over a network. When the most significant byte of a multibyte quantity is transmitted first in a serial stream, it is referred to as big-endian byte order. When the least significant byte is transmitted first, it is called little-endian. Some computers with byte-addressable memory use big-endian order internally, while others use little-endian. If one type of byte ordering is used for data transmission and the other type is used internally, byte swapping must be performed to handle the data correctly. The original design of the X Windows protocol [9] circumvents the byte swapping problem by providing separate network access connection ports for each byte order. A simpler, more suitable approach for damage control systems is to provide the byte order information within the data packet, similar to later versions of X Windows [10]. This approach allows sensor interface units (SIUs) to handle data in their most efficient manner and puts the burden of byte swapping on the operator's workstation or console device. Since these workstations will have powerful processing capabilities, minimal impact is incurred.

Word alignment is another computer architecture issue that must be addressed. Information being passed by the protocol may contain data comprising 16- and 32-bit quantities. Some computers require 16-bit quantities to be aligned on 16-bit boundaries, and 32-bit quantities to be aligned on 32-bit boundaries in memory. To allow efficient implementations of the protocol on these machines, blocks of data must always be multiples of 32 bits, and all 16- and 32-bit quantities within a block must be aligned on 16- and 32-bit boundaries, respectively. This restriction poses no problem for smaller 8-bit machines and creates only a small overhead in memory and block size. Performance increases in larger machines may be significant with this approach.

PROTOCOL ISSUES

Several significant issues must be considered in the design of a damage control systems protocol. To provide sensor-, interface-, and console-specific message passing, a coherent addressing scheme must be established. A method of prioritizing data must be established to permit quick handling of mission-critical information. To define the functionality of the protocol, the types of data packets permitted and their relationships to each other must be specified. Provisions for interaction with networked databases and other future capabilities must be made.

Addressing

In a networked environment, an addressing scheme must be established that can provide both individual and group addresses. Shipboard damage control systems must be able to access information pertaining to individual sensors, consoles, or interface multiplexers. The use of a 16-bit address field provides 65,536 individual addresses, but some allowance must be made for different addressing modes. A broadcast address must be available to transfer the information in a single message to all devices on the network. Typically, the address composed of all-ones bits (i.e., the address with a value of 65,535) is reserved for a broadcast address. The all-zeros address is sometimes used as an alternate broadcast address, but this address may cause problems when an uninitialized device incorrectly uses zero as its own address. To prevent this problem, the all-zeros address is an invalid address, and the all-ones address is used to conform to normal networking practices.

Being able to address a group of devices with a single address is often convenient and can reduce the amount of network traffic considerably. Addresses in the 32,768 through 65,534 range are reserved for group addresses. This allows maximum flexibility in the assignment of both group and individual addresses. Group addressing would typically be used to address all consoles, all devices on the main deck, all devices in engineering spaces, all smoke detectors, or any other grouping that may be used to make system management more efficient.

Multiplexers that provide interfaces to several devices are assigned a range of addresses to respond to. The number of addresses used is equal to the maximum number of interfaces supported by the multiplexer plus two additional addresses. The first address in the multiplexer range is assigned to the multiplexer controller (not to any attached device), and the last device in the range signifies all devices attached to the multiplexer (effectively a multiplexer group address). The addresses for the devices attached to the multiplexer are formed by adding the multiplexer address to the number of the interface used by the device.

Priority Service

The speed at which a response to an event is required varies widely for DC systems. Some events are simply notification of a change in status of a device, such as a hatch being opened, while others may threaten the survivability of the ship, such as a fire being detected in a magazine. Prioritized service for messages is required to prevent critical messages from being delayed by insignificant ones already in the message processing queue. Low priority is assigned to messages that do not affect the normal operations of the ship, such as messages associated with information logging. Normal priority is used for normal DC system operations such as changing sensor configurations. Warning priority is used to alert DC personnel to a problem that may lead to an emergency condition. Color displays would normally present this information as a "yellow alert". Alarm priority means that an emergency condition (a "red alert") that requires immediate action has occurred. Each message contains a priority field that indicates the criticality of the information it contains.

Packet Types

The functionality of the protocol is determined by the types of data packets used and their association to each other. Packets are used to send commands, requests, replies, notifications, and acknowledgments between devices on the network.

A *command* packet instructs a device to take a particular action. Commands are generally sent from an operator's console to an SIU to configure, activate, deactivate, modify, or test sensors, interfaces, multiplexers, or associated equipment. Commands may also be sent from one console to another to assign operational responsibility, reconfigure console functions, or pass pertinent information. Commands are a unilateral transfer from the source to the destination, and neither require nor expect any reply. An example of a command from a console to an SIU would be to set the alarm threshold of a heat sensor to a particular temperature.

A *request* packet solicits information. Requests are typically sent from an operator's console to an SIU to determine the current value of a sensor reading, an interface configuration, or the operational status of a particular device. Requests may also be sent between consoles to solicit operational control, interoperator messages, or system configuration status. All request packets require a reply from the device to which the request was addressed. A time-out feature is used with all request packets. The time-out period is begun when a request is issued, and if no reply is received before the expiration of the time-out period, an exception function is executed. This exception function is typically the issuance of an alert message to the operator indicating that no reply was received to a request. An example of a request from a console to an SIU would be to request the current temperature of a heat sensor.

A *reply* packet provides the information solicited in a request packet. The purpose of the reply packet is to cancel the time-out function initiated by the issuance of a request packet. An example of a reply from an SIU to a console would be to report the current temperature of a heat sensor.

A *notification* packet provides unsolicited information to a console. These packets are usually initiated by changes in status, alarm conditions, or other events that require operator notification or action. Notification packets require an acknowledgment from a console. This acknowledgment can be automatic (through software control), or may require operator action to initiate it. A time-out feature, similar to the one used with request packets, is used by the initiating device to assure delivery of the notification. Upon expiration of the time-out period, the notification is issued again, assuring response to an alarm condition at an operator's console. An example of a notification packet from an SIU to a console would be to report a high-temperature alarm condition from a heat detector.

An *acknowledgment* packet acknowledges receipt of a notification packet. The purpose of the acknowledgment packet is to cancel the time-out function initiated by the issuance of a notification packet. An example of an acknowledgment packet from a console to an SIU would be to acknowledge a high-temperature alarm condition from a heat sensor.

Packet structure

Each packet is structured to provide certain message processing functions in the packet header, along with variable length, variable context data in the message block. The header contains the priority, packet type number, protocol version number, flags for configuration indicators, addressing information, and a unique packet identification number (Fig. 6).

Priority	Packet type	Flags F1	F2	F3	F4	Version number	Source address	Destination address	Unique identifier
4 bits	4 bits	1 bit	1 bit	1 bit	1 bit	4 bits	16 bits	16 bits	16 bits

Fig.6 — Header information

The first byte of the header contains the packet priority and type number. The most significant four bits of the first byte provide the packet priority, and the least significant four bits are the packet type. The second byte of the header contains flags that convey information such as byte order, along with the protocol version number of the device sending the packet. Flags are single-bit indicators of the presence of a condition if the bit value is one, and the absence of the condition if the bit value is zero. The minimal configuration presented here requires at least the byte order flag. A value of zero for the byte order flag indicates that little-endian byte ordering is used, and a value of one indicates big-endian. The protocol version number may be used as upgrades, and revisions of the protocol provide enhancements that may not be supported by older devices. Table 1 shows the values used for the first two header bytes. The remainder of the header is composed of three 16-bit quantities that convey the source address, destination address, and a unique identifier number for the packet. The unique identifier is generated in the source device for each new packet transmission, but repeated packets that are sent because of time-outs use the original identifier, thereby preventing multiple alarms for a single-notification packet.

Table 1 — Priority, Packet Type, and Configuration Flag Values Used in the First Byte of the Header

Priority values	
0	Low
1	Normal
2	Warning
3	Alarm

Packet type values	
0	Reserved
1	Command packet
2	Request packet
3	Reply packet
4	Notification packet
5	Acknowledgment packet
6-15	Unassigned

Flag values	
F1	Big-endian byte order
F2-F4	Unassigned

The message block that follows the header is composed of two 16-bit fields that provide the function code for the data and the length (in 32-bit longwords) of the appended data block, followed by the variable length data (Fig. 7). The data being transferred depends on the function and format of the information, and can be any length, but the length of the packet is still required to be a multiple of 32 bits. Any excess bytes appended to the block to meet this requirement are ignored. Note that zero is a valid length for the data block, since some function codes provide no amplifying data.

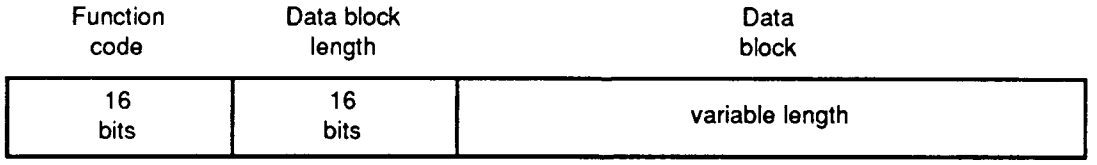

Fig. 7 — Message block information

The function codes indicate the action to take, information to transfer, or type of notification to act on, and provide the mechanism through which consoles and SIUs interact. These function codes will be defined as required to include the capabilities of sensors and interfaces under development. Command function codes sent from consoles to SIUs include instructions such as set alarm value, reset interface, enable built-in test (BIT) checking, disable alarm, activate device, or calibrate sensors. Command functions transmitted between consoles include, for example, assume operational command, display operator message, update console database, and switch to alternate network interface. Request function codes sent from consoles to SIUs perform queries of data such as sensor values, alarm settings, interface status, BIT test results, and number of active sensor interfaces. Interconsole requests include queries for system configuration status, interoperator messages, and database information. Notification packets have function codes for events such as alarm threshold exceeded, device control assumed by local control panel, interface fault detected, and power-up initialization complete.

A well-defined, comprehensive assignment of function codes is required to have an effective interaction between the sensors, SIUs, and consoles. Because current sensors provide little more than a pass/fail indication of their alarm status, and development of advanced sensors is in its infancy, the assignment of function codes to devices or interfaces would be premature at this time. These assignments will be defined after the development of the sensors and interfaces has progressed.

Database Interaction

Databases that allow data retrieval via network access have become an efficient and convenient way of obtaining information without requiring the data to be encapsulated within an application program. This allows the database information to change as required, without changing the application programs that use the information. Application programs with encapsulated databases require software upgrades on all workstations that run the programs when the data change, but with networked databases, none of the application programs require modification if the data change. Future shipboard systems will have databases for engineering drawings, DC diagrams, equipment status, maintenance records, duty rosters, and other information that would be helpful to DC personnel. The ability to transfer these data must be provided for in the interface protocol.

Networked databases will be based on commercial products with standard access techniques and protocols such as structured query language (SQL). These protocols should be used for retrieval of database information, and knowledge of the protocols must be included in application programs that require database interaction. Access control such as passwords or other security measures that may be used by these databases can be accommodated with a request-and-reply verification procedure prior to allowing data retrieval. Functions codes to support these actions will be added to the protocol as networked database access is made available on ships.

FUTURE CONSIDERATIONS

The most important attribute of the protocol needed to meet future requirements is the ability to expand its capabilities to support new equipment designs, technologies, or techniques. Expandability of the protocol has been included in the design through the use of functions codes and version numbers. As new equipment is developed with features and functions that are unique to it, new function numbers will be added to the protocol to support them.

To keep the complexity of the protocol low, function codes are generally designed to be as generic as possible. For example, a request for a sensor reading can be used for virtually all sensors, but the interpretation of the data is dependent on the particular sensor. The sensor reading may be given in degrees Fahrenheit for a heat sensor, percent per foot obscuration for a smoke detector, inches of depth for a flood detector, type of combustion material for a flame detector, or a simple open/closed indication for a hatch closure sensor. In each case the reporting mechanism (the protocol packet type and function code) is the same, even though the data have entirely different meanings.

The use of protocol version numbers is specifically designed to determine the capabilities of specific devices as they are upgraded and enhanced. Early versions of many devices may report readings as simply above or below a fixed threshold setting, while later versions will provide the actual sensor value. Version numbers are used to determine if the value provided uses the format specified for the older equipment or the newer equipment. The capability of a particular device to respond to a particular function code is determined by its version number. When new function codes are added to the protocol, a corresponding new version number is created. Older equipment will be backward compatible, but it will not have the capability to respond to the newer function codes.

CONCLUSIONS

Damage control systems are evolving rapidly into large, complex systems with a potential of collecting vast quantities of detailed information. To handle this information effectively, DC systems will be transformed from their traditional centralized architectures to more efficient and more survivable distributed architectures. A coherent method of transferring DC information across these distributed networks is required in the form of an interface protocol that can be used by DC systems and personnel to quickly assess damage situations and initiate quick corrective actions.

In this report, we have described the requirements of the mechanism through which advanced damage control sensors and consoles interact, and have addressed issues that may impact the use of the protocol in various system configurations. The impact of networks of various media types, topological structures, data transmission speeds, and management techniques has been discussed with respect to their use as an underlying structure upon which the interface protocol is built. Careful attention has been taken to address current and future computer system architecture requirements in the design of the protocol structure. Specific protocol requirements for device and group addressing, prioritized service, and database access have also been described. Details of specific requirements for the ability to send commands, requests, replies, notifications, and acknowledgments between devices has been presented, and the method for expanding the protocol to meet the needs of the future has been given. The concepts and recommendations presented in this report will provide a fast and effective way for complex damage control systems to be managed and operated quickly and efficiently.

ACKNOWLEDGMENTS

The author thanks Dr. Patricia Tatem of the Navy Technology Center for Safety and Survivability (NRL Code 6180) for her support of this effort. This report is submitted in partial fulfillment of Milestone 3, Task 6 of the Damage Control Project (RH21S22) of the Surface Ship Technology Block Program (ND1A/PE62121N). The work described herein is sponsored by the Office of Naval Technology (ONT211) and performed by the Naval Research Laboratory (NRL Code 5533).

REFERENCES

1. H. K. Whitesel, W. F. Zeller, and C. A. Miller, "Sensor Requirements and Technology for Damage Control," David Taylor Research Center Report DTRC/PAS-90-7, June 1990.

2. D. L. Tate, "A Graphical User Interface Design for Shipboard Damage Control," NRL Report 9355, Aug. 1991.

3. "Damage Control Management System (DCMS) Baseline Definition (DDG 51)," Naval Sea Systems Command, Code 55X22, May 1989.

4. S. Herman and C. T. Loeser, "Damage Control – The Last Line of Shipboard Defense," *Naval Engineers Journal* **104**(1), 63-79 (1992).

5. J. M. McQuillan and V. G. Cerf, *Tutorial: A Practical View of Computer Communications Protocols* (Institute for Electrical and Electronics Engineers, New York, 1978).

6. T. D. West, "The NRL Integrated Communications Environment Network (NICENET) Tutorial," Version I, Naval Research Laboratory, September 1989.

7. *Survivable Adaptable Fiber Optic Embedded Network II*, Military Handbook MIL-HDBK-0036 (DRAFT), Naval Ocean Systems Center, January 1991.

8. "Department of Defense Human Computer Interface Style Guide,Version 1.0," Defense Information Systems Agency, Center for Information Management, February 1992.

9. R. W. Scheifler and J. Gettys, "The X Window System," *ACM Trans. Graphics* 5(2), 79-109 (1986).

10. A. Nye, *Volume 0: X Protocol Reference Manual* (O'Reilly and Associates, Inc., Sebastopol, CA, May 1990).