No. 3174

APPLICATION LAYER COMMUNICATION PROTOCOLS FOR BUILDING ENERGY MANAGEMENT AND CONTROL SYSTEMS

S.T. Bushby
ASHRAE Associate Member

ABSTRACT

The requirements for an industry standard communication protocol for energy management and control systems (EMCS) are discussed in terms of the International Organization for Standardization's (ISO) Open Systems Interconnection (OSI) Basic Reference Model with emphasis on the application layer.

The information exchange requirements of commonly used control strategies are analyzed to develop a list of the minimum application level services required for an industry standard. This list was augmented to include additional desirable services, based on several years of experience building and operating an EMCS at the National Bureau of Standards.

Two public EMCS protocols, believed to be representative of the current state of commercially available EMCS, are described and analyzed to determine their ability to meet the application service requirements developed and their compatibility with the OSI Reference Model. Both protocols were found to meet the minimum requirements, but neither provides all of the services considered desirable. Neither protocol was found to be compatible with the OSI model.

INTRODUCTION

The Arab oil embargo of 1973 and the resulting escalation in world prices for primary energy resources caused a major rethinking of the uses of energy in the United States and the western world. One of the many sectors of the economy affected by this was the building industry. As long as fuel and electricity prices were low there was little interest in modernizing climate control systems in buildings. Sharp increases in energy costs stimulated a great deal of interest in finding ways to provide occupant comfort while reducing costs. Much effort was put into development of "energy management and control systems" (EMCS) in the late 1970s and early 1980s to meet this need.

The computer industry was undergoing rapid development during the same time period. Microprocessor technology made large advances while prices dropped significantly, opening up many new economically viable applications for microprocessor technology. One of these applications was and continues to be EMCS. The early focus of EMCS research efforts was in the development of direct digital control (DDC) techniques and algorithms to implement them. DDC equipment and control algorithms are now readily available from many commercial sources. The trend in the building industry now is toward distributed DDC systems where control functions are implemented by a network of microprocessors. Other building services are also being integrated into these microprocessor networks, such as security and fire systems.

Although microprocessor-based EMCS are now available from many manufacturers, there is at the present time no industry standard for communication protocols. Most companies in the building controls business have opposed development of standards on the basis that it would restrict their ability to market a product with unique capabilities. It was also feared that standards would lock the technology into the lowest common denominator of its present form and force all manufacturers to produce equipment at a fixed level of mediocrity.

Steven T. Bushby is a research engineer in the Mechanical Systems and Controls Group, Building Environment Division, Center for Building Technology, National Bureau of Standards, Gaithersburg, MD 20899.

Pressure from customers not wanting to be locked into using a single vendor began to mount. Some company managers began to decide that it may be a desirable marketing feature to be able to interface with various companies' equipment. With these developments the resistance to standards began to fade. In January, 1987, the American Society of Heating, Refrigerating, and Air Conditioning Engineers, Inc. (ASHRAE) approved formation of a committee to develop an industry standard for EMCS communication protocols (EUN Feb. 23, 1987).

There is considerable controversy in the industry about whether such standards are a good idea and what form they should take if they are adopted (EUN March 2, March 9, 1987). A great deal of the controversy stems from a lack of consensus about what the scope of such a protocol should be. There is much talk in the building industry about "intelligent buildings." Intelligent buildings have computer systems which integrate control of many building services including security systems, lighting systems, fire alarm systems, telephone communications, data processing, and climate control systems. A future communication protocol standard will probably have to address the possibility of integrating many building services into one package.

This paper will address only protocols for climate control systems, usually referred to as EMCS. An analysis of the application needs in the context of the International Organization for Standardization's (ISO) Open Systems Interconnection (OSI) Basic Reference Model, will be made and two public EMCS protocols will be examined. Neither protocol conforms to the OSI Model, but they do represent the current state-of-the-art in commercially available systems.

DISTRIBUTED EMCS ARCHITECTURES

Distributed, microprocessor-based, energy management and control systems generally utilize a hierarchical control structure as depicted in Figure 1. The highest tier in the control hierarchy will be referred to as the host or central control unit (CCU). This level provides supervisory control functions, such as changing setpoint schedules or parameters used in the control algorithms of lower level controllers. The CCU is also used to log performance data and alarms, and serves as the operator interface of the EMCS.

The second tier contains controllers characterized by the ability to handle multiple control loops. These controllers are generally a microprocessor/multiplexor combination and are called by various names. For the purpose of this paper, they will be referred to as field interface devices, or simply FIDs. A FID may be used to provide multiple DDC loops or it may be used to supervise a group of lower level controllers.

The lowest tier contains simple controllers designed to provide a single DDC loop. These controllers are called unitary controllers and are often built into hardware devices like a fan or compressor.

The distribution of software throughout the various levels of the control system may vary from installation to installation. Two major factors in deciding where control software should reside are the hardware capabilities of the microprocessors and the need to be able to maintain control of equipment, even if it is not optimal control, in the event of a failure at the supervisory level.

The communication architecture is completely distinct from the control hierarchy described above. Two types of communication architecture may be found in current energy management and control systems. The first closely follows the control hierarchy. The CCU communicates and exchanges information, on a master - slave basis, with FIDs and UCs. There is no peer-to-peer communication and no sharing of information between FIDs or unitary controllers.

The second type of communication architecture is shown in Figure 2. In this case peer-to-peer communication is allowed and information may be shared by the various controllers. This architecture can increase system reliability. A CCU of some sort is still needed to provide an operator interface and supervisory control, but the CCU no longer controls communication between processors on the network. This reduces the possibility for a single source failure disrupting all network communication, making shared data and processing capability more feasible. This architecture is not common, but a few systems are now available. It is likely that this will become the standard approach in future systems.

COMMONLY USED CONTROL ELEMENTS AND THEIR INFORMATION EXCHANGE REQUIREMENTS

An analysis of the type and quantity of information to be exchanged in a network is a necessary precursor to defining communication protocols. Several control algorithms commonly implemented in building EMCS software are discussed below. A typical EMCS will combine several of these algorithms into one unified package. Their purpose is briefly described and the information exchange requirements for each is analyzed. Combining this information with the other special functions required in a distributed EMCS will provide a basis for defining the application specific services needed in an application layer protocol.

ECONOMIZER cycles are control algorithms for minimizing the use of air conditioning equipment by using outdoor air for cooling a building's interior when outdoor weather conditions permit. Two basic types of economizer algorithms are used. A dry-bulb economizer utilizes the outdoor and return air dry-bulb temperatures to position the outdoor, return, and relief air dampers. An enthalpy economizer performs the same function, but the bases for the control decisions are the enthalpies of the return and outdoor air. The enthalpy economizer is more complicated and requires additional sensors, but it has been shown to save more energy in some climates. A detailed description of economizer algorithms may be found elsewhere (Kao 1983, Park et al. 1984, May and Kelly 1985, Spitler et al. 1987).

The enthalpy economizer can be used to define the information exchange requirements for economizer cycles:

INPUTS	OUTPUTS
temperatures:	damper control actions
supply air, return air,	chilled water valve actions
outdoor air, changeover,	
set point	
humidities:	
return air RH (or dew point)	
outdoor air RH (or dew point)	
position sensors:	
outdoor air damper	
return air damper	
relief air damper	
chilled water valve	
tuning constants:	
eg. PID constants	
sequence delay time	

The economizer usually resides at the FID level of the control hierarchy. The values for temperatures, humidities and positions are determined from the sensor outputs and control signals are sent to device actuators.

Communication with the CCU level is needed to allow the operator to set scaling factors and alarm limits for sensor outputs, and to pass data and alarms up to the CCU. Some sensor data may also need to be exchanged between FIDs.

DUTY CYCLING and DEMAND LIMITING algorithms are used to cycle equipment on and off for reducing electricity consumption during part load conditions and to switch electrical loads off during peak utility rate periods respectively. A detailed description of these algorithms is given by May (1983), Park (1984), and May and Kelly (1985). The information exchange requirements are as follows:

INPUTS	OUTPUTS
duty cycle interval	on/off switching for
off period	effected devices
phase	
minimum on time	
minimum off time	

Duty cycling/demand limiting software usually resides at the FID level of the control hierarchy. All of the inputs are parameters which must be exchanged with the CCU level. The outputs are control signals to device actuators.

TIME OF DAY CONTROL algorithms are used to initiate tasks at a particular time on specified days (May 1983). Examples would be nighttime setback, weekend scheduling, and holiday scheduling. This software usually resides at the FID level of the control hierarchy and the outputs are on/off commands to controlled devices, starting and stopping of software tasks residing in the FID, or changing parameter values. The information exchange requirements are:

INPUTS	OUTPUTS
Task ID	implement FUNCTION
FUNCTION	
time	
day (of week or year)	

Task ID identifies which task is to be addressed and FUNCTION specifies what action is to be taken. It is also necessary to be able to reset FID clocks from the CCU in order to synchronize them.

OPTIMUM START/STOP algorithms use prediction techniques to determine the time of day to shut equipment down, but still maintain building comfort levels until the end of the occupancy period, or the time of day to start equipment to achieve comfort levels by the time the occupancy period begins (Park 1983). There is considerable variation in the complexity of the techniques used to determine the start/stop times. An optimum start/stop algorithm might be implemented at either the supervisory level of the CCU or at the FID level. Typical information exchange requirements would be:

INPUTS	OUTPUTS
outdoor air temperature	on/off switching for effected devices
zone air temperature	
zone air temperature setpoint	
hours of zone occupation	
max. permissible zone temperature	
min. permissible zone temperature	
earliest permissible shut off time	

There are functions that an application layer protocol needs to support in addition to the requirements dictated by the control algorithms discussed above. These functions may not be required in the sense that mechanical equipment cannot be controlled without them, but experience operating an EMCS suggests that they are highly desirable and should be included in a standard. The functions include:

1. Requesting a specific group of data values from the FID on a periodic basis for the purpose of logging trends. In the extreme case this group of data values would represent all sensors connected to the FID.

2. Remote reset capabilities to refresh FID software if it develops errors.

3. The ability to download software from the CCU to the FID or upload software from FID to CCU.

4. Troubleshooting information from the FID in the form of indications of what control actions the software is trying to execute, device status reports, and a communications test.

5. Access control restrictions to prevent unauthorized changes to control parameters.

THE ISO OPEN SYSTEMS INTERFACE MODEL

Establishing computer-to-computer communication between heterogeneous machines is a difficult task. Different vendors use different data formats and different data exchange conventions. Even within one vendor's product line different models often communicate in unique ways. As the use of computer to computer communication and networking proliferates, a one-at-a-time special-purpose approach to communication software development becomes prohibitively expensive. The only solution is to adopt appropriate standards.

Before standards can be developed, the complex problem of heterogeneous, distributed communication needs to be broken down into manageably sized tasks. A structure or architecture defining these tasks must be developed. In 1977, the ISO formed a committee to address this problem and define a reference architecture to be used for developing communication standards. The result of this work is the Open Systems Interconnection - Basic Reference Model (ISO 1984).

The ISO committee chose to adopt a seven-layered architecture as shown in Figure 3. The layers are arranged in a hierarchical fashion with each layer performing a unique function necessary for the communication task. A given layer relies on lower layers to perform more primitive functions. The key to understanding layered architecture is to think of each layer as a black box with interfaces on both sides of the box carefully defined. The user's application program connects to the OSI application layer and communicates with a second, remote user application program. This communication appears to take place between the two applications as if they were directly connected through their application-to-OSI application layer interfaces. No knowledge or understanding of the other layers is required. In a similar manner, each layer of the protocol relies on lower layers to provide communication services and establishes a virtual peer-to-peer communication with its companion layer on the other system. The only real connection between the two systems takes place at the physical layer.

Because any given layer of the protocol interfaces only with the layers above and below, changes in the protocol become easier. Ideally, the layers should be defined such that changes in one layer do not require changes in the other layers. In practice, changes in one layer may sometimes require modification to the interface with an adjacent layer.

It is beyond the scope of this paper to describe the features of each layer. Numerous descriptions can be found in the literature and ISO standard 7498. It is the highest layer, the application layer, which is the most appropriate place to begin EMCS protocol development. It is here that the specific requirements of EMCS are addressed. After these requirements are understood, it will become more clear what services from lower layers in the OSI model will need to be implemented.

THE ISO APPLICATION LAYER STRUCTURE

An "application" may be defined as a set of information processing requirements. An "application process" is a logical element within a system which performs the information processing required for a particular application. In a distributed system, each distributed portion of the application is a separate application process (Bartoli 1981). In the context of EMCS, the application is to control the climate of a building or a group of buildings, while minimizing cost. Each element in the distributed architecture (CCU, FID, UC) is a distinct application process.

Each application process has associated with it a universe of information, which can be thought of in terms of its semantics or the meaning of the information, and its representation or syntax. In order for two application processes to communicate with each other there must be an overlap or sharing of at least a portion of these semantics. This amounts to both parties having some knowledge and understanding of the topic to be discussed. This concept is illustrated in Figure 4.

Each application process in the EMCS has its own distinct universe of information, only a portion of which will be in common with a different application process. That common portion may itself vary, depending on which of the other application processes it is communicating with. For example, the semantics that a FID has in common with a second FID may be different from the semantics that the same FID has in common with the CCU. All that is required is that the common region contain semantics that are necessary and sufficient to establish control of a distributed EMCS.

The application layer of the OSI model provides for the definition of application service elements which are responsible for the transfer of information in the overlapping region of Figure 4. Thus, a communication protocol standard only addresses that portion of the application processes which overlap. It is entirely possible to have an open protocol standard and still permit individual companies to maintain proprietary software in each of the application processes which make use of the information exchanged. The protocol merely provides a common syntax which permits exchange of the semantics which are in common for the two application processes.

The application entity structure considered by the ISO is shown in Figure 5. Two categories of service elements are recognized: common application service elements (CASE); and specific application service elements (SASE) (Bartoli 1981). Common application service elements provide capabilities required for information transfer which are independent of the nature of the application. Specific application service elements provide information transfer capabilities to satisfy the particular needs of a specific application process. The user element represents capabilities needed as an interface between the application layer service elements and the remainder of the application process.

Common application service elements defined by the ISO can be divided into nine categories based on the services they provide. These categories are listed in Table 1 along with their associated service primitives (Bartoli 1981). The EMCS information exchange requirements discussed in the previous section provide a basis for determining the services which need to be provided by specific application service elements. These services are summarized in Table 2.

Two public protocols for EMCS's will be examined to determine how well they provide the capabilities identified as necessary for EMCS. One protocol will be referred to as protocol A and the other as protocol B. The last column in Table 2 indicates which application service elements are provided by each protocol.

DESCRIPTION OF PROTOCOL A

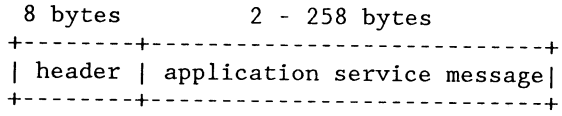
The developer of protocol A made a corporate decision to install distributed direct digital control energy management systems in some of the company's facilities based on expectations for improved performance and reduced energy costs. Specifications were drawn up based on the use of personal computers (PC) as field processors. These processors are linked together along with a mainframe host computer via a communication network. The specifications and protocols developed for in-house use were made public with the intent that control system vendors might adopt protocol A for their commercial systems, giving the company a presence in the EMCS market (IBM 1985).

Protocol A uses a master-slave approach to control access to the network. The mainframe host computer serves as the master providing direction to one or more PC-DDC building control systems. The system architecture is shown in Figure 6. The host communicates with PCs over multidropped leased telephone lines using either synchronous or asynchronous communication protocols.

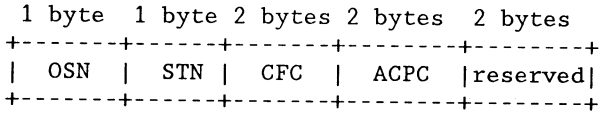
Protocol A defines a set of "application messages" which can be imbedded as data in the transmission. The same application messages are used in both the synchronous and asynchronous versions of the protocol. This is a step towards a layered architecture in the sense that a distinction is made between the application messages and the means of transmitting the message. The application message is self contained and can be used with either transmission scheme without change.

Protocol A Application Message Services

Protocol A specifies an application message with the following format:



The header contains the following fields:



where:
- OSN - originating system number
- STN - system transmission number (used in messages initiated by host)
- CFC - communication function code (1 = no response expected; 2 = response expected)
- ACPC - application command processor code, essentially a code to indicate the type of application service message which is to follow

Table 3 is a list of application message services provided by protocol A.

DESCRIPTION OF PROTOCOL B

A second company has released a draft EMCS protocol which will be referred to as protocol B (Fisher 1986). Protocol B is described by its author as a generalized version of a previous "standard" used for four years in the company's EMCS products.

Protocol B is based on a master-slave medium access control. In terms of the protocol definitions, there is a "host" which is a requester of information and represents either human operators or control programs. There are also one or more "targets" which are field devices or systems which are being queried or commanded. Communications are always initiated by the host, even for alarm and exception reporting. Messages are composed of ASCII characters transmitted serially using standard asynchronous data frames. The transmission medium may be either dial-up commercial telephone lines or a multidropped hard-wired bus.

Protocol B provides for two basic types of access to a target: protocol transactions, where the host sends a query in a specified format and a single target responds, and interactive terminal mode. Interactive terminal mode is initiated by the host and permits the host to emulate an interactive dumb terminal connected to the target. A simple control sequence is used to terminate this mode.

One of the most innovative features of protocol B is the use of symbolic-referencing for data access.[1] In the context of this paper, symbolic referencing refers to using a symbol, or name, to reference a particular sensor value or algorithm parameter. Symbolic-referencing relieves the host from maintaining tables of sensor addresses in the various and possibly dissimilar target systems. All sensor points and control parameters are represented by a twelve-character point name. Symbols also have properties which are given two-character names. A simple example would be a temperature sensor point which might have properties for Current Temperature (CT), High Alarm Limit (HL), and Low Alarm Limit (LL). The host can request information by specifying the name and the desired property. Knowledge of the physical point addressing structure of the target is unnecessary. However, requesting information by specifying a target port address is allowed in the protocol.

Protocol B provides for a "name cache" to efficiently transfer a group of data values which need to be fetched repeatedly. Once the cache has been set up it is possible to retrieve all of the values with one short message (9 bytes). This significantly reduces the protocol overhead compared to fetching the values by repeated use of the read point command. It also significantly reduces the number of transactions on the network which can improve overall network performance if high loading conditions exist.

General Message Format

Messages from the host to any given target begin with the ASCII semicolon (hex 3B) and end with the ASCII carriage return (hex 0D). The initial character of the response message from the target varies depending on the command from the host, but always terminates with an ASCII carriage return. The general message format for host commands is:

```
;uxx...ccct
where:
     ; =  the start of message indicator
     u =  unit number (This is the only exception to
          the seven bit ASCII character convention.
          All 8 bits are utilized allowing unit codes
          from 80 Hex to FF Hex)
    xx =  command code as a two hex digit number
   ... =  command dependent part
  cccc =  CRC16 of all bytes from the ; up to the CRC bytes
     t =  message terminator (ASCII carriage return
```

Under appropriate circumstances, error codes are returned in response to host commands.

---

[1] The terminology used in the protocol specification is "object oriented data access." "Objects" are defined with specified "attributes." This terminology, however, suggests features which do not appear to be included in the protocol and therefore is not used in this paper.

Protocol B Command Descriptions

In protocol B the command code and its associated data correspond to the role of an application layer service. Table 4 is a brief description of the host commands defined in protocol B. All commands need not be implemented in all targets. Any command sent to a target that does not implement that command will generate an error response (Fisher 1986).

ANALYSIS OF PROTOCOLS A AND B

Table 2 includes a comparison between the list of desired application services and the corresponding services actually implemented by the two public protocols examined. Neither of the protocols presented supply all of the services desired and some of the services that are provided may not be implemented in an optimal way. However, both of these protocols can be implemented in such a way as to provide all essential EMCS services. This is to be expected since they are actually being used in operating systems.

The symbolic-referencing approach permitted in protocol B is attractive from the standpoint of interconnectivity of field units supplied by different vendors. It eliminates the need to maintain tables of physical point addressing structure in the host for each FID and UC in the network. Host software based on a consistent set of symbol names and characteristics could communicate with any field device without any knowledge of the instrumentation wiring in the field device.

Symbolic-referencing can also simplify the process of passing control parameters. Instead of specifying table formats to pass sets of related parameters from CCU to FID for each control algorithm that might be implemented, these parameters could be considered as properties of a symbol defined for the algorithm being implemented. For example, PID control constants could be handled by a symbol, "PIDCONTROL", with properties "PC" (proportional constant), "IC" (integral constant), "DC" (derivative constant). The advantage is that it is not necessary to have a standardized table format and a specific application service to handle each case. A small number of application services which exchange values for symbol properties can be used for nearly all parameters. This approach will, however, require standardized symbols and properties.

Both of the protocols examined have some serious shortcomings. Master slave medium access control is a simple scheme but it is also restrictive. Peer-to-peer communication between field devices can be very advantageous from an instrumentation perspective because it permits sharing of data from sensors. High quality sensors can be used, where appropriate, to increase reliability and the cost can be distributed over all of the devices which need that particular data. In a carefully designed network with peer-to-peer communication, the impact from failure of a single processor can be kept to a minimum.

Both protocols have ignored current LAN technology and the seven layer OSI reference model for communication protocols. Protocol B has specifically rejected the OSI seven layer model for communication protocols. The specification states (Fisher 1981):

> "[Protocol B] is in no way an attempt to be completely universal, such as the OSI seven layer communications model...[protocol B] bi-passes the seven layer protocol by using a single layer Functional Architecture. MAP and TOP do not address functional architectures, except to provide an undefined 'applications layer'. In this regard, it would be possible to utilize a specialized applications layer for a MAP network which permitted [protocol B] devices to communicate over MAP. This approach is similar to using the space shuttle to cross the street."

Ignoring what the rest of the world is doing with respect to communication protocols may have some short term benefits but it does not give long term flexibility and durability, which is needed for an industry standard.

It is true that all of the services of the OSI reference model are probably not needed for EMCS applications. However, it is premature to reject the OSI model out of hand and start from scratch. A more reasonable approach would be to carefully analyze the current an future requirements of a building management control system protocol standard and see which layers of the OSI model may apply. A protocol can then be developed which is based on these layers and in effect null the unused layers. This collapsed communication architecture approach was used to develop Enhanced Performance Architecture (EPA) and mini-MAP (SME 1987).

The advantages of this approach are twofold. Such a standard would be more easily modified as the scope of the application or communication technology changes. Only layers that are affected need to be changed, leaving most of the standard intact. A variety of physical mediums could also be accepted in the standard without changing higher layers. This approach was adopted by ANSI and IEEE in the 800 series LAN standards. The second advantage is potential hardware cost reductions due to economies of scale if printed circuits being manufactured for LAN applications can be incorporated into EMCS technology.

If lower layers of an EMCS protocol are compatible with LAN standards, this would make it possible for the EMCS network and other LAN applications to be combined into a single communication system. This may not be desirable in all situations, but in others it could help reduce the effective cost of installing a modern EMCS.

CONCLUSIONS

Energy management and control systems have been discussed in terms of the information exchange requirements for distributed processing. The Application Layer of the ISO Open Systems Interface Model was discussed and a set of specific application service elements developed for EMCS. Two public communication protocols were examined in the context of the application requirements and the OSI Reference Model.

The two protocols were found to be adequate to meet the essential requirements of energy management and control systems but lacked some desirable application features. Neither is structured in terms of the OSI Reference Model and as a result offer neither the flexibility that a layered standard would have for future revision, or compatibility with other LAN applications that could bring hardware cost reductions due to economies of scale.

The process of developing an industry standard has begun under the auspices of ASHRAE. The standard committee can best serve the industry by carefully examining the current and future requirements of the application, examining current protocols in light of these requirements, taking the best features of the current protocols and recasting them in a structure compatible with both the ISO Model and other developing network technologies.

REFERENCES

Bartoli, P. D. 1981. "The Application Layer of the Reference Model of Open Systems Interconnection." Proceedings of the IEEE, Vol. 71, No. 12.

Energy User News February 23, 1987. "Standardizing EMCS Protocols:Panel Discussion - Part I."

Energy User News March 2, 1987. "Standardizing EMCS Protocols: "Standardizing EMCS Protocols Panel Discussion - Part II."

Energy User News March 9, 1987. "Standardizing EMCS Protocols:Panel Discussion Part III."

Fisher, D.M. 1986. "Public Host Protocol Guidelines Version 6. "American Auto-Matrix, Inc., One Technology Drive, Export, PA.

IBM. "IBM Facilities Automation Communication Network Specification, Revision: January 18, 1985." IBM-NAD, Facilities Automation System Center, Atlanta Georgia.

ISO. 1984. ISO Standard 7498, "Information Processing Systems-Open Systems Interconnection-Basic Reference Model." International Standards Organization. Specifications available from ANSI, 1400 Broadway, New York, New York 10018.

Kao, J., 1983. "Strategies for Energy Conservation for a Large Office Building." National Bureau of Standards. NBSIR 83-2746.

May, W.B.; Kelly, G.E., 1985. "Verification of Public Domain Control Algorithms for Building Energy Management and Control Systems." National Bureau of Standards, NBSIR 85-3285.

May, W.B., 1983. "Time of Day Control and Duty Cycling Algorithms for Building Management and Control Systems." National Bureau of Standards, NBSIR 83-2713.

Park, C., 1983. "An Optimum Start/Stop Control Algorithm for Heating and Cooling Systems in Buildings." National Bureau of Standards, NBSIR 83-2720.

Park, C., 1984. "Demand Limiting Algorithms for Energy Management and Control Systems." National Bureau of Standards, NBSIR 84-2826.

Park, C.; Kelly, G.E.; Kao, J., 1984. "Economizer Algorithms for Energy Management and Control Systems." National Bureau of Standards, NBSIR 84-2832.

SME.W1987. "Manufacturing Automation Protocol Specification Version 3.0." General Motors. Published by Society of Manufacturing Engineers, One SME Drive, P.O.Box 930, Dearborn, MI 48121.

Spitler, J.D.; Hittle, D.C.; Johnson, D.L.; Pederson, C.O., 1987. "A Comparative Study of the Performance of Temperature - Based Economy Cycles." ASHRAE Transactions. Vol 93, pt. 2.

TABLE 1

Summary of ISO Common Application Service Elements

Services Required	Service Elements
Origination	A_ASSOCIATE
Termination	A_RELEASE A_USER_ABORT A_PROVIDER_ABORT
Context	A_CONTEXT_DEFINE A_CONTEXT_SELECT
Interruption	A_SUSPEND A_RESUME
Information Transfer	A_TRANSFER A_TRANSFER_CONFIRM A_TRANSFER_EXPEDITED A_PURGE
Status	A_STATUS A_EXCEPTION_REPORT
Dialogue Control	A_PASS_TOKEN A_REQUEST_TOKEN
Synchronization	A_MARK A_SYNCHRONIZE A_RELEASE_MARK
Message	A_SEND_MESSAGE

TABLE 2

Summary of Specific Application Service Element Capabilities

Services Required	Service Elements	Implemented By
Data Transfer	A_REQUEST_POINT_VALUE	A, B+
	A_REQUEST_GROUP_VALUES	A, B
	A_REQUEST_ALL_VALUES	A, B
	A_SEND_POINT_VALUE	A, B+
	A_SEND_GROUP_VALUES	A, B
	A_SEND_ALL_VALUES	A, B
Parameter Transfer	A_REQUEST_TIME_OF_DAY_TABLE	A*, B*
	A_REQUEST_DUTY_CYCLE_TABLE	A*, B+
	A_REQUEST_PID_TABLE	A*, B+
	A_REQUEST_RESET_SCHEDULE	A*, B+
	A_REQUEST_ALARM_LIMITS	A*, B+
	A_REQUEST_SCALING_FACTORS	A*, B+
	A_SEND_TIME_OF_DAY_TABLE	A*, B*
	A_SEND_DUTY_CYCLE_TABLE	A*, B+
	A_SEND_PID_TABLE	A*, B+
	A_SEND_RESET_SCHEDULE	A*, B+
	A_SEND_ALARM_LIMITS	A*, B+
	A_SEND_SCALING_FACTORS	A*, B+
Alarm Reports	A_SEND_ALARM	A, B
	A_ERROR_REPORT	A, B
Set Clocks	A_SET_TIME	A, B
Troubleshooting	A_COMM_TEST	A, B
	A_CONTROL_ECHO_ON	
	A_CONTROL_ECHO_OFF	
	A_REBOOT_FID	B
	A_DOWNLOAD_SOFTWARE	B
	A_UPLOAD_SOFTWARE	B
Access Control	A_AUTHENTICATION	
	A_ACCESS_DECISION	
Broadcasting	A_BROADCAST	B
Event Messages	A_SEND_EVENT_MESSAGE	A, B

---

* A general file transfer procedure is defined in the protocol. Presumably, it can be used in some manner to perform these functions.

\+ These services are all combined into one request and one send service which accepts a symbol and a property as arguments.

TABLE 3

Application Message Services Provided by Protocol A

Service	Description	Length bytes
IPL and control	Host sends current time and date to PC to synchronize PC and subsystem clocks	8
Host-PC-COS reference table	Host transmits a reference table defining the points to be used in the control system	10-250
Host output command request	Host commands an output point on the PC	12
PC all points value request	Host requests transmission of the values of all points defined in the system	2
PC point value request	Host requests transmission of the values of all points in a specified list	8-128
PC point value response	Response to PC point value request	12-244
Event messages	PC sends an unsolicited event message to host (eg. alarm)	10-N
Unsolicited PC reports	Reports in the form of text sent to host for printing	8-N
Host/remote data request	Host or PC requests data from the the other.(eg. time,COS table, data)	2-46
Host/remote data transfer	Transmit data from a resident file in response to a host/remote data request	8-240
Communications test	Tests the communications link between host and PC	8
Application response	This message is used by the host/PC to respond to various messages by indicating error states regarding the application message received	2

TABLE 4

Application Services Provided by Protocol B*

Service	Description	Length bytes
Read Channel	Read an attribute of a hardware point	17
Write Channel	Write to an attribute of a hardware point	25
Read Point	Read an attribute of a named point	23
Write Point	Write an attribute of a named point	31
Acknowledge Transaction	Acknowledge the occurrences of a specified alarm transaction	13
Read Next Channel Attribute	Read the next attribute of a hardware point after the named one	17
Read Next Point Attribute	Read next attribute of named point after the attribute specified	23
Read Storage Record	Read 128 byte record from currently open storage file	13
Write Storage Record	Write 128 byte record to currently open storage file	270
Read Value and Status	Retrieve the named attribute of a point and its alarm status	23
Read Message Text	Read queued message text	9
Set Time and Date	Set current date and time	21
Restart	Cause a "soft" restart of FID	9
Say Hello	Communications Test.	9
Open Storage File	Open specified storage file.	26
Close Storage File	Close currently open storage file.	10
Initialize Name Cache	Preset a specified slot in a name cache to a certain point name and attribute name.	24
Fetch Through Cache	Fetch all of the values specified by active cache slots	9
Broadcast Commands		

---

* Virtual terminal services are not included

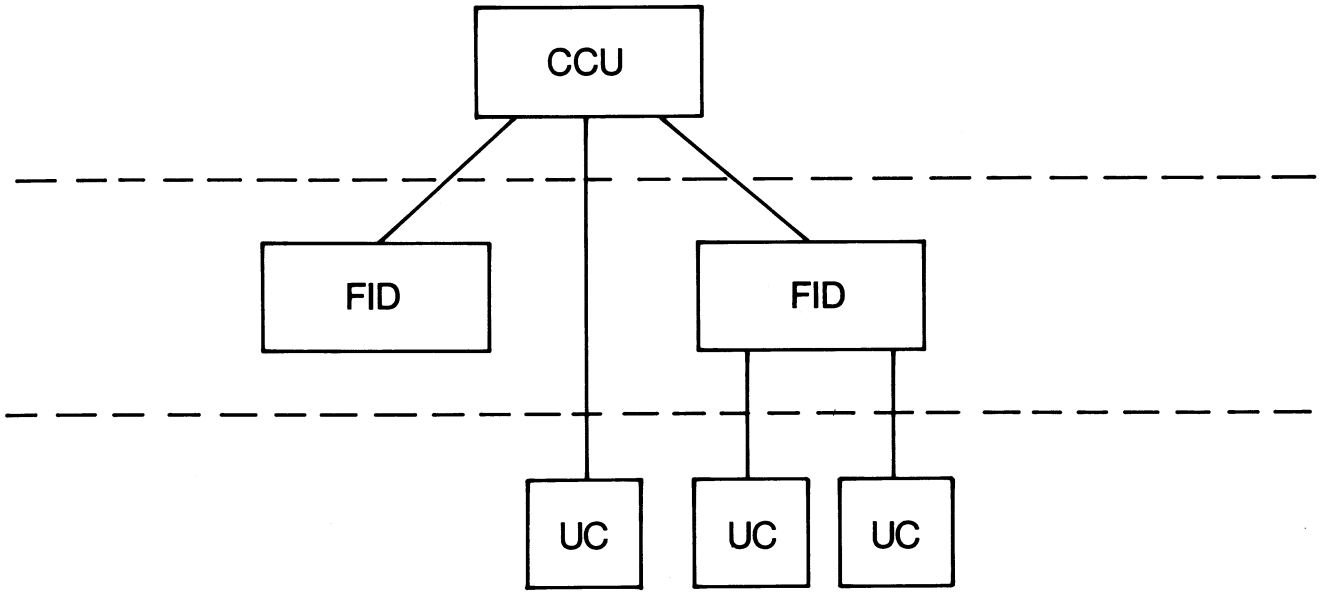


Figure 1 Hierarchical control structure for a typical EMCS. Note: CCU-central control unit, F10-field interface device, UC-unitary controller.

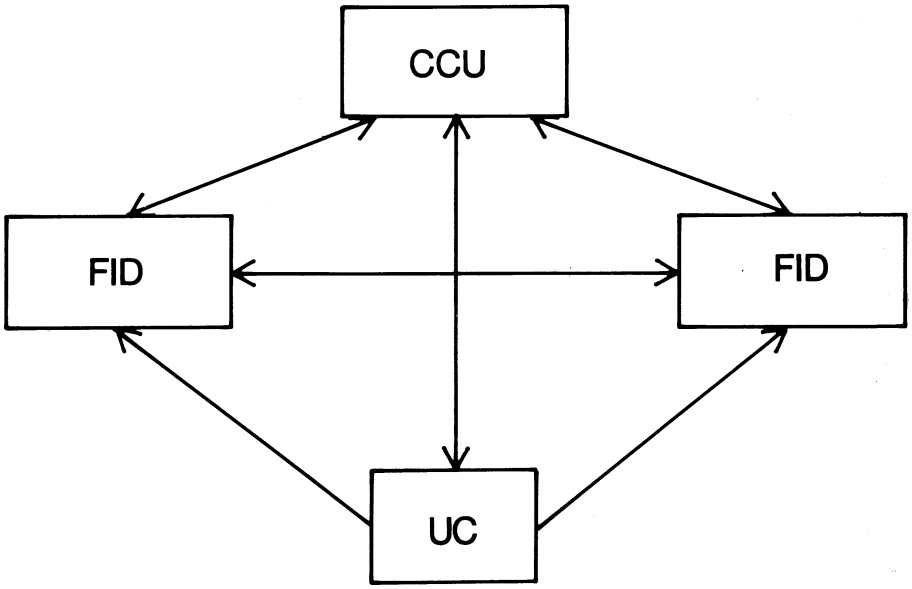


Figure 2 Advanced EMCS communication architecture with peer-to-peer communication

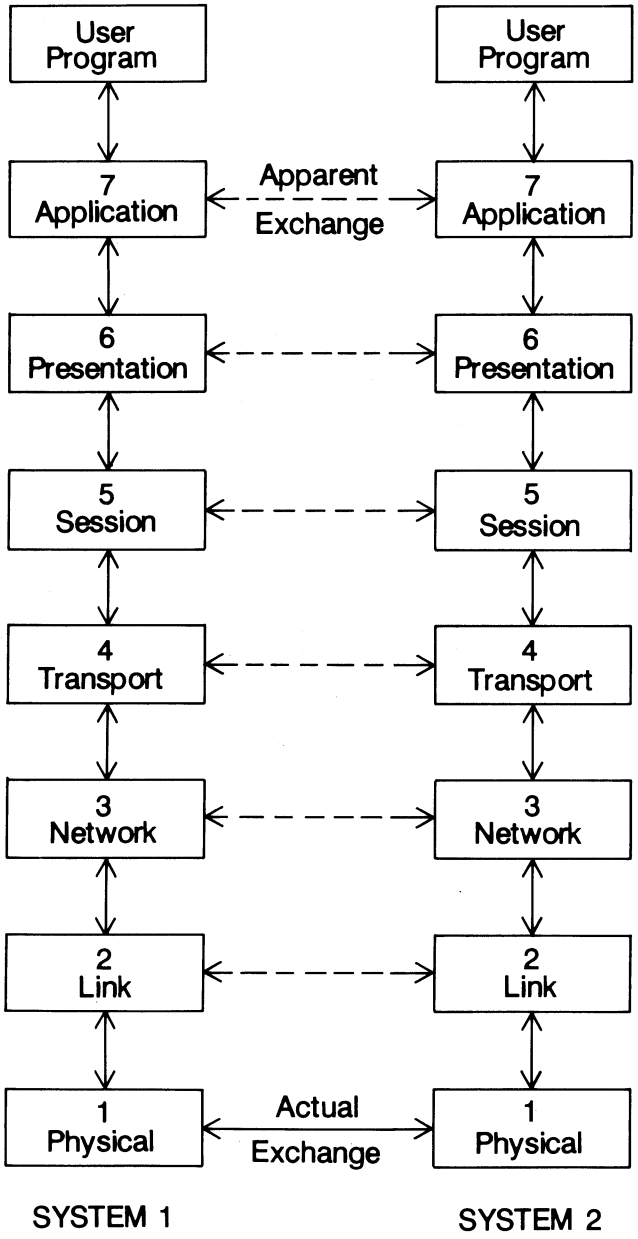


Figure 3 The layered architecture of the OSI reference model.

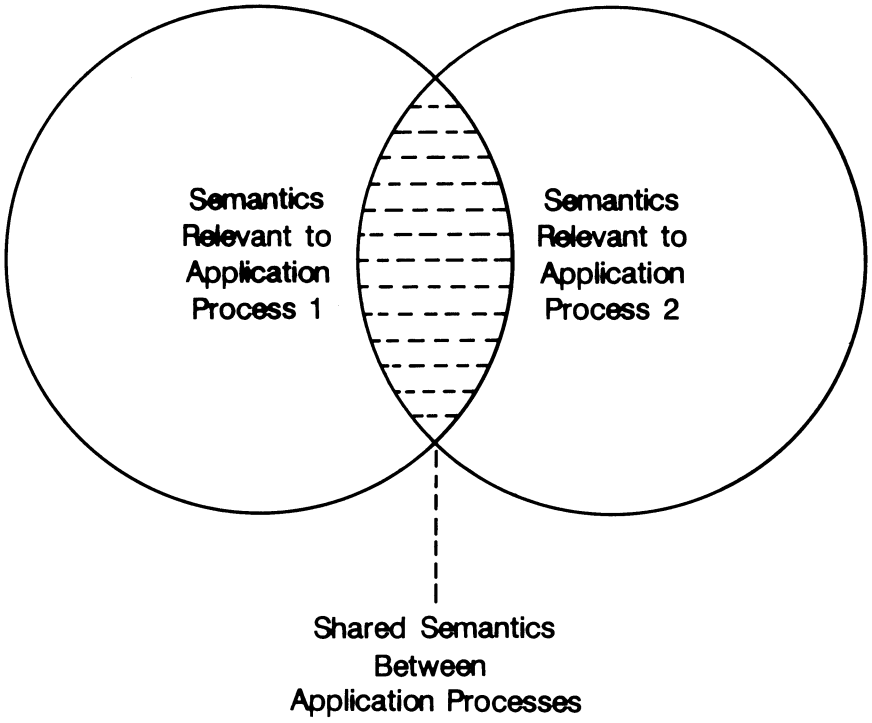


Figure 4 Shared semantics between application processes are necessary for communication.

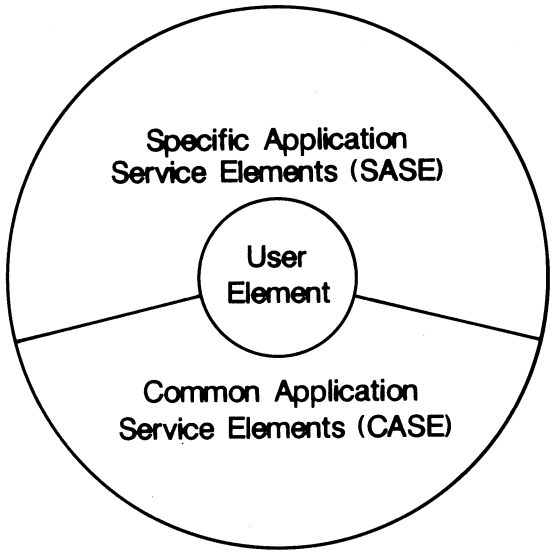
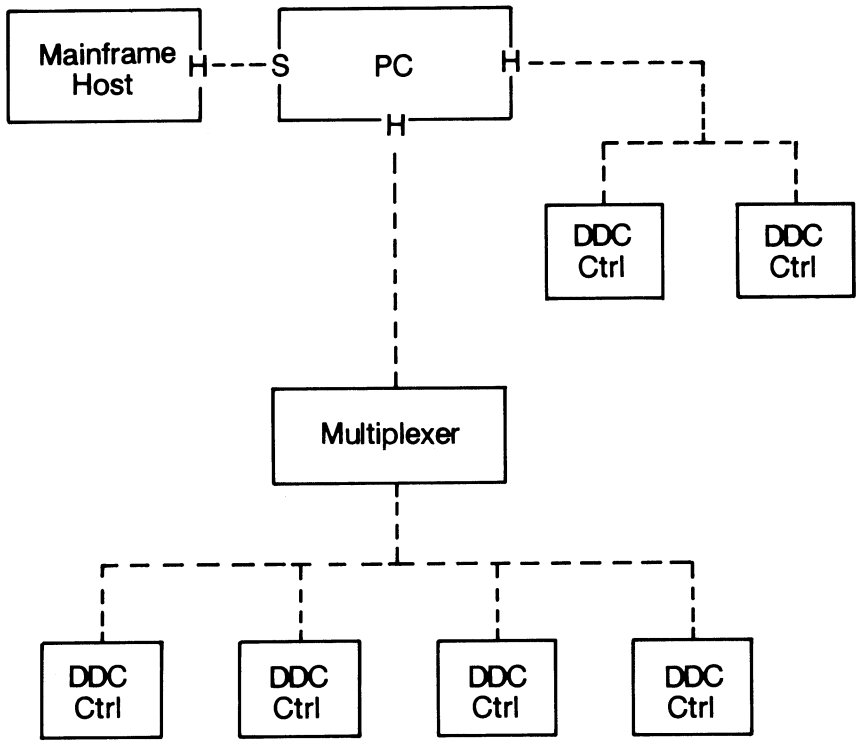


Figure 5 Structure of the ISO application entity



Legend:

H – Host

S – Tributary/Slave

– – – Communications Network

Figure 6 System architecture for protocol A.

DISCUSSION

D. Schwenk, Principal Investigator, U.S. Army Corp of Engineers, Champaign, IL: What is the impact of your work on ASHRAE's development of a building energy management standard protocol and how long do you think it will be until a protocol is available?

S. Bushby: The ASHRAE standards project committee (SPC) to which you referred has several active working groups. One of these groups is called the Application Services Working Group. The ideas presented in this paper are directly related to the activities of this working group and have been presented to them. The working group has already taken action to adopt a set of "functions" that need to be provided by the application layer of the protocol that correspond almost exactly to what I have presented here.

It is impossible to predict when a draft standard will be completed. I can say that all members of the SPC are active and contributing to the effort. Good progress was made in the first year but much work remains to be done.

P.R. Armstrong, Partner, Colorado Energy Associates, Ft. Collins, CO: "Command Primitives" were mentioned in reference to the "state" of various communications services. The "state machine" formalism was also mentioned in this regard. Could you please elaborate?

Bushby: The question refers to remarks I made about the next step in the process of developing an application layer protocol, that is, a more formal description of what the application services are and precisely how they work. A finite state machine is a commonly used abstraction that describes how a process works, in this case a protocol service implementation. A series of logical "states" are defined that describe the possible relationships between the communicating entities. Command primitives cause the transition from one state to another. Using these abstractions, a map can be made that describes the dynamic interactions that can take place during a communication. This type of analysis is useful both for describing the dynamic interactions involved and also for checking the service specification for internal consistency and completeness.