

Specification for

ControlNet —

Part 6: Object model

ICS 25.040.40; 35.100.70

NO COPYING WITHOUT BSI PERMISSION EXCEPT AS PERMITTED BY COPYRIGHT LAW

BSI
Documents

AVA

Document Sharing Hub

Committees responsible for this Draft for Development

The preparation of this Draft for Development was entrusted to Technical Committee AMT/7, Monitoring and control aspects of AMT, upon which the following bodies were represented:

- ABB Kent plc
- Cegelec Industrial Controls Ltd.
- Department of Trade and Industry
- Fisher-Rosemount Ltd.
- Foxboro Great Britain
- Institution of Electrical Engineers
- Measurement Technology Ltd.
- Phoenix Contact Ltd.
- Process Automation and Computer Systems Ltd.
- Rockwell Automation Ltd.
- Schneider Ltd.
- Siemens Automation Ltd.
- University of Warwick
- Wago Ltd.

This Draft for Development, having been prepared under the direction of the Advanced Manufacturing Technology Assembly, was published under the authority of the Standards Board and comes into effect on 15 November 1997

© BSI 05-1999

The following BSI reference relates to the work on this publication:
Committee reference AMT/7

ISBN 0 580 28675 4

Amendments issued since publication

Amd. No.	Date	Comments

Contents

	Page
Committees responsible	Inside front cover
Foreword	ii
0 Introduction	1
1 Scope	5
2 Object format and specification rules	5
2.1 Object specification format	5
2.2 Object Name and Class Code	6
2.3 Object Scope	6
2.4 Object Revision History	7
2.5 Attributes	7
2.6 Common services	10
2.7 Get_Attributes_All response	11
2.8 Set_Attributes_All request	13
2.9 Object-specific services	14
2.10 Behaviour	16
2.11 Accessing application object data	18
2.12 Revision situations	18
3 Class, Attribute and Service Code ranges	18
3.1 Defined ranges	18
3.2 Class code ID ranges	19
3.3 Attribute ID ranges	19
3.4 Service Code ranges	19
4 Common services	19
4.1 TNL_message service primitives	19
4.2 Extended status format definitions	20
4.3 Common services definitions	23
4.4 Member Service Protocols	41
5 General status codes	43
6 Vendor-specific extensions to existing objects	46
6.1 Vendor specific attributes	46
6.2 Defining a new Vendor-specific object (informative)	47
Figure 6-1 — A class of objects within a device	2
Figure 6-2 — Addressing format using MAC, Class, Instance and Attribute IDs	3
Figure 6-3 — Object model	3
Figure 6-4 — Relationship to the OSI/ISO model	5
Figure 6-5 — Instance Attribute Table format and terms	10
Figure 6-6 — State Transition Diagram (STD)	17
Figure 6-7 — Request service data field parameters	35
Table 6-1 — Class, object, instance, attribute example	1
Table 6-2 — Object specification terms	4
Table 6-3 — Object specification format	6
Table 6-4 — Object Revision History Table Format	7
Table 6-5 — Object Revision History Table example — Informative	7
Table 6-6 — Class Attribute table format and terms	8
Table 6-7 — Reserved Class Attributes for all object class definitions	10
Table 6-8 — Common Services Table format and terms	11
Table 6-9 — Get_Attributes_All response service rules	12

	Page
Table 6-10 — Class level object/service specific reply data of Get_Attributes_All — Informative	12
Table 6-11 — Get_Attributes_All data array method — Informative	13
Table 6-12 — Set_Attributes_All request service rules	13
Table 6-13 — Set_Attributes_All attribute ordering method — Informative	13
Table 6-14 — Set_Attributes_All data array method — Informative	14
Table 6-15 — Object specific: services	14
Table 6-16 — Object specific service request parameter	15
Table 6-17 — Object specific service response data	16
Table 6-18 — State Event Matrix — Informative	17
Table 6-19 — Attribute Access Rules Table	18
Table 6-20 — Addressing categories	18
Table 6-21 — Class code ID ranges	19
Table 6-22 — Attribute ID ranges	19
Table 6-23 — Service Code ranges	19
Table 6-24 — Common Services list	22
Table 6-25 — Service data for Get_Attributes_All success response	23
Table 6-26 — Service response general status codes for Get_Attributes_All response	23
Table 6-27 — Service data for Set_Attributes_All request	24
Table 6-28 — Service response general status codes for Set_Attributes_All response	24
Table 6-29 — Service data for Get_Attributes_List request	25
Table 6-30 — Service data for Get_Attributes_List response	25
Table 6-31 — Structure components of the data_of_attributes	25
Table 6-32 — Service response general status codes for Get_Attributes_List response	25
Table 6-33 — Service Data for Set_Attributes_List request	26
Table 6-34 — Structure component of the data_of_attributes	26
Table 6-35 — Service data for Set_Attributes_List request	26
Table 6-36 — Structure components of the data_of_attributes	26
Table 6-37 — Service response general status codes for Set_Attributes_List response	27
Table 6-38 — Service Data for Reset request	27
Table 6-39 — Service data for Reset success response	27
Table 6-40 — Service response general status codes for Reset response	28
Table 6-41 — Service Data for Start request	28
Table 6-42 — Service Data for Start Success response	29
Table 6-43 — Service response general status codes for Start response	29
Table 6-44 — Service data for Stop request	29
Table 6-45 — Service Data for Stop Success response	30
Table 6-46 — Service response general status codes for Stop response	30
Table 6-47 — Service data for Create request	30
Table 6-48 — Service data for Create success response	31
Table 6-49 — Service response general status codes for Create response	31
Table 6-50 — Service Data for Delete request	31
Table 6-51 — Service Data for Delete success response	32
Table 6-52 — Service response general status codes for Delete response	32

	Page
Table 6-53 — Service Data for Apply_Attributes request	32
Table 6-54 — Service Data for Apply_Attributes success response	33
Table 6-55 — Service response general status codes for Apply_Attributes response	33
Table 6-56 — Service Data for Get_Attribute_Single success response	33
Table 6-57 — Service response general status codes for Get_Attribute_Single response	34
Table 6-58 — Service Data for Set_Attribute_Single request	34
Table 6-59 — Service Data for Set_Attribute_Single success response	34
Table 6-60 — Service response general status codes for Set_Attribute_Single response	34
Table 6-61 — Service Data for Find_Next_Object_Instance request	36
Table 6-62 — Service Data for Find_Next_Object_Instance Success response	36
Table 6-63 — Service response general status codes for Find_Next_Object_Instance response	36
Table 6-64 — Service Data for Restore request	37
Table 6-65 — Service Data for Restore Success response	37
Table 6-66 — Service response general status codes for Restore response	37
Table 6-67 — Service Data for Save request	38
Table 6-68 — Service Data for Save Success response	38
Table 6-69 — Service response general status codes for Save response	38
Table 6-70 — Service response general status codes for Save response	39
Table 6-71 — Service data for Get_Member request	39
Table 6-72 — Service data for Get_Member success response	39
Table 6-73 — Service data for Set_Member request	40
Table 6-74 — Service data for Set_Member success response	40
Table 6-75 — Service data for Insert_Member request	40
Table 6-76 — Service data for Insert_Member success response	41
Table 6-77 — Service data for Remove_Member request	41
Table 6-78 — Service data for Remove_Member success response	41
Table 6-79 — Member ID/EX description	41
Table 6-80 — Service data for Basic format Member request messages	42
Table 6-81 — Service data for Basic format Member response messages	42
Table 6-82 — Service data for Extended format Member request messages	42
Table 6-83 — Service data for Extended format Member response messages	43
Table 6-84 — Extended Protocol ID	43
Table 6-85 — Service data for Multiple Sequential Member request messages	43
Table 6-86 — Service data for Member response messages (Extended Protocol)	43
Table 6-87 — General status codes	44
Table 6-88 — Example vendor specific instance attributes — Informative	46
Table 6-89 — Example vendor specific services — Informative	47
Table 6-90 — Vendor specific advantages/disadvantages	47

Foreword

The DD 241 specification was developed by the Allen-Bradley Company, Inc. as “ControlNet”. Allen-Bradley has granted BSI a right to publish the specification as an open communications standard for digital communications networks in industrial control applications.

This part of DD 241 describes the General Description of the deterministic control network.

This publication consists of the following parts:

- *Part 1: General description;*
- *Part 2: Physical layer and media;*
- *Part 3: Data link layer;*
- *Part 4: Network and transport layer;*
- *Part 5: Data management;*
- *Part 6: Object model;*
- *Part 7: Device profiles;*
- *Part 8: Electronic data sheet;*
- *Part 9: Station management;*
- *Part 10: Object library.*

This Draft for Development is not to be regarded as a British Standard.

It is being issued in the Draft for Development series of publications and is of a provisional nature because the subject matter is still under development requiring wider exposure. It should be applied on this provisional basis so that information and experience of its practical application may be obtained.

A review of this Draft for Development will be carried out not later than 2 years after its Publication. Notification of the start of the review period, with a request for the submission of comments from users of this Draft for Development, will be made in an announcement in *Update Standards*.

According to the replies received the responsible BSI Committee will judge whether the Draft for Development can be converted into a British Standard or what other action should be taken.

Observations which it is felt should receive attention before the official call for comments will be welcomed.

These should be sent to the Secretary of BSI Technical Committee AMT/7 at BSI, 389 Chiswick High Road, London, W4 4AL, giving the document reference and clause number and proposing, where possible, an appropriate revision of the text.

Disclaimer of warranties

This document is provided on an “as is” basis without warranty by BSI and Allen-Bradley as to the accuracy or completeness of the information it contains or as to the results which may be achieved through the use of this information. Further, BSI and Allen-Bradley are not providing any warranties as to whether products produced based on DD 241 may infringe the intellectual property rights of any third parties except that, at the date of publication, to the best of BSI’s and Allen-Bradley’s knowledge, no written claims have been made against BSI and/or Allen-Bradley alleging that any such products would infringe the intellectual property rights of any third parties. In no event shall BSI and/or Allen-Bradley be responsible for any losses, damages, costs or expenses incurred as a result of the information contained in DD 241.

Summary of pages

This document comprises a front cover, an inside front cover, pages i to vi, pages 1 to 47 and a back cover.

This standard has been updated (see copyright date) and may have had amendments incorporated. This will be indicated in the amendment table on the inside front cover.

0 Introduction

In this standard, object modelling represents the network visible behaviour of a device. This part provides information on

- object modelling and terminology;
- object addressing;
- the object model;
- object specification format and rules;
- how to extend object specifications;
- and how to create new object specifications.

A device is modelled as a collection of objects. Object modelling organises related data and procedures into one entity: the object. An object is a collection of related services; and attributes. Services are procedures an object performs. Attributes are characteristics of objects represented by values or variables. Typically, attributes provide status information or govern the operation of an object. The value associated with an attribute is able to affect the behaviour of an object. An object's behaviour is an indication of how the object responds to particular events.

A class is a group of objects all of which represent a particular system component. A class defines a particular type of object and defines the characteristics shared by all the objects within a class. For example, the human race could be represented by the class "Human" with millions of objects within it. All objects in a class are identical in form and behaviour, but usually contain different attribute values.

Objects within a class are called object instances. An object instance is the actual representation of a particular object within a class. Each instance of a class has the same set of attributes, but has its own set of attribute values, which makes each instance in the class unique. As Figure 6-1 illustrates, multiple object instances within a particular class can reside in a device. The terms object, instance, and object instance all refer to a specific instance.

An object instance and/or an object class has attributes, provides services and implements a behaviour. As stated above, attributes are characteristics of an object and/or an object class which provides a description of externally visible characteristics or features of an object. Services are invoked to trigger the object/class to perform a task. Services provide a function supported by an object and/or object class. A set of common services are defined as well as provisions for the definition of object-specific services. Object-specific services are those which are defined by a particular object class to perform a required function not covered by a common service.

The behaviour of an object indicates how it responds to particular events. Actions result from different events of which the object is aware, such as receiving service requests, detecting internal faults, or elapsing timers. Each class specification contains a definition of how the object within the class acts.

Figure 6-1 shows an example of the object concept. Every person in the human race can be represented by an instance within the class "Human." All humans have the same set of attributes: eyes, ears, age, gender, etc. Yet, because the values of each attribute vary, everyone looks different and performs in distinct ways.

Table 6-1 — Class, object, instance, attribute example

Class	Instances	Attributes	Attribute values
Human	Mary	Sex	Female
		Age	31
	Joe	Sex	Male
		Age	50

Abstract object modelling is used to describe

- the suite of communication services available;
- the externally visible behaviour of a device;
- a common means by which information within devices is accessed and exchanged.

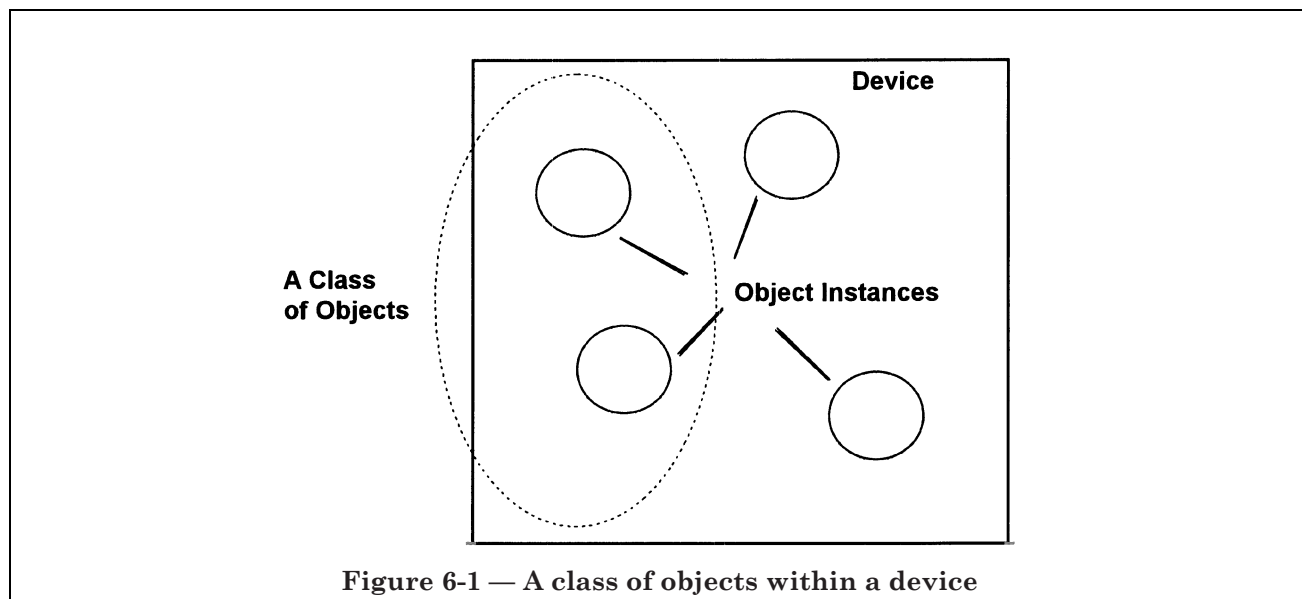


Figure 6-1 — A class of objects within a device

A device is modelled as a collection of objects. An object provides an abstract representation of a particular component within a device. The realisation of this abstract object model within a device is implementation dependent. In other words, a device internally maps this object model in a fashion specific to its implementation.

The information presented here provides a common basis for logically addressing separate physical components across the network. These addressing terms are used in the Object Library. All references to Object Library in this part refer to Part 10, Object Library, of this standard. Figure 6-2 should be referenced in the following discussion.

The term “node” is used to refer to that portion of a device which contains the link interface. The term device refers to the whole device. A device is able to contain multiple nodes.

The Class ID is a unique integer identification value assigned to each object class accessible from the network. The object class may be referenced with this Class ID. In this standard, class code is synonymous with Class ID.

The Instance ID is an integer identification value assigned to an *object instance* when it is created that identifies it among all *Instances* of the same *Class*). This integer is unique within the *Node : Class* in which it resides.

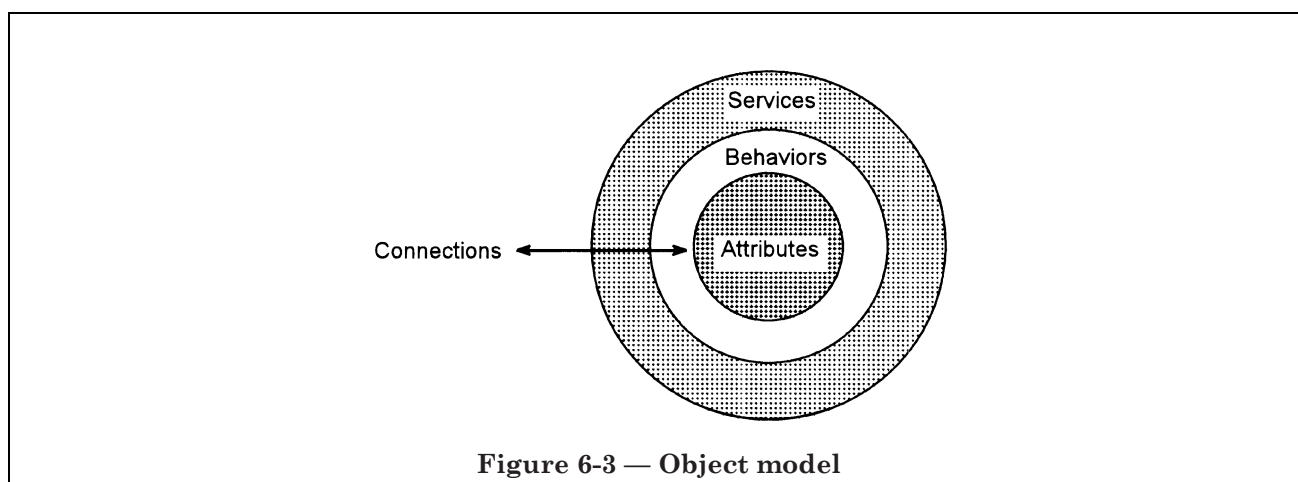
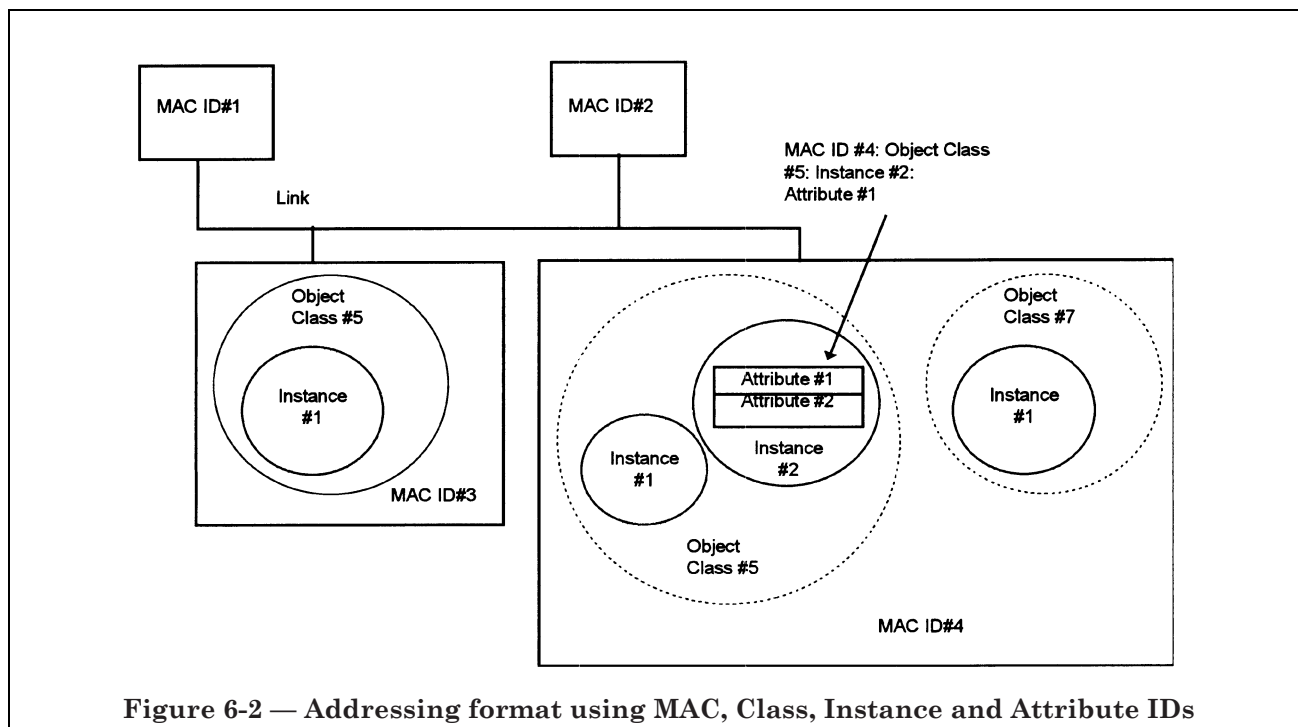
The Attribute ID is an integer identification value that is unique among all other attributes of that object. A detailed definition of an object’s attributes is provided with that object’s definition (object specification).

The address of attribute number 1 of instance 2 in class 5 in the device with a MAC ID of 4 is *MAC ID#4: Object Class #5: Instance #2: Attribute #1* as shown in Figure 6-2. This nomenclature is referred to as

Class/Instance/Attribute addressing.

Each object specification contained in the Object Library is defined based on the contents of an object. An object consists of the following, as illustrated in Figure 6-3:

- a set of closely related **attributes** (data);
- defined **behaviours**;
- **services** (common or object-specific);
- supported **connections**.



The object specifications in the Object Library are defined using the terms shown in Table 6-2.

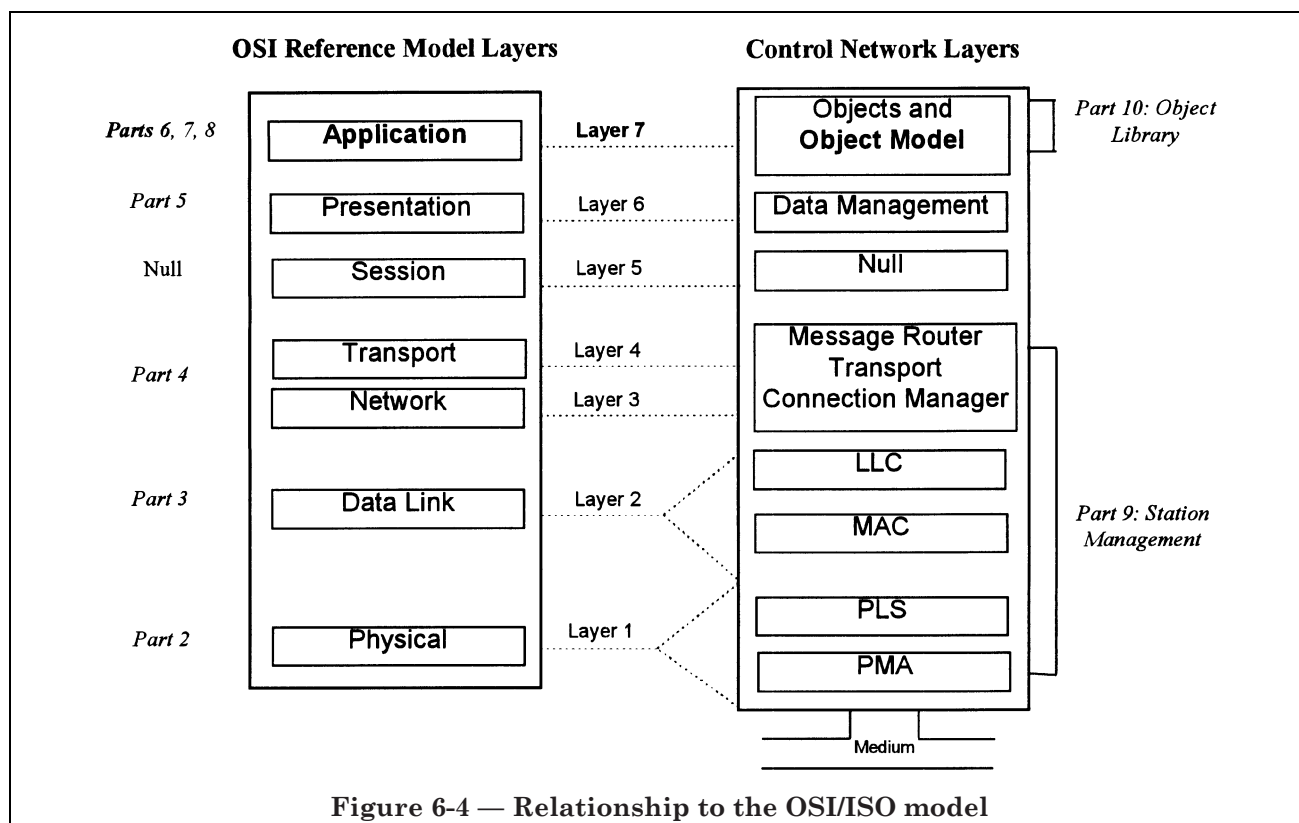
Table 6-2 — Object specification terms

Term	Description
Object	An abstract representation of a particular component within a device.
Class	A set of objects, all of which represent the same kind of system component. A class is a generalisation of the object; a template for defining variables and methods. All objects in a class are identical in form and behaviour, but usually contain different data in their attributes.
Class Code	A unique identifier assigned to each object class.
Scope	The Scope in the object specification contains a brief functional description of the object class.
Instance	A specific and real (physical) occurrence of an object. For example: California is an instance of the object class state. The terms object, instance, and object instance refer to a specific Instance.
Attributes	A description of an externally visible characteristic or feature of an object. The attributes of an object contain information about variable portions of an object. Typically, they provide status information or govern the operation of an object. Attributes may or may not affect the behaviour of an object. Attributes are divided into Class Attributes and Instance Attributes
Class Attributes	An attribute that is shared by all objects within the same class.
Instance Attributes	An attribute that is unique to an object instance and not shared by the object class.
Service	A function supported by an object and/or object class. A set of common services is defined and provisions for the definition of object-specific services are provided. Object-specific services are those which are defined by a particular object class to perform a function not covered by a common service.
Common Services	A list of the common services defined for this object class.
Object-specific Services	The full specifications of any services unique to this object class.
Behaviour	Indication of how the object responds to particular events. Its description includes the relationship between attribute values and services.

1 Scope

This part of the standard defines the requirements for the Object Model of a device on the deterministic control network. The Common Services and General Status Codes are also defined.

The Object Model of the deterministic control network corresponds to the definition of Layer 7 of the seven layer OSI model in accordance with ISO 7498. Figure 6-4 shows the location of the Object Model within the OSI model.



2 Object format and specification rules

2.1 Object specification format

The object specifications in the Object Library shall be defined using the format specified in Table 6-3. The order in which the sections appear in Table 6-3 are the same order in which they shall appear in each object specification of the Object Library.

Table 6-3 — Object specification format

Required?	Section name	Section information
Yes	Object name and class code	See clause 2.2 for details.
Yes	Object Scope	See clause 2.3 for details.
Conditional	Object Revision History Table	See clause 2.6 for details. Not required for revision 1 of an object. Required for revisions other than revision 1.
Yes	Attributes	Attributes are divided into Class Attributes and Instance Attributes. See clause 2.5 for details.
Yes	— Class Attribute Table;	See clause 2.5.2 for details. Class Attribute ID 1 shall be required to be listed in all object specifications.
Yes	— Instance Attribute Table.	See clause 2.5.3 for details.
Yes Conditional	Common Services — Common services listed in the Common Services Table	See clause 2.6 for details. Common service specific request parameter(s) and responses defined/described.
Conditional Conditional	— Get_Attributes_All response — Set_Attributes_All request	See clause 2.7 for details. See clause 2.8 for details.
Yes Conditional	Object Specific Services — Object specific services listed in the Object Specific Services Table.	See clause 2.9 for details. Object specific service(s) and responses listed, defined and described. — Object specific service specific request parameter(s) and defined/described. — Object specific service specific response(s) defined/described
Yes Conditional Conditional	Object Behaviour Description — State Transition Diagram (STD); — State Event Matrix (SEM).	See clause 2.10 for details. See clause 2.10.2 for details. See clause 2.10.3 for details.
Conditional	Attribute Access Rules Table	See clause 2.10.5 for details. If attribute access rules vary with the state of an object then an attribute access table shall be required.

2.2 Object Name and Class Code

Each object in the Object Library shall have a unique name and a unique class code. The class code shall be a hexadecimal value unique to an object. The valid ranges for class codes shall be as specified in clause 3. The class code shall be used to identify the object class when accessing an object in a device. This standard defines the allocation of the class codes. The format of the Object name and class code shall be of the form:

Object name (class code = 0xCC)

where CC shall represent this object's class code in hexadecimal format, e.g. the Object name and class code for the Assembly Object is: Assembly Object (class code = 0x04). Valid class code ranges shall be as specified in Table 6-21.

The object name and class code shall be the first line in the object specification.

2.3 Object Scope

Every object specification shall begin with a brief functional description of the object being defined. This shall be placed immediately following the object name and class code in the object specification.

2.4 Object Revision History

2.4.1 Location and when required

If an object specification has been changed, these changes shall be tracked using an **Object Revision History Table**. Table 6-4 specifies the format of the Object Revision History Table. The Object Revision Table shall immediately follow the object scope in the object specification.

Table 6-4 — Object Revision History Table Format

Class Revision	Description of Changes
See clause 2.4.2	See clause 2.4.3

2.4.2 Object specification revision

The object specification class revision is a decimal number which shall start at one and shall be incremented by one with every revision to the object specification. Class attribute 1, Revision, shall be the object specification revision.

2.4.3 Description of object specification changes

This part of the Object Revision History Table shall specify and describe the reasons for the object specification changes.

Object specifications with a revision of 1 need not have the object revision table.

NOTE Table 6-5 is an example of an object revision table.

Table 6-5 — Object Revision History Table example — Informative

Class revision	Description of changes
01	Initial Release
02	1) Class specific service codes 0×4B (Add_Member) and 0×4C (Delete_Member) obsoleted. 2) Added support of common services 0×18,19,1A and 1B. 3) List of all major changes continues.

2.5 Attributes

2.5.1 General

The Attribute part of an object specification shall be divided into two clauses

- Class attributes, followed by;
- Instance attributes.

The attributes shall be placed immediately following the Object Revision History Table if present, or if the Object Revision History Table is not present, then immediately after the object scope.

2.5.2 Class attribute specification

2.5.2.1 Format

An attribute whose value is shared by all objects within the same class shall be referred to as a **Class Attribute**. Class Attributes shall be defined in a Class Attribute Table using the format and terms defined in Table 6-6.

Every object specification shall list Class Attribute 1, Revision, causing the Class Attribute Table to be in every object specification.

Table 6-6 — Class Attribute table format and terms

Attribute ID	Need in implementation	Access rule	Name	Data type	Description of attribute	Semantics of values
Note 1	Note 2	Note 3	Note 4	Note 5	Note 6	Note 7
NOTE 1 See clause 2.5.2.2 NOTE 2 See clause 2.5.2.3 NOTE 3 See clause 2.5.2.4 NOTE 4 See clause 2.5.2.5 NOTE 5 See clause 2.5.2.6 NOTE 6 See clause 2.5.2.7 NOTE 7 See clause 2.5.2.8						

2.5.2.2 Attribute ID

The **Attribute ID** shall be a unique integer identification value assigned to an attribute. The decimal number shall be used in the object specification. The valid ranges for Attribute IDs shall be as specified in clause 3. The Attribute ID shall identify the particular attribute being accessed. The term *Get_Attributes* shall refer to all such services, such as the *Get_Attribute_Single* and *Get_Attributes_All* services. The term *Set_Attributes* shall refer to all such services, such as the *Set_Attribute_Single* and *Set_Attributes_All* services.

2.5.2.3 Need in implementation

Need in implementation shall specify whether or not the attribute is necessary in the object class implementation. An attribute shall be defined as *optional*, *required*, or *conditional*. Conditional shall mean that the attribute's *need in implementation* changes depending on circumstances. When an attribute is defined as conditional, the attribute shall either be optional, required or not applicable (n/a). The conditions upon which the attribute assumes these different *need in implementation* shall be specified.

If a Class Attribute is *optional*, then a default value or a special case processing method shall be defined such that the Client (Requester) can process the error message that occurs when accessing those objects that choose not to implement the class attribute.

2.5.2.4 Access rule

The **Access rule** shall specify how a requester accesses an attribute. The definitions for access rules shall be

- Settable (Set) — The attribute may be accessed by one of the *Set_Attribute* services (if the behaviour of the device does not require a *Set_Attribute* service, then a device implementation of that object is not required to implement the attribute as settable);
- Settable attributes are also Gettable and may be accessed by *Get_Attribute* services;
- Gettable (Get) — The attribute may be accessed by one of the *Get_Attribute* services, but shall not be modified by any of the *Set_Attribute* services.

If the access rule of any attributes are conditioned on the state of the object, then the object specification shall include an **Attribute Access Rules Table**. The Attribute Access Rules Table shall list the access rules for all object states for each attribute which has varying access rules.

NOTE See clause 2.10.5 for the format of the attribute access rules table.

2.5.2.5 Name

Name shall refer to the name assigned to the attribute.

An attribute may contain sub-elements which may also have names, as is the case with the **STRUCT** of data type. The attribute name shall be the name which appears first, or at the top row in the name column (the row that contains the Attribute ID).

2.5.2.6 Data type

Every attribute shall be assigned a **Data type** which shall be either an elementary or a derived data type. The data types specified for the defined attributes shall be used in all implementations.

NOTE The elementary data types are defined in part 5 of this standard.

2.5.2.7 *Description of attribute*

Description of attribute shall provide general information about the attribute. If the description needs more room than can fit in the attribute table, the description shall immediately follow the Class Attribute table. Included in the Class Attribute table shall be an appropriate reference to this additional information.

2.5.2.8 *Semantics of values*

Semantics of values shall specify the meaning of the value(s) of the attribute. If this information needs more room than can fit in the table it shall immediately follow the Class Attribute table. Included in the Class Attribute table shall be an appropriate reference to this information.

2.5.2.9 *Predefined class attributes*

Seven predefined Class Attribute IDs shall be reserved for class object definition. The seven predefined/reserved class attributes shall have the definitions listed in Table 6-7. Because these attributes are reserved, class Attribute ID numbers 1 through 7 shall always be reserved. Therefore, if a class attribute is added to an object specification, it shall start with Attribute ID #8.

Each object specification shall specify which of the seven predefined class attributes are supported by that object class. Class attribute 1, Revision, shall be required in every object specification.

2.5.3 *Instance attribute specification*

An **Instance attribute** shall be unique to an object instance and not shared by the object class. Instance Attributes in the Object Library shall be defined in the Instance Attribute Table using the same format and terms as Class Attributes (see Table 6-6, but substitute the word “instance” for the word “class”). Instance ID = 0 shall be a special case. A service directed to Instance ID = 0 shall be applied to the class, not to a particular instance. The header for the Instance Attribute Table shall be as shown in Figure 6-5.

Each instance shall have a unique set of attributes. Instance attribute ID's may be numbered from 1 onwards without any correlation to the Attribute ID's of the class of which the object is an instance. There shall be no reserved Instance Attributes.