EIBA Handbook Series

Release 3.0

Volume 3:
System Specifications

Part 4: Application Environment

Chapter 1: User Layer

23.03.1999

Table of Contents

1. Overview

The user layer is the layer between the application layer and the application. It contains the EIB communication relevant tasks of the application. It eases the communication task of the application by offering a communication interface that abstracts from many application layer details.

EIB allows single-processor and dual-processor EIB device designs.

- The single-processor device design is based on a BAU that runs the EIB communication stack and the internal user application. Between the seven-layered EIB communication stack and the internal user application there is the internal user layer.
- The dual-processor device design is based on a BAU that communicates via a serial PEI protocol with the external user application run at the second processor. The external user layer shall also be run at the second processor. The message protocol is either the default External Message Interface (EMI), as specified in Chapter 3/6/3 "External Message Interface" or a manufacturer specific format.

The following clauses define the client and server functionality and the communication interface of the internal user application loacted in the BAU. No definitions are made in respect to the external user layer that is located between the External Message Interface and the external user application.

The internal user layer contains the following objects and the access routines to them:

- the group-communication-objects, which can be accessed via application service access points on multicast communication relationships, see the corresponding clause in Chapter 3/3/7 "Application Layer".
- the EIB objects, which can be accessed via application services on one-to-one connection-less and one-to-one connection-oriented communication relationships. The EIB objects are either system objects or application objects.

System objects are the device object, the address table object, the association table object and the application object. System objects are relevant for network management, (see Part 3/5 "Network Management").

Application objects are EIB objects defined in the user application. They may be defined by the internal or external user application, based on EIB object structure rules defined in this document.

Note: The physical address object, which can be accessed via broadcast communication relationships, is a data link layer object. Therefore this object and its access routines are not a matter of the user layer.

The following clauses explain the data structures of each of the user layer objects. Additionally they define by which application services those objects are accessible. Both the object client and object server functionality may be implemented by the external or the internal user layer. It is recommended to locate the group communication objects, the system objects, the mask version object and the manufacturing information object in the internal user layer; the application objects may be situated in either of the two user layers.

Note: In case of the default EMI to the external user application then, if any of those objects is situated in the external user application, the internal and external user application together must realize the corresponding object server functionality. The internal user application's task is then to redirect the layer-7 service indications to the external user application.

1.1 Communicating Information via the EIB Media - an Overview

The EIB transport layer provides the possibility to establish communication relationships with remote EIB devices. These communication relationships may be broadcast, multicast, one-to-one connection-oriented and one-to-one connection-less ones.

The broadcast communication relationship is needed only during the EIB device configuration phase. Its purpose is to supply the remote EIB device with a unique physical address - either with the help of the programming button or with a unique serial number already existing in the remote EIB device. See Chapter 3/3/4 "Transport Layer" for the explanation of this communication relationship, Chapter 3/3/7 "Application Layer" for the explanation of the services allowed on this communication relationship and Part 3/5 "Network Management" for an explanation of the physical address configuration procedure of EIB network management.

The multicast communication relationships are usable during the operational phase of the EIB network by the user application. They allow to broadcast group-object data of up to 14 bytes to those devices which belong to the same group. In an EIB network the corresponding A_GroupValue_Read and A_GroupValue_Write services are the standard services during the operational phase. See Chapter 3/3/4 "Transport Layer" for the explanation of this communication relationship and Chapter 3/3/7 "Application Layer" for the explanation of the services allowed on this communication relationship. Clause 2 "General Structure of Group-Communication-Objects" of this document explains the data structure of group-objects and the sophisticated user layer interface between the group-objects and the internal user application. Chapter 3/6/3 "External Message Interface" explains the message interface between the group-objects and the external user application.

The one-to-one connection-oriented communication relationship in an EIB network serves basically for network management during the EIB device configuration and commissioning phase and for maintenance during the operational phase. It allows downloading the address table, the association table and the internal user application. The internal user application comprises the user application program and the group-object table. Address table, association table and group-object table are the commissioning information needed to establish multicast communication relationships. See Chapter 3/3/4 "Transport Layer" for the explanation of the one-to-one communication relationship and Chapter 3/3/7 "Application Layer" for the explanation of the services allowed on this communication relationship. Clause 5 "Object Servers and their Object Structures"of this document explains the data structure of the accessible objects and the corresponding object servers.

The one-to-one connection-less communication relationships serve to transfer property descriptions and property values of EIB objects during the operational phase. Property values may be of a size of up to 2 kb because the A_PropertyValue_Read and A_PropertyValue_Write services allow segmented data transfer. Therefore transfer of files and lists is possible. See Chapter 3/3/4 "Transport Layer" for the explanation of this communication relationship and Chapter 3/3/7 "Application Layer" for the explanation of the services allowed on this communication relationship. Clause 3 "General Structure of EIB Objects"of this document explains the data structure of EIB objects; clause 5.8 "Property Server" of this document explains the object property server.

1.2 User Layer's Network Management Server Functionality

Seen from a client-server architecture point of view there exists at least one management client application and a management server application. While the management server application shall be implemented by the BAU firmware, the management client applications are located beyond the PEI connection terminal, i.e. they are external user applications.

Consequence: The management client as an external user application shall communicate with the local BAU via transport layer messages. The external user application shall include completely encoded APDU's (i.e. application layer PDU´s) in the transport layer messages as user data. The management server shall be part of the BAU and therefore shall autonomously „serve“ the client requests without the necessity of any help by the internal user application. This is automatically true, because the management server functionality is based on the object servers to be implemented in the internal user layer.

2. General Structure of Group-Communication-Objects

2.1 Overview

Group-communication-objects may be distributed to a number of EIB end devices. Each EIB end device may be transmitter for group-communication-object values. More than one group-object may exist in an EIB end device and a group-communication object in an EIB end device may be assigned to one or more group addresses. Group-objects of an EIB end device may belong to the same or to different groups. Each group has a network wide unique group address. The group address is mapped to a local cr_id that is unique for the communication relationships of the EIB end device in layer-4. The application layer maps the cr_id to service access points that access the group-communication-objects.

Fig. 3/4/1-1 shows an example with several multicast communication relationships:

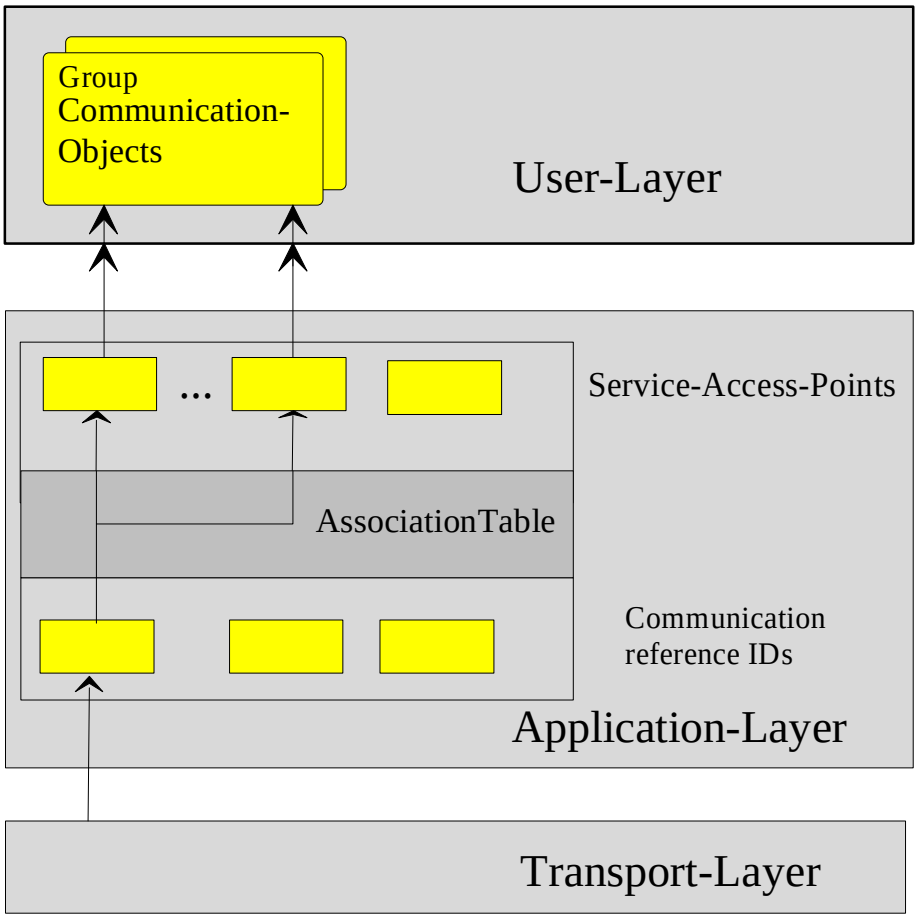

Fig. 3/4/1-1: Multicast Communication Relationship

2.2 General Data Structure of the Group-Communication Object Table

In the sense of the previous clause a group-communication-object consists of a three parts. The group-communication-object description, the object-value and the communication-flags ("RamFlags").

The group communication object description must at least include the object-type and the transmission priority.

The communication flags show the state of a group communication object. Following states are possible: update, read_request, write_request, transmitting, ok, error.

2.3 Triggering Group-Object Value Transfers and Receiving A_GroupValue_Read.ind and A_GroupValue_Write

It is the responsibility of the internal or external user application program to trigger group-object value transmissions.

2.3.1 Reading a Group-Communication-Object Value

The following figure sketches in vertical direction the flow of action to read a group-communication-object value:

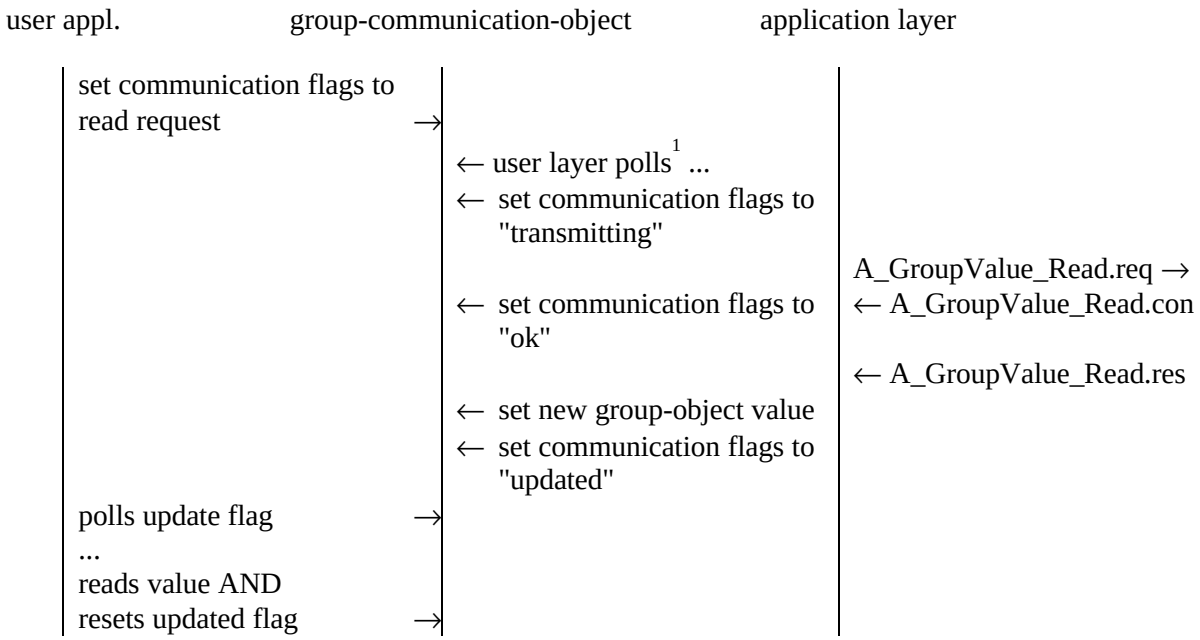

Fig. 3/4/1-2: Reading a Group-Communication-Object Value

[1] "Polls" here means that the user layer reads the Group Communication Objects flags. It shall not be confused with the Layer-2 polling service primitive.

If a negative A_GroupValue_Read.con returns, then the transmission status shall be set to "error". Furthermore the updated flag shall not be set to "updated" and no new group-object value shall be set.

2.3.2 Receiving an A_GroupValue_Read.ind

The following figure sketches in vertical direction the flow of action of the user layer during reception of an A_GroupValue_Read.ind:

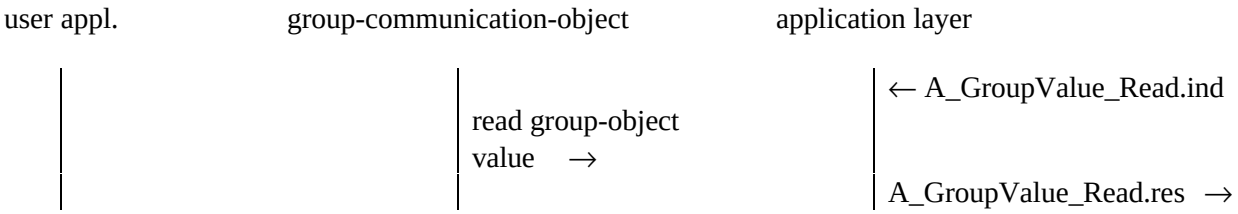

Fig. 3/4/1-3: Receiving an A_GroupValue_Read ind

2.3.3 Transmitting a Group-Object Value

The following figure sketches in vertical direction the flow of action to transmit a group-object value:

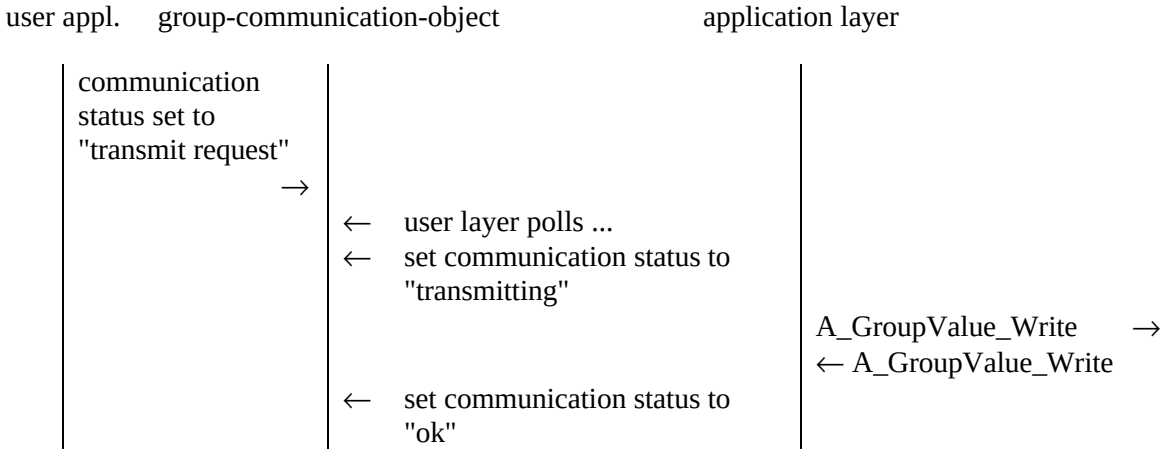

Fig. 3/4/1-4: Transmitting a Group-Object Value

If a negative A_GroupValue_Write.con returns then the transmission status shall be set to "error".

2.3.4 Receiving an A_GroupValue_Write.ind

The following figure sketches in vertical direction the flow of action of the user layer during reception of an A_GroupValue_Write.ind:

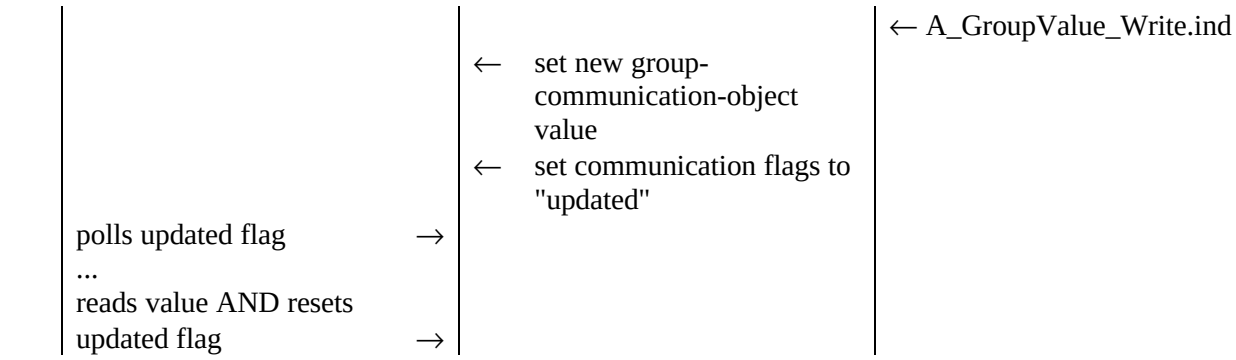

Fig. 3/4/1-5: Receiving an A_GroupValue_Write.ind

3. General Structure of EIB Objects

EIB Objects have a common general structure. Instances of that structure are called EIB objects. EIB objects can be located either in the internal or in the external user application. Each object instance in an EIB end device has a unique identifier, the object_id.

An EIB object has the following structure:

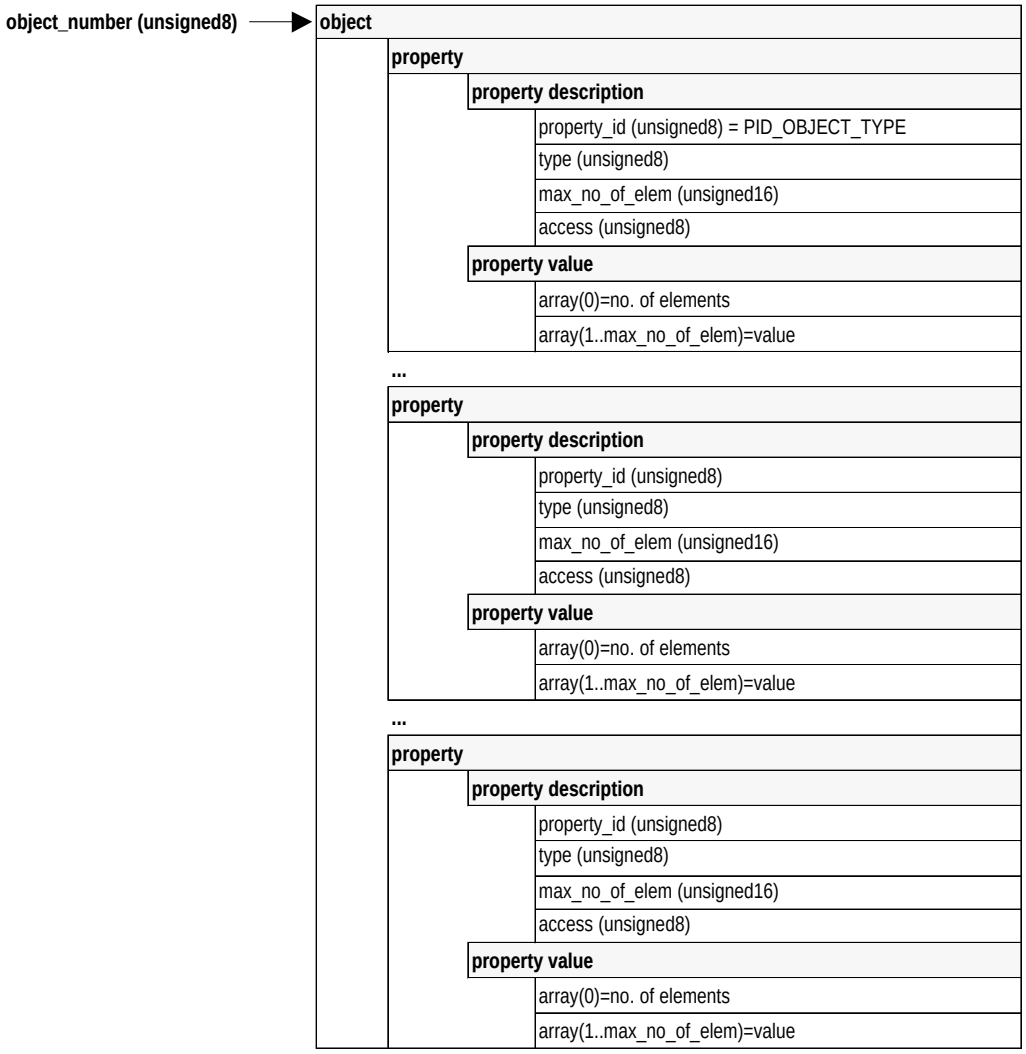

Fig. 3/4/1-6: General Structure of EIB Objects

Every object consists of at least one property. Every property consists of a property description and a property value. The property description consists of a property_id, a type, max_no_of_elem and access level. The type describes the data type of the property.

The property_id must be unique for the object. The property with ID 1 is the description of the object itself. This property is mandatory for every object.

The attribute access in the property description indicates the necessary access level to read from or write to the property value (see Chapter 3/3/7 "Application Layer", clause "A_Authorize_Request").

The value of a property is an array with array index 1... max_no_of_elem. The maximum number of elements of the array is defined in max_no_of_elem of the property description. The array element ´0´ contains the current number (unsigned16) of valid array elements. The array can be reset to no elements by writing zero on element ´0´. The array is automatically extended if an element is written beyond the currently last element, but within the maximum allowed number of entries.

3.1 EIB Object Interworking

Standards for EIB Objects have been set for indication of the property data types, Ids and the object-types. Please refer to Part 3/7 "Interworking".

4. System EIB Objects

System objects are EIB objects usable for network management. No system objects are mandatory for an EIB Device. There are four system objects:

1. the device object
2. the address table object
3. the association table object
4. the application program object.

The purpose of system objects is to enable uniform network management. Network management is supporting the end user during the configuration of EIB end devices through predefined system objects that offer the necessary properties. See Part 3/5 "Network Management" for more details.

Note: Every object has its own distinct behaviour that is described in an accompanying status machine. The status machine defines the rules to access each property or group of properties from devices. See Part 3/5 "Network Management" for the status machines of the system objects.

4.1 Device Object (object_id 0)

The device object contains global information about a device. It provides device specific management operations. The device object contains among other things the following information:

- the object ID
- the list of enabled services and service elements (e.g. verify mode)
- the operating system (firmware) including manufacturer, type, and version
- optionally the hardware device type, i.e. order number
- the actual detected PEI-Type

4.2 Address Table Object (object_id 1)

The address table object provides information about and management operations on the address table. See Part 3/5 "Network Management" on how it is used. The address table object contains among other things the following information:

- the object ID
- the load state

4.3 Association Table Object (object_id 2)

The association table object provides information about and management operations on the association table. See Part 3/5 "Network Management" how it is used. The association table object contains among other things the following information:

- the object ID
- the load state

4.4 Application Program Object (object_id 3)

The application program object contains global information about the internal user application program. It provides management operations for the internal user application program. See Part 3/5 "Network Management" how it is used for network management. If an application program can be loaded then an application program object must exists. This object contains among other things the following information:

- the object ID
- the load state
- the run state
- the application program ID
- the expected PEI-Type

4.5 Defining Own Application Objects (object_id 4, ...)

It is possible to create own application objects. They shall be created according to the structures and data types given in clause 3 "General Structure of EIB Objects"of this document.

4.5.1 Property Server for own Application Objects

Because own application objects are uniform to the system objects the same property server may be used for internal application objects, provided the application object structure is given to the property server implemented in the internal user layer. See clause 5.8 "Property Server"of this document.

4.5.2 Property Client for Accessing Remote Application Objects

The interface of the property client at the external user application depends on the EMI chosen by the LM_Switch message, see Chapter 3/6/3 "External Message Interface", clause "Layer Access Management". If the transport layer EMI is chosen, then A_PropertyValue_Read, A_PropertyDescription_Read and A_PropertyValue_Write service requests must generated by entering respective APDU's as user data to the T_Data.req or to T_Data_Unack.req messages.

If the user layer EMI is chosen then the accessing remote application objects depends on private communication between the external user application and the local internal one on the basis of U_Userdata.req messages, see Chapter 3/6/3 "External Message Interface", clause "User Layer EMI".

An internal property client may be based on the internal message format of A_PropertyValue_Read, A_PropertyDescription_Read and A_PropertyValue_Write request and confirmation messages. The major obstacle will be the restricted EEPROM space for the internal user application comprising this internal property client.

5. Object Servers and their Object Structures

Some of the object servers described in the following sub-clauses are parts of the network management server. The network management server, see its implicit description in Part 3/5 "Network Management", shall be completely contained in the user layer implementation.

5.1 Physical Address Server

The purpose of the A_PhysicalAddress_Read, A_PhysicalAddress_Write, A_PhysicalAddressSerialNumber_Read and A_PhysicalAddressSerialNumber_Write services on broadcast communication relationships is to read respectively change the physical address of a remote communication partner.

The user layer implementation shall realize at least the server part of those services as described in Chapter 3/3/7 "Application Layer" in the respective clauses and in Part 3/5 "Network Management", clause „Setting the Physical Address“. The data format of the physical address is explained in Chapter 3/3/2 "Data Link Layer General".

5.2 AD Converter Value Server

The purpose of the A_ADC_Read service on one-to-one connection-oriented communication relationships is to read one of up to 64 AD channels for read_count times in a remote communication partner.

The user layer implementation shall realize at least the server part of that service as described in Chapter 3/3/7 "Application Layer" in the respective clause.

5.3 BAU Memory Access Server

The purpose of the A_Memory_Read, A_Memory_Write and the optional A_MemoryBit_Write services on one-to-one connection-oriented communication relationships is to read respectively change BAU memory of a remote communication partner.

The user layer implementation shall realize at least the server part of those services as described in Chapter 3/3/7 "Application Layer" in the respective clauses.

The memory access server is a basic element of the network management server, see Part 3/5 "Network Management". A special purpose of the A_Memory_Write service is to download pieces of system release 1.x network management objects during the device configuration phase. The whole download procedure must be realized by a sequence of A_Memory_Write services. The following network management objects can be changed by A_Memory_Write services.

- Address Table
- Association Table
- Application Program (incl. Group-Communication-Objects and Loadable PEI Protocol)
- Various System Parameter

5.4 External Memory Access Server

The purpose of the optional A_UserMemory_Read, A_UserMemory_Write and A_UserMemoryBit_Write services on one-to-one connection-oriented communication relationships is to read respectively change external memory of a remote communication partner. The external memory is potentially located in the address space of the second processor running the external user application. The actual details about that address space depend on the EIB device providing this service.

The user layer implementation shall realize the external memory access server at least in compliance with the definition of those services in Chapter 3/3/7 "Application Layer".

5.5 Information Server about the Manufacturer of the External User Application

The purpose of the optional A_UserManufacturerInfo_Read service on one-to-one connection-oriented communication relationships is to read the two-byte information generated by the manufacturer of the external user application. The data format shall be the same as that of the BAU manufacturer data.

The user layer implementation shall realize the information server about the manufacturer of the external user application at least in compliance with the definition of that service in Chapter 3/3/7 "Application Layer".

5.6 Mask Version and Mask Type Server

The purpose of the A_DeviceDescriptor_Read service on one-to-one connection-oriented communication relationships is to read the mask version and the mask type of a remote communication partner.

The user layer implementation shall realize at least the server part of that service as described in Chapter 3/3/7 "Application Layer" in the respective clause.

5.7 BAU Restart Server

The purpose of the A_Restart service on one-to-one connection-oriented communication relationships is to reset (hardware reset) the BAU of a remote communication partner. Depending on the EIB device that may lead also to a reset of the adapter hardware and the external user application.

The user layer implementation shall realize at least the server part of that service as described in Chapter 3/3/7 "Application Layer" in the respective clause.

5.8 Property Server

The purpose of the A_PropertyDescription_Read service on one-to-one connection-less and one-to-one connection-oriented communication relationships is to read the description of an object property situated at a remote communication partner. The purpose of the A_PropertyValue_Read and A_PropertyValue_Write services on one-to-one connection-less and one-to-one connection-oriented communication relationships is respectively to read and change the value of an object property situated at a remote communication partner.

The user layer implementation shall realize at least the server part of those services as described in Chapter 3/3/7 "Application Layer" in the respective clauses. The general structure of EIB objects, for which those services only make sense, is explained in clause 3 "General Structure of EIB Objects" of this document. Explanations of the data format of property descriptions and property values can also be found there. System objects, which are special kinds of EIB-objects, and their properties are explained in clause 4 "System EIB Objects" of this document.

Note: The property server serves both the system objects and the application objects.

Features of the A_PropertyValue_Read, A_PropertyValue_Write and A_PropertyDescription_Read services:

- All three services can use either the connection-less or the connection-oriented transport layer.
- The property object server shall be encoded in the local management server, so that the internal user application must not be involved.
- Those services allow accesses to the whole object or to a single property. To read/write several but not all properties is not possible.
- The property server for application Objects is active even if the internal user application is “Loaded”.

5.9 Service Information Server

The purpose of the A_ServiceInformation_Indication_Write service on broadcast communication relationships is to broadcast a call to the service people containing information about the service need.

The user layer implementation shall realize at least the server part of this service as described in Chapter 3/3/7 "Application Layer" in the respective clause. Furthermore it shall pass automatically the A_ServiceInformation_Indication_Write.req primitive to layer-7 in case

- a running internal user application is stopped by the removal of the local PEI adapter.
- a source address is detected in a layer-2 frame, which is identical to the own physical address.

6. Externally Accessible User Layer Interface

The group-communication-objects described in clause 2 "General Structure of Group-Communication-Objects" can be made available via the User Layer EMI to the external user application. Furthermore the internal user application can communicate with the external one by the U_Userdata service.

See Chapter 3/6/3 "External Message Interface", clause "User Layer EMI" for the user layer message formats and clause "Layer Access Management" how to switch to the user layer EMI.

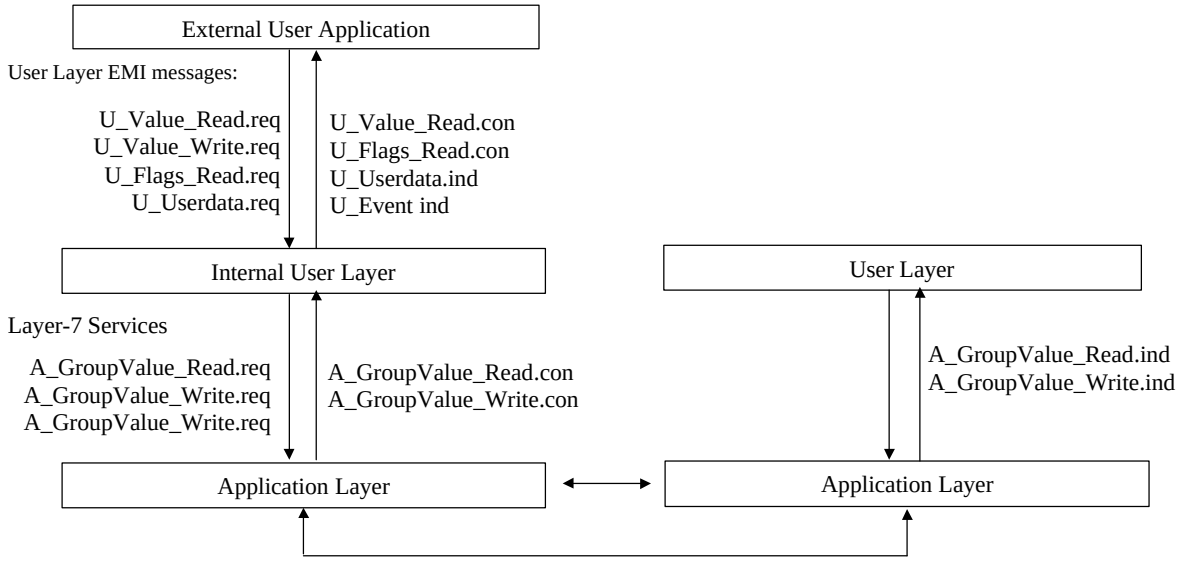

Fig. 3/4/1-7: Scheme: Message Interface to the External User Application

The U_Value_Read_ and U_Value_Write messages allow respectively reading and writing on the group-object value and on the group-object's Communication. If U_Event.ind generation is enabled (parameter of layer-8) then group-communication-object value changes are reported to the external user application.

The definition and the parameters of a group-communication-object can be read by the U_FLAGS_READ-message.

For transparent data transmission from and to the external user application the U_Userdata message must be used.

7. User Layer Parameters

7.1 U_Event.ind generation flag

If the U_Event.ind generation flag is set then automatically whenever the updated flag of the communication flags is set by the internal user layer, the internal user layer shall also generate an U_Event.ind message to the external user application.