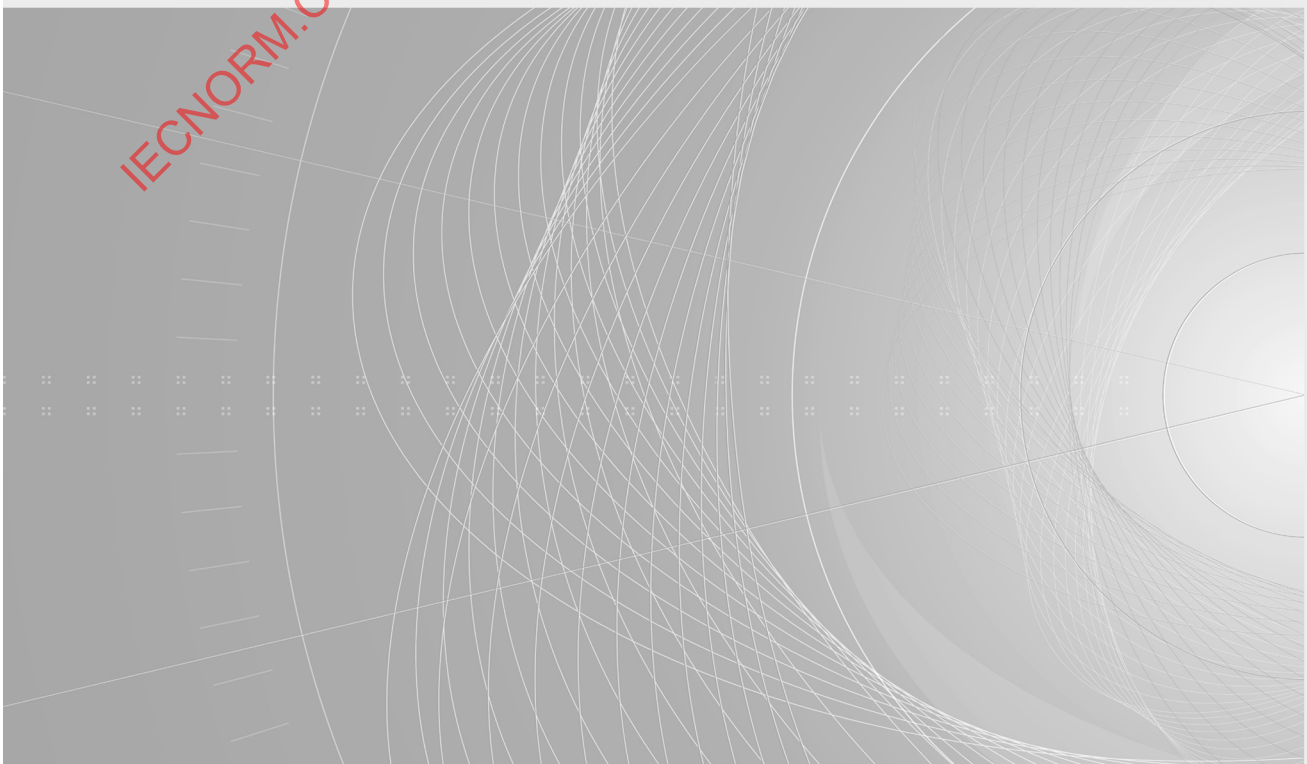
ISO/IEC 14543-3-4

Edition 1.0 2007-01

INTERNATIONAL STANDARD

Information technology – Home electronic system (HES) architecture – Part 3-4: System management – Management procedures for a network based control of HES Class 1



ISO/IEC 14543-3-4:2007-01(en)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
info@iec.ch
www.iec.ch

About the IEC
The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications
The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigendum or an amendment might have been published.

IEC publications search - webstore.iec.ch/advsearchform
The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished
Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and once a month by email.

IEC Customer Service Centre - webstore.iec.ch/csc
If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: sales@iec.ch.

Electropedia - www.electropedia.org
The world's leading online dictionary on electrotechnology, containing more than 22 000 terminological entries in English and French, with equivalent terms in 16 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary
67 000 electrotechnical terminology entries in English and French extracted from the Terms and definitions clause of IEC publications issued between 2002 and 2015. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

ISO/IEC 14543-3-4

Edition 1.0 2007-01

INTERNATIONAL STANDARD

Information technology – Home electronic system (HES) architecture – Part 3-4: System management – Management procedures for a network based control of HES Class 1

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 35.240.67

ISBN 2-8318-8953-7

CONTENTS

FOREWORD 7
INTRODUCTION 9
1 Scope 10
2 Normative references 10
3 Terms, definitions and abbreviations 11
3.1 Terms and definitions 11
3.2 Abbreviations 13
4 Conformance 13
5 Network management procedures 13
5.1 General 13
5.2 NM_IndividualAddress_Read 14
5.2.1 Description 14
5.2.2 Management service used 14
5.2.3 Sequence 14
5.2.4 Exception handling 14
5.3 NM_IndividualAddress_Write 14
5.3.1 Description 14
5.3.2 Management services used 15
5.3.3 Sequence 15
5.3.4 Exception handling 16
5.4 NM_SerialNumberDefaultIA_Scan 16
5.4.1 Description 16
5.4.2 Management service used 16
5.4.3 Sequence 17
5.4.4 Exception handling 17
5.5 NM_IndividualAddress_SerialNumber_Read 17
5.5.1 Description 17
5.5.2 Management service used 17
5.5.3 Sequence 17
5.5.4 Exception handling 18
5.6 NM_IndividualAddress_SerialNumber_Write 18
5.6.1 Description 18
5.6.2 Management services used 18
5.6.3 Sequence 18
5.6.4 Exception handling 18
5.7 NM_DomainAddress_Read 19
5.7.1 Description 19
5.7.2 Management service used 19
5.7.3 Sequence 19
5.7.4 Exception handling 19
5.8 NM_DomainAddress_Write 19
5.8.1 Description 19
5.8.2 Management services used 20
5.8.3 Sequence 20

5.8.4 Exception handling 21
5.9 NM_DomainAddress_Scan 21
5.9.1 Description 21
5.9.2 Management service used 21
5.9.3 Sequence 22
5.9.4 Exception handling 22
5.10 NM_Router_Scan 22
5.10.1 Description 22
5.10.2 Management service used 22
5.10.3 Sequence 23
5.11 NM_SubnetworkDevices_Scan 23
5.11.1 Description 23
5.11.2 Management service used 23
5.11.3 Sequence 23
5.12 NM_SubnetworkAddress_Read 24
5.12.1 Description 24
5.12.2 Management service used 24
5.12.3 Sequence 24
5.12.4 Exception handling 24
5.13 NM_IndividualAddress_Reset 24
5.13.1 Description 24
5.13.2 Management services used 24
5.13.3 Sequence 25
5.14 NM_IndividualAddress_Scan 25
5.14.1 Description 25
5.14.2 Management services used 25
5.14.3 Sequence 26
5.14.4 Possible reactions 26
5.15 NM_IndividualAddress_Check 26
5.15.1 Description 26
5.15.2 Management services used 26
5.15.3 Sequence 27
5.16 NM_IndividualAddress_Check_LocalSubnetwork 27
5.16.1 Description 27
5.16.2 Management service used 27
5.16.3 Sequence 28
5.17 NM_GroupAddress_Check 28
5.17.1 Description 28
5.17.2 Management service used 28
5.17.3 Sequence 28
5.17.4 Exception handling 29
5.18 NM_FunctionalBlock_Scan 29
5.18.1 Description 29
5.18.2 Management service used 29
5.18.3 Sequence 29
5.18.4 Exception handling 29
6 Device management procedures 30
6.1 General 30
6.2 General exception handling 30

6.3 DM_Connect 30
6.3.1 General Description 30
6.3.2 Procedure: DMP_Connect_RCo 30
6.3.3 Procedure: DMP_Connect_RCl 31
6.4 DM_Disconnect 32
6.4.1 General description 32
6.4.2 Procedure: DMP_Disconnect_RCo 32
6.4.3 Procedure: DMP_Disconnect_RCl 32
6.5 DM_Authorize 33
6.5.1 General description 33
6.5.2 Procedure: DMP_Authorize_RCo 33
6.6 DM_SetKey 33
6.6.1 General description 33
6.6.2 Procedure: DM_SetKey_RCo 34
6.7 DM_Restart 34
6.7.1 General description 34
6.7.2 Procedure: DM_Restart_RCo 34
6.8 DM_Delay 35
6.8.1 Description 35
6.8.2 Procedure: DMP_Delay 35
6.9 DM_IndividualAddressRead 35
6.10 DM_IndividualAddressWrite 35
6.11 DM_DomainAddressRead 36
6.12 DM_DomainAddressWrite 36
6.13 DM_ProgMode_Switch 36
6.13.1 Description 36
6.13.2 Procedure: DMP_ProgModeSwitch_RCo 36
6.14 DM_GroupObject_Link_Read 37
6.14.1 Description 37
6.14.2 Management service used 37
6.14.3 Sequence 37
6.14.4 Exception handling 37
6.15 DM_GroupObject_Link_Write 37
6.15.1 Description 37
6.15.2 Management services used 38
6.15.3 Sequence 38
6.15.4 Exception handling 38
6.16 DM_MemWrite 38
6.16.1 General description 38
6.16.2 Procedure: DMP_MemWrite_RCo 39
6.16.3 Procedure: DMP_MemWrite_RCoV 40
6.17 DM_MemVerify 41
6.17.1 General description 41
6.17.2 Procedure: DMP_MemVerify_RCo 42
6.18 DM_MemRead 42
6.18.1 General description 42
6.18.2 Procedure: DMP_MemRead_RCo 43
6.19 DM_UserMemWrite 43
6.19.1 General description 43

6.19.2 Procedure: DMP_UserMemWrite_RCo 44
6.19.3 Procedure: DMP_UserMemWrite_RCoV 45
6.20 DM_UserMemVerify 46
6.20.1 General description 46
6.20.2 Procedure: DMP_UserMemVerify_RCo 46
6.21 DM_UserMemRead 47
6.21.1 General description 47
6.21.2 Procedure: DMP_UserMemRead_RCo 47
6.22 DM_InterfaceObjectWrite 48
6.22.1 General description 48
6.22.2 Procedure: DMP_InterfaceObjectWrite_R 48
6.23 DM_InterfaceObjectVerify 49
6.23.1 General description 49
6.23.2 Procedure: DMP_InterfaceObjectVerify_R 50
6.24 DM_InterfaceObjectRead 51
6.24.1 General description 51
6.24.2 Procedure: DMP_InterfaceObjectRead_R 51
6.25 DM_InterfaceObjectScan 52
6.25.1 General description 52
6.25.2 Procedure: DMP_InterfaceObjectScan_R 53
6.26 DM_LoadStateMachineWrite 54
6.26.1 General description 54
6.26.2 Procedure: DMP_LoadStateMachineWrite_RCo_Mem 56
6.26.3 Procedure: DMP_LoadStateMachineWrite_RCo_IO 59
6.27 DM_LoadStateMachineVerify 62
6.27.1 General description 62
6.27.2 Procedure: DM_LoadStateMachineVerify_RCo_Mem 63
6.27.3 Procedure: DMP_LoadStateMachineVerify_R_IO 64
6.28 DM_LoadStateMachineRead 64
6.28.1 General description 64
6.28.2 Procedure: DMP_LoadStateMachineRead_RCo_Mem 65
6.28.3 Procedure: DMP_LoadStateMachineRead_R_IO 66
6.29 DM_RunStateMachineWrite 67
6.29.1 General description 67
6.29.2 Procedure: DMP_RunStateMachineWrite_RCo_Mem 67
6.29.3 Procedure: DMP_RunStateMachineWrite_R_IO 68
6.30 DM_RunStateMachineVerify 69
6.30.1 General description 69
6.30.2 Procedure: DMP_RunStateMachineVerify_RCo_Mem 70
6.30.3 Procedure: DMP_RunStateMachineVerify_R_IO 70
6.31 DM_RunStateMachineRead 71
6.31.1 General description 71
6.31.2 Procedure: DMP_RunStateMachineRead_RCo_Mem 72
6.31.3 Procedure: DMP_RunStateMachineRead_R_IO 72
6.32 DM_LCSlaveMemWrite 73
6.32.1 General description 73
6.32.2 Procedure: DMP_LCSlaveMemWrite_RCo 74
6.33 DM_LCSlaveMemVerify 75
6.33.1 General description 75

6.33.2 Procedure: DMP_LCSlaveMemVerify_RCo 75
6.34 DM_LCSlaveMemRead 76
6.34.1 General description 76
6.34.2 Procedure: DMP_LCSlaveMemRead_RCo 76
6.35 DM_LCExtMemWrite 77
6.35.1 General description 77
6.35.2 Procedure: DMP_LCExtMemWrite_RCo 78
6.36 DM_LCExtMemVerify 79
6.36.1 General description 79
6.36.2 Procedure: DMP_LCExtMemVerify_RCo 80
6.37 DM_LCExtMemRead 80
6.37.1 General description 80
6.37.2 Procedure: DMP_LCExtMemRead_RCo 81
6.38 DM_LCExtMemOpen 81
6.38.1 General description 81
6.38.2 Procedure: DMP_LCExtMemOpen_RCo 82
6.39 DM_LCRouteTableStateWrite 82
6.39.1 General description 82
6.39.2 Procedure: DMP_LCRouteTableStateWrite_RCo 82
6.40 DM_LCRouteTableStateVerify 83
6.40.1 General description 83
6.40.2 Procedure: DMP_LCRouteTableStateVerify_RCo 83
6.41 DM_LCRouteTableStateRead 84
6.41.1 General description 84
6.41.2 Procedure: DMP_LCRouteTableStateRead_RCo 84
Bibliography 86

Table 1 – Resulting states after each event 55
Table 2 – Overview state machine types and tables 55
Table 3 – Overview addresses for the load management controls 56
Table 4 – Addresses of the load state controls 63
Table 5 – Addresses of the load state controls 65
Table 6 – Run state events and resulting run states 67
Table 7 – Addresses of the run state controls 68

INFORMATION TECHNOLOGY – HOME ELECTRONIC SYSTEM (HES) ARCHITECTURE –

Part 3-4: System management – Management procedures for network based control of HES Class 1

FOREWORD

1) ISO (International Organization for Standardization) and IEC (International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards. Their preparation is entrusted to technical committees; any ISO and IEC member body interested in the subject dealt with may participate in this preparatory work. International governmental and non-governmental organizations liaising with ISO and IEC also participate in this preparation.

2) In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

3) The formal decisions or agreements of IEC and ISO on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC and ISO member bodies.

4) IEC, ISO and ISO/IEC Publications have the form of recommendations for international use and are accepted by IEC and ISO member bodies in that sense. While all reasonable efforts are made to ensure that the technical content of IEC, ISO and ISO/IEC Publications is accurate, IEC or ISO cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.

5) In order to promote international uniformity, IEC and ISO member bodies undertake to apply IEC, ISO and ISO/IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any ISO/IEC Publication and the corresponding national or regional publication should be clearly indicated in the latter.

6) ISO and IEC provide no marking procedure to indicate their approval and cannot be rendered responsible for any equipment declared to be in conformity with an ISO/IEC Publication.

7) All users should ensure that they have the latest edition of this publication.

8) No liability shall attach to IEC or ISO or its directors, employees, servants or agents including individual experts and members of their technical committees and IEC or ISO member bodies for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication of, use of, or reliance upon, this ISO/IEC publication or any other IEC, ISO or ISO/IEC publications.

9) Attention is drawn to the normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.

IEC and ISO draw attention to the fact that it is claimed that compliance with this document may involve the use of a patent concerning a particular way of using the Network Management Mechanism stated in this standard.

Hager Control SAS has informed IEC and ISO that it has the following patent that is not essential for the implementation of any particular clause of this standards but may concern specific combinations thereof:

EP 0817 423A1

ISO and IEC take no position concerning the evidence, validity and scope of this putative patent right. The holder of this putative patent right has assured IEC and ISO that they are willing to negotiate free licences or licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this putative patent right is registered with IEC and ISO. Information may be obtained from:

Hager Control SAS
33, rue Saint-Nicolas
PB 154
F-67704 Saverne Cedex
France

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. IEC and ISO shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 14543-3-4 was prepared by subcommittee 25: Interconnection of information technology equipment, of ISO/IEC joint technical committee 1: Information technology.

This International Standard is a product family standard. It shall be used in conjunction with ISO/IEC 14543-2-1, 14543-3-1, 14543-3-2, 14543-3-3, 14543-3-5, 14543-3-6 and 14543-3-7.

This International Standard has been approved by vote of the member bodies, and the voting results may be obtained from the address given on the title page.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

INTRODUCTION

The management procedures capture the dynamics of managing distributed resources on the network in terms of abstract procedures. On the network itself, a procedure consists of a sequence of telegrams, exchanged between two partners, the management client and the management server.

The management client is a powerful device with 'controller' function, typically, but not exclusively, PC-based. Except for network-oriented management, the server is always a 'target device'. In the former case, it is in fact the network as a whole which acts as partner or server. Ultimately, of course, the response to a client request is always generated by the individual devices connected to the network, either one or many. In addition to its run-time behaviour (based on group communication), every device moreover supports a rich management server profile for this purpose. An important objective of this part “Management Procedures” is to allow a concise description of such a profile. It is clear that the information about the full set of management procedures supported by a particular device or implementation tells us significantly more about the device than merely the list of services through which this is realised.

In general, one single device may well implement both client as well as server function. For and during the execution of a particular management procedure, however, one device takes on one single role.

Currently, ISO/IEC 14543, *Information technology – Home Electronic System (HES) architecture*, consists of the following parts:

Part 2-1: *Introduction and device modularity*
Part 3-1: *Communication layers – Application layer for network based control of HES Class 1*
Part 3-2: *Communication layers – Transport, network and general parts of data link layer for network based control of HES Class 1*
Part 3-3: *User process for network based control of HES Class 1*
Part 3-4: *System management – Management procedures for network based control of HES Class 1*
Part 3-5: *Media and media dependent layers – Power line for network based control of HES Class 1*
Part 3-6: *Media and media dependent layers – Twisted pair for network based control of HES Class 1*
Part 3-7: *Media and media dependent layers – Radio frequency for network based control of HES Class 1*
Part 4: *Home and building automation in a mixed-use building (technical report)*
Part 5-1: *Intelligent grouping and resource sharing for HES Class 2 and Class 3 – Core protocol (under consideration)*
Part 5-2: *Intelligent grouping and resource sharing for HES Class 2 and Class 3 – Device certification (under consideration)*

Additional parts may be added later.

INFORMATION TECHNOLOGY – HOME ELECTRONIC SYSTEM (HES) ARCHITECTURE –

Part 3-4: System management – Management procedures for network based control of HES Class 1

1 Scope

This part of ISO/IEC 14543 establishes general principles for network and device management shared by all installation modes for network based control of HES Class 1 and independent of the installation mode used. The aim is to standardize the interaction between a management client and a management server which leads to the successful configuration of the devices. The management procedures thus specify the highest level communication requirements between a management client and a management server. These requirements specify

a) the **sequence** of messages that shall be exchanged between a management client and a management server,

b) the **contents** and **interpretation** of the transported data,

c) the **action** to take based on this data (setting internal resources, state machines, physical actions, …), and

d) the error and exception handling.

The management procedures are based on the application layer services.

Some management procedures are solely based on the use of one or a sequence of dedicated application layer services to achieve the required goal. For these, ISO/IEC 14543-3-1 and ISO/IEC 14543-3-2 provide sufficient information concerning the underlying mechanisms.

Other management procedures additionally use the application layer services to access internal data in the management server to achieve the required goal. This data is defined as objects as specified in ISO/IEC 14543-3-3.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 14543-2-1, *Information technology – Home electronic system (HES) architecture – Part 2-1: Introduction and device modularity*

ISO/IEC 14543-3-1, *Information technology – Home electronic system (HES) architecture – Part 3-1: Communication layers – Application layer for network based control of HES Class 1*

ISO/IEC 14543-3-2, *Information technology – Home electronic system (HES) architecture – Part 3-2: Communication layers – Transport, network and general parts of data link layer for network based control of HES Class 1*

ISO/IEC 14543-3-3, *Information technology – Home electronic system (HES) architecture – Part 3-3: User process for network based control of HES Class 1*

ISO/IEC 14543-3-5, *Information technology – Home electronic system (HES) architecture – Part 3-5: Media and media dependent layers – Power line for network based control of HES Class 1*

ISO/IEC 14543-3-6, *Information technology – Home electronic system (HES) architecture – Part 3-6: Media and media dependent layers – Twisted pair for network based control of HES Class 1*

ISO/IEC 14543-3-7, *Information technology – Home electronic system (HES) architecture – Part 3-7: Media and media dependent layers – Radio frequency for network based control of HES Class 1*

3 Terms, definitions and abbreviations

3.1 Terms and definitions

For the purposes of this International Standard the terms and definitions given in ISO/IEC 14543-2-1 and the following apply.

3.1.1
network
combination of several transmission links connected at individual points by electrical or optical means as part of an installation, system, appliance or component

3.1.2
bus access unit (BAU)
contains all protocol layers plus the optional internal user application

3.1.3
device
product
HES products consist of devices in the form of hardware, firmware and their associated software

3.1.4
management procedures
the dynamics of managing distributed resources on the network in terms of abstract procedures between two partners, the management client and the management server

3.1.5
management client
powerful device with 'controller' function, typically but not exclusively PC-based

3.1.6
management server
a particular device that acts as target device; except for network-oriented management, where the network as a whole acts as partner or server

3.1.7
network management
device-independent management procedures on the network as for example reading/writing the individual address and scanning the network. For these procedures no knowledge of the single devices is required

3.1.8
device management
procedures to access one specific device. These procedures describe for example the load procedures or reading the state. A detailed knowledge of the device is required for these procedures

3.1.9
communication mode
mode describing the relationship between communication points upon which the communication relies: one-to-many connectionless (multicast), one-to-all connectionless (broadcast), one-to-one connectionless, one-to-one connection-oriented

3.1.10
Group Address Table (GrAT)
shared resource of both the Link Layer and the group-oriented Transport Layer; used by the Link Layer as a look-up reference to check whether it should pass a received frame to the upper layers or not and used by the group-oriented Transport Layer to map an incoming LSAP (Group Address) to a TSAP in receiving direction and vice versa in sending direction

3.1.11
group object association table
resource of the Application Layer that stores the relationship between Transport Layer Service Access Points (TSAPs) and Application Layer Service Access Points (ASAP), as needed when mapping the Multicast Communication Mode messages A_GroupValue_Read and A_GroupValue_Write to T_Data_Group messages and vice versa

NOTE 1 The TSAP is an index in the Group Address Table. The ASAP is the Group Object number. The lowest ASAP is 0.

NOTE 2 The ASAP is a unique identifier for a group object to the Application Layer. Please also refer to the Application Layer specifications in ISO/IEC 14543-3-1. The ASAP is thus a group object number.

3.1.12
application program
element within an installed system (i.e. in a device) which performs information processing for a particular application and ensures the operations needed to execute the application

3.1.13
physical external interface (PEI)
physical and electrical interface situated in a device between the bus access unit and any hardware performing an application function

3.1.14
PEI type
physical and logical identifier of the configuration of the PEI to enable hardware compatibility recognition

3.1.15
external message interface (EMI)
collection of messages that together build a generic message interface to each protocol layer of a BAU and any application function

3.2 Abbreviations

ASAP	Application Layer Service Access Points
BAU	Bus Access Unit
DoA	Domain Address field in the frame
DoA_Device	Domain Addresses of the Device of which the individual address is read; it is contained in the response if the device is on Powerline
EMI	External Message Interface
GrAT	Group Address Table
IA	Individual Address of the sender
PEI	Physical External Interface
PPPP	Individual address of the device, in the response
RCo	Point-to-Point, Connection-oriented Communication Mode to a remote device
RCoV	Point-to-Point, Connection-oriented Communication Mode with verification to a remote device
RCl	Point-to-Point, Connectionless Communication Mode to a remote device
SA	Source Address of the sender
SN	Serial Number field in the frame
SN_Device	Serial Number of the Device of which the individual address is to be read
TL	Transport Layer
TSAPs	Transport Layer Service Access Points

4 Conformance

A management server conforming to this International Standard shall support all the network management procedures specified in clause 5 which contain the services it supports and all the device management procedures specified in clause 6 which contain the services it supports.

5 Network management procedures

5.1 General

The network management procedures describe the device-independent management procedures. These procedures shall be used to configure the network and to obtain information on the configuration of the network and connected devices.

For these procedures no knowledge of the single devices is required. They will work with every device connected to the network with the management server function implemented. Both management server and management client shall be based on the use of the dedicated application layer services which are specified in ISO/IEC 14543-3-1 for this purpose. Every individual management procedure below contains a dedicated subclause “Management services used” referencing, by name, the application layer services used. The procedures work independently of the location of the management client in the network. Some procedures require the previous configuration of routers and domain addresses via other procedures.

5.2 NM_IndividualAddress_Read

5.2.1 Description

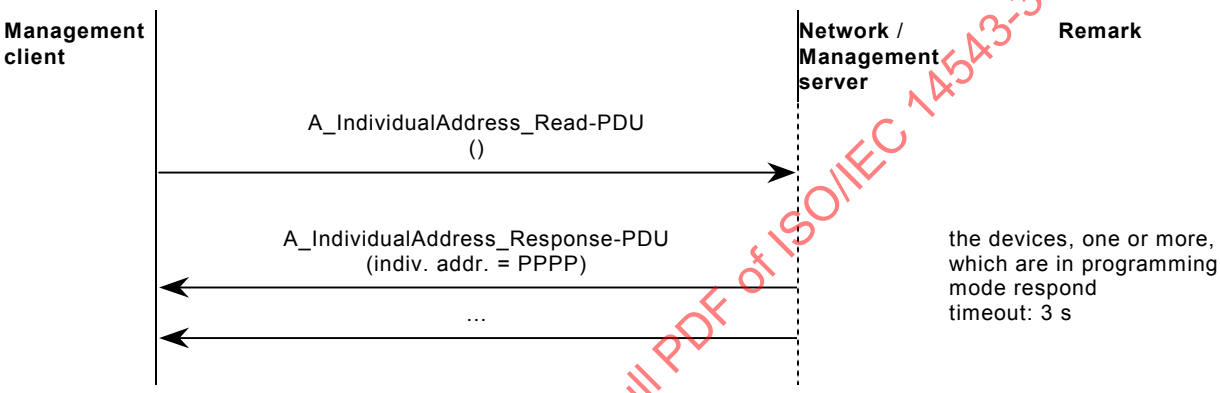
This network management procedure shall be used to read out the individual addresses of all devices which are in programming mode.

This procedure works independently of the configuration of the individual address of the router. When applicable, this procedure shall be preceded by the configuration of the domain address.

5.2.2 Management service used

The NM_IndividualAddress_Read procedure shall use the following management service:
– A_IndividualAddress_Read.

5.2.3 Sequence



5.2.4 Exception handling

Always wait until the timeout has elapsed. Collect all responses during this timeout.
If no A_IndividualAddress_Response is received, no device is in programming mode.
If one A_IndividualAddress_Response is received, exactly one device is in programming mode.
If more than one response is received, several devices are in programming mode.
If two or more responses with the same individual address are received, there is more than one device with the same individual addresses.
Do not evaluate Layer-2 repetitions.

5.3 NM_IndividualAddress_Write

5.3.1 Description

This network management procedure shall be used to write the individual address of one single device which is in programming mode.

The procedure shall wait until exactly one device is in programming mode. It shall check that no other device has the same individual address and only one device is in programming mode. The procedure shall check if the programming was successful and switch off the programming mode by executing a restart of the device.

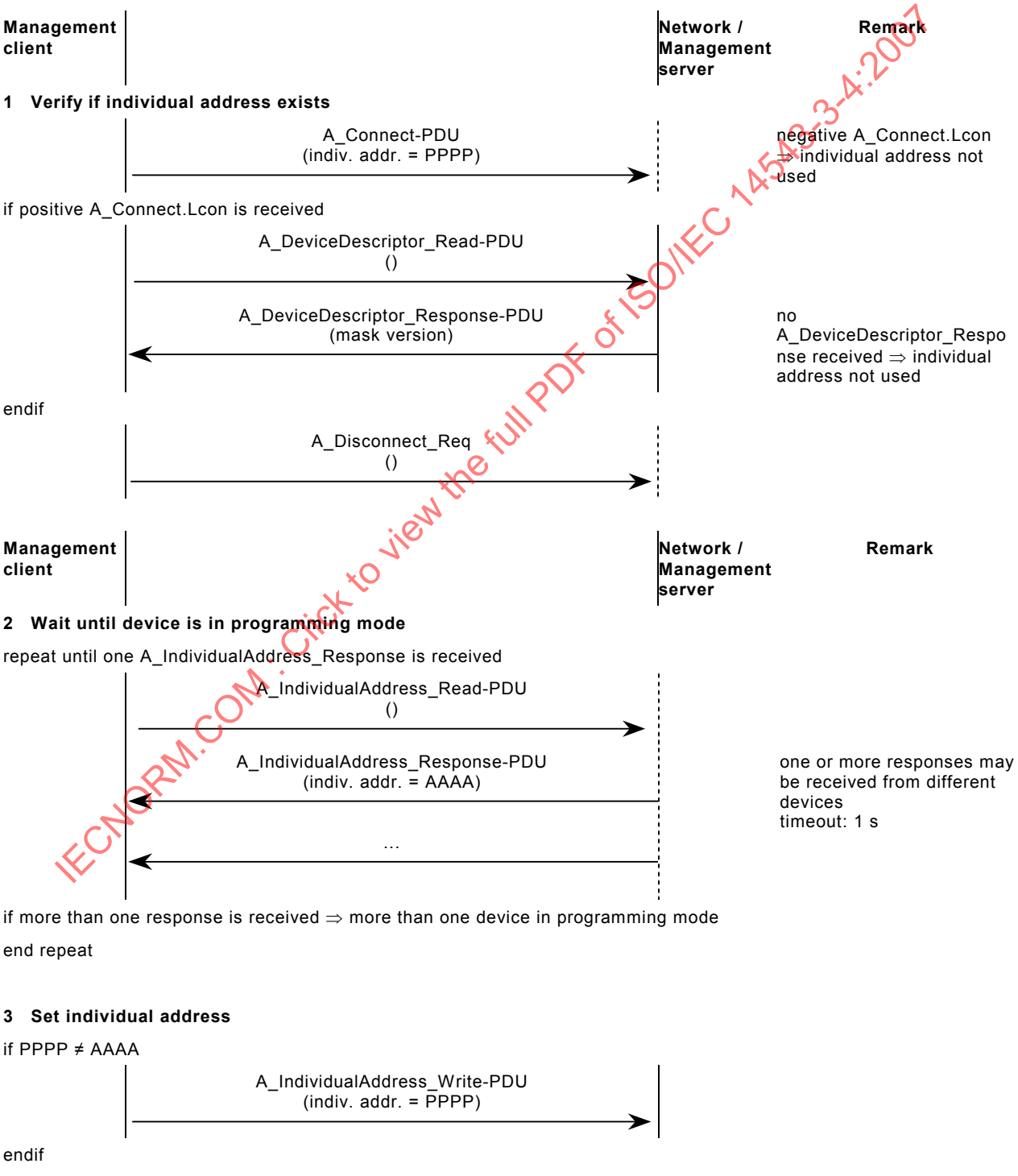
When applicable, this procedure shall be preceded by the configuration of the individual addresses of the installed routers and the domain addresses.

5.3.2 Management services used

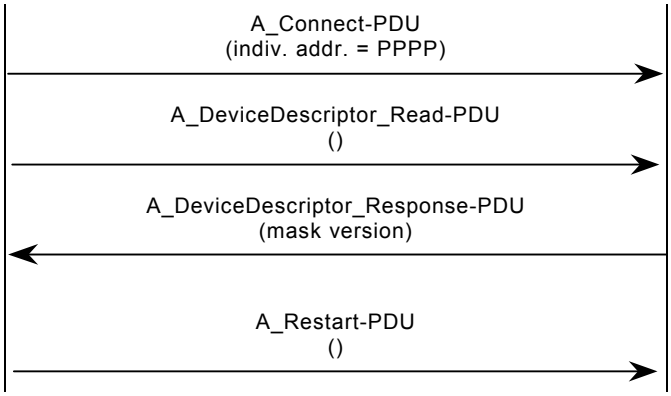
The NM_IndividualAddress_Write procedure shall use the following management services:

- A_IndividualAddress_Read;
- A_IndividualAddress_Write;
- A_DeviceDescriptor_Read;
- A_Restart;
- A_Connect.

5.3.3 Sequence



4 Verify and switch LED off



abort the connection of the client side transport layer

5.3.4 Exception handling

to 1: If an A_Disconnect.ind is received instead of an A_DeviceDescriptor_Response-PDU, then a device with this individual address exists but it may either use another connection or does not support connection-oriented mode. Continue with the procedure in every case.

to 2: The management client shall always wait until the timeout has elapsed. It shall collect all the responses during this timeout. This procedure shall wait until one device is in programming mode.

NOTE: The user of the management client should receive notification of how many devices are in programming mode (none or more than one).

The following cases may occur at this point:

– A device with the individual address exists, but it is not the one which is in programming mode
  ⇒ do not continue with the procedure.

– A device with the individual address exists, and it is the one which is in programming mode
  ⇒ continue with the procedure.

– No device with the individual address exists
  ⇒ continue with the procedure.

to 4: If no A_DeviceDescriptor_Response-PDU is received, then the programming of the individual address may have failed, or the system (router) was not configured correctly.

5.4 NM_SerialNumberDefaultIA_Scan

5.4.1 Description

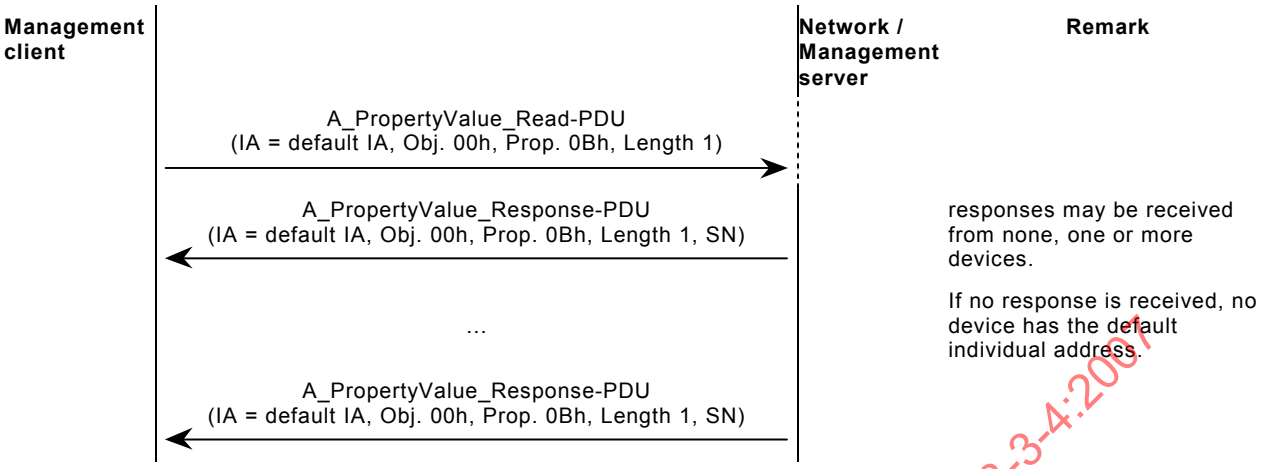
This network management procedure shall be used to obtain the serial number of each device of which the individual address (IA) is the default individual address.

5.4.2 Management service used

The NM_SerialNumberDeafultIA_Scan procedure shall use the following management service:

– A_PropertyValue_Read.

5.4.3 Sequence



5.4.4 Exception handling

The general exception handling of 5.2 shall be applied.

5.5 NM_IndividualAddress_SerialNumber_Read

5.5.1 Description

This network management procedure shall be used to read the individual address of one single device of which the serial number is known.

The serial number of the device (SN_Device) shall be known in advance.

If the individual address of more than one device has to be read, this network management procedure “NM_IndividualAddress_SerialNumber_Read” shall be repeated for each device, using each device’s serial number.

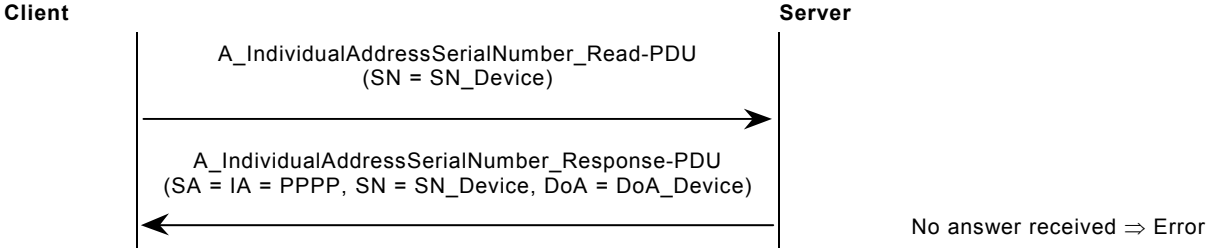
5.5.2 Management service used

The NM_IndividualAddress_Read procedure shall use the following management service:
– A_IndividualAddressSerialNumber_Read.

5.5.3 Sequence

SN = serial number field in the frame
SN_Device = serial number of the device of which the individual address is to be read
SA = Source Address of the sender
IA = individual address of the sender
PPPP = the individual address of the device, in the response
DoA = domain address field in the frame
DoA_Device = the domain addresses of the device of which the individual address is read; it is contained in the response if the device is on Powerline.

1 Get individual address of server



The individual address is contained as the source address of the response frame.

5.5.4 Exception handling

If no answer is received, there is no device present in the network with the given serial number.

5.6 NM_IndividualAddress_SerialNumber_Write

5.6.1 Description

This network management procedure shall be used to write the individual address of one single device of which the serial number is known. The procedure shall ensure that the assigned individual address is unique, and shall check whether the programming was successful.

If applicable this procedure shall be preceded by the configuration of the individual addresses of the installed routers and the domain addresses.

The serial number of the device to be programmed shall be known in advance, either by the procedure NM_SerialNumberDefaultIA_Scan or by any other means.

If the individual address of more than one device has to be programmed, this network management procedure shall be repeated for each device, using that device's serial number.

5.6.2 Management services used

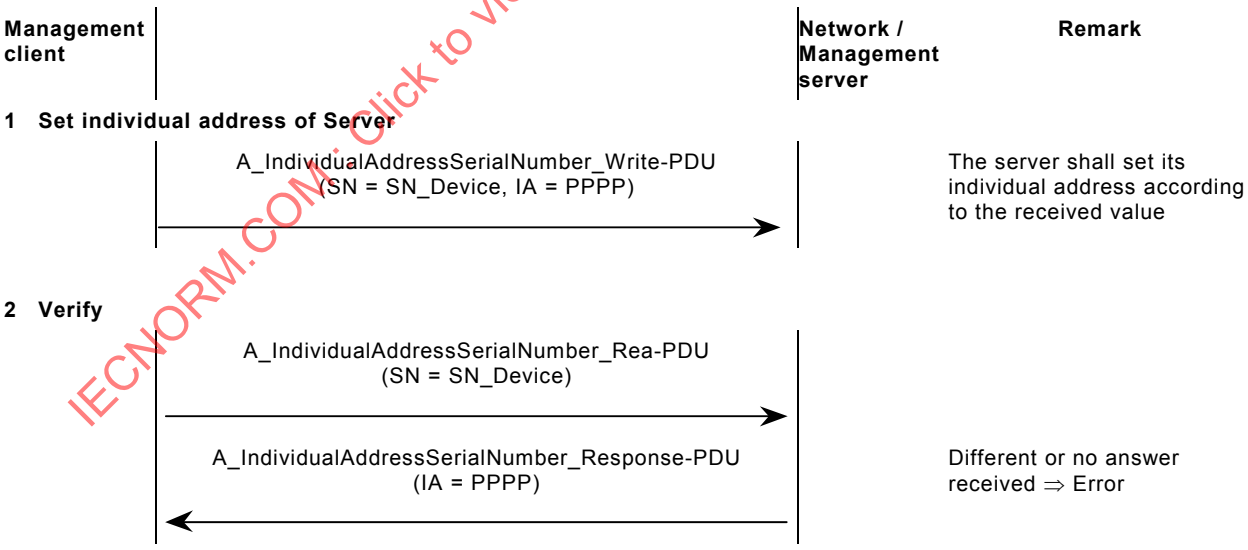
The NM_IndividualAddress_SerialNumber_Write procedure shall use the following management services:

– A_IndividualAddressSerialNumber_Write;

– A_IndividualAddressSerialNumber_Read.

5.6.3 Sequence

PPPP = individual address to be programmed
SN_Device = serial number of device to which the individual address will be assigned



NOTE In contrast to the procedures "NM_IndvidualAddress_Write" and "NM_DomainAndIndividualAddress_Write", both requiring devices to be in 'programming mode' and using the A_IndividualAddress_Write service, this procedure does not reset the device after assigning the individual address.

5.6.4 Exception handling

The general exception handling of 5.2 shall be applied.

5.7 NM_DomainAddress_Read

5.7.1 Description

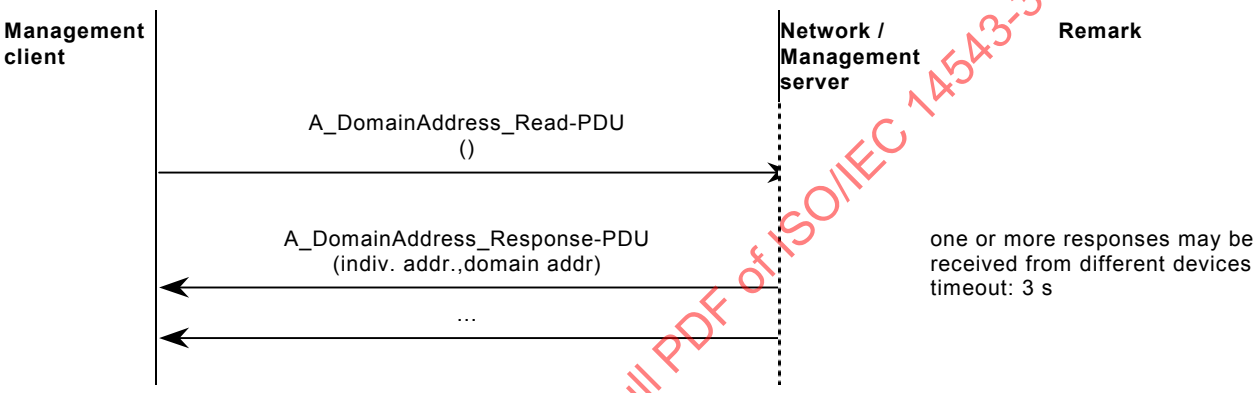
This network management procedure shall be used to read out the domain addresses of all the devices which are in programming mode.

This procedure shall be used independently of the configuration of the domain address and the individual address of the router.

5.7.2 Management service used

The NM_DomainAddress_Read procedure shall use the following management service:
– A_DomainAddress_Read.

5.7.3 Sequence



5.7.4 Exception handling

Always wait until the timeout period has elapsed. Collect all responses during this timeout period.

If no A_DomainAddress_Response is received, no device is in programming mode.

If one A_DomainAddress_Response is received, exactly one device is in programming mode.

If more than one response is received, several devices are in programming mode.

If two or more responses with the same domain address and individual addresses are received, there is more than one device with the same domain address and the same individual addresses.

Layer-2 repetitions shall not be evaluated.

5.8 NM_DomainAddress_Write

5.8.1 Description

This network management procedure shall be used to set the domain address and the individual address of one single device which is in programming mode.

This procedure shall ensure that no other device has the same individual address and wait until exactly one device is in programming mode. It shall verify whether the programming is successful and switch the device into normal operation mode by executing a restart of the device.

For this procedure the management server has to provide a free domain address.

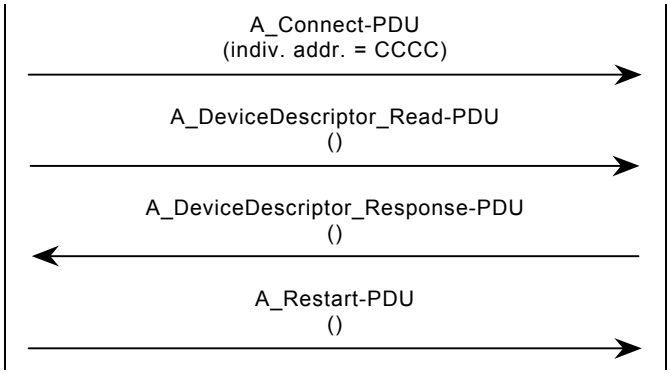
5.8.2 Management services used

The NM_DomainAddress_Write procedure shall use the following management services:

- A_DomainAddress_Read;
- A_DomainAddress_Write;
- A_IndividualAddress_Write;
- A_DeviceDescriptor_Read;
- A_Restart;
- A_Connect.

5.8.3 Sequence

4 Verify and switch LED off



abort the connection of the client side transport layer

5.8.4 Exception handling

to 1: If an A_Disconnect-PDU is received instead of an A_DeviceDescriptor_Response-PDU, then a device with this individual address exists but either it may be using another connection, or it does not support connection-oriented mode. The procedure shall be continued in any case.

to 2: The management server shall always wait until the timeout has elapsed. It shall collect all responses during this timeout. This procedure shall wait until one device is in programming mode.

NOTE The user of the management client should receive notification of how many devices are in programming mode (none or more than one).

The following cases may occur at this point:

– A device with the individual address exists, but it is not the one which is in programming mode.
  ⇒ The procedure shall be aborted in this case.

– A device with the individual address exists, and it is the one which is in programming mode.
  ⇒ The procedure shall be continued in this case.

– No device with the individual address exists.
  ⇒ The procedure shall be continued in this case.

to 4: If no A_DeviceDescriptor_Response-PDU is received, then the programming of the individual address may have failed or the system (router) has not been configured correctly.

5.9 NM_DomainAddress_Scan

5.9.1 Description

This management procedure shall be used to provide a free domain address that is used by no device within the physical reach of this medium.

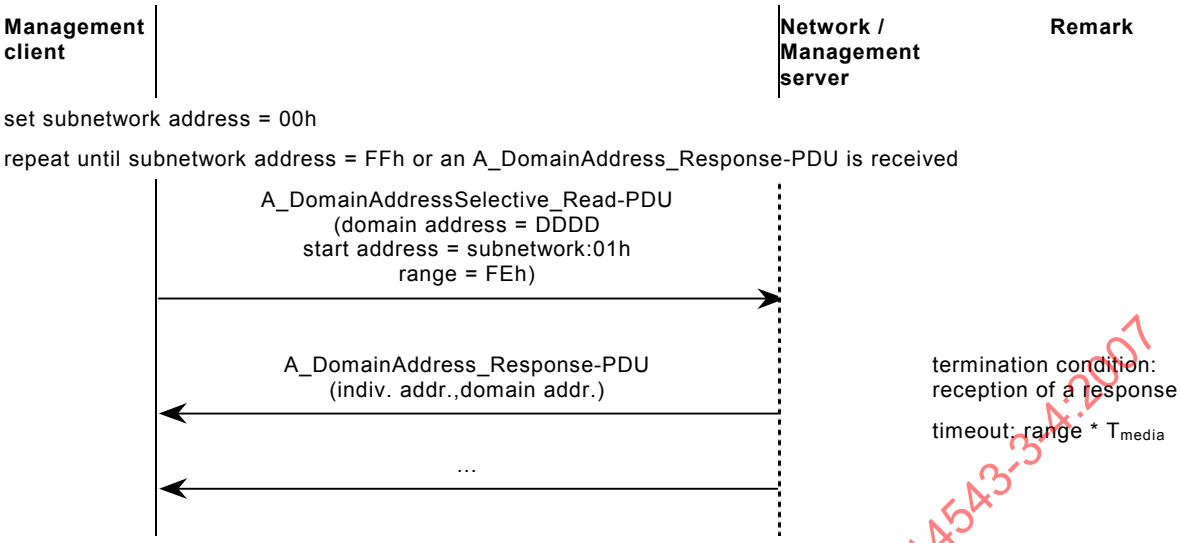
Therefore the management server shall check for the existence of every individual address with this domain address.

5.9.2 Management service used

The NM_DomainAddress_Scan procedure shall use the following management service:

– A_DomainAddressSelective_Read.

5.9.3 Sequence



5.9.4 Exception handling

The management client shall wait until the timeout period has expired. If no response is received during this time, the subnetwork address of the start address shall be incremented. If the subnetwork address exceeds FFh, the domain address DDDD shall be accepted as a free domain address.

If a response is received the management server shall obtain a new domain address from the set of available domain addresses. The subnetwork address of the start address shall be reset to 00h and the scanning procedure shall be executed until a free domain address is found.

5.10 NM_Router_Scan

5.10.1 Description

This network management procedure shall be used to determine which routers are installed in a network.

This procedure tries to build up a connection with each router. It shall collect all A_Disconnect-PDUs. All routers from which an A_Disconnect-PDU is received exist in the network.

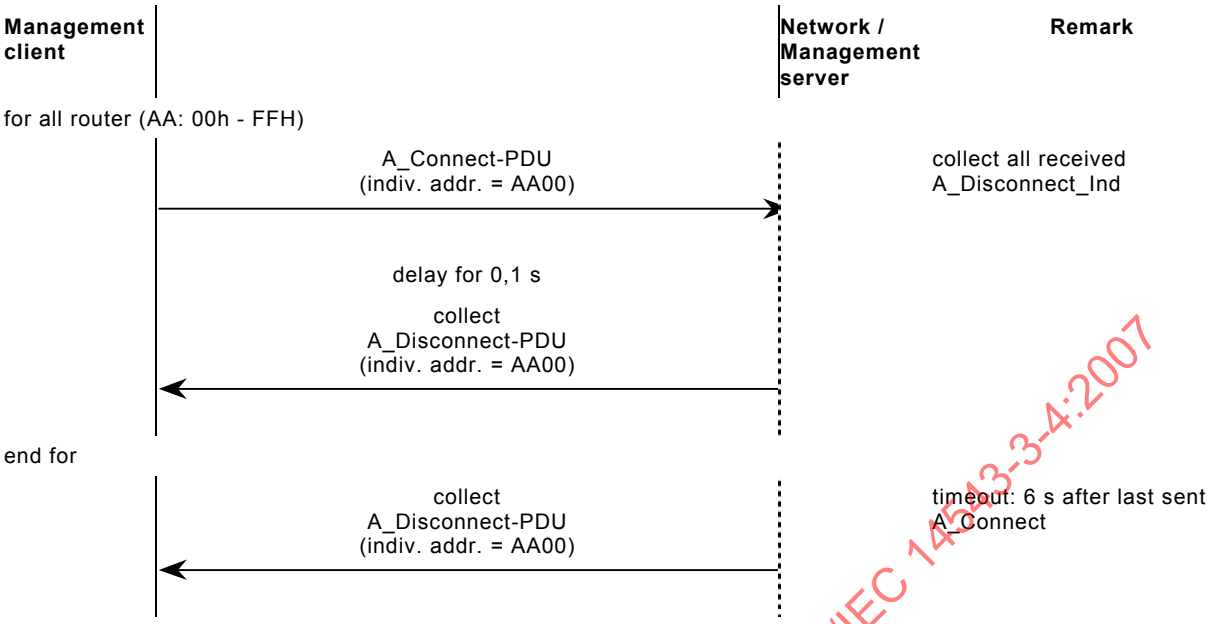
For this procedure the individual address of the routers and the domain address shall be configured.

5.10.2 Management service used

The NM_Router_Scan procedure shall use the following management service:

– A_Connect.

5.10.3 Sequence



5.11 NM_SubnetworkDevices_Scan

5.11.1 Description

This network management procedure shall be used to determine which devices exist on a line.

This procedure tries to build up a connection using every possible individual address in this line. It shall collect all A_Disconnect-PDUs. All the devices from which an A_Disconnect-PDU is received exist on the line.

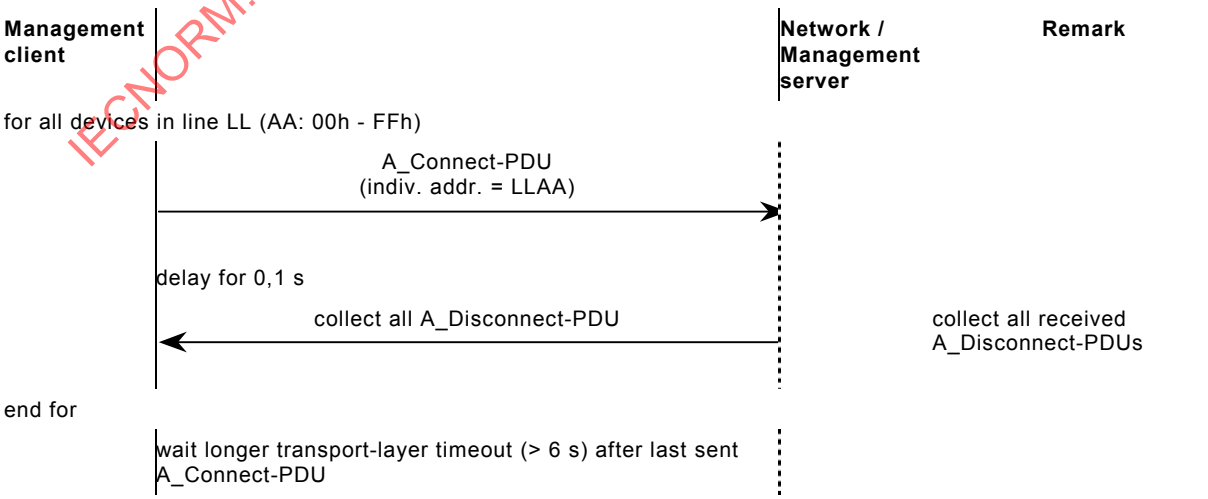
For this procedure the individual addresses of the routers used and the domain address shall be configured.

5.11.2 Management service used

The NM_SubnetworkDevices_Scan procedure shall use the following management service:

– A_Connect.

5.11.3 Sequence



5.12 NM_SubnetworkAddress_Read

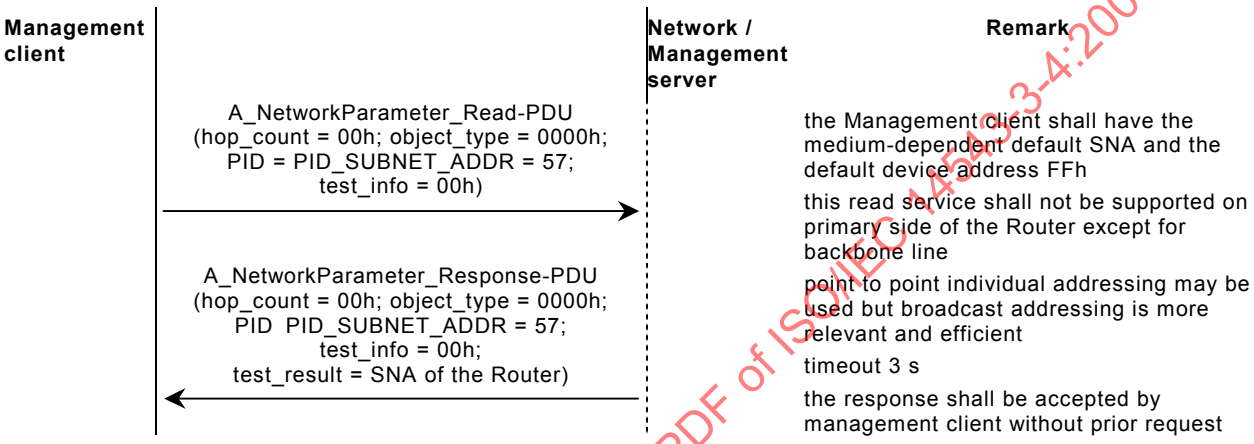
5.12.1 Description

This network management procedure shall be used to determine the address of the subnetwork to which the management client that performs this procedure is connected.

5.12.2 Management service used

The NM_SubnetworkAddress_Read procedure shall use the following management service:
– A_NetworkParameter_Read.

5.12.3 Sequence



5.12.4 Exception handling

The management client shall wait until the timeout period has expired. If no response is received during this time, it shall keep the current SNA.

In case of reception of multiple responses the last one shall be kept.

5.13 NM_IndividualAddress_Reset

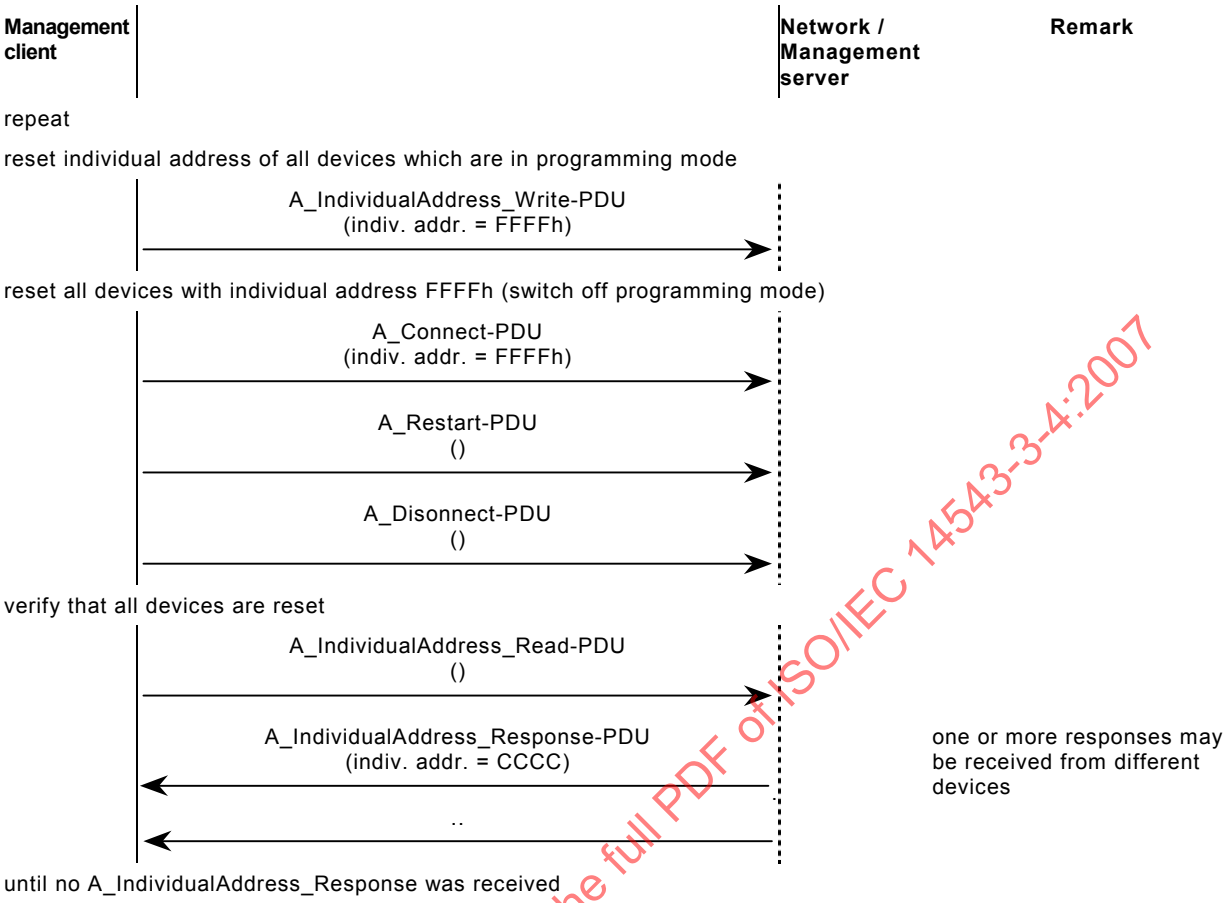
5.13.1 Description

This network management procedure shall be used to set back the individual address of one or more devices which are in programming mode, to the default individual address FFFFh.

5.13.2 Management services used

The NM_IndividualAddress_Reset procedure shall use the following management services:
– A_IndividualAddress_Write;
– A_Restart;
– A_IndividualAddress_Read;
– A_Connect;
– A_Disconnect.

5.13.3 Sequence



NOTE Do not evaluate any local confirms, or received telegrams, except the A_IndividualAddress_Read.Lcon and the received A_IndividualAddress_Response-PDU.

5.14 NM_IndividualAddress_Scan

5.14.1 Description

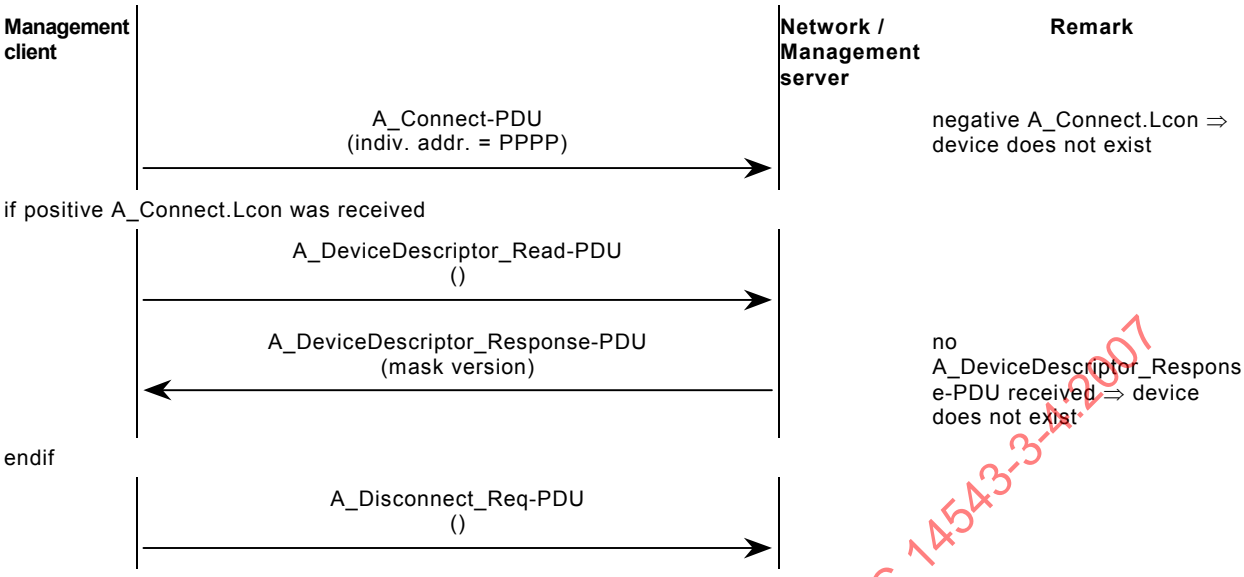
This network management procedure shall be used to check for the existence of a device with a given individual address.

5.14.2 Management services used

The NM_IndividualAddress_Scan procedure shall use the following management services:

- A_Connect;
- A_DeviceDescriptor_Read;
- A_Disconnect.

5.14.3 Sequence



5.14.4 Possible reactions

Device reacts on A_DeviceDescriptor_Read – connection-oriented ⇒ device with this individual address exists and supports connection-oriented communication.

Device sends a A_Disconnect-PDU and no A_DeviceDescriptor_Response-PDU ⇒ a device with this individual address exists, but it supports no connection-oriented communication.

5.15 NM_IndividualAddress_Check

5.15.1 Description

The procedure shall be used by a device or management client to check whether an individual address is occupied on the network.

5.15.2 Management services used

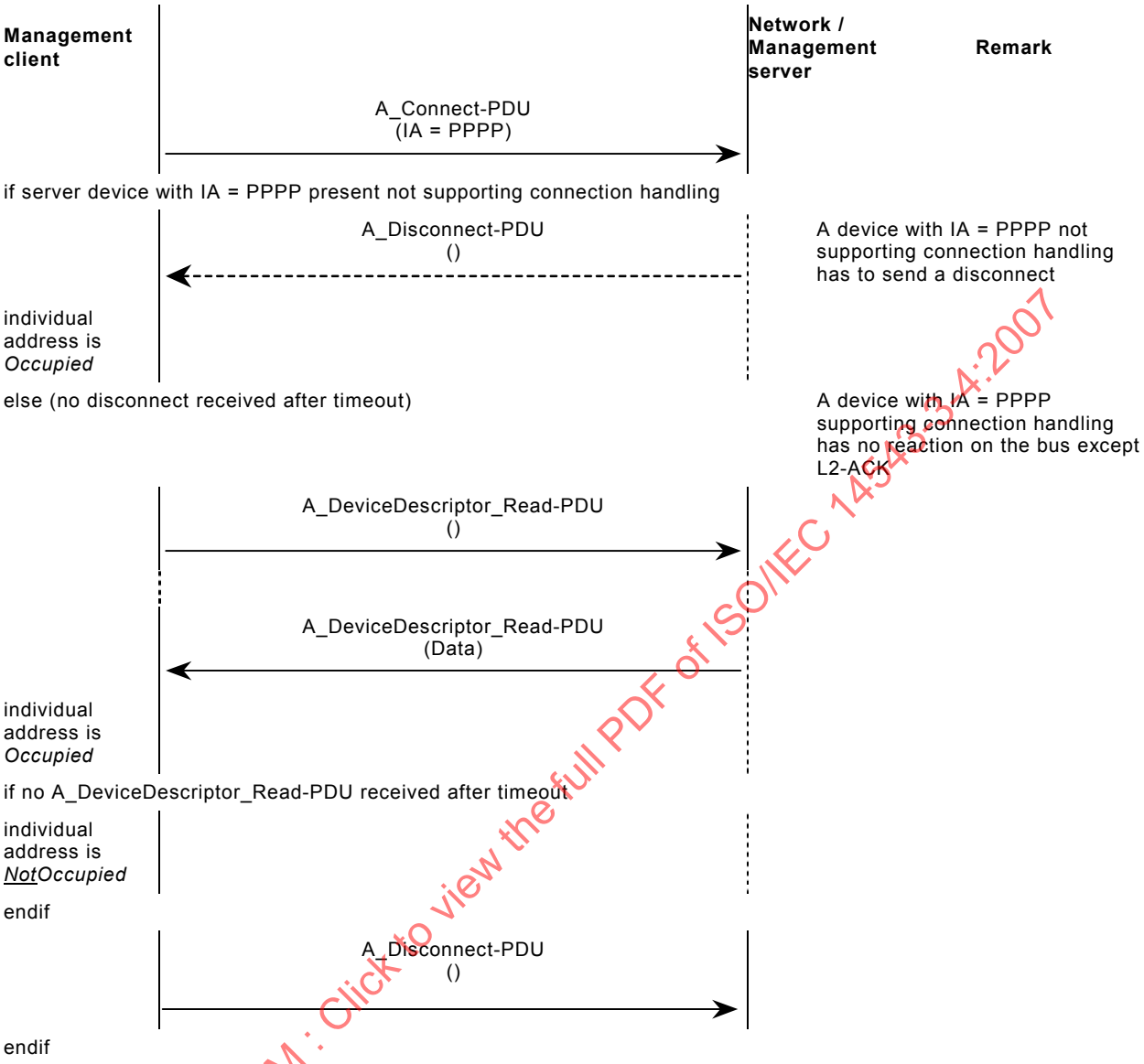
The NM_IndividualAddress_Check procedure shall use the following management services:

– A_DeviceDescriptor_Read;

– A_Connect;

– A_Disconnect.

5.15.3 Sequence



5.16 NM_IndividualAddress_Check_LocalSubnetwork

5.16.1 Description

The procedure shall be used by a management client to check whether a device with a given individual address is present on the subnetwork where the management client is itself located.

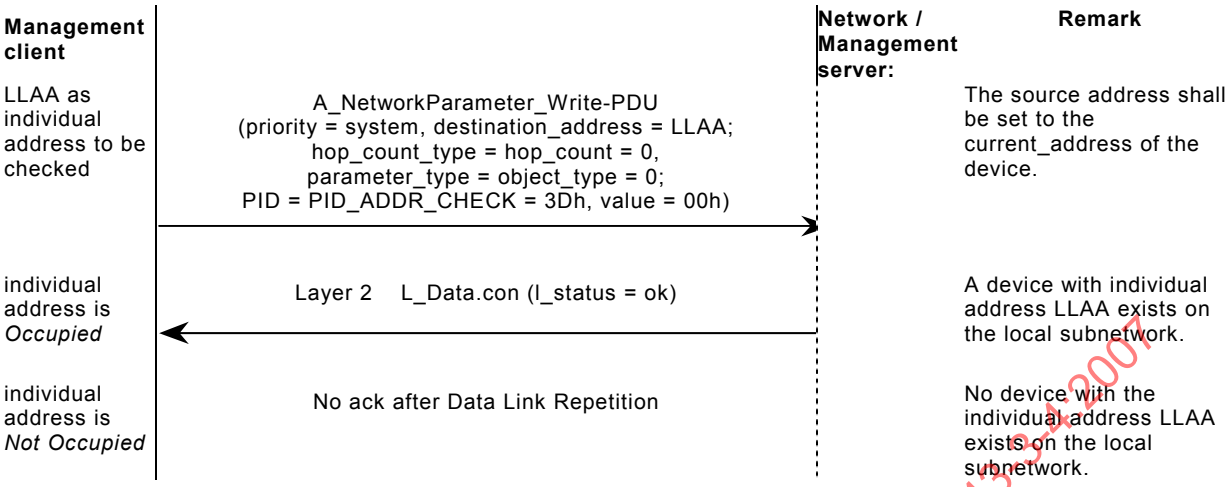
To check if an individual address LLAA (LL is the subnetwork address) is used by another device on a local subnetwork, a network procedure shall request a point-to-point message A_NetworkParameter_Write to this address LLAA.

5.16.2 Management service used

The NM_IndividualAddress_Check_LocalSubnetwork procedure shall use the following management service:

– A__NetworkParameter_Write.

5.16.3 Sequence



NOTE For some medium communication protocols, the flag l_satus can be set to value "not_ok", due to BUSY or NACK , after specific link layer repetitions. In this case, unsuccessful transmission shall be recovered by the constant individual address conflict detection procedure.

5.17 NM_GroupAddress_Check

5.17.1 Description

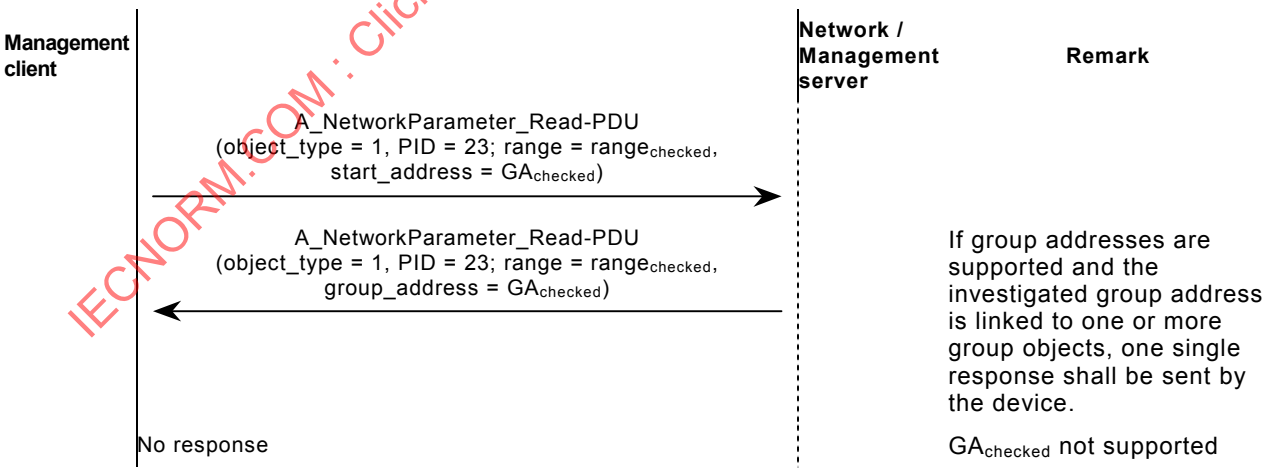
This management procedure shall be used by a management client to check whether a group address is used within a given address range on the network. To check for a single group address, the range shall be set to 1.

5.17.2 Management service used

The NM_GroupAddress_Check procedure shall use the following management service:

– A_NetworkParameter_Read.

5.17.3 Sequence



If the $range_{checked}$ equals one, only one single group address ($GA_{checked}$) shall be checked. The remote server either does not answer, if it does not support any group address within the range $GA_{checked}$ to $Ga_{checked}$ + $range_{checked}$ − 1, or it answers with one single response, if it has for one or more group addresses in the range one or more associations. So, even if one group address in the range is associated with more than one group object or if more than one group address is (multiple times) associated, it shall only reply with one answer.

The service shall be sent in broadcast mode and the response shall use point-to-point connectionless communication mode.

5.17.4 Exception handling

The management client shall wait until the timeout period has expired. This timeout period is the sum of the frame transmission time based on the medium used and the maximum time allowed for the device to process the frame.

If no response is received, the management client shall assume that $GA_{checked}$ is not used in the network.

5.18 NM_FunctionalBlock_Scan

5.18.1 Description

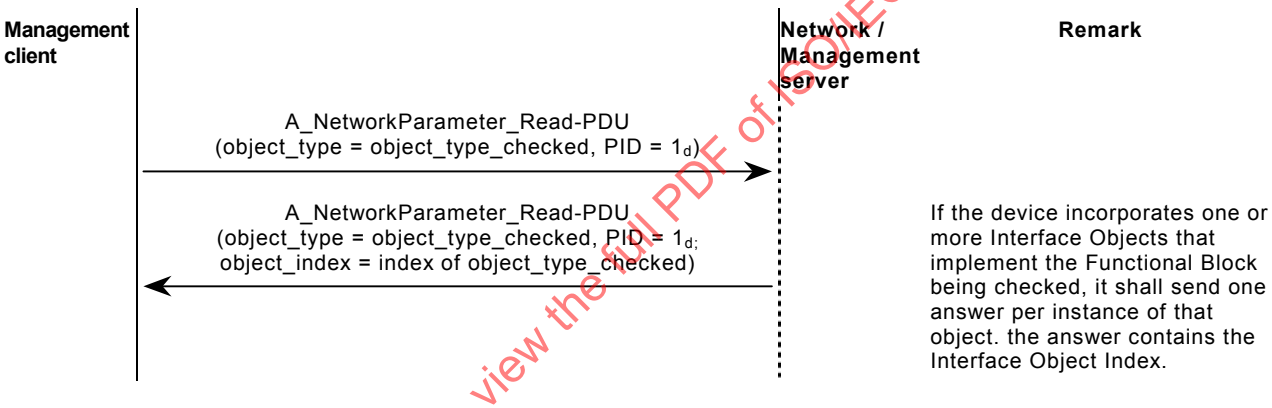
This management procedure shall be used by a management client to find if a Functional Block of a given type is available in the network.

5.18.2 Management service used

The NM_FunctionalBlock_Scan procedure shall use the following management service:

– A_NetworkParameter_Read.

5.18.3 Sequence



5.18.4 Exception handling

The management client shall wait until the timeout period has expired. This timeout period shall be the sum of the frame transmission time based on the medium used and the maximum time allowed for the device to process the frame.

If no response is received, the management client shall assume that the investigated functionality (Interface Object of given type) is not available in the network.

If one or more answers are received, the management client shall use the contained Interface Object Index(es) for further management.

6 Device management procedures

6.1 General

The device management procedures describe the rules and procedures for managing a single device. These procedures are device-dependent. Therefore a detailed knowledge of the device is required.

The knowledge of the resources (for example, volatile and non-volatile memory and control variables) is implemented in the management server. These resources can be implemented as interface objects, as defined in ISO/IEC 14543-3-3. The following paragraphs describe the general device management procedures. Some parameters depend on the actual BAU-type of the management server.

The device management procedures can be used to read out the state of a device, write parameters and to load or unload a device with an application.

6.2 General exception handling

In general, if an error is detected, the download shall be interrupted and an error message shall be raised.

If the exception handling for a device management procedure differs from this, this shall be stated in the description of the procedure.

6.3 DM_Connect

6.3.1 General Description

This device management procedure shall be used to establish a connection to a management server with a specific individual address and to check its existence by reading the mask version.

The connection is closed with a DM_Disconnect.

DM_Connect (flags)
flags bit 0 use connection-oriented / connectionless communication
0: connection-oriented communication
1: connectionless communication
All other bits are reserved. These shall be set to 0.
This shall be tested by the management client.

6.3.2 Procedure: DMP_Connect_RCo

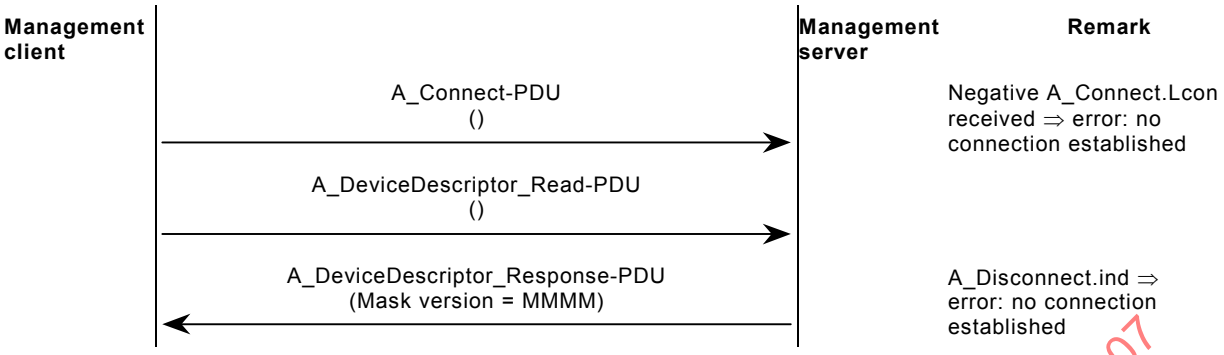
6.3.2.1 Description

This method shall use the connection-oriented remote communication.

6.3.2.2 Management services used

The DMP_Connect_RCo procedure shall use the following management services:
– A_Connect;
– A_DeviceDescriptor_Read.

6.3.2.3 Sequence



6.3.2.4 Exception handling

There are several error situations when a connection-oriented communication is being set up.

– If the T_Connect.reqtelegram is not acknowledged by a Link Layer ACKframe (negative A_Connect.Lcon) then the management server with the individual address does not exist or the system is not configured correctly. (e.g. coupler, domain address).

– If a T_Disconnect-telegram is sent out by the management server but no A_DeviceDescriptor-_Response, the device with the individual address exists. The reason for this may either be that the management server is already using another connection, doesn't support connection-oriented mode, or the timeout period has elapsed.

– If a T_Disconnect-telegram is sent out by the transport layer of the management client, then the system may not be configured correctly, or the management server doesn't exist.

– If more than one A_DeviceDescriptor_Response-PDU is received, there is more than one device with the target address.

If the above indicates the network may be configured incorrectly, then the network topology shall be checked. In all other cases, the DM_Connect may be repeated several times. If this procedure is not successful, the procedures depending on it shall be aborted.

6.3.3 Procedure: DMP_Connect_RCI

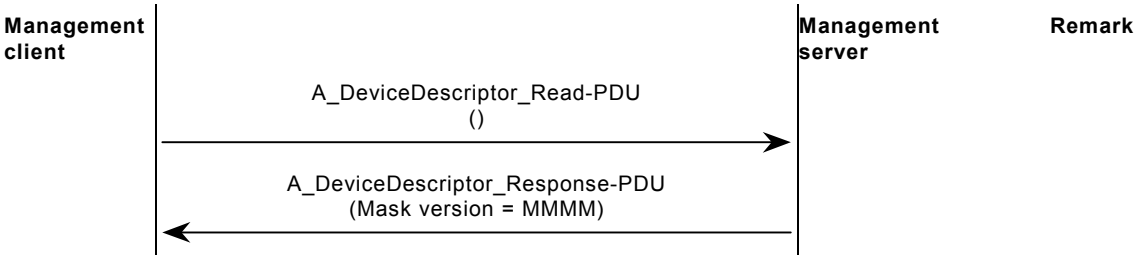
6.3.3.1 Description

This procedure shall use connectionless remote communication.

6.3.3.2 Management service used

The DMP_Connect_RCI procedure shall use the following management service:
– A_DeviceDescriptor_Read.

6.3.3.3 Sequence



6.3.3.4 Exception handling

There are several error situations when a connectionless communication is being set up.

– If no A_DeviceDescriptor_Response-PDU is received, the management server with the individual address does not exist or the network is not configured correctly.

– If more than one A_DeviceDescriptor_Response-PDU is received, there is more than one device with the target address.

If the above indicates the network may be configured incorrectly, then the network topology shall be checked. In all other cases, the DM_Connect may be repeated several times. If this procedure is not successful, the procedures depending on it shall be aborted.

6.4 DM_Disconnect

6.4.1 General description

This device management procedure is used to close a connection to the management server which was built up with DM_Connect.

A DM_Connect shall be executed before executing this management procedure.

DM_Disconnect (flags)

flags	All bits are reserved. These shall be set to 0. This shall be tested by the management client

6.4.2 Procedure: DMP_Disconnect_RCo

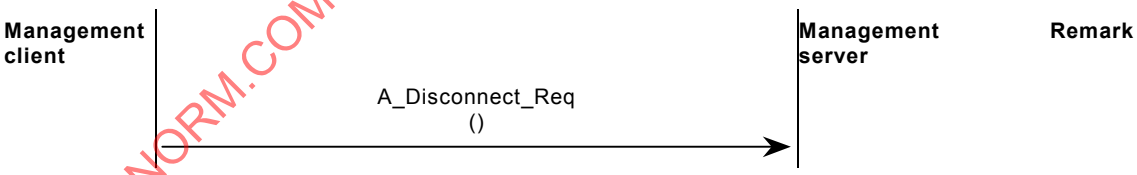
6.4.2.1 Description

This method shall use connection-oriented remote communication.

6.4.2.2 Management service used

The DMP_Disconnect_RCo procedure shall use the following management service:
A_Disconnect.

6.4.2.3 Sequence



6.4.2.4 Exception handling

The general exception handling is applicable.

6.4.3 Procedure: DMP_Disconnect_RCI

6.4.3.1 Description

This procedure is the complement of the DMP_Connect_RCI, as specified in 5.3.3.

NOTE This procedure has been set up for a management client implemented as software on a computer. Such a management client may keep track of devices with which it has built a connection. This procedure allows any user of the management client to indicate that he/she is no longer interested in the device, which allows the management client to update its state.

6.4.3.2 Management services used

None.

6.4.3.3 Sequence

No messages are exchanged on the medium.

6.4.3.4 Exception handling

No exceptions are possible to this procedure.

6.5 DM_Authorize

6.5.1 General description

This device management procedure shall be used to obtain access authorization. The authorization shall be executed only when it is required by the management server. DM_Connect shall be executed before executing this management procedure.

DM_Authorize (flags, keys)

flags	All bits are reserved. These shall be set to 0. This shall be tested by the management client.
key	key for authorization

6.5.2 Procedure: DMP_Authorize_RCo

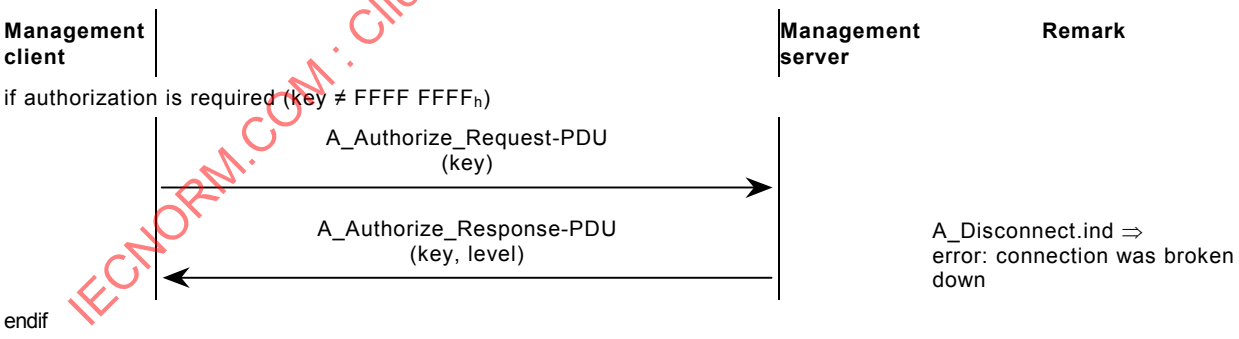
6.5.2.1 Description

This method shall use the connection-oriented remote communication.

6.5.2.2 Management service used

The DMP_Authorize_RCo procedure shall use the following management service:
A_Authorize_Request.

6.5.2.3 Sequence



6.5.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.6 DM_SetKey

6.6.1 General description

This device management procedure shall be used to set the key in the management server. A DM_Connect shall be executed before executing this management procedure.

DM_SetKey (flags, keys, level)

flags	All bits are reserved. These shall be set to 0. This shall be tested by the management client.
key	key for authorization
level	level for which the key is to be set

6.6.2 Procedure: DM_SetKey_RCo

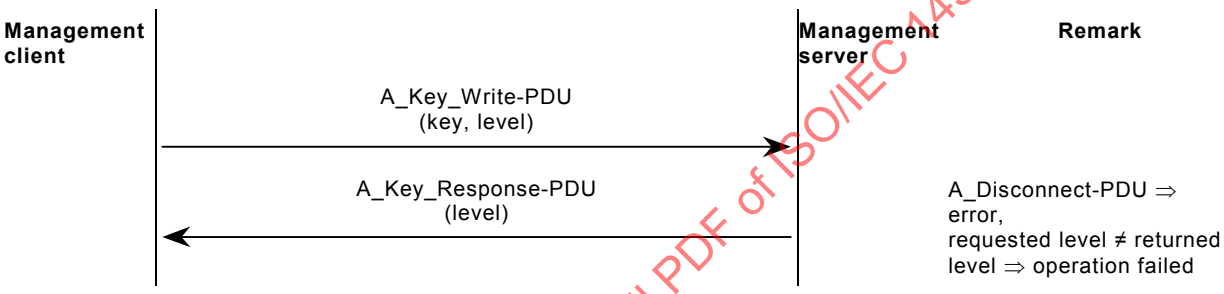
6.6.2.1 Description

This procedure shall use the connection-oriented remote communication.

6.6.2.2 Management service used

The DM_SetKey_RCo procedure shall use the following management service:
A_Key_Write.

6.6.2.3 Sequence



6.6.2.4 Exception handling

If the level returned in A_Key_Response-PDU is not the same as in the A_Key_Write-PDU, the operation was not successful. Probably an authorization is required.

6.7 DM_Restart

6.7.1 General description

This device management procedure shall be used to execute a reset of the management server. The transport layer connection of the management server is broken down. A DM_Restart shall be followed by an explicit DM_Disconnect. A new connection can be established with the DM_Connect management procedure.

A DM_Connect shall be executed before executing this management procedure.

DM_Restart (flags)

flags	All bits are reserved. These shall be set to 0. This shall be tested by the management client.

6.7.2 Procedure: DM_Restart_RCo

6.7.2.1 Description

This method shall use the connection-oriented remote communication.

6.7.2.2 Management service used

The DMP_Restart_RCo procedure shall use the following management service:

A_Restart

6.7.2.3 Sequence

Management client		Management server	Remark
	A_Restart-PDU () →		
	abort the connection of the client side transport layer		

6.7.2.4 Exception handling

All telegrams sent out by the management server shall be ignored, except negative transport layer confirmations.

6.8 DM_Delay

6.8.1 Description

This device management procedure shall be used to wait a specified time before starting the next action.

NOTE This procedure is thought for a management client implemented as software on a computer. This procedure allows any user of the management client to indicate that it should wait for a certain period before doing the next action.This procedure does not provoke any communication on the bus system.

DM_Delay (flags, delay time)

delay time	Time in milliseconds
flags	All bits are reserved. These shall be set to 0. This shall be tested by the management client.

6.8.2 Procedure: DMP_Delay

6.8.2.1 Management services used

None.

6.8.2.2 Sequence

Management client		Management server	Remark
	delay for specified time		

6.8.2.3 Exception handling

The general exception handling of 5.2 is applicable.

6.9 DM_IndividualAddressRead

This device management procedure shall be used to read out the individual addresses of the local device, independent of the programming mode. For remote procedures, see 4.2.

A DM_Connect shall be executed before executing this management procedure.
DM_IndividualAddressRead (individual address)

individual address	contains the individual address of the device

6.10 DM_IndividualAddressWrite

This device management procedure shall be used to write the individual addresses of the local device, independent of the programming mode. For remote procedures, see 4.2.

A DM_Connect shall be executed before executing this management procedure.
DM_IndividualAddressWrite (individual address)
individual address contains the individual address of the device

6.11 DM_DomainAddressRead

This device management procedure shall be used to read the domain address of the local device, independent of the programming mode. For remote procedures, see 4.2.

A DM_Connect shall be executed before executing this management procedure.
DM_DomainAddressRead (DomainAddress)
DomainAddress contains the domain addresses of the device

6.12 DM_DomainAddressWrite

This device management procedure shall be used to write the domain addresses of the local device, independent of the programming mode. For remote procedures, see 4.2.

A DM_Connect shall be executed before executing this management procedure.

DM_DomainAddressWrite (DomainAddress)

DomainAddress contains the domain addresses of the device

6.13 DM_ProgMode_Switch

6.13.1 Description

This device management procedure switches the programming mode of the device.

A DM_Connect shall be executed before executing this management procedure.

DM_ProgMode_Switch (flags, mode)

flags	All bits are reserved. These shall be set to 0. This shall be tested by the management client.
mode	0: switch programming mode off 1: switch programming mode on

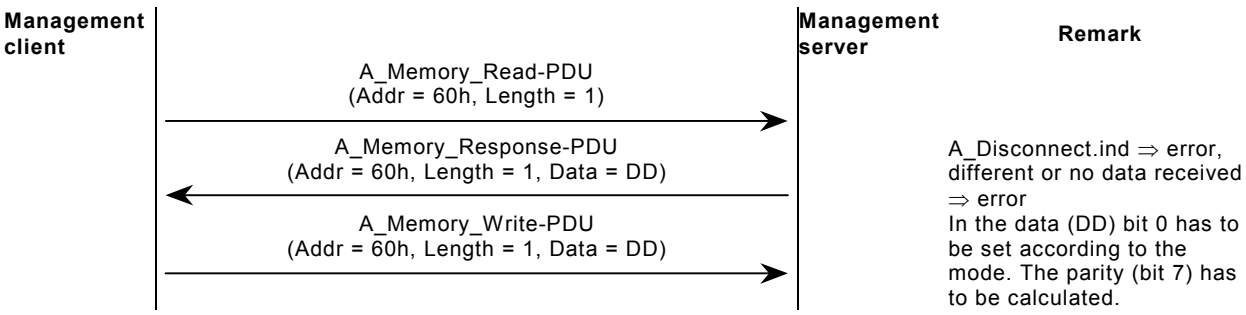
6.13.2 Procedure: DMP_ProgModeSwitch_RCo

This method shall use the connection-oriented remote communication. The state of the programming mode is located at memory address 60h.

6.13.2.1 Management services used

The DMP_ProgModeSwitch_RCo procedure shall use the following management services:
– A_Memory_Read
– A_Memory_Write

6.13.2.2 Sequence



6.13.2.3 Exception handling

The general exception handling of 5.2 is applicable.

6.14 DM_GroupObject_Link_Read

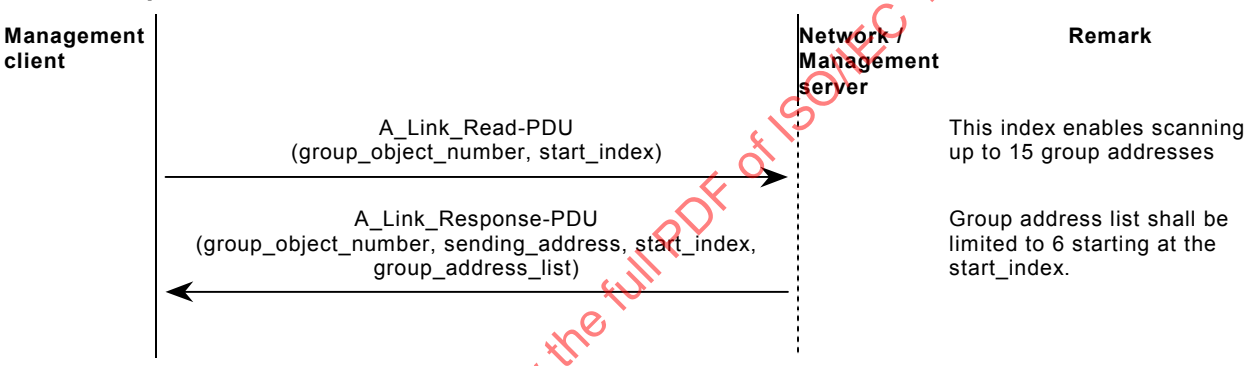
6.14.1 Description

This device management procedure shall be used by the management client to read the list of the links attached to a group object in a management server. It shall indicate the start_index from which group addresses shall be read in the list of links. The management server shall respond with the updated list of links on the group object and, if any, the sending group address.

6.14.2 Management service used

The DM_GroupObject_Link_Read procedure shall use the following management service:
– A_Link_Read

6.14.3 Sequence



The management client shall use the minimal value “one” as value for the start_index.

A management server may not support another value for the start_index than one. The positive response by the management server shall contain the requested group_object_number, the index of the sending address, the start_address and the list of up to six group addresses associated to the questioned group object.

If the management server has no sending group address linked to the tested Group Address, it shall indicate this by setting the value of the parameter sending_address to null in the A_Link_Response-PDU; otherwise that value shall be the value of the index in the group address list that contains the sending group address.

The service shall use point-to-point connectionless communication mode.

6.14.4 Exception handling

Any failure to execute the request by the management server, like non-existing group_object_number, invalid reserved bits in the received A_Link_Read-PDU or invalid start_index, shall be answered by the negative response that shall be indicated by a value null for the start_index and no group addresses contained in the group address list.

6.15 DM_GroupObject_Link_Write

6.15.1 Description

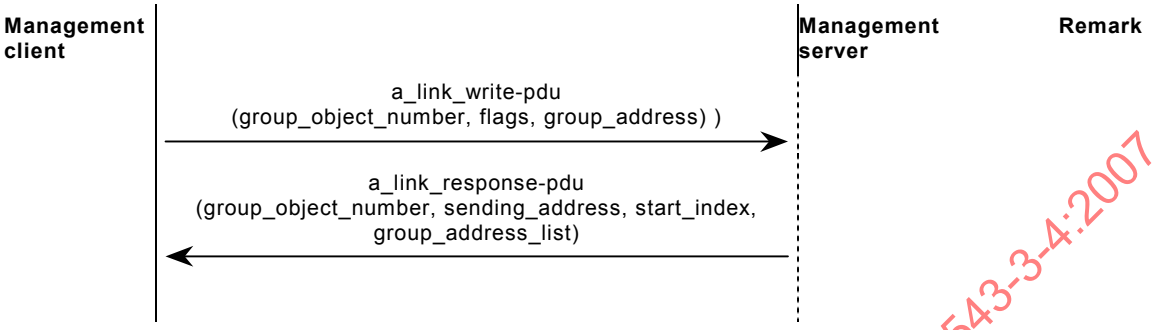
This device management procedure shall be used to add a link between a given group object and the group address of management server. It shall also be used to delete an existing link.

6.15.2 Management services used

The DM_GroupObject_Link_Write procedure shall use the following management services:

– A_Link_Write
– A_Link_Read

6.15.3 Sequence



flags s bit 0

In case the action is “add”, this bit shall indicate whether the newly linked group address shall be the sending group address for the group object or not.

0: This group address shall *not be the sending* group address for this group object.

1: This group address shall be *the sending* group address for this group object.

In case the action is “delete”, the value of this bit shall be zero.

d bit 1

This bit shall specify the action to be done.

0: *Add* the contained group address to the list of group addresses associated with this group object.

1: *Delete* the contained group address from the list of group addresses associated with this group object.

All other bits are reserved. These shall be set to 0.
This shall be tested by the management client.

The response shall indicate the updated group address list linked to the group object, and sending address.

The service shall use point-to-point connectionless communication mode.

6.15.4 Exception handling

Any failure to the execution of the request at network management level (such as table full, non-existent group_object_number, invalid reserved bits, unknown start_index and others) shall be answered by a negative response.

6.16 DM_MemWrite

6.16.1 General description

This device management procedure writes a contiguous block of data to the specified memory addresses.

The data shall be located either in the management control or in the data block. Only the data that is specified in the data block is written. Depending on the flag the data is verified immediately. If the deviceStartAddress is higher than the deviceEndAddress this management procedure shall be skipped.

A DM_Connect shall be executed before executing this management procedure.

DM_MemWrite (flags, dataBlockStartAddress, deviceStartAddress, deviceEndAddress, data)

flags	bit 0 location of data 0: in data block 1: in management procedure bit 1 verify enabled / disabled 0: disabled 1: enabled All other bits are reserved. These shall be set to 0. This shall be tested by the management client.
dataBlockStartAddress	specifies the address where the data are located in the data block. If the data are located in the management procedure, this field is set to 0.
deviceStartAddress	address of first memory octet that is written by this management procedure.
deviceEndAddress	address of the last octet that is written by this management procedure.
data	the data that are transferred by this management procedure. The data can be located in the data block or in the management procedure.

Data format

code	flags	dataBlockStartAddress	deviceStartAddress	deviceEndAddress	reserved / data
20h	*FF*h	*BBBB BBBB*	*SSSS SSSS*	*EEEE EEEE*	00h / *DD*
1 octet	1 octet	4 octet	4 octet	4 octet	18 octet
NOTE Values in italics are examples.					

6.16.2 Procedure: DMP_MemWrite_RCo

6.16.2.1 Description

This method shall use the connection-oriented remote communication. The verify mode of the management server is not used.

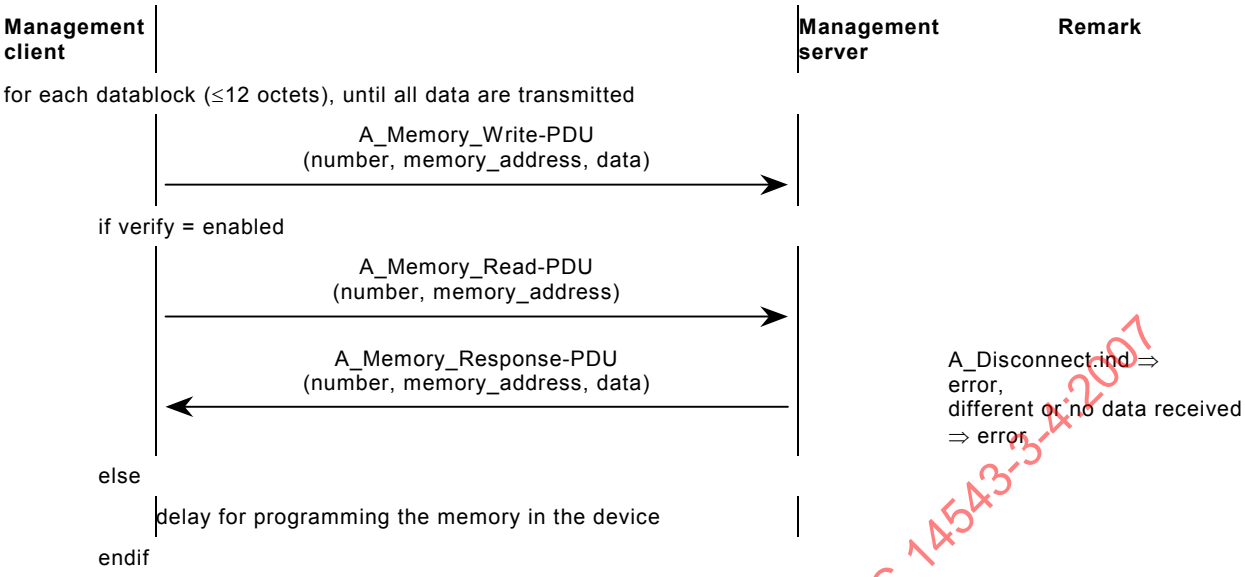
6.16.2.2 Management services used

The DMP_MemWrite_RCo procedure shall use the following management services:

– A_Memory_Write

– A_Memory_Read

6.16.2.3 Sequence



NOTE The delay for programming the memory in the device depends on the management server and on the number of octets written.

6.16.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.16.3 Procedure: DMP_MemWrite_RCoV

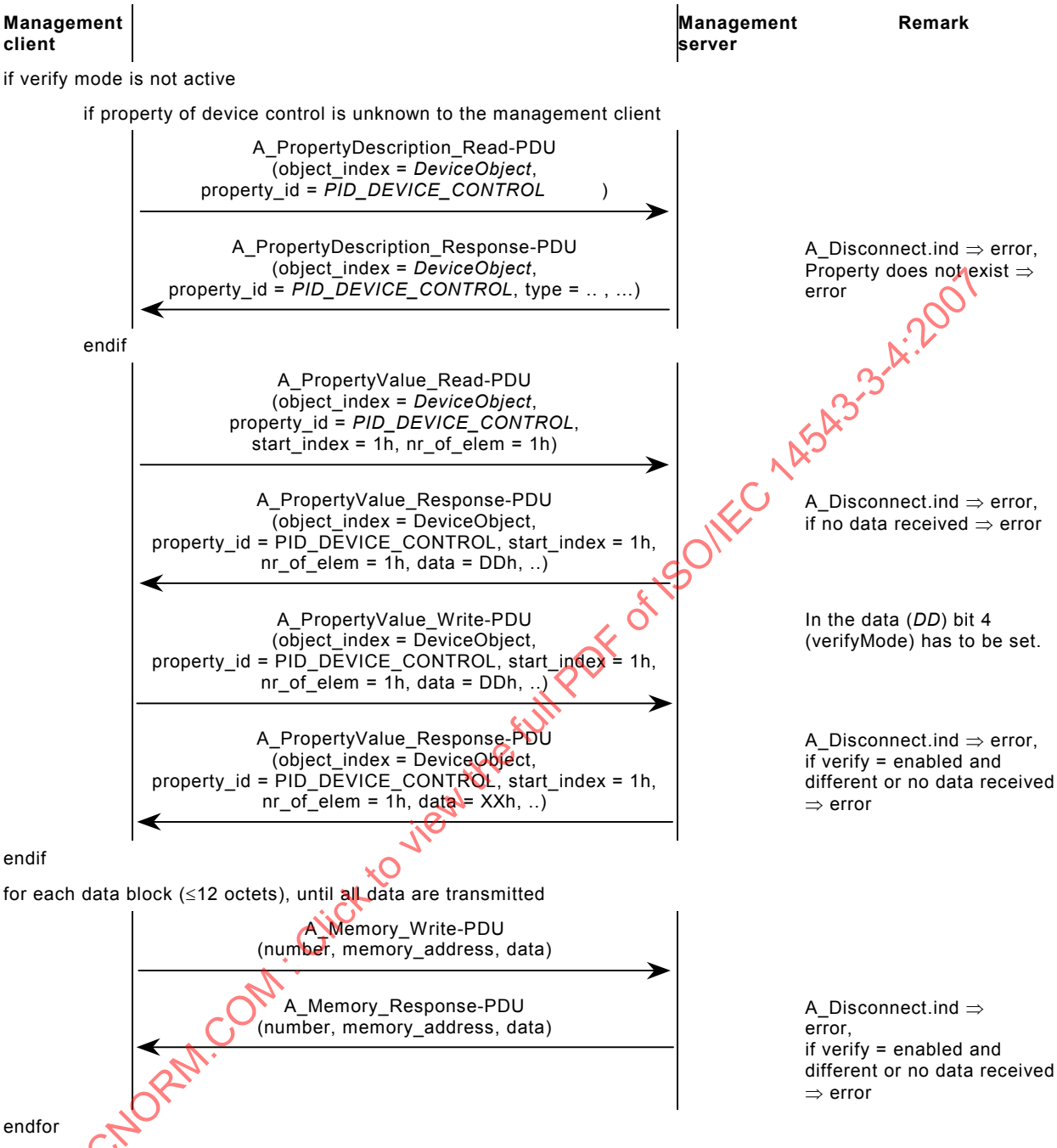
6.16.3.1 Description

This method shall use the connection-oriented remote communication. The verify mode of the management server is used.

6.16.3.2 Management service used

The DMP_MemWrite_RCoV procedure shall use the following management service:
A_Memory_Write

6.16.3.3 Sequence



6.16.3.4 Exception handling

The general exception handling of 5.2 is applicable.

6.17 DM_MemVerify

6.17.1 General description

This device management procedure shall read a contiguous block of memory and compare it with the specified data. The data shall be located either in the management control or in the data block. Only the data that are specified in the data block shall be compared. If the deviceStartAddress is higher than the deviceEndAddress this management procedure shall be skipped.

A DM_Connect shall be executed before executing this management procedure.

DM_MemVerify		(flags, dataBlockStartAddress, deviceStartAddress, deviceEndAddress, data)
flags	bit 0	location of data 0: in data block 1: in management control
		All other bits are reserved. These shall be set to 0. This shall be tested by the management client.
dataBlockStartAddress		specifies the address where the data are located in the data block. If the data are located in the management procedure, this field is set to 0.
deviceStartAddress		address of first memory octet that is compared by this management procedure
deviceEndAddress		address of the last octet that is compared by this management procedure
data		the data that are compared by this management procedure. The data can be located in the data block or in the management procedure.

6.17.2 Procedure: DMP_MemVerify_RCo

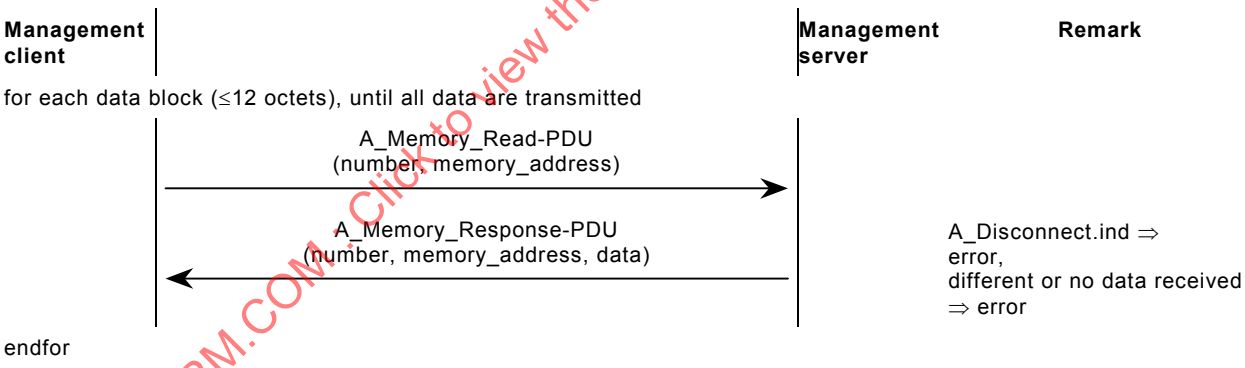
6.17.2.1 Description

This method shall use the connection-oriented remote communication.

6.17.2.2 Management service used

The DMP_MemVerify_RCo procedure shall use the following management service:
A_Memory_Read

6.17.2.3 Sequence



6.17.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.18 DM_MemRead

6.18.1 General description

This device management procedure shall read a contiguous block of memory and store it in the data block. If the deviceStartAddress is higher than the deviceEndAddress this management procedure shall be skipped.

A DM_Connect shall be executed before executing this management procedure.

DM_MemRead (flags, dataBlockStartAddress, deviceStartAddress, deviceEndAddress, data)

flags	bit 0 location of data 0: in data block 1: - All other bits are reserved. These shall be set to 0. This shall be tested by the management client.
dataBlockStartAddress	specifies the address where the data are located in the data block. If the data are located in the management procedure, this field is set to 0
deviceStartAddress	address of first memory octet that is read by this management procedure
deviceEndAddress	address of the last octet that is read by this management procedure
data	the data that are read by this management procedure. The data are stored in the data block.

6.18.2 Procedure: DMP_MemRead_RCo

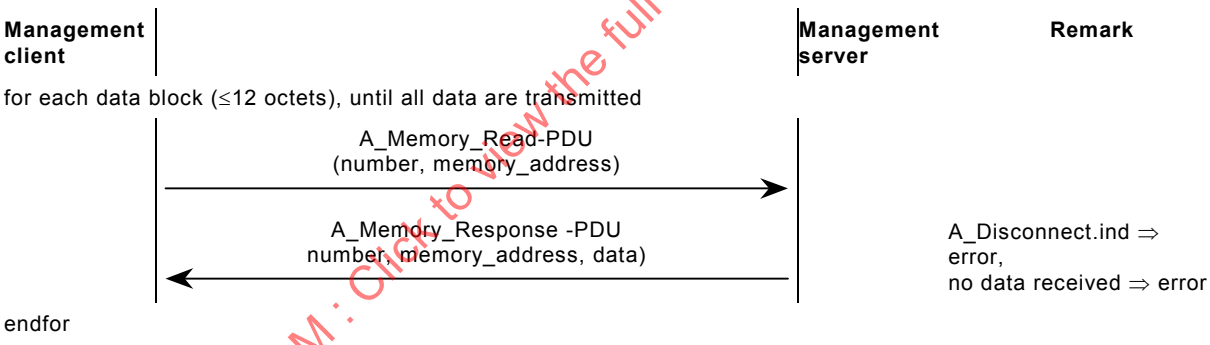
6.18.2.1 Description

This method shall use the connection-oriented remote communication.

6.18.2.2 Management service used

The DMP_MemRead_RCo procedure shall use the following management service:
A_Memory_Read

6.18.2.3 Sequence



6.18.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.19 DM_UserMemWrite

6.19.1 General description

This device management procedure shall write a contiguous block of data to the specified memory addresses of the user memory in the management server. The data shall be located either in the management control or in the data block. Only the data that are specified in the data block shall be written. Depending on the flag, the data shall be verified immediately. If the deviceStartAddress is higher than the deviceEndAddress this management procedure shall be skipped.

A DM_Connect shall be executed before executing this management procedure.

DM_UserMemWrite (flags, dataBlockStartAddress, deviceStartAddress, deviceEndAddress, data)

flags	bit 0 location of data 0: in data block 1: in management procedure bit 1 verify enabled / disabled 0: disabled 1: enabled All other bits are reserved. These shall be set to 0. This shall be tested by the management client.
dataBlockStartAddress	specifies the address where the data are located in the data block. If the data are located in the management procedure, this field is set to 0.
deviceStartAddress	address of first user memory octet that is written by this management procedure
deviceEndAddress	address of the last user memory octet that is written by this management procedure
data	the data that are transferred by this management procedure. The data can be located in the data block or in the management procedure.

6.19.2 Procedure: DMP_UserMemWrite_RCo

6.19.2.1 Description

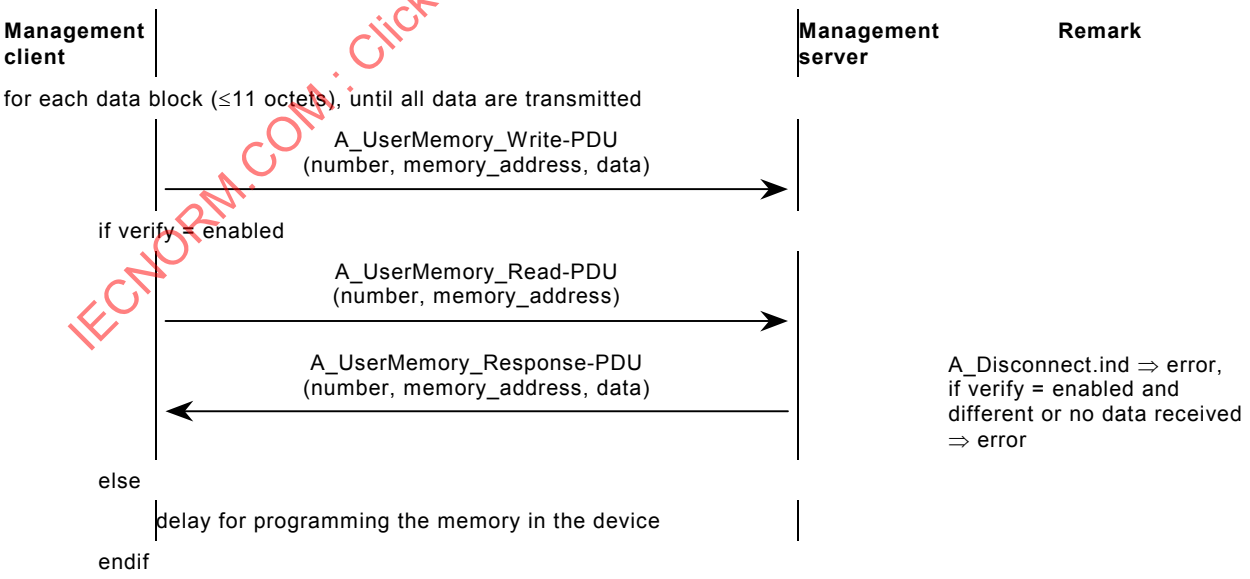
This method shall use the connection-oriented remote communication. The verify mode of the management server is not used.

6.19.2.2 Management services used

The DMP_UserMemWrite_RCo procedure shall use the following management services:
A_UserMemory_Write

A_UserMemory_Read

6.19.2.3 Sequence



NOTE The delay for programming the memory in the device depends on the management server and on the amount of written octets.

6.19.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.19.3 Procedure: DMP_UserMemWrite_RCoV

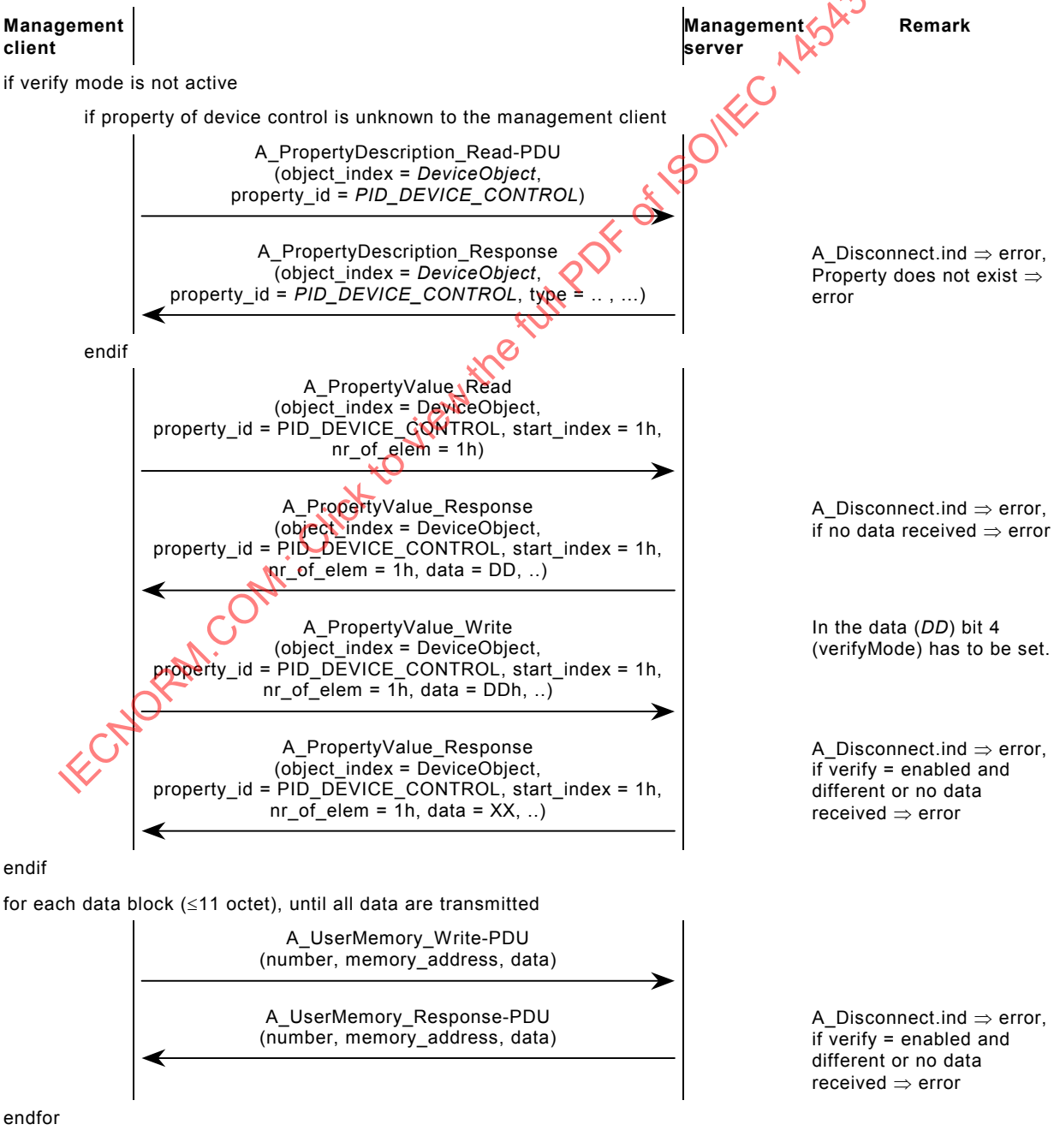
6.19.3.1 Description

This method shall use the connection-oriented remote communication. The verify mode of the management server is used.

6.19.3.2 Management service used

The DMP_UserMemWrite_RCoV procedure shall use the following management service:
A_UserMemory_Write

6.19.3.3 Sequence



6.19.3.4 Exception handling

The general exception handling of 5.2 is applicable.

6.20 DM_UserMemVerify

6.20.1 General description

This device management procedure shall read a contiguous block of user memory in the management server and compare it with the specified data. The data shall be located either in the management control or in the data block. Only the data that are specified in the data block shall be compared. If the deviceStartAddress is higher than the deviceEndAddress this management procedure shall be skipped.

A DM_Connect shall be executed before executing this management procedure.

DM_UserMemVerify (flags, dataBlockStartAddress, deviceStartAddress, deviceEndAddress, data)

flags	bit 0 location of data 0: in data block 1: in management control All other bits are reserved. These shall be set to 0. This shall be tested by the management client.
dataBlockStartAddress	specifies the address where the data are located in the data block. If the data are located in the management procedure, this field is set to 0.
deviceStartAddress	address of first user memory octet that is compared by this management procedure
deviceEndAddress	address of the last user memory octet that is compared by this management procedure
data	the data that are compared by this management procedure. The data can be located in the data block or in the management procedure.

6.20.2 Procedure: DMP_UserMemVerify_RCo

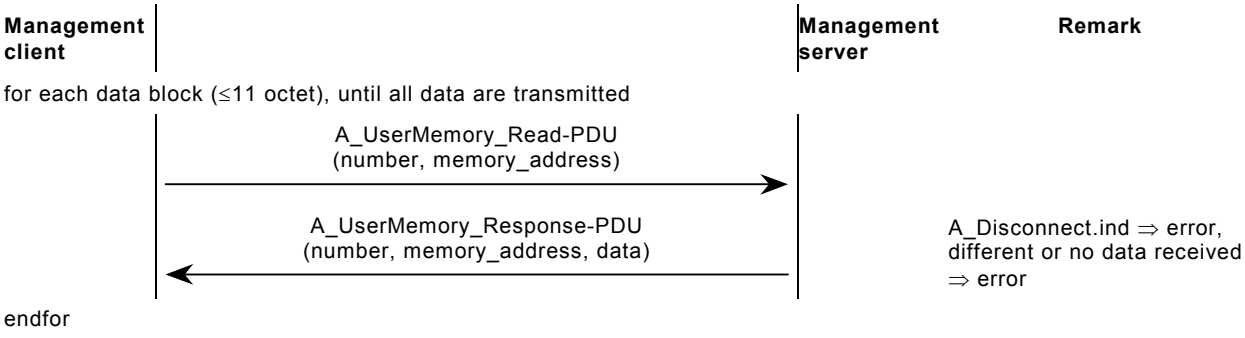
6.20.2.1 Description

This method shall use the connection-oriented remote communication.

6.20.2.2 Management service used

The DMP_UserMemVerify_RCo procedure shall use the following management service:
A_Memory_Read

6.20.2.3 Sequence



6.20.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.21 DM_UserMemRead

6.21.1 General description

This device management procedure shall read a contiguous block of memory in the management server and store it in the data block. If the deviceStartAddress is higher than the deviceEndAddress, this management procedure shall be skipped.

A DM_Connect shall be executed before executing this management procedure.

DM_UserMemRead (flags, dataBlockStartAddress, deviceStartAddress, deviceEndAddress, data)

flags	bit 0 location of data 0: in data block 1: - All other bits are reserved. These shall be set to 0. This shall be tested by the management client.
dataBlockStartAddress	specifies the address where the data are located in the data block.
deviceStartAddress	address of first memory octet that is read by this management procedure
deviceEndAddress	address of the last octet that is read by this management procedure
data	the data that are read by this management procedure. The data are stored in the data block.

6.21.2 Procedure: DMP_UserMemRead_RCo

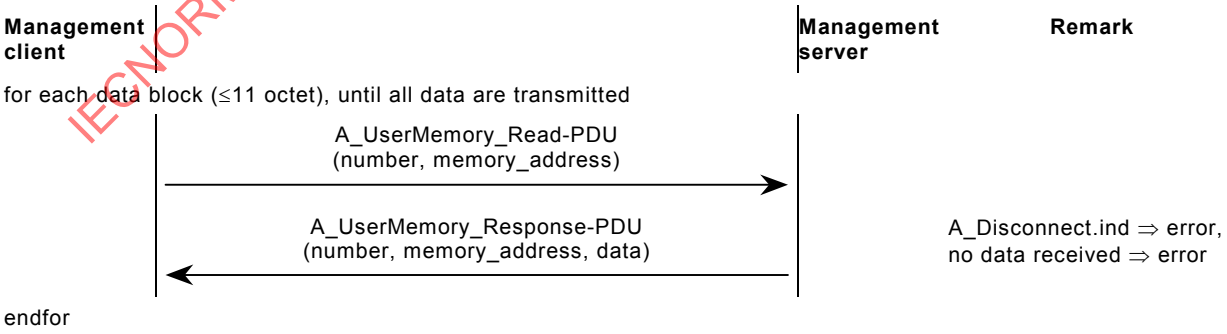
6.21.2.1 Description

This method shall use the connection-oriented remote communication.

6.21.2.2 Management service used

The DMP_UserMemRead_RCo procedure shall use the following management service:
A_UserMemory_Read

6.21.2.3 Sequence



6.21.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.22 DM_InterfaceObjectWrite

6.22.1 General description

This device management procedure shall write to the specified property value of an interface object. The data shall be located either in the management control or in the data block. Depending on the flag, the data shall be verified immediately. The interface object can be addressed via the object type and index (e.g. first object of type “polling master”) or via the object index independent of the type.

A DM_Connect shall be executed before executing this management procedure.

DM_InterfaceObjectWrite (flags, dataBlockStartAddress, objectType, objectIndex, property_id, startIndex, noElements, data)

flags	bit 0: location of data 0: in data block 1: in management control bit 1: verify enabled / disabled 0: disabled 1: enabled bit 2: address mode 0: address via object type / index 1: address via object index All other bits are reserved. These shall be set to 0. This shall be tested by the management client.
dataBlockStartAddress	specifies the address where the data are located in the data block. If the data are located in the management procedure, this field shall be set to 0.
objectType	type of the interface object
objectIndex	index of the interface object of one type. This index shall start counting from 0.
property_id	identifier of the property
startIndex	start element of the property
noElements	number of elements that are transferred
data	the data that are transferred by this management procedure. The data can be located in the data block or in the management procedure.

6.22.2 Procedure: DMP_InterfaceObjectWrite_R

6.22.2.1 Description

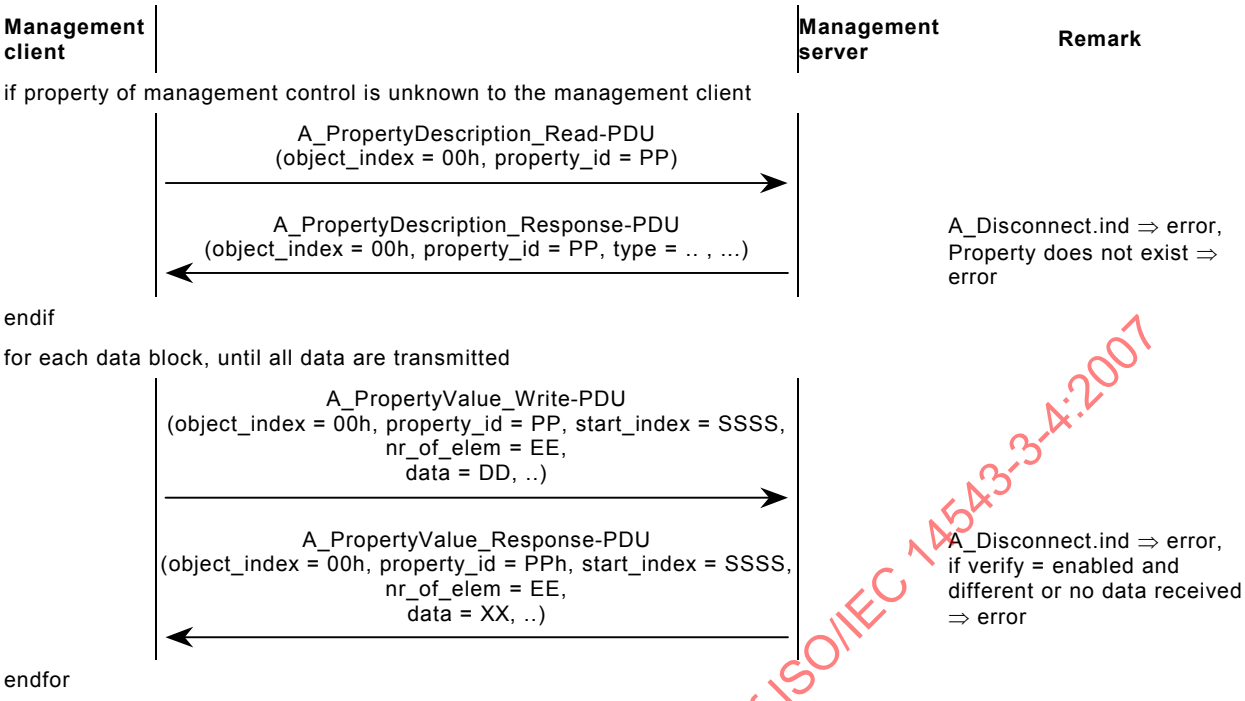
This method shall use the connection-oriented or connectionless remote communication.

6.22.2.2 Management services used

The DMP_InterfaceObject_R procedure shall use the following management services:
A_PropertyDescription_Read

A_PropertyValue_Write

6.22.2.3 Sequence



6.22.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.23 DM_InterfaceObjectVerify

6.23.1 General description

This device management procedure shall read the property value of an interface object and compare it with the specified data. The data shall be located either in the management control or in the data block. The interface object can be addressed via the object type and index (e.g., first object of type “polling master”) or via the object index independent of the type.

A DM_Connect shall be executed before executing this management procedure.

DM_InterfaceObjectVerify (flags, dataBlockStartAddress, objectType, objectIndex, property_id, startIndex, noElements, data)

flags	bit 0: location of data 0: in data block 1: in management control bit 2: address mode 0: address via object type / index 1: address via object index All other bits are reserved. These shall be set to 0. This shall be tested by the management client.
dataBlockStartAddress	specifies the address where the data are located in the data block. If the data are located in the management procedure, this field shall be set to 0.
objectType	type of the interface object
objectIndex	index of the interface object of one type. This index shall start counting from 0.
property_id	identifier of the property
startIndex	start element of the property
noElements	number of elements, which are compared
data	the data that are compared by this management procedure. The data can be located in the data block or in the management procedure.

6.23.2 Procedure: DMP_InterfaceObjectVerify_R

6.23.2.1 Description

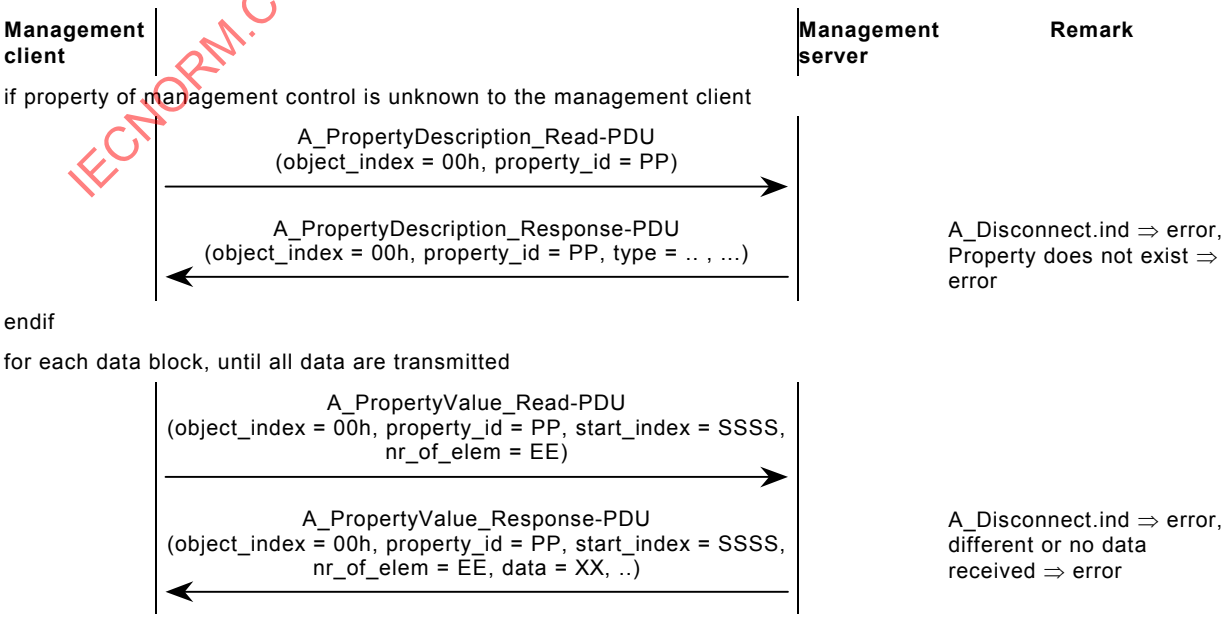
This method shall use the connection-oriented or connectionless remote communication.

6.23.2.2 Management services used

The DMP_InterfaceObjectVerify_R procedure shall use the following management services:
A_PropertyDescription_Read

A_PropertyValue_Read

6.23.2.3 Sequence



6.23.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.24 DM_InterfaceObjectRead

6.24.1 General description

This device management procedure reads the property value of an interface object and stores it in the data block. The interface object can be addressed via the object type and index (e.g., first object of type “polling master”) or via the object index independent of the type.

A DM_Connect shall be executed before executing this management procedure.

DM_InterfaceObjectRead	(flags, dataBlockStartAddress, objectType, objectIndex, property_id, startIndex, noElements, data)	
flags	bit 0:	location of data
		0: in data block 1: -
	bit 2:	address mode
		0: address via object type / index 1: address via object index
	All other bits are reserved. These shall be set to 0. This shall be tested by the management client.	
dataBlockStartAddress	specifies the address where the data are located in the data block. If the data are located in the management procedure, this field shall be set to 0.	
objectType	type of the interface object	
objectIndex	index of the interface object of one type. This index shall start counting from 0.	
property_id	identifier of the property	
startIndex	start element of the property	
noElements	number of elements that are read	
data	the data that are read by this management procedure. The data shall be stored in the data block.	

6.24.2 Procedure: DMP_InterfaceObjectRead_R

6.24.2.1 Description

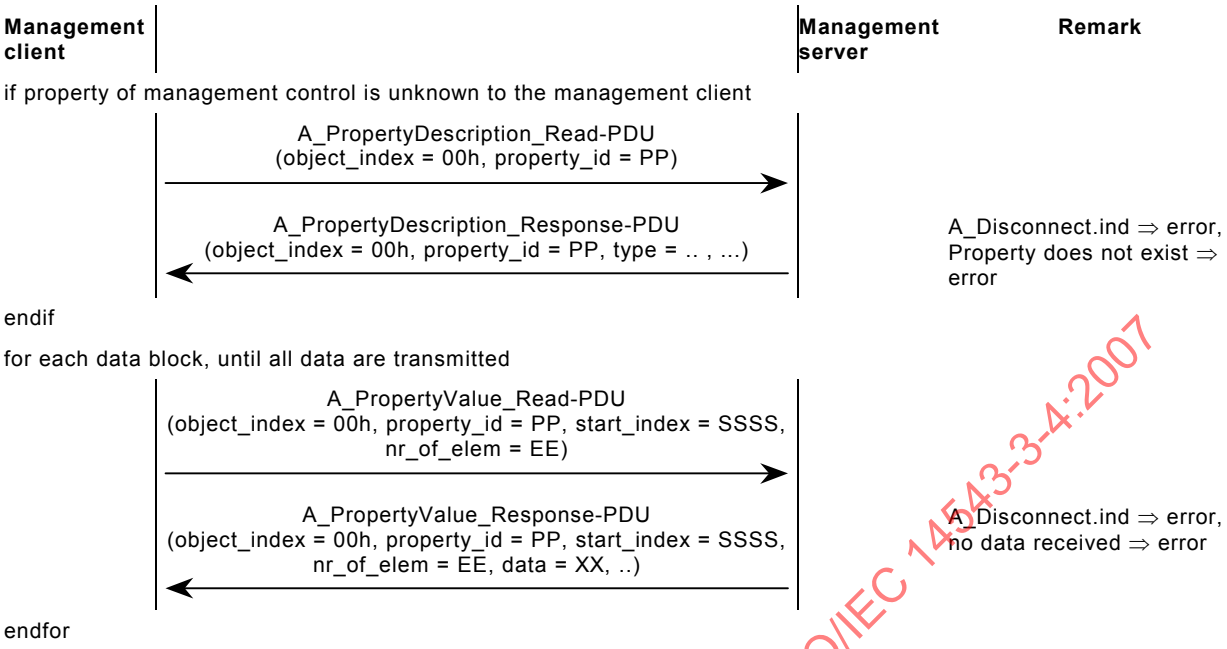
This method shall use the connection-oriented or connectionless remote communication.

6.24.2.2 Management services used

The DMP_InterfaceObjectRead_R procedure shall use the following management services:
A_PropertyDescription_Read

A_PropertyValue_Read

6.24.2.3 Sequence



6.24.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.25 DM_InterfaceObjectScan

6.25.1 General description

This device management procedure shall scan for available interface objects in one management server and return the description of each found interface object.

A DM_Connect shall be executed before executing this management procedure.

DM_InterfaceObjectScan (flags, dataBlockStartAddress, object_index, data)

flags

bit 0: location of data

- 0: in data block
- 1: no data returned

bit 2: scan all properties

- 0: read only the type of the interface object
- 1: scan all the properties of the interface object

bit 3: scan interface objects

- 0: read only the data of one object
- 1: scan all interface objects

All other bits are reserved. These shall be set to 0.
This shall be tested by the management client.

dataBlockStartAddress — specifies the address where the data are located in the data block. If the data are located in the management procedure, this field shall be set to 0.

object_index — index of the interface object. If interface object scan is enabled the value shall be 0.

data — the data that are read by this management procedure. The data shall be stored in the data block.

For each found interface object the following data shall be returned; the end of the list shall be marked by interface object number 0:

- object_index;
- object_type;
- PropertyCount.

For each found property the following data is returned:

- property_index;
- property_id;
- DataType;
- Number of elements;
- Access rights / write enable.

6.25.2 Procedure: DMP_InterfaceObjectScan_R

6.25.2.1 Description

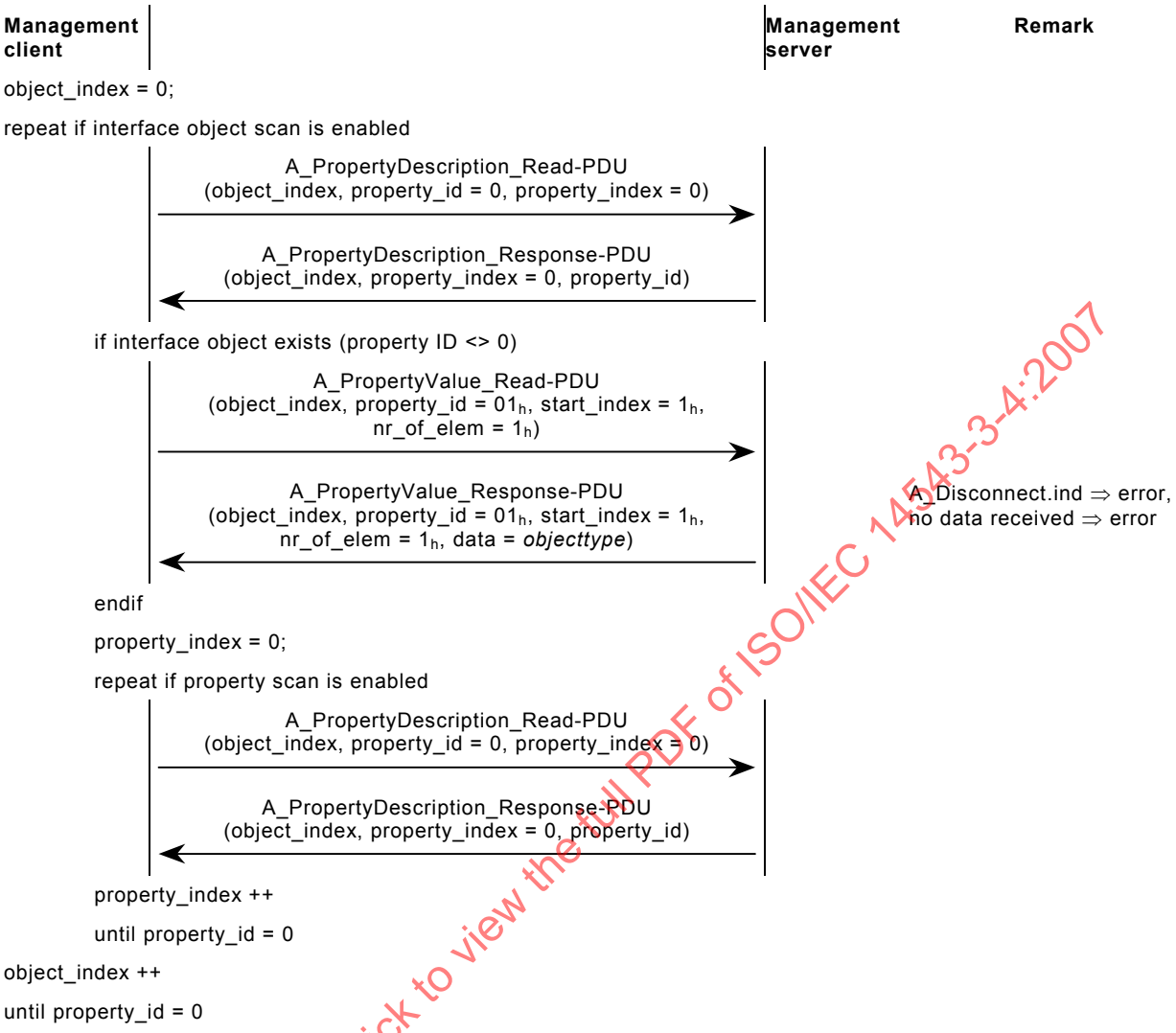
This method shall use the connection-oriented or connectionless remote communication.

6.25.2.2 Management services used

The DMP_InterfaceObjectScan_R procedure shall use the following management services:
A_PropertyDescription_Read

A_PropertyValue_Read

6.25.2.3 Sequence



6.25.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.26 DM_LoadStateMachineWrite

6.26.1 General description

This device management procedure shall be used to write to the load state machine of a management server. The data shall be located in the management procedure. Depending on the flag, the resulting state shall be verified immediately.

Table 1 – Resulting states after each event

Event	Resulting state
Unload	Unloaded
Load	Loading
LoadCompleted	Loaded
AllocAbsDataSeg	Loading
AllocAbsStackSeg	
AbsTaskSeg	
TaskPtr	
TaskCtrl1	
TaskCtrl2	

A DM_Connect shall be executed before executing this management procedure.

DM_LoadStateMachineWrite (flags, stateMachineType, stateMachineNr, event, eventData)

flags bit 0: location of data
0: -
1: in management control
bit 1: verify resulting state enabled / disabled
0: disabled
1: enabled
All other bits are reserved. These shall be set to 0.
This shall be tested by the management client.

stateMachineType type of the object that contains the state machine:

Type	State machine
0001	Address table
0002	Association table
0003	Application program
0004	PEI program

stateMachineNr index to the state machine. For this index only the state machines of this type are relevant. This index starts counting from 0.

event code of the event

eventData data of the event

Different events can be sent to the load state machines.

Table 2 – Overview state machine types and tables

Event	State machine type			
	Address table	Association table	Application program	PEI program
Unload	X	X	X	X
Load	X	X	X	X
LoadCompleted	X	X	X	X
AllocAbsDataSeg	X	X	X	X
AllocAbsStackSeg			X	X
AllocAbsTaskSeg	X	X	X	X
TaskPtr			X	
TaskCtrl1			X	
TaskCtrl2			X	

6.26.2 Procedure: DMP_LoadStateMachineWrite_RCo_Mem

6.26.2.1 Description

This method shall use the connection-oriented remote communication. The control and state of the load state machine shall be located in the memory of the management server and can be accessed via direct memory access. The verify mode of the management server shall not be used.

This procedure shall support only one state machine of each type.

The address of the management control is 0104h. The address (AAAA) of the load state depends on the load state machine.

Table 3 – Overview addresses for the load management controls

State machine	Address of load state (AAAA)
Address table	B6EAh
Association table	B6EBh
Application program	B6ECh
PEI program	B6EDh

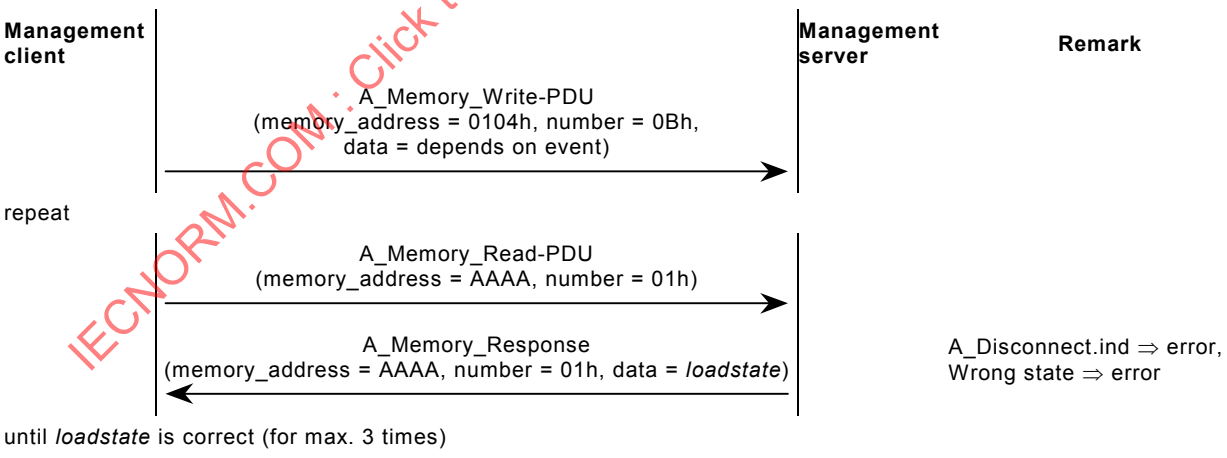
6.26.2.2 Management services used

The DMP_LoadStateMachineWrite_RCo_Mem procedure shall use the following management services:

A_Memory_Write

A_Memory_Read

6.26.2.3 Sequence



The transmitted data depend on the event.

- LoadEvent: Unload

	Data
State machine / Event	Reserved
L4h	00h
1 octet	10 octet

- LoadEvent: Load

	Data
State machine / Event	Reserved
L1h	00h
1 octet	10 octet

- LoadEvent: LoadComplete

	Data
State machine / Event	Reserved
L2h	00h
1 octet	10 octet

- LoadEvent: AllocAbsDataSeg (segment type 0)

	Data							
State machine / Event	Segment type	Segment ID	Start address	Length	Access attributes	Memory type	Memory attributes	Reserved
L3h	00h	00h	SSSS	EEEE - SSSS +1	AA	TT	MM	00h
1 octet	1 octet	1 octet	2 octets	2 octets	1 octet	1 octet	1 octet	1 octet

Access Attributes — contains the access level of the segment
- bit 0…3 write access level
- bit 4…7 read access level

Memory type — contains the type of the memory of the segment
- bit 0…2 memory type
  - 1 Zero page RAM
  - 2 RAM
  - 3 EEPROM
- bit 3…7 Reserved. Shall be zero

Memory Attributes — additional memory configuration
- bit 0…6 Reserved. Shall be zero
- bit 7
  - 0 Checksum control disabled
  - 1 Checksum control enabled

- LoadEvent: AllocAbsStackSeg (segment type 1)

Data								
State machine / Event	Segment type	Segment ID	Start address	Length	Access attributes	Memory type	Memory attributes	Reserved
L3h	01h	00h	SSSS	EEEE - SSSS +1	AA	TT	MM	00h
1 octet	1 octet	1 octet	2 octets	2 octets	1 octet	1 octet	1 octet	1 octet

Access Attributes — contains the access level of the segment
- bit 0…3 write access level
- bit 4…7 read access level

Memory type — contains the type of the memory of the segment
- bit 0…2 memory type
  - 1 Zero page RAM
  - 2 RAM
  - 3 EEPROM
- bit 3…7 Reserved. Shall be zero

Memory Attributes — additional memory configuration
- bit 0…6 Reserved. Shall be zero
- bit 7
  - 0 Checksum control disabled
  - 1 Checksum control enabled

- LoadEvent: AllocAbsTaskSeg (segment type 2)

Data					
State machine / Event	Segment type	Segment ID	Start address	PEI type	Application ID / Table ID – Version
L3h	02h	00h	SSSS	PP	MM MM TT TT VV
1 octet	1 octet	1 octet	2 octets	1 octet	5 octets

Application ID / Table ID — Application Software Type
- MM MM Software Manufacturer ID
- TT TT Manufacturer Specific Application Software ID
- VV Version of the Application Software

- LoadEvent: TaskPtr (segment type 3)

Data						
State machine/ Event	Segment type	Segment ID	initAddr	SaveAddr	PEIHandler	Reserved
L3h	03h	00h	IIII	SSSS	PPPP	00h
1 octet	1 octet	1 octet	2 octets	2 octets	2 octets	2 octets

- LoadEvent: TaskCtrl1 (segment type 4)

Data					
State machine/ Event	Segment type	Segment ID	Interface object address	Interface object count	Reserved
L3h	04h	00h	AAAA	NN	00h
1 octet	1 octet	1 octet	2 octets	1 octet	5 octets

- LoadEvent: TaskCtrl2 (segment type 5)

Data						
State machine/ Event	Segment type	Segment ID	callbackAddr	CommObjPtr	CommObjSegPtr1	CommObjSegPtr2
L3h	05h	00h	*CCCC*	*0000h*	*1111h*	*2222h*
1 octet	1 octet	1 octet	2 octets	2 octets	2 octets	2 octets

NOTE The values in italics are examples.

6.26.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.26.3 Procedure: DMP_LoadStateMachineWrite_RCo_IO

6.26.3.1 Description

This method shall use connection-oriented remote communication. The control and state of the load state machine shall be located in interface objects of the management server and can be accessed via property services.

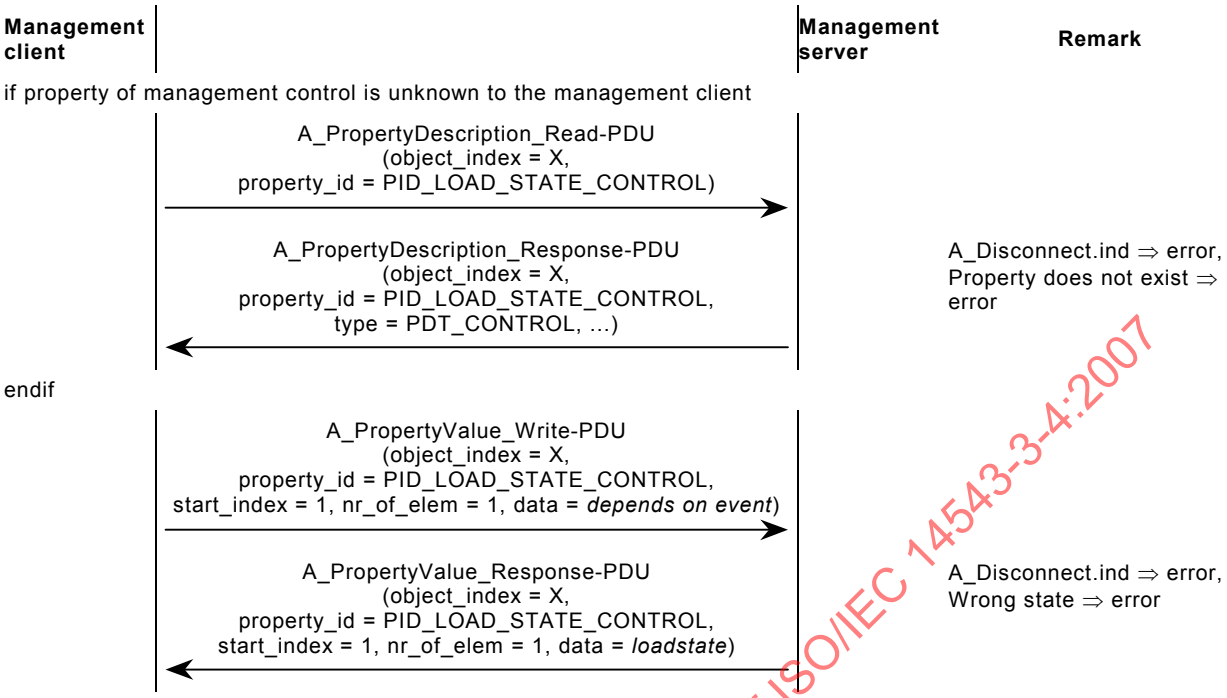
The management client shall search the corresponding interface object in the management server.

6.26.3.2 Management services used

The DMP_LoadStateMachineWrite_RCo_IO procedure shall use the following management services:
A_PropertyDescription_Read

A_PropertyValue_Write

6.26.3.3 Sequence



The transmitted data depends on the event.

- LoadEvent: Unload

	Data
Event	Reserved
04h	00h
1 octet	9 octets

- LoadEvent: Load

	Data
Event	Reserved
01h	00h
1 octet	9 octets

- LoadEvent: LoadCompleted

	Data
Event	Reserved
02h	00h
1 octet	9 octets

- LoadEvent: AllocAbsDataSeg (segment type 0)

Data							
Event	Segment type	Start address	Length	Access attributes	Memory type	Memory attributes	Reserved
03h	00h	SSSS	EEEE - SSSS +1	AA	TT	MM	00h
1 octet	1 octet	2 octets	2 octets	1 octet	1 octet	1 octet	1 octet

Access Attributes contains the access level of the segment
- bit 0…3 write access level
- bit 4…7 read access level

Memory type contains the type of the memory of the segment
- bit 0…2 memory type
  - 1 Zero page RAM
  - 2 RAM
  - 3 EEPROM
- bit 3…7 Reserved. Shall be zero

Memory Attributes additional memory configuration
- bit 0…6 Reserved. Shall be zero
- bit 7 0 Checksum control disabled

- LoadEvent: AllocAbsStackSeg (segment type 1)

Data							
Event	Segment type	Start address	Length	Access attributes	Memory type	Memory attributes	Reserved
03h	01h	SSSSh	EEEE - SSSS +1	AA	TT	MM	00h
1 octet	1 octet	2 octets	2 octets	1 octet	1 octet	1 octet	1 octet

Access Attributes contains the access level of the segment
- bit 0…3 write access level
- bit 4…7 read access level

Memory type contains the type of the memory of the segment
- bit 0…2 memory type
  - 1 Zero page RAM
  - 2 RAM
  - 3 EEPROM
- bit 3…7 Reserved. Shall be zero

Memory Attributes additional memory configuration
- bit 0…6 Reserved. Shall be zero
- bit 7 0 Checksum control disabled

- LoadEvent: AllocAbsTaskSeg (segment type 2)

Data				
Event	Segment type	Start address	PEI type	Application ID / Table ID - Version
03h	02h	SSSS	PP	MM MM TT TT VV
1 octet	1 octet	2 octets	1 octet	5 octets

Application ID /Table ID — Application Software Type

MM MM	Software Manufacturer ID
TT TT	Manufacturer Specific Application Software ID
VV	Version of the Application Software

- LoadEvent: TaskPtr (segment type 3)

Data					
Event	Segment type	initAddr	SaveAddr	PEIHandler	Reserved
03h	03h	IIII	SSSS	PPPP	00h
1 octet	1 octet	2 octets	2 octets	2 octets	2 octets

- LoadEvent: TaskCtrl1 (segment type 4)

Data				
Event	Segment type	Interface object address	Nr. of interface objects	Reserved
03h	04h	AAAA	NN	00h
1 octet	1 octet	2 octets	1 octet	5 octets

- LoadEvent: TaskCtrl2 (segment type 5)

Data					
Event	Segment type	callbackAddr	CommObjPtr	CommObjSegPtr1	CommObjSegPtr2
03h	05h	CCCC	*0000h*	*1111h*	*2222h*
1 octet	1 octet	2 octets	2 octets	2 octets	2 octets
NOTE Values in italics are examples.					

6.26.3.4 Exception handling

The general exception handling of 5.2 is applicable.

6.27 DM_LoadStateMachineVerify

6.27.1 General description

This device management procedure shall be used to verify the state of a load state machine of a management server. The state shall be located in the management procedure.

A DM_Connect shall be executed before executing this management procedure.

DM_LoadStateMachineVerify (flags, stateMachineType, stateMachineNr, state)

flags	bit 0: location of data 0: - 1: in management control All other bits are reserved. These shall be set to 0. This shall be tested by the management client.
stateMachineType	type of the object that contains the state machine:

Type	State machine
0001	Address table
0002	Association table
0003	Application program
0004	PEI program

stateMachineNr	index to the state machine. For this index only the state machines of this type are relevant. This index shall start counting from 0.
state	state of the state machine

6.27.2 Procedure: DM_LoadStateMachineVerify_RCo_Mem

6.27.2.1 Description

This method shall use the connection-oriented remote communication. The control and state of the load state machine shall be located in the memory of the management server and can be accessed via direct memory access.

This procedure shall support only one state machine of each type.

The address of the management control shall be 0104h. The address (AAAA) of the load state depends on the load state machine.

Table 4 – Addresses of the load state controls

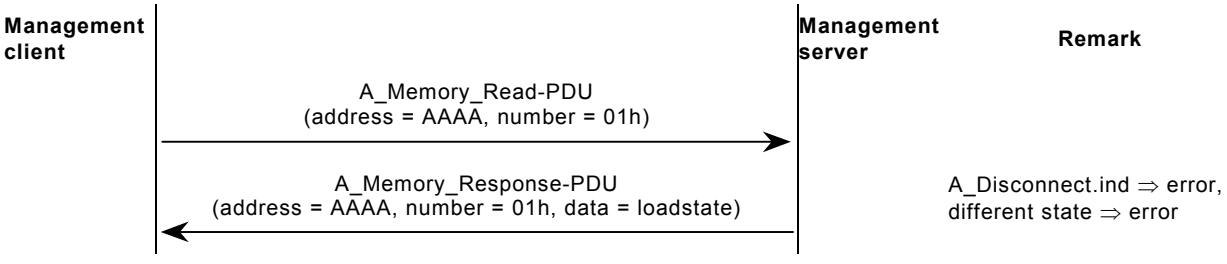
State machine	Address of load state (AAAA)
Address table	B6EAh
Association table	B6EBh
Application program	B6ECh
PEI program	B6EDh

This procedure shall not be used for further developments of management servers.

6.27.2.2 Management service used

The DM_LoadStateMachineVerify_RCo_Mem procedure shall use the following management service:
A_Memory_Read

6.27.2.3 Sequence



6.27.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.27.3 Procedure: DMP_LoadStateMachineVerify_R_IO

6.27.3.1 Description

This method shall use either the connection-oriented or connectionless remote communication. The control and state of the load state machine shall be located in interface objects of the management server and can be accessed via property services.

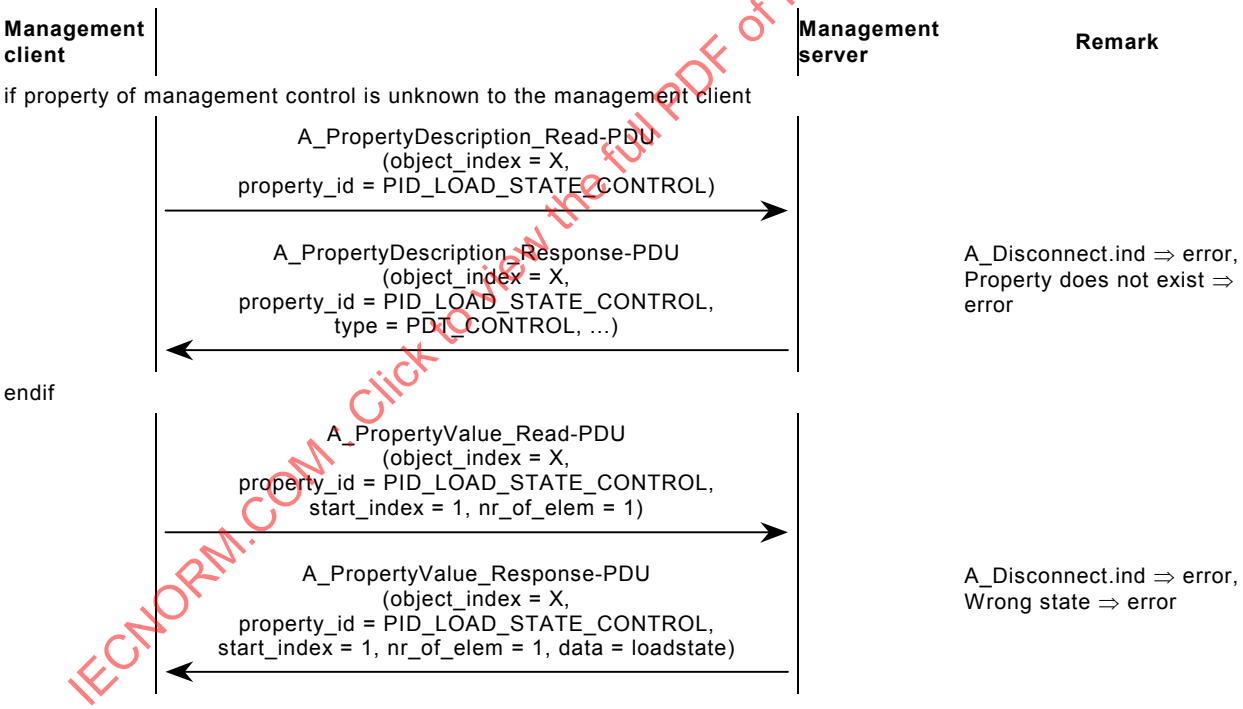
The management client shall search the corresponding interface object in the management server.

6.27.3.2 Management services used

The DMP_LoadStateMachineVerify_R_IO procedure shall use the following management services:
A_PropertyDescription_Read

A_PropertyValue_Read

6.27.3.3 Sequence



6.27.3.4 Exception handling

The general exception handling of 5.2 is applicable.

6.28 DM_LoadStateMachineRead

6.28.1 General description

This device management procedure shall be used to read the state of a load state machine of a management server. The state shall be located in the management procedure.

A DM_Connect shall be executed before executing this management procedure.

DM_LoadStateMachineRead (dataBlockStartAddress, flags, stateMachineType, stateMachineNr, state)

dataBlockStartAddress: specifies the address where the data are stored in the data block

flags: bit 0: location of data

0: in data block
1: -

All other bits are reserved. These shall be set to 0. This shall be tested by the management client.

stateMachineType: type of the object that contains the state machine:

Type	State machine
0001	Address table
0002	Association table
0003	Application program
0004	PEI program

stateMachineNr: index to the state machine. For this index only the state machines of this type are relevant. This index shall start counting from 0.

state: state of the state machine

6.28.2 Procedure: DMP_LoadStateMachineRead_RCo_Mem

6.28.2.1 Description

This method shall use the connection-oriented remote communication.

The control and state of the load state machine shall be located in the memory of the management server and can be accessed via direct memory access.

This procedure shall support only one state machine of each type.

The address of the management control shall be 0104h. The address (AAAA) of the load state depends on the load state machine.

Table 5 – Addresses of the load state controls

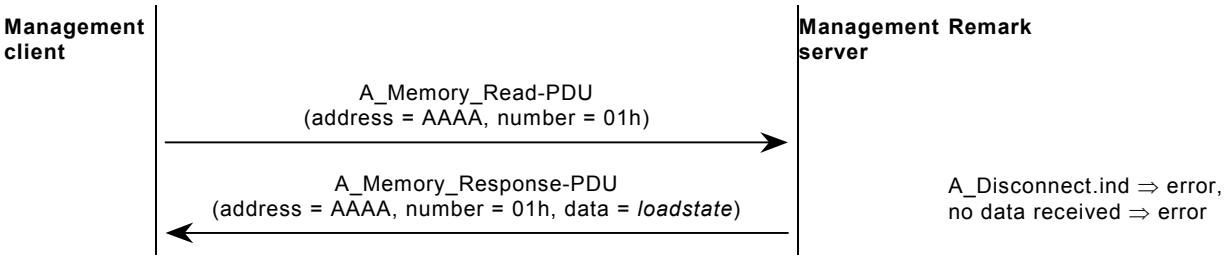
State machine	Address of load state (AAAA)
Address table	B6EAh
Association table	B6EBh
Application program	B6ECh
PEI program	B6EDh

6.28.2.2 Management service used

The DMP_LoadStateMachineRead_RCo_Mem procedure shall use the following management service:

A_Memory_Read

6.28.2.3 Sequence



6.28.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.28.3 Procedure: DMP_LoadStateMachineRead_R_IO

6.28.3.1 Description

This method shall use either the connection-oriented or connectionless remote communication. The control and state of the load state machine shall be located in interface objects of the management server and can be accessed via property services.

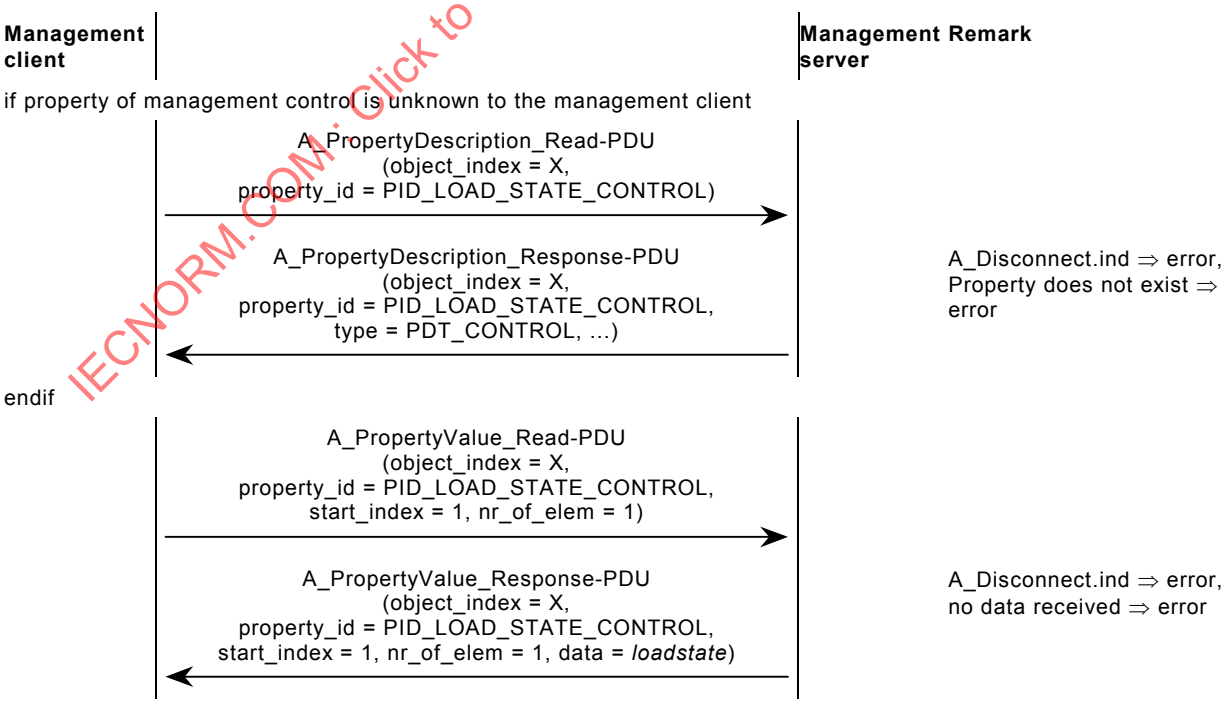
The management client shall search the corresponding interface object in the management server.

6.28.3.2 Management services used

The DMP_LoadStateMachineRead_R_IO_Mem procedure shall use the following management services:
A_PropertyDescription_Read

A_PropertyValue_Read

6.28.3.3 Sequence



6.28.3.4 Exception handling

The general exception handling of 5.2 is applicable.

6.29 DM_RunStateMachineWrite

6.29.1 General description

This device management procedure shall be used to write to the run state machine of a management server. The data shall be located in the management procedure. Depending on the flag, the resulting state shall be verified immediately.

Table 6 – Run state events and resulting run states

Event	Resulting state
Restart	Ready or Running
Stop	Terminated

A DM_Connect shall be executed before executing this management procedure.

DM_RunStateMachineWrite (flags, stateMachineType, stateMachineNr, event)

flags: bit 0: location of data

0: -
1: in management control

bit 1: verify the resulting state enabled / disabled

0: disabled
1: enabled

All other bits are reserved. These shall be set to 0. This shall be tested by the management client.

stateMachineType: type of the object that contains the state machine:

Type	State machine
0003	Application program
0004	PEI program

stateMachineNr: index to the state machine. For this index only the state machines of this type are relevant. This index shall start counting from 0.

event: code of the event

6.29.2 Procedure: DMP_RunStateMachineWrite_RCo_Mem

6.29.2.1 Description

This procedure shall use the connection-oriented remote communication. The control and state of the run state machine shall be located in the memory of the management server and can be accessed via direct memory access. The verify mode of the management server shall not be used.

This procedure shall support only one state machine of each type.

The address of the run control shall be 0103h. The address (AAAA) of the run state depends on the run state machine.

Table 7 – Addresses of the run state controls

State machine	Address of run state (AAAA)
Application program	0101h
PEI program	0102h

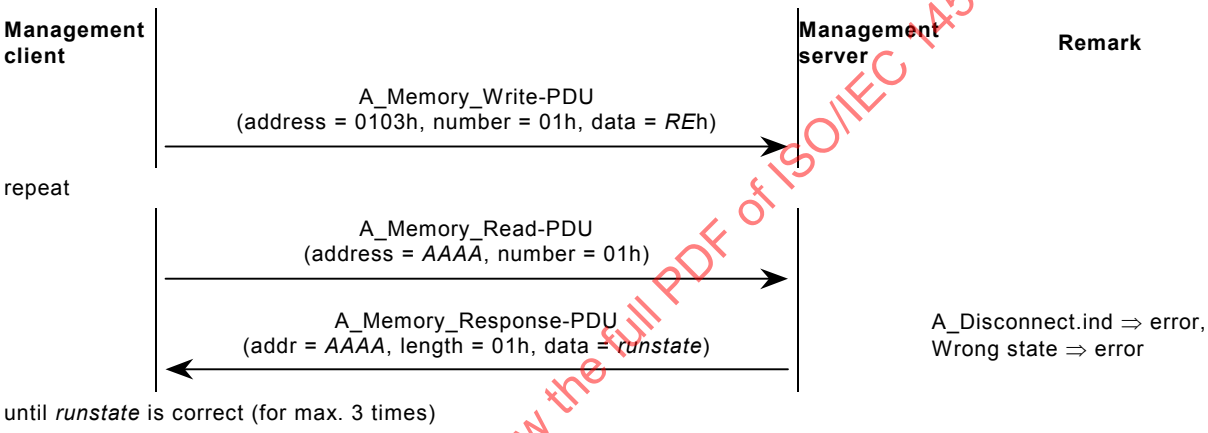
This procedure shall not be used for further developments of management servers.

6.29.2.2 Management services used

The DMP_RunStateMachineWrite_RCo_Mem procedure shall use the following management services:
A_Memory_Write

A_Memory_Read

6.29.2.3 Sequence



6.29.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.29.3 Procedure: DMP_RunStateMachineWrite_R_IO

6.29.3.1 Description

This method shall use either the connection-oriented or connectionless remote communication. The control and state of the run state machine shall be located in interface objects of the management server and can be accessed via property services.

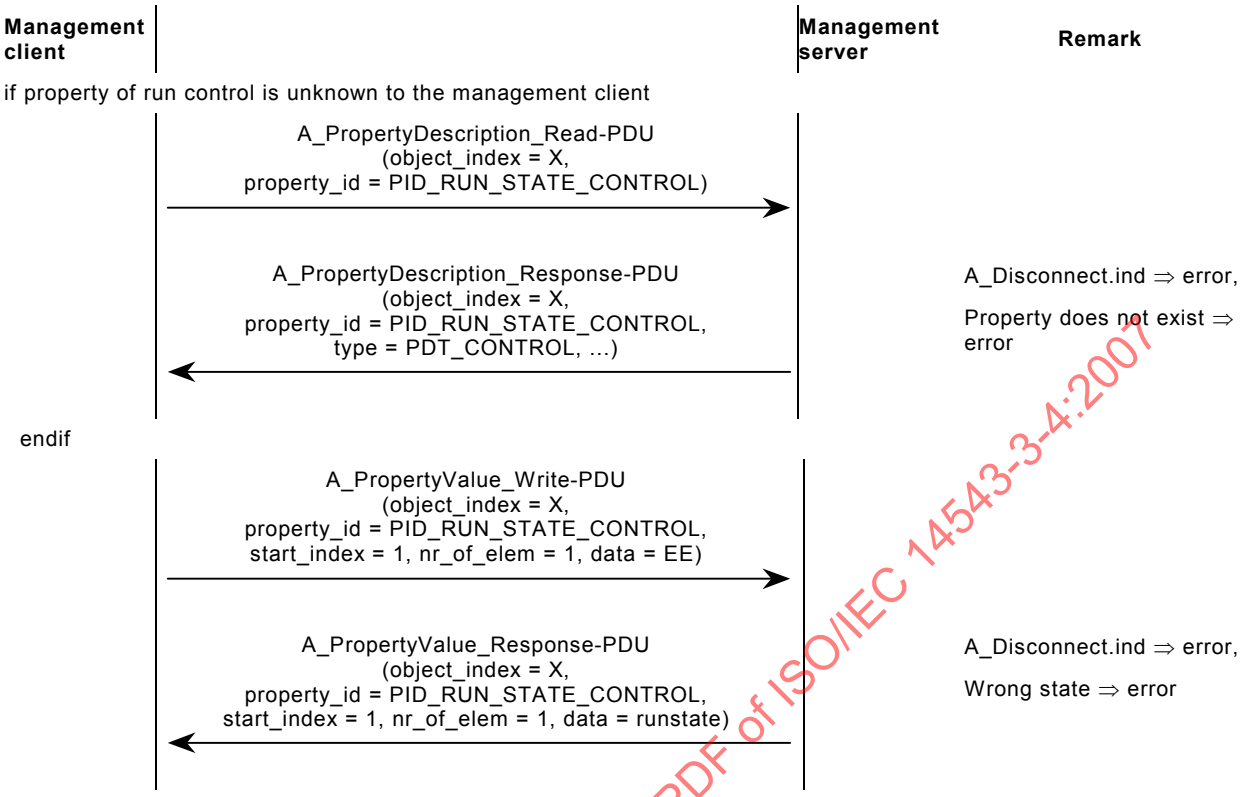
The management client shall search the according interface object in the management server.

6.29.3.2 Management services used

The DMP_RunStateMachineWrite_R_IO procedure shall use the following management services:
A_PropertyDescription_Read

A_PropertyValue_Write

6.29.3.3 Sequence



6.29.3.4 Exception handling

The general exception handling of 5.2 is applicable.

6.30 DM_RunStateMachineVerify

6.30.1 General description

This device management procedure shall be used to verify the state of a run state machine of a management server. The state shall be located in the management procedure.

A DM_Connect shall be executed before executing this management procedure.

DM_RunStateMachineVerify (flags, stateMachineType, stateMachineNr, state)

flags: bit 0: location of data

0: -
1: in management control

All other bits are reserved. These shall be set to 0. This shall be tested by the management client.

stateMachineType: type of the object that contains the state machine:

Type	State machine
0003	Application program
0004	PEI program

stateMachineNr: index to the state machine. For this index only the state machines of this type are relevant. This index shall start counting from 0.

state: state of the state machine

6.30.2 Procedure: DMP_RunStateMachineVerify_RCo_Mem

6.30.2.1 Description

This method shall use the connection-oriented remote communication. The control and state of the run state machine shall be located in the memory of the management server and can be accessed via direct memory access.

This procedure shall support only one state machine of each type.

The address (AAAA) of the run state depends on the run state machine.

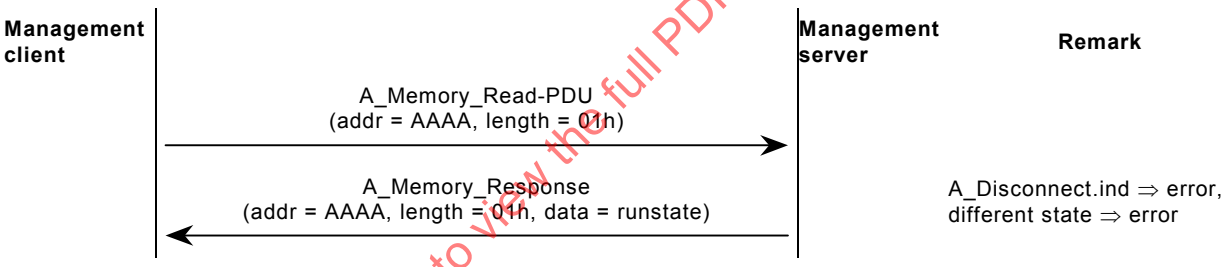
State machine	Address of run state
Application program	0101h
PEI program	0102h

This procedure shall not be used for further developments of management servers.

6.30.2.2 Management service used

The DMP_RunStateMachineVerify_RCo_Mem procedure shall use the following management service:
A_Memory_Read

6.30.2.3 Sequence



6.30.2.4 Exception handling

The general exception handling of 5.2 is applicable.

6.30.3 Procedure: DMP_RunStateMachineVerify_R_IO

6.30.3.1 Description

This method shall use either the connection-oriented or connectionless remote communication. The control and state of the run state machine shall be located in interface objects of the management server and can be accessed via property services.

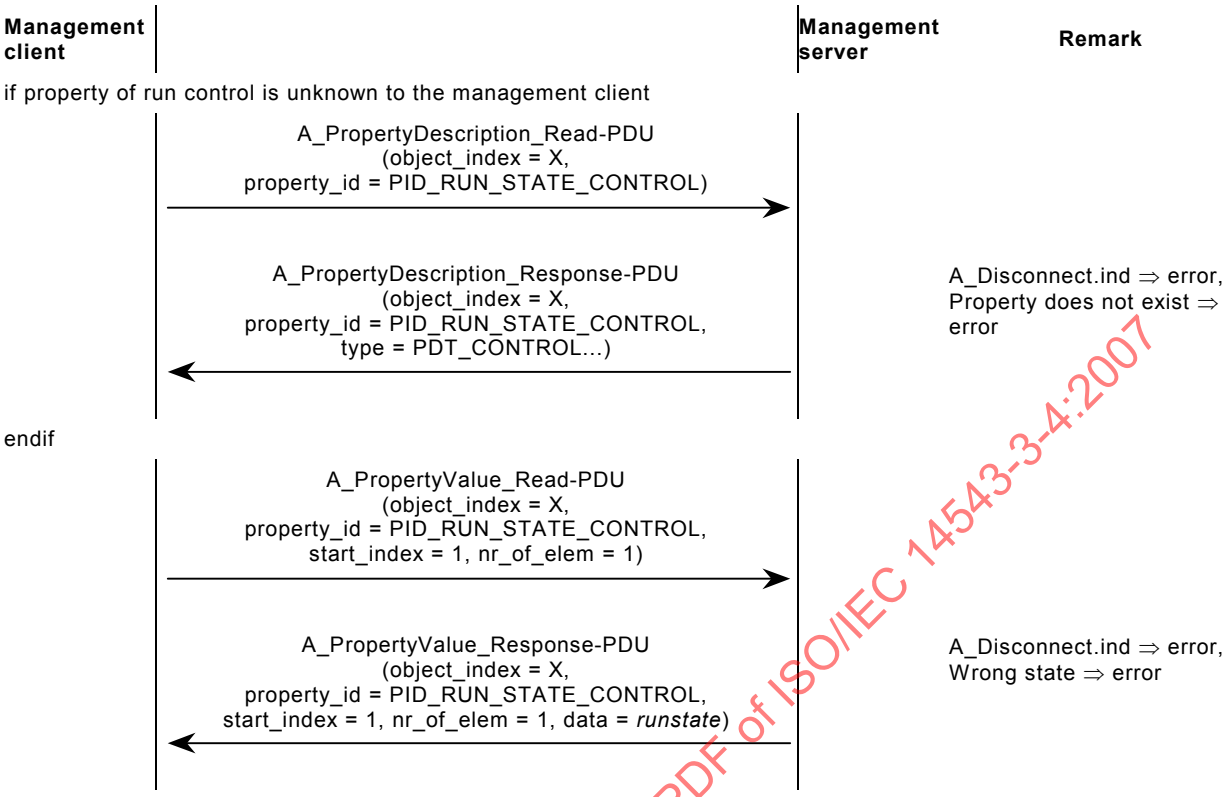
The management client shall search the corresponding interface object in the management server.

6.30.3.2 Management services used

The DMP_RunStateMachineVerify_R_IO procedure shall use the following management services:
A_PropertyDescription_Read

A_PropertyValue_Read

6.30.3.3 Sequence



6.30.3.4 Exception handling

The general exception handling of 5.2 is applicable.

6.31 DM_RunStateMachineRead

6.31.1 General description

This device management procedure shall be used to read the state of a run state machine of a management server. The state shall be located in the management procedure.

A DM_Connect shall be executed before executing this management procedure.

DM_RunStateMachineRead (dataBlockStartAddress, flags, stateMachineType, stateMachineNr, state)

dataBlockStartAddress: specifies the address where the data are stored in the data block

flags: bit 0: location of data

0: in data block
1: -

All other bits are reserved. These shall be set to 0. This shall be tested by the management client.

stateMachineType: type of the object that contains the state machine:

Type	State machine
0003	Application program
0004	PEI program

stateMachineNr: index to the state machine. For this index only the state machines of this type are relevant. This index shall start counting from 0.

state: state of the state machine