

INTERNATIONAL STANDARD

NORME INTERNATIONALE



**Devices and integration in enterprise systems – Function blocks (FB) for
process control and electronic device description language (EDDL) –
Part 4: EDD interpretation**

**Les dispositifs et leur intégration dans les systèmes de l'entreprise – Blocs
fonctionnels (FB) pour les procédés industriels et le langage de description
électronique de produit (EDDL) –
Partie 4: Interprétation EDD**



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2020 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester. If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

Droits de reproduction réservés. Sauf indication contraire, aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de l'IEC ou du Comité national de l'IEC du pays du demandeur. Si vous avez des questions sur le copyright de l'IEC ou si vous désirez obtenir des droits supplémentaires sur cette publication, utilisez les coordonnées ci-après ou contactez le Comité national de l'IEC de votre pays de résidence.

IEC Central Office
3, rue de Varembe
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 It also gives information on projects, replaced and withdrawn publications.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 since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

A propos de l'IEC

La Commission Electrotechnique Internationale (IEC) est la première organisation mondiale qui élabore et publie des Normes internationales pour tout ce qui a trait à l'électricité, à l'électronique et aux technologies apparentées.

A propos des publications IEC

Le contenu technique des publications IEC est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente, un corrigendum ou amendement peut avoir été publié.

Recherche de publications IEC -

webstore.iec.ch/advsearchform

La recherche avancée permet de trouver des publications IEC en utilisant différents critères (numéro de référence, texte, comité d'études,...). Elle donne aussi des informations sur les projets et les publications remplacées ou retirées.

IEC Just Published - webstore.iec.ch/justpublished

Restez informé sur les nouvelles publications IEC. Just Published détaille les nouvelles publications parues. Disponible en ligne et une fois par mois par email.

Service Clients - webstore.iec.ch/csc

Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions contactez-nous: sales@iec.ch.

Electropedia - www.electropedia.org

Le premier dictionnaire d'électrotechnologie en ligne au monde, avec plus de 22 000 articles terminologiques en anglais et en français, ainsi que les termes équivalents dans 16 langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (IEV) en ligne.

Glossaire IEC - std.iec.ch/glossary

67 000 entrées terminologiques électrotechniques, en anglais et en français, extraites des articles Termes et Définitions des publications IEC parues depuis 2002. Plus certaines entrées antérieures extraites des publications des CE 37, 77, 86 et CISPR de l'IEC.

INTERNATIONAL STANDARD

NORME INTERNATIONALE



**Devices and integration in enterprise systems – Function blocks (FB) for
process control and electronic device description language (EDDL) –
Part 4: EDD interpretation**

**Les dispositifs et leur intégration dans les systèmes de l'entreprise – Blocs
fonctionnels (FB) pour les procédés industriels et le langage de description
électronique de produit (EDDL) –
Partie 4: Interprétation EDD**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

ICS 25.040.40; 35.240.50

ISBN 978-2-8322-8445-2

**Warning! Make sure that you obtained this publication from an authorized distributor.
Attention! Veuillez vous assurer que vous avez obtenu cette publication via un distributeur agréé.**

CONTENTS

FOREWORD.....	8
INTRODUCTION.....	11
1 Scope.....	12
2 Normative references	12
3 Terms, definitions, abbreviated terms acronyms and conventions	12
3.1 General terms and definitions	12
3.2 Terms and definitions related to modular devices.....	13
3.3 Abbreviated terms and acronyms	14
3.4 Conventions.....	14
4 EDDL user interface description	15
4.1 Overview.....	15
4.2 Menu conventions for handheld applications	15
4.3 Menu conventions for PC-based applications	16
4.3.1 Overview	16
4.3.2 Online root menus	16
4.3.3 Offline root menu	17
4.3.4 Example of EDD menu structure	17
4.3.5 User interface	22
4.4 Label concatenation for indirect variable references.....	25
4.4.1 General	25
4.4.2 Simple variable references	26
4.4.3 Complex variable references	26
4.5 Help concatenation	28
4.5.1 General	28
4.5.2 Simple variable references	28
4.5.3 Complex variable references	29
4.6 Containers and contained items	30
4.6.1 Overview	30
4.6.2 Permitted and default STYLES.....	30
4.6.3 Containers	32
4.6.4 Contained items.....	34
4.7 Layout rules	40
4.7.1 Overview	40
4.7.2 Controlling the layout by LAYOUT_TYPE attribute	41
4.7.3 Layout rules for WIDTH and HEIGHT	45
4.7.4 Layout rules for COLUMNBREAK and ROWBREAK.....	48
4.7.5 Layout examples	54
4.7.6 Conditional user interface	69
4.8 Graphical elements	75
5 EDDL data description.....	79
5.1 EDDL application stored device data.....	79
5.1.1 Overview	79
5.1.2 FILE	79
5.1.3 LIST	81
5.2 Exposing data items outside the EDD application.....	88
5.3 Initialization of EDD instances.....	88

5.3.1	Overview	88
5.3.2	Initialization support	88
5.3.3	TEMPLATE	88
5.4	Device model mapping	89
5.4.1	BLOCK_A	89
5.4.2	BLOCK_B	89
6	EDDL METHOD programming and usage of builtins	90
6.1	Method environment	90
6.1.1	General	90
6.1.2	Security	90
6.1.3	Device data	90
6.1.4	Method TYPE and parameters	91
6.1.5	Abort processing	91
6.2	Implementation requirements	92
6.3	Builtin MenuDisplay	92
6.4	Division by zero and undetermined floating values	95
6.4.1	Integer and unsigned integer values	95
6.4.2	Floating-point values	95
7	Modular devices	96
7.1	Overview	96
7.2	EDD identification	96
7.3	Instance object model	96
7.4	Offline configuration	97
7.5	Online configuration	97
7.6	Simple modular device example	97
7.6.1	General	97
7.6.2	Separate EDD file example with direct EDD referencing	98
7.6.3	Separate EDD file example with classification EDD referencing and interfaces	100
7.6.4	One EDD file example	102
7.6.5	Combination of single and separate modular device example	104
7.7	Upload and download for modular devices	104
7.8	Diagnostic	104
7.9	Reading modular device topology	105
7.9.1	SCAN	105
7.9.2	Detect module type	107
7.10	Configuration check	107
8	Session management	108
8.1	Overview	108
8.2	Data management	108
8.2.1	Overview	108
8.2.2	Caching for online session	109
8.2.3	Caching for offline session	110
8.2.4	Caching for dialogs and windows	111
8.2.5	Caching for METHODS	112
8.3	UI aspects of editing sessions	115
8.4	User roles	116
9	Offline and online configuration	116
9.1	Overview	116

9.2	Offline dataset	116
9.3	Offline configuration.....	116
9.4	Online dataset	116
9.5	Online configuration.....	116
9.6	Upload and download	117
9.6.1	Overview	117
9.6.2	Error recovery.....	118
9.6.3	Upload procedure	118
9.6.4	Download procedure.....	120
10	EDDL communication description	122
10.1	General.....	122
10.2	Parsing data received from the device	123
10.3	Parsing complex data items	123
10.4	Foundation Fieldbus	123
10.5	ISA100_Wireless communication model.....	127
Annex A	(normative) Device simulation	131
Annex B	(informative) Predefined identifiers	132
Annex C	(informative) Description of EDDL profiles	135
C.1	Communication Server (CS).....	135
C.2	Foundation Fieldbus (FF).....	135
C.3	Generic Protocol Extension (GPE)	135
C.4	HART.....	135
C.5	ISA100.....	135
C.6	PROFIBUS (PB).....	135
C.7	PROFINET (PN).....	136
Annex D	(normative) Upload/download caching model	137
Bibliography		139
Figure 1	– EDD example of root menus.....	22
Figure 2	– Example of an EDD application for diagnostics	22
Figure 3	– Example of an EDD application for process variables.....	23
Figure 4	– Example of an EDD application for primary variables	23
Figure 5	– Example of an EDD application for process-related device features	24
Figure 6	– Example of an EDD application for device features	24
Figure 7	– Example of an EDD application for maintenance features	25
Figure 8	– Usage of COLLECTION MEMBERS in MENUs of STYLE GROUP	33
Figure 9	– Displaying single bits of BIT_ENUMERATED	35
Figure 10	– Displaying multiple bits of BIT_ENUMERATED.....	36
Figure 11	– Example of an EDD application for a variable of type BIT_ENUMERATED	36
Figure 12	– EDD example with a "write-only" variable (HANDLING WRITE)	37
Figure 13	– Basic layout elements	40
Figure 14	– Example of layout with equal column width.....	42
Figure 15	– Example of layout with optimized column width	42
Figure 16	– Cell body in a layout with optimized column width (label to the left).....	43
Figure 17	– Cell body in a layout with optimized column width (label on top).....	43
Figure 18	– EDD source code for a layout with VARIABLES spanning columns	47

Figure 19 – Layout with VARIABLES spanning multiple columns	47
Figure 20 – EDD source code for layout for protruding elements example.....	49
Figure 21 – Layout for protruding elements	49
Figure 22 – EDD source code for layout for partially filled rows example.....	50
Figure 23 – Layout for partially filled rows	50
Figure 24 – EDD source code for layout for partially filled rows example.....	51
Figure 25 – Layout for partially filled rows	51
Figure 26 – EDD source code for layout for oversized elements example.....	52
Figure 27 – Oversized element in a layout with equal column width	52
Figure 28 – Oversized element in a layout with optimized column width	52
Figure 29 – EDD source code example for a layout for columns in stacked group	53
Figure 30 – Layout for columns in stacked group	53
Figure 31 – EDD source code for layout for columns with GRAPHS in stacked group example	54
Figure 32 – Layout for columns with GRAPHS in stacked group	54
Figure 33 – Example of an EDD for an overview menu.....	55
Figure 34 – Example of an EDD application for an overview window	55
Figure 35 – EDD source code for a layout with menu items spanning a single column	55
Figure 36 – Example of a layout with menu items spanning a single column	56
Figure 37 – Example of an EDD using COLUMNBREAK	56
Figure 38 – Example of an EDD application for an overview window	57
Figure 39 – EDD example for an overview window	57
Figure 40 – Example of an EDD application for an overview window	58
Figure 41 – EDD source code for a layout with small in-line images.....	58
Figure 42 – Example of a layout with small in-line images.....	59
Figure 43 – EDD source code for a multi-column layout with GROUP	60
Figure 44 – Example of a multi-column layout with GROUP	61
Figure 45 – Example of an EDD for in-line graphs and charts	61
Figure 46 – Example of an EDD application for an in-line graph.....	62
Figure 47 – Example of an EDD for full-width graphs and charts	62
Figure 48 – Example of an EDD application for a full-width graph in a layout with equal column width.....	63
Figure 49 – Example of an EDD application for a full-width graph in a layout with optimized column width.....	64
Figure 50 – Example of an EDD for nested containers	65
Figure 51 – Example of an EDD application for nested containers	65
Figure 52 – Example of an EDD for EDIT_DISPLAYS	66
Figure 53 – Example of an EDD application for EDIT_DISPLAYS.....	67
Figure 54 – Example of an EDD for images.....	67
Figure 55 – Example of an EDD application for images	68
Figure 56 – Example of an EDD for large inline-images	68
Figure 57 – Example of layout with a large inline-image.....	69
Figure 58 – EDD example for VALIDITY in online session.....	70
Figure 59 – Example of an EDD application for a gauge with limit regions	70

Figure 60 – Example of an EDD for a gauge with limit regions	78
Figure 61 – Example of a file declaration	80
Figure 62 – Example of comparing valve signatures.....	81
Figure 63 – Example of more complex file declaration	82
Figure 64 – Example of reviewing the stored radar signals.....	83
Figure 65 – Example of an EDD that inserts, replaces, or compares radar signals	88
Figure 66 – Example of a BLOCK_A	89
Figure 67 – Example of a wizard	94
Figure 68 – The different relations of a module	97
Figure 69 – Components and possible configuration of the modular devices	98
Figure 70 – Separate EDD file example with direct EDD referencing.....	99
Figure 71 – EDD example for module1.....	99
Figure 72 – EDD example for module2.....	100
Figure 73 – EDD example for modular device	101
Figure 74 – EDD example for module1.....	102
Figure 75 – EDD example for module2.....	102
Figure 76 – EDD example for module2.....	103
Figure 77 – Upload/download order of a modular device	104
Figure 78 – Example of a SCAN METHOD	106
Figure 79 – Example of a DETECT METHOD.....	107
Figure 80 – Example of a CHECK_CONFIGURATION METHOD	108
Figure 81 – Data caching for an online session.....	110
Figure 82 – Data caching for an offline session.....	111
Figure 83 – Sub dialogs or windows using a shared edit cache	111
Figure 84 – Sub dialogs or windows using separate edit caches	112
Figure 85 – Data caching for nested METHODS	112
Figure 86 – Data caching for a METHOD invoked within a dialog or window	113
Figure 87 – Data caching for a METHOD invoking a dialog using an edit cache	113
Figure 88 – Data caching for a METHOD invoking a dialog	113
Figure 89 – Data flow for download to the device	117
Figure 90 – Data flow for upload from the device	118
Figure 91 – Example device with 2 unique BLOCK_A definitions.....	124
Figure 92 – Example EDD for a device with 2 unique BLOCK_A definitions	125
Figure 93 – BLOCK_A example with PARAMETER_LISTS.....	126
Figure 94 – Example EDD for a BLOCK_A with PARAMETER_LISTS	127
Figure 95 – Example ISA100_Wireless device objects representation.....	128
Figure 96 – Example EDD for a ISA100_Wireless device with 2 unique BLOCK_A definitions	129
Figure 97 – BLOCK_A example with PARAMETER_LISTS.....	129
Figure 98 – Example EDD for a BLOCK_A with PARAMETER_LISTS	130
Figure D.1 – Upload caching model	137
Figure D.2 – Download caching model	138

Table 1 – List of defined root menu identifiers for handhelds.....	14
--	----

Table 2 – List of defined root menu identifiers for PC-based devices	16
Table 3 – Fall back alternatives for online root menus.....	16
Table 4 – Fall back alternatives for offline root menus	17
Table 5 – Label rule summary for simple variable references	26
Table 6 – Label rule summary for simple variable references	26
Table 7 – Prefix rule summary for complex variable references.....	27
Table 8 – Prefix rule summary for complex variable references.....	27
Table 9 – Body rule summary for complex variable references	27
Table 10 – Body rule summary for complex variable references	27
Table 11 – Suffix rule summary for complex variable references	28
Table 12 – Suffix rule summary for complex variable references	28
Table 13 – Help rule summary for simple variable references	28
Table 14 – Help rule summary for simple variable references	28
Table 15 – Help prefix rule summary for complex variable references	29
Table 16 – Help prefix rule summary for complex variable references	29
Table 17 – Help suffix rule summary for complex variable references	29
Table 18 – Help suffix rule summary for complex variable references	29
Table 19 – Permitted contained items and default STYLES.....	31
Table 20 – Uninitialized state of VARIABLES on user interface	34
Table 21 – Example of "write-only" variable in an online dialog	38
Table 22 – Description of layout content	41
Table 23 – Minimum and maximum width for input fields spanning one column	43
Table 24 – WIDTH and HEIGHT span and applicability	45
Table 25 – Example 1 VALIDITY in an online session	71
Table 26 – Example 2 VALIDITY in an online session	72
Table 27 – Example 3 VALIDITY in an online session	73
Table 28 – Example 4 VALIDITY in an online session	74
Table 29 – Examples of floating-point results.....	95
Table 30 – Usages of COMPONENT_PATH.....	96
Table 31 – Diagnostic classifications	105
Table 32 – Terminology for session management	108
Table 33 – Terminology used in data management	109
Table 34 – Builtins for method cache controlling	114
Table 35 – List of defined upload menu identifiers	118
Table 36 – List of defined download menu identifiers	120
Table B.1 – ARRAY predefined identifiers.....	132
Table B.2 – COLLECTION predefined identifiers.....	132
Table B.3 – COMMAND predefined identifiers.....	132
Table B.4 – IMAGE predefined identifiers	133
Table B.5 – MENU predefined identifiers	133
Table B.6 – METHOD predefined identifiers.....	134
Table B.7 – VARIABLE predefined identifiers.....	134

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**DEVICES AND INTEGRATION IN ENTERPRISE SYSTEMS –
FUNCTION BLOCKS (FB) FOR PROCESS CONTROL AND
ELECTRONIC DEVICE DESCRIPTION LANGUAGE (EDDL) –****Part 4: EDD interpretation****FOREWORD**

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC 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 National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees 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, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) 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.
- 9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61804-4 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation.

This second edition cancels and replaces the first edition published in 2015. This edition constitutes a technical revision.

This edition was developed by merging material from multiple variants of existing EDDL specifications including those from FieldComm Group (Foundation™ Fieldbus¹, HART®²), PROFIBUS™³ Nutzerorganisation e.V. (PNO), and ISA100_Wireless™⁴ Compliance Institute (ISA100 WCI). When a profile deviation exists, it is now indicated in the context where the related deviation is found. As a result, the formatting and numbering of this edition may be different from any of the individual specifications from which this edition was derived.

This edition includes the following significant technical changes with respect to the previous edition:

- communication profiles ISA100 and GPE were added;
- description of rules for optimized-column-width layout have been added;
- description of the concatenation of labels and help was added;
- color banding for meter type charts was added.

The text of this International Standard is based on the following documents:

CDV	Report on voting
65E/633/CDV	65E/690/RVC

Full information on the voting for the approval of this International Standard can be found in the report on voting indicated in the above table.

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

A list of all parts in the IEC 61804 series, published under the general title *Devices and integration in enterprise systems – Function blocks (FB) for process control and Electronic Device Description Language (EDDL)*, can be found on the IEC website.

Future standards in this series will carry the new general title as cited above. Titles of existing standards in this series will be updated at the time of the next edition.

¹ FOUNDATION™ Fieldbus is the trademark of FieldComm Group. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

² HART® is the registered trademark of FieldComm Group. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

³ PROFIBUS and PROFINET are the trademarks of the PROFIBUS Nutzerorganisation e.V. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

⁴ ISA100_Wireless™ is the trademark of ISA100 Wireless Compliance Institute. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on IEC web site under "<http://webstore.iec.ch>" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

This part of IEC 61804

- contains an overview of the use of EDDL;
- provides examples demonstrating the use of the EDDL constructs;
- shows how the use cases are fulfilled; and
- shows the proper EDD application interpretation for each example.

This part of IEC 61804 is not an EDDL tutorial and is not intended to replace the EDDL specification.

Instructions are provided for the EDD application, which describe what will be performed without prescribing the technology used in the host implementation. For example, the FILE construct describes data that is stored by the EDD application on behalf of the EDD. The FILE construct does not specify how the data is stored. The EDD application can use a database, a flat file, or any other implementation it chooses.

EDDL features are limited by profile for each of the communication technologies. The descriptions in this part of IEC 61804 refer to these features in a general sense and not all communication technologies will support all of the features described. The profile definitions in IEC 61804-3 are referred to in order to understand the features supported by each communication technology.

DEVICES AND INTEGRATION IN ENTERPRISE SYSTEMS – FUNCTION BLOCKS (FB) FOR PROCESS CONTROL AND ELECTRONIC DEVICE DESCRIPTION LANGUAGE (EDDL) –

Part 4: EDD interpretation

1 Scope

This part of IEC 61804 specifies EDD interpretation for EDD applications and EDDs to support EDD interoperability. This document is intended to ensure that field device developers use the EDDL constructs consistently and that the EDD applications have the same interpretations of the EDD. It supplements the EDDL specification to promote EDDL application interoperability and improve EDD portability between EDDL applications.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements 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.

IEC 61784-1, *Industrial communication networks – Profiles – Part 1: Fieldbus profiles*

IEC 61784-2, *Industrial communication networks – Profiles – Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC/IEEE 8802-3*

IEC 61804-3, *Devices and integration in enterprise systems – Function blocks (FB) for process control and electronic device description language (EDDL) – Part 3: EDDL syntax and semantics*

IEC 61804-5, *Devices and integration in enterprise systems – Function blocks (FB) for process control and electronic device description language (EDDL) – Part 5: EDDL Built-in library*

IEC 62734, *Industrial networks – Wireless communication network and communication profiles – ISA 100.11a*

IEC 62769-4⁵, *Field Device Integration (FDI) – Part 4: FDI Packages*

IEC 62769-7⁶, *Field Device Integration (FDI) – Part 7: FDI Communication devices*

3 Terms, definitions, abbreviated terms, acronyms and conventions

3.1 General terms and definitions

For the purposes of this document, the terms and definitions given in IEC 61804-3 and the following apply.

⁵ Under preparation. Stage at the time of publication: IEC RFDIS 62769-4:2020.

⁶ Under preparation. Stage at the time of publication: IEC RFDIS 62769-7:2020.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at <http://www.electropedia.org/>
- ISO Online browsing platform: available at <http://www.iso.org/obp>

3.1.1

EDD developer

individual or team that develops an EDD

3.1.2

container

user interface elements that contain other user interface elements

Note 1 to entry: Containers can include menus, windows, dialogs, tables, pages, groups, and other containers.

3.1.3

contained item

user interface elements that can be contained in containers

Note 1 to entry: Contained items can include variables, methods, graphs, charts, images, static text.

3.1.4

device developer

individual or team that develops a device and an EDD that describes the device

3.1.5

handheld

device with limited display resolution that restricts EDD applications user interface

3.2 Terms and definitions related to modular devices

3.2.1

channel

connection to a process that is being measured or controlled

3.2.2

component

software or hardware item contained within the modular device concept

Note 1 to entry: A component cannot function separately from a modular device hosting it. A component may support one or more types of modular devices.

3.2.3

interface

basic declarations of basic constructs

Note 1 to entry: An interface defines all public parts that components may use.

3.2.4

modal window

child window that requires users to interact with it before they can return to operating the parent application, thus preventing the workflow on the application main window

3.2.5

modular device

device that can contain a variety of software and/or hardware components

3.3 Abbreviated terms and acronyms

CP	Communication Profile
CPF	Communication Profile Family
CS	Communication Server
EDD	Electronic Device Description
EDDL	Electronic Device Description Language
FDI	Field Device Integration
FF	Foundation Fieldbus
GPE	Generic Protocol Extension
HART	Highway Addressable Remote Transducer
ISA100	ISA100_Wireless™
PB	PROFIBUS
PC	Personal computer
PI	PROFIBUS and PROFINET International
PI PROFILE PA	PI specific profile for Process Automation field devices
PN	PROFINET

3.4 Conventions

There exists some differences using and interpreting EDDL based on the used communication network and device model. Annex C includes a description of each of the EDDL profiles.

The different communication networks used within this part of IEC 61804 are:

- HART (according to IEC 61784-1 CPF9)
- FOUNDATION fieldbus (according to IEC 61784-1 CPF1)
- PROFIBUS (according to IEC 61784-1 CPF3)
- PROFINET (according to IEC 61784-2 CPF3)
- Communication Server
 - IEC 62769-4 FDI Technical Specification: Part 4: FDI Packages
 - IEC 62769-7 FDI Technical Specification: Part 7: Communication Devices
- ISA100_Wireless (according to IEC 62734)
- GPE (Generic Protocol Extension)
 - IEC 62769-4 FDI Technical Specification: Part 4: FDI Packages
 - IEC 62769-7 FDI Technical Specification: Part 7: Communication Devices.

EDD examples in this document show parts of EDD implementations and are incomplete.

EDDL keywords are written in upper case letters. If more than one basic construct item is meant the keyword is written in uppercase letters added with a lower case 's' for example VARIABLES.

The capitalized word Builtin refers to functions specified in IEC 61804-5.

EXAMPLE Builtin MenuDisplay refers to the Builtin named MenuDisplay specified in IEC 61804-5.

4 EDDL user interface description

4.1 Overview

Most EDD applications can be characterized as either a PC application or a handheld application. Due to the relatively small screen of a handheld device, handheld applications can only display a small amount of information at any given time. On the other hand, PC applications can provide a much more beneficial user interface, largely due to their larger screen size.

To support the capabilities of PC applications, the MENU construct has been extended in IEC 61804-3 compared to previous definitions in IEC 61804-2:2004. Due to the differences in the user interfaces of PC applications and handheld applications, it is expected that many devices will define two MENU hierarchies – one for handheld applications and the other for PC applications. Some MENUs may be used in both hierarchies. Therefore, the entire hierarchy does not need to be specified twice.

Different menu structures for different classes of applications are possible. This document shall be used to create menu structures in an EDD that are interpreted by applications in an unambiguous way. To provide interoperability across applications, this document shall be followed. Annex B lists predefined identifiers that may be used by EDD applications to access standardized information.

4.2 Menu conventions for handheld applications

EDD applications use specific menus in the EDDs to show the user interface of the device (see Table 1). In addition, user interface items can be described for handhelds and PCs at once. For handhelds, strings can have beside a language code, a specific country code zz to specify shorter strings or lower resolution images (see IEC 61804-3). In order to fit in a limited display space, a handheld application may render all menus using STYLE TABLE. Handheld applications shall render references to images in MENUs of STYLE TABLE always as hyperlink.

Table 1 – List of defined root menu identifiers for handhelds

Menu identifier	Default STYLE	Short description
root_menu	TABLE	Handheld root menu for HART devices. This is mandatory for all HART EDDs.
Menu_Top*	TABLE	Legacy root menu prefix for BLOCK_A MENU_ITEMS for FOUNDATION fieldbus and ISA100_Wireless devices e.g. Menu_Top_TB.
offline_root_menu	TABLE	Offline configuration. Handhelds may optionally support this menu.
hh_process_variables_root_menu	TABLE	Process variable views for FOUNDATION fieldbus and HART devices.
hh_device_root_menu	TABLE	Device feature views for set up for FOUNDATION fieldbus and HART devices.
hh_diagnostic_root_menu	TABLE	Diagnostic views for FOUNDATION fieldbus and HART devices.

4.3 Menu conventions for PC-based applications

4.3.1 Overview

EDD applications use special menus in the EDDs to show the user interface of the device. Such menus are defined for diagnostic, process variables, device features, and offline configuration. Table 2 defines identifiers for the different root menus with its default STYLES. The default STYLE is used by the EDD application if the STYLE attribute is not defined in the MENU. The “Usage” column in Table 2 defines how the EDD application has online or offline access to the device parameter (see Clause 8).

Table 2 – List of defined root menu identifiers for PC-based devices

MENU identifier	Default STYLE	Usage	Short description
device_root_menu	MENU	Online	Device feature views for set up
diagnostic_root_menu	MENU	Online	Diagnostic views
maintenance_root_menu	MENU	Online	Maintenance feature views
offline_root_menu	TABLE	Offline	Offline configuration
process_variables_root_menu	MENU	Online	Process variable views

For HART EDDs, the EDD application shall display offline_root_menu with a default STYLE WINDOW if STYLE is not defined.

PROFIBUS and PROFINET allow in submenus to define different parameter access by defining ACCESS ONLINE or ACCESS OFFLINE.

The EDDs should contain the online root menus from Table 2. The online root menus are optional for PROFIBUS, PROFINET and FOUNDATION fieldbus. The offline root menu is optional for FOUNDATION fieldbus and ISA100_Wireless. HART EDDs shall have root_menu, in addition to all root menus in Table 2. GPE shall have at least one Online menu. GPE shall have the offline_root_menu. Communication Server shall have at least one Online menu.

4.3.2 Online root menus

4.3.2.1 General

If none of the online root menus exists then communication profile specific entry points shall be used (see Table 3).

Table 3 – Fall back alternatives for online root menus

Communication profile	Description of fall back alternatives
FF, ISA100	Menus declared in the BLOCK_A MENU_ITEMS attribute, e.g. device_root_menu_aiblock. If block based menus are not defined, then a MENU of STYLE TABLE shall be generated from the BLOCK_A PARAMETERS attribute.
HART	root_menu, in STYLE TABLE.
PB, PN	Menu_Main_Specialist if exists or Menu_Main_Maintenance. These menus include online and offline sub-menus. These would be rendered as STYLE MENU.

4.3.2.2 Diagnostic root menu

The diagnostic_root_menu includes views that show the device state, detailed diagnostic information and may include graphical views that show, for example, a valve signature.

4.3.2.3 Process variables root menu

The `process_variables_root_menu` includes views that show process measurements and set points with their quality and important information for process operators, for example, ranges.

4.3.2.4 Device root menu

The `device_root_menu` includes features for a device. These features can be split into process-related and device-specific features. This structure is not required if the number of features is too small for splitting. Submenus that represent any structuring are allowed on the device feature menu. In case of such submenus, the menus that are underneath can be split into process-related and device-specific features.

4.3.2.5 Maintenance root menu

The `maintenance_root_menu` should contain features for maintaining the device during the runtime phase and e.g. showing information about last maintenance inspection.

4.3.3 Offline root menu

The `offline_root_menu` is a menu hierarchy including e.g. data items and methods for offline configuration. It contains, in particular, all of the application-specific parameters of the device and may also contain important read-only and writeable variables. For more information about application-specific parameters, see 9.3. The menu can have offline methods, for example, configuration assistants.

If the offline root menu does not exist, the communication profile specific entry point shall be used (see Table 4).

Table 4 – Fall back alternatives for offline root menus

Communication profile	Description of fall back alternatives
FF, ISA100	BLOCK_A PARAMETERS
HART	MENU upload_variables
PB, PN	Table_Main_Specialist if exists or Table_Main_Maintenance

4.3.4 Example of EDD menu structure

The EDD example in Figure 1 includes additional menus that can be added to an EDD for PC applications. These menus are additional to the existing menus for the applications. This example is specific for a device with an interface according to HART. Examples for devices with an interface according FOUNDATION fieldbus, ISA100_Wireless, PROFIBUS, PROFINET, Communication Server profile and GPE profile would be very similar.

```

MENU diagnostic_root_menu
{
    LABEL "Diagnostics";
    STYLE MENU;                                /* not required to define STYLE in this case, */
                                              /* because of default STYLE of the root menu */

    ITEMS
    {
        status_window,                        /* menu: style=window */
        self_test                             /* method */
    }
}

MENU status_window
{
    LABEL "Status";
    STYLE WINDOW;
    ITEMS
    {
        standard_diagnostics_page,           /* menu: style=page */
        devspec_diagnostics_page             /* menu: style=page */
    }
}

MENU standard_diagnostics_page
{
    LABEL "Standard";
    STYLE PAGE;
    ITEMS
    {
        device_status                         /* variable */
    }
}

MENU devspec_diagnostics_page
{
    LABEL "Device Specific";
    STYLE PAGE;
    ITEMS
    {
        xmtr_specific_status_1,              /* variable */
        xmtr_specific_status_2               /* variable */
    }
}

METHOD self_test
{
    LABEL "Self Test";
    DEFINITION
    {
        /* elided */
    }
}

MENU maintenance_root_menu
{
    LABEL "Maintenance";
    STYLE MENU;                                /* not required to define STYLE in this case, */
                                              /* because of default STYLE of the root menu */

    ITEMS
    {
        device_mode_dialog,                  /* menu: style=dialog */
        teach_in                             /* method */
    }
}

MENU device_mode_dialog
{
    LABEL "Device Mode";
    STYLE DIALOG;
    ITEMS
    {
        mode_page                             /* menu: style=page */
    }
}

MENU mode_page
{

```

```

    LABEL "Process Variables";
    STYLE PAGE;
    ITEMS
    {
        transducer_group,      /* menu: style=group */
        function_group         /* menu: style=group */
    }
}

MENU transducer_group
{
    LABEL "Transducer";
    STYLE GROUP;
    ITEMS
    {
        trans_target_mode,    /* variable */
        trans_actual_mode     /* variable */
    }
}

MENU function_group
{
    LABEL "Function";
    STYLE GROUP;
    ITEMS
    {
        func_target_mode,     /* variable */
        func_actual_mode      /* variable */
    }
}

METHOD teach_in
{
    LABEL "Teach-in";
    DEFINITION
    {
        /* elided */
    }
}

MENU process_variables_root_menu
{
    LABEL "Process Variables";
    STYLE MENU;
    /* not required to define STYLE in this case, */
    /* because of default STYLE of the root menu */

    ITEMS
    {
        overview_window,      /* menu: style=window */
        primary_vars_window   /* menu: style=window */
    }
}

MENU overview_window
{
    LABEL "Overview";
    STYLE WINDOW;
    ITEMS
    {
        process_vars_page     /* menu: style=page */
    }
}

MENU process_vars_page
{
    LABEL "Process Variables";
    STYLE PAGE;
    ITEMS
    {
        pressure_group,       /* menu: style=group */
        temperature_group     /* menu: style=group */
    }
}

MENU pressure_group
{
    LABEL "Pressure";
    STYLE GROUP;

```

```

ITEMS
{
    pv_digital_value,          /* variable */
    pv_upper_range_value,     /* variable */
    pv_lower_range_value      /* variable */
}
}

MENU temperature_group
{
    LABEL "Temperature";
    STYLE GROUP;
    ITEMS
    {
        sv_digital_value,          /* variable */
        sv_upper_range_value,     /* variable */
        sv_lower_range_value      /* variable */
    }
}

MENU primary_vars_window
{
    LABEL "Primary Variables";
    STYLE WINDOW;
    ITEMS
    {
        pressure_chart_page,      /* menu: style=page */
        temperature_chart_page    /* menu: style=page */
    }
}

MENU pressure_chart_page
{
    LABEL "Pressure";
    STYLE PAGE;
    ITEMS
    {
        pressure_chart            /* chart */
    }
}

CHART pressure_chart
{
    /* elided */
}

MENU temperature_chart_page
{
    LABEL "Temperature";
    STYLE PAGE;
    ITEMS
    {
        temperature_chart        /* chart */
    }
}

CHART temperature_chart
{
    /* elided */
}

MENU device_root_menu
{
    LABEL "Device";
    STYLE MENU;
    /* not required to define STYLE in this case, */
    /* because of default STYLE of the root menu */

    ITEMS
    {
        process_related_window,    /* menu: style=window */
        device_specific_window,    /* menu: style=window */
        master_reset               /* method */
    }
}

MENU process_related_window
{
    LABEL "Process Related";

```

```

STYLE WINDOW;
ITEMS
{
    identification_page,      /* menu: style=page */
    output_info_page         /* menu: style=page */
}
}

MENU identification_page
{
    LABEL "Identification";
    STYLE PAGE;
    ITEMS
    {
        tag,                  /* variable */
        manufacturer,         /* variable */
        device_type,          /* variable */
        device_revision,      /* variable */
        descriptor,           /* variable */
        message               /* variable */
    }
}

MENU output_info_page
{
    LABEL "Output Information";
    STYLE PAGE;
    ITEMS
    {
        range_values_group,    /* menu: style=group */
        sensor_limits_group    /* menu: style=group */
    }
}

MENU range_values_group
{
    LABEL "Range Values";
    STYLE GROUP;
    ITEMS
    {
        pv_units,             /* variable */
        pv_urv,               /* variable */
        pv_lrv                /* variable */
    }
}

MENU sensor_limits_group
{
    LABEL "Sensor Limits";
    STYLE GROUP;
    ITEMS
    {
        sensor_units,         /* variable */
        upper_sensor_limit,    /* variable */
        lower_sensor_limit     /* variable */
    }
}

MENU device_specific_window
{
    LABEL "Device Specific";
    STYLE WINDOW;
    ITEMS
    {
        identification_page,    /* menu: style=page */
        calibration_page        /* menu: style=page */
    }
}

MENU calibration_page
{
    LABEL "Calibration";
    STYLE PAGE;
    ITEMS
    {
        sensor_limits_group,    /* menu: style=group */
        sensor_trim_group       /* menu: style=group */
    }
}

```

```

    }
}

MENU sensor_trim_group
{
    LABEL "Sensor Trim";
    STYLE GROUP;
    ITEMS
    {
        upper_sensor_trim_point, /* variable */
        lower_sensor_trim_point, /* variable */
        sensor_trim               /* method */
    }
}

METHOD master_reset
{
    LABEL "Master Reset";
    DEFINITION
    {
        /* elided */
    }
}

```

Figure 1 – EDD example of root menus

4.3.5 User interface

4.3.5.1 Diagnostics

Figure 2 shows an example of an EDD application for diagnostics.

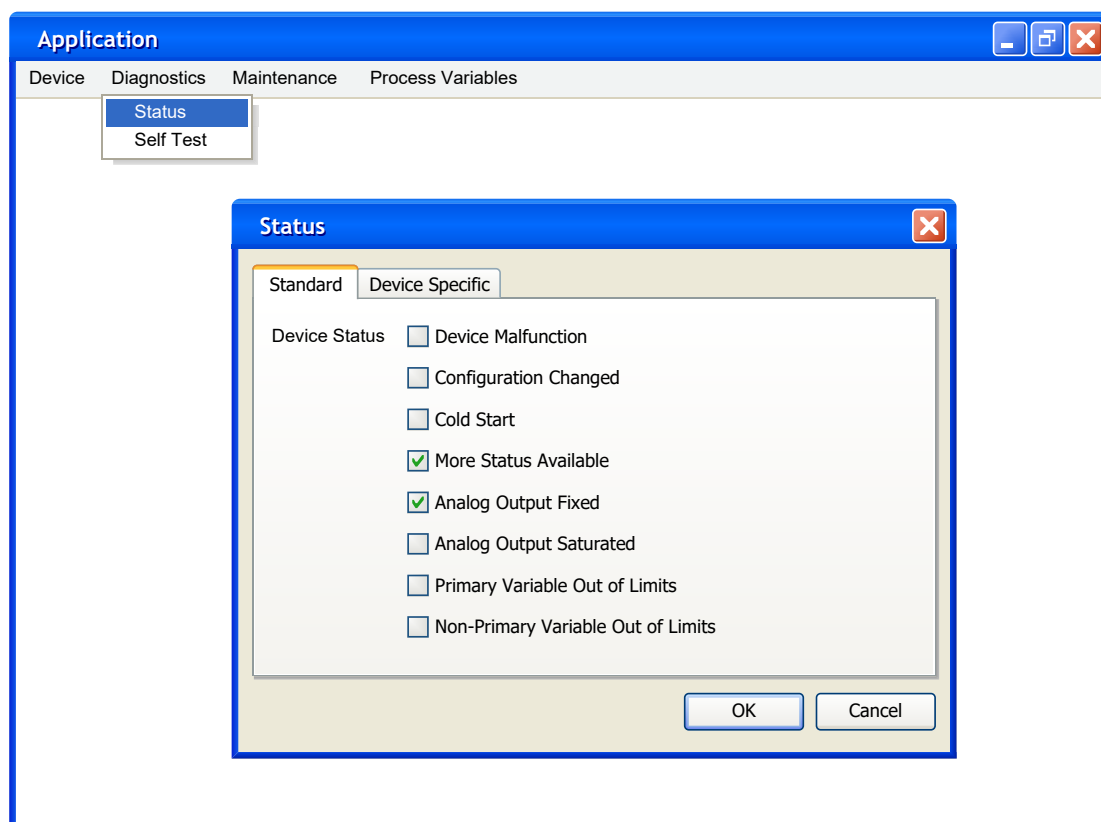


Figure 2 – Example of an EDD application for diagnostics

4.3.5.2 Process variables

Figure 3 and Figure 4 show examples of EDD applications for process variables.

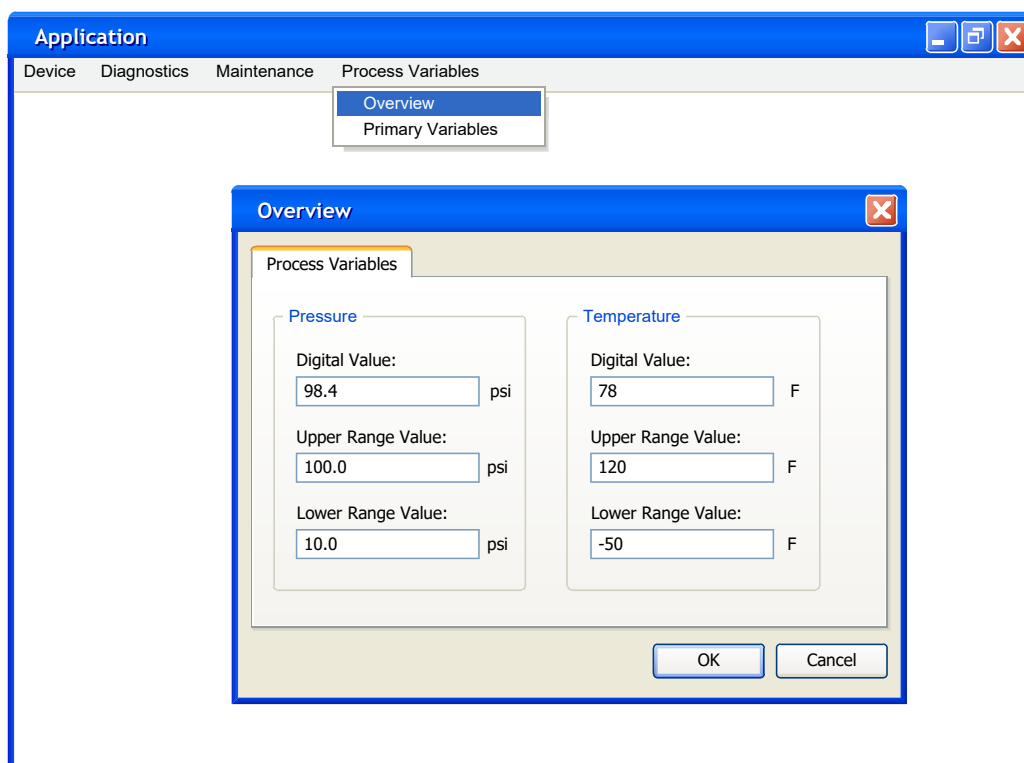


Figure 3 – Example of an EDD application for process variables

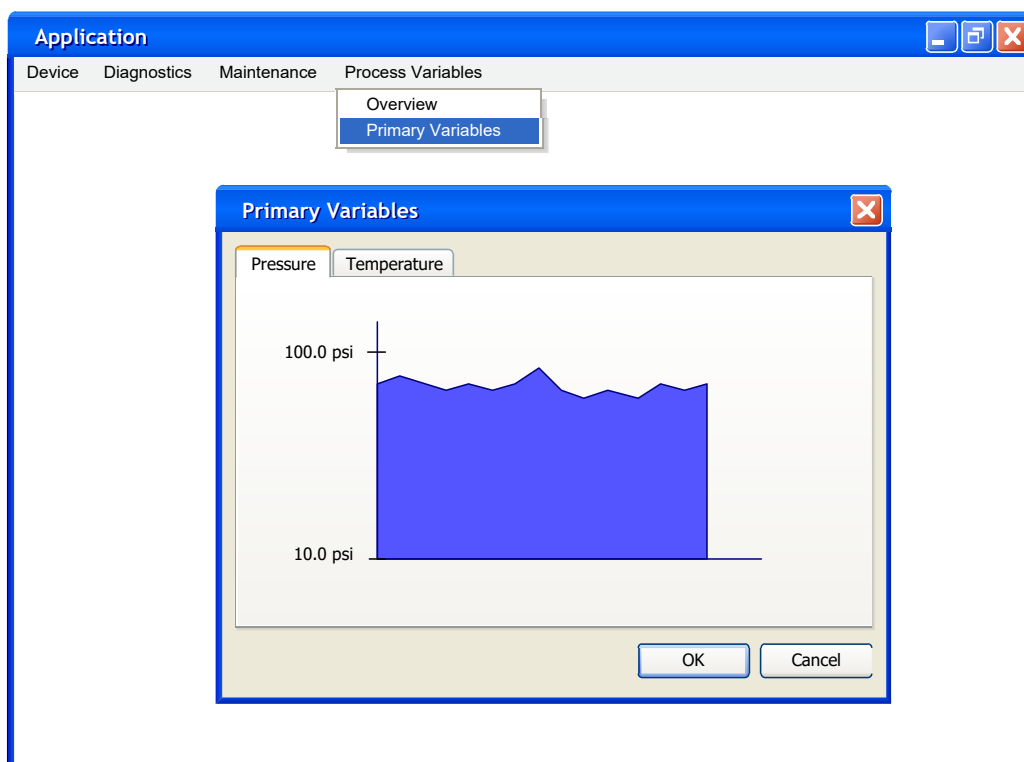


Figure 4 – Example of an EDD application for primary variables

4.3.5.3 Device features

Figure 5, Figure 6 and Figure 7 show examples of EDD applications for device features.

The screenshot shows the 'Application' window with tabs for Device, Diagnostics, Maintenance, and Process Variables. The 'Process Variables' tab is active, and a sub-menu is open showing 'Process Related', 'Device Specific', and 'Master Reset'. The 'Process Related' dialog box is open, showing the 'Output Information' tab. It contains two sections: 'Range Values' and 'Sensor Limits'. Both sections have a 'Units' dropdown set to 'psi'. The 'Range Values' section has 'Upper Range Value' set to 100.0 and 'Lower Range Value' set to 10.0. The 'Sensor Limits' section has 'Upper Sensor Limit' set to 250 and 'Lower Sensor Limit' set to 0. 'OK' and 'Cancel' buttons are at the bottom.

Figure 5 – Example of an EDD application for process-related device features

The screenshot shows the 'Application' window with tabs for Device, Diagnostics, Process Variables, and Maintenance. The 'Diagnostics' tab is active, and a sub-menu is open showing 'Process Related', 'Device Specific', and 'Master Reset'. The 'Device Specific' dialog box is open, showing the 'Calibration' tab. It contains two sections: 'Sensor Limits' and 'Sensor Trim'. Both sections have a 'Units' dropdown set to 'psi'. The 'Sensor Limits' section has 'Upper Sensor Limit' set to 250 and 'Lower Sensor Limit' set to 0. The 'Sensor Trim' section has 'Upper Trim Point' set to 250 and 'Lower Trim Point' set to 0, with a 'Trim Sensor' button below. 'OK' and 'Cancel' buttons are at the bottom.

Figure 6 – Example of an EDD application for device features

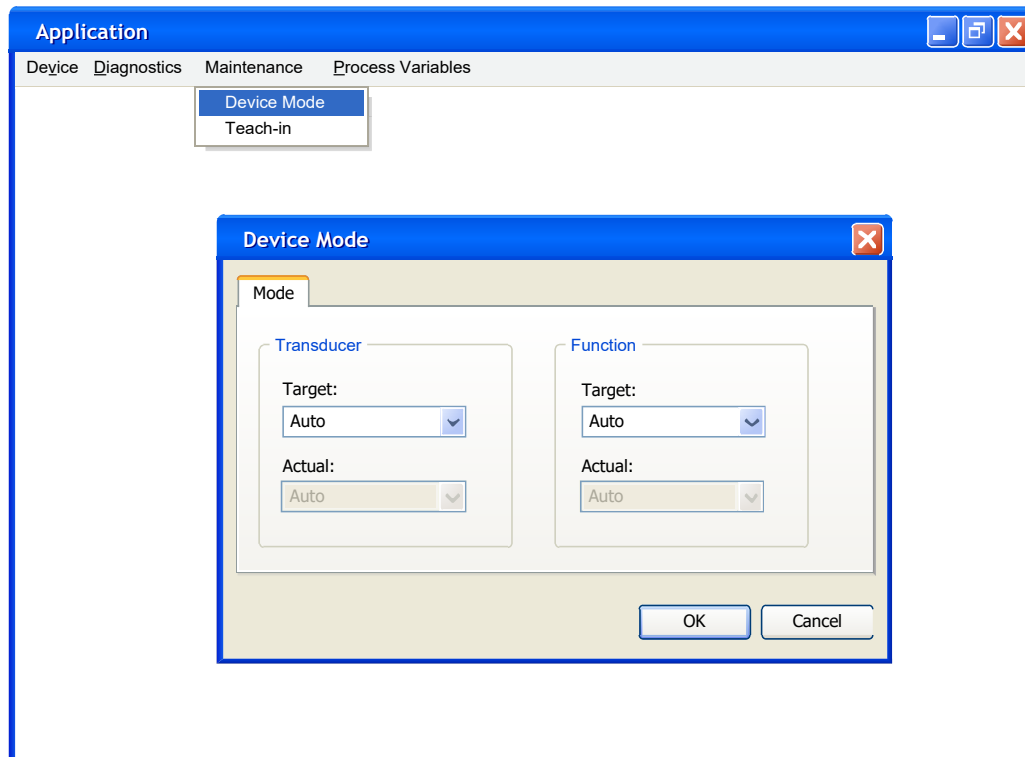


Figure 7 – Example of an EDD application for maintenance features

4.4 Label concatenation for indirect variable references

4.4.1 General

For indirect variable-references only, the LABEL displayed shall be a concatenation of the strings found in the EDD item containers through which the VARIABLE was referenced. This occurs, for example, when a VARIABLE is referenced via a COLLECTION MEMBER or ARRAY ELEMENT. For example, PV.DEVICE_VARIABLE.DIGITAL_VALUE is an indirect reference to a VARIABLE via both arrays and collections. A reference is defined as:

```
reference:= [ identifier | reference [ expression ] |
             reference . identifier ]
```

To specify Label concatenation, variable-references will be classified as a direct, simple, or complex reference.

A "direct reference" is:

```
direct-reference:= identifier
```

String concatenation is not applicable to direct references.

A "simple reference" is:

```
simple-reference:= [ identifier [ expression ] | identifier . identifier ]
```

And a "complex reference" is:

```
complex-reference:= [ simple-reference . identifier |
                     simple-reference [ expression ] |
                     complex-reference . identifier |
                     complex-reference [ expression ] ]
```

Label concatenation is only applicable to simple and complex (indirect) variable-references. Direct variable-references do not have concatenated Labels.

In the following subclauses, any label or description that holds an empty string ("") shall be treated as if it does not exist.

4.4.2 Simple variable references

For simple variable-references, the Label and Description of the simple variable-reference are space concatenated together. See the summary in Table 5 and Table 6.

Table 5 – Label rule summary for simple variable references

Reference type	Label + description exists	Only label exists	Only description exists	Neither label nor description exists
FILE REFERENCE_ARRAY COLLECTION	Label shall be the space concatenation of the term's Label and the term's Description.	Label shall be the term's Label.	Label shall be the term's Description.	Same Label as direct variable-reference.
RECORD	Label shall be the term's Description.	Label shall be the term's Label.	Label shall be the term's Description.	Same Label as direct variable-reference.

Table 6 – Label rule summary for simple variable references

Reference type	Label exists	
	Yes	No
LIST VALUE_ARRAY	Label shall be Label "[n]" where n is the value of the index.	Label shall be "[n]" where n is the value of the index.

4.4.3 Complex variable references

The LABEL for a complex variable-reference is constructed by space concatenating the Prefix, Body and Suffix strings together.

Prefix-string

The LABEL Prefix-string is generated based on the first term in the complex variable-reference. In the case of a file, reference-array, block, record, or collection reference, the first non-null Description-string is used. See the Prefix rules summarized in Table 7 and Table 8.

Notice that if a FILE, REFERENCE_ARRAY, BLOCK or COLLECTION reference is the first term in the complex variable-reference and it does not have a corresponding Description-string, the search for a Prefix-string continues with the next term in the complex variable-reference. This process is repeated until the Prefix-string is generated or final term in the complex variable-reference is encountered. If the Prefix-string is generated the process continues with the construction of the Body and Suffix strings.

If the final term is encountered with no Prefix-string being generated then the rules for simple variable-references are used to generate the LABEL using the final term in the complex variable-reference.

Table 7 – Prefix rule summary for complex variable references

Reference type	Description exists	
	Yes	No
FILE REFERENCE_ARRAY COLLECTION RECORD BLOCK_A	Prefix is Description.	Skip this term. Continue to next term in the complex variable-reference to locate Prefix string.

Table 8 – Prefix rule summary for complex variable references

Reference type	Label exists	
	Yes	No
LIST VALUE_ARRAY	Prefix shall be Label "[n]" where n is the value of the index.	Prefix shall be "[n]" where n is the value of the index.

Body-string

The Body-string is a sequential concatenation of strings generated from the intermediate terms in the complex variable-reference. See Table 9 and Table 10 for a summary of the rules for constructing the Body-string.

Generation of the Body-string ceases when the final term in the complex variable-reference is reached. The Body-string shall be the null (i.e., "") string if the final term is reached before a Body-string is created. The final term in the complex variable-reference shall be used to generate the Suffix.

Table 9 – Body rule summary for complex variable references

Reference type	
FILE COLLECTION RECORD	Nothing is concatenated to the Body-string.
LIST VALUE_ARRAY	[n] is concatenated to the Body-string where n is the value of the index found in the reference.

Table 10 – Body rule summary for complex variable references

Reference type	Prefix is from a FILE, COLLECTION or REFERENCE_ARRAY	
	Yes	No
REFERENCE_ARRAY	Nothing is concatenated to the Body-string.	[n] is concatenated to the Body-string where n is the index.

Suffix-string

The Suffix-string is generated from the final term in the complex variable-reference. See Table 11 and Table 12 for a summary of the rules for constructing the Suffix-string.

Table 11 – Suffix rule summary for complex variable references

Reference type	
LIST VALUE_ARRAY	[n] is Suffix where n is the value of the index.

Table 12 – Suffix rule summary for complex variable references

Reference type	Description exists	
	Yes	No
FILE REFERENCE_ARRAY COLLECTION RECORD	Suffix is Description.	No Suffix (i.e., an empty string).

4.5 Help concatenation

4.5.1 General

The Help-string for a simple or complex variable-reference is generated in a manner similar to that used for the Label-string and depends on whether it is a simple or complex variable-reference.

In the following clauses, any help that holds an empty string (“”) shall be treated the same as if the help does not exist.

4.5.2 Simple variable references

For simple variable-references, the help string is generated using the rules in Table 13 and Table 14.

Table 13 – Help rule summary for simple variable references

Reference type	Container's help + member/element help exists	Only container's help exists	Only member/element help exists	Neither help exists
FILE REFERENCE_ARRAY COLLECTION RECORD	Help shall be space concatenation of both Help-strings.	Help shall be Container's Help.	Help shall be Member/Element Help.	Help shall be same as with direct variable-reference.

Table 14 – Help rule summary for simple variable references

Reference type	Container's help exists	
	Yes	No
LIST VALUE_ARRAY	Help shall be Container's Help.	Help shall be an empty string.

4.5.3 Complex variable references

For a complex variable-reference, in general, there is possibly a Help-string Prefix and Suffix. Unlike Label concatenation, there is no Help-string Body. Basically, the Help string is the first-found Help space-concatenated to the Help from the final term in the complex variable-reference. If both the Help Prefix and Suffix are empty strings then the Help-string shall be an empty string.

The Help Prefix is generated based on the first term in the complex variable-reference. In the case of a FILE, REFERENCE_ARRAY, or COLLECTION reference, the first non-empty Member/Element Help is used. See the Prefix rules summarized in Table 15 and Table 16.

If the final term is encountered with no Prefix being generated then the rules for simple variable-references are used to generate the Help based on the final term in the complex variable-reference.

Table 15 – Help prefix rule summary for complex variable references

Reference type	
LIST VALUE_ARRAY	Prefix is Help string from the list or value-array (possibly empty).

Table 16 – Help prefix rule summary for complex variable references

Reference type	Member/element help exists	
	Yes	No
FILE REFERENCE_ARRAY COLLECTION RECORD	Help Prefix is Member/Element Help string.	Skip this reference. Continue to next reference to locate HELP Prefix string.

The Help suffix is generated from the final term in the complex variable-reference (see Table 17 and Table 18).

Table 17 – Help suffix rule summary for complex variable references

Reference type	
LIST VALUE_ARRAY	Suffix is Help string from referenced VARIABLE (possibly empty).

Table 18 – Help suffix rule summary for complex variable references

Reference type	Member/element help exists	
	Yes	No
FILE REFERENCE_ARRAY COLLECTION RECORD	Help Suffix is Member/Element Help string.	No Suffix (i.e., an empty string).

4.6 Containers and contained items

4.6.1 Overview

The user interface extensions are based on a simple user interface model. The model consists of two concepts:

- containers; and
- contained items.

Containers are so named because they contain other user interface elements. Containers may include: menus, windows, dialogs, tables, pages, groups, and other containers. Containers correspond to a MENU. They are distinguished from one another via the STYLE attribute. This STYLE attribute indicates how the MENU will be displayed.

Contained items include e.g. variables, methods, edit displays, graphs, charts, images, static text.

4.6.2 Permitted and default STYLEs

Table 19 defines permitted user interface items in containers, substitutes and default STYLEs are not defined. If the style of a menu is not defined, then a default style may be used depending on the menu where the menu is contained and the content of the menu.

General rules:

If the ACCESS online or offline of a contained MENU is different to the access of the container, the EDD application shall use STYLE WINDOW in a window otherwise shall use STYLE DIALOG.

Table 19 – Permitted contained items and default STYLES

Container (Parent)	Permitted contained items (Child items)																Default STYLE of contained items (used if STYLE is not defined or in case of conflicts)	
	MENU	DIALOG, WINDOW	PAGE	GROUP	TABLE	EDIT_DISPLAY	VARIABLE	COLLECTION, RECORD	ARRAY, LIST	CHART, GRAPH, GRID	IMAGE	static text	METHOD	PLUGIN	COLUMNBREAK, ROWBREAK	SEPARATOR		
MENU	✓	1	✗	✗	10	✓	8	D	8	2	2	✓	✓	✓	6	6	Communication profiles: CS, FF, GPE, ISA100, PB, PN MENU if no data item is in the menu, otherwise DIALOG. Communication profiles: HART MENU	
DIALOG, WINDOW	✓	1	✓	✓	7	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	5	PAGE	
PAGE	✓	1	✗	✓	7	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	5	GROUP	
GROUP	✓	1	✗	✓	7	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	5	GROUP if the parent of the parent GROUP is not of STYLE GROUP. DIALOG rendered as button or hyperlink if the parent of the parent GROUP is of STYLE GROUP.	
TABLE	✗	9	✗	✗	✓	✓	3	D	✓	2	3	✓	✓	✓	6	6	TABLE	

Key

- ✓ Shall be rendered as defined for the contained item; e.g., for a “MENU” according to its STYLE attribute.
- ✗ Not allowed. Default STYLE shall be used to solve this conflict (see column “Default STYLE of contained items”).
- D Default STYLE shall be used (see column “Default STYLE of contained items”).
- 1 Shall be rendered as button or hyperlink by using the LABEL. The item shall be presented when the button or hyperlink is selected.
- 2 Shall be rendered as a button or hyperlink by using the LABEL. The item shall be presented on a dialog (i.e., a modal window), when the button or hyperlink is selected.
- 3 Shall be rendered as button or hyperlink by using the LABEL, or inline. If not rendered inline, then the item shall be presented on a dialog (i.e., a modal window), when the button or hyperlink is selected.
- 4 Shall be rendered as a button or hyperlink by using the LABEL. The item shall be presented on a modeless window, when the button or hyperlink is selected.
- 5 Shall be interpreted as COLUMNBREAK.
- 6 Shall be rendered as a line or ignored.
- 7 **Communication profiles: CS, FF, GPE, ISA100, PB, PN**
Shall be rendered according to note “✓”.
- Communication profiles: HART**
Shall be rendered according to note “✗”.
- 8 **Communication profiles: CS, FF, GPE, ISA100, PB, PN**
Shall be rendered according to note “2”.
- Communication profiles: HART**
Shall be ignored and not presented.
- 9 **Communication profiles: CS, FF, GPE, ISA100, PB, PN**
Shall be rendered according to note “1”.
- Communication profiles: HART**
Shall be rendered according to note “✗”.
- 10 **Communication profiles: CS, FF, GPE, ISA100, PB, PN**
Shall be rendered according to note “4”.
- Communication profiles: HART**
Shall be rendered according to note “D” if subordinate to DIALOG or WINDOW, otherwise according to note “4”.

4.6.3 Containers

4.6.3.1 Menu

Communication profiles: CS, FF, GPE, ISA100, PB, PN

A menu of STYLE MENU takes the form of a drop-down menu, a pop-up menu or a navigation bar.

Communication profiles: HART

A menu of STYLE MENU takes the form of a drop-down menu or a pop-up menu.

4.6.3.2 Window

A menu of STYLE WINDOW takes the form of a modeless window. The LABEL of the menu shall be used as the caption for the window.

4.6.3.3 Dialog

A MENU of STYLE DIALOG shall be a modal window. If a dialog contains a subordinated window, the EDD application shall manage the window as a modal subdialog. The LABEL of the menu shall be used as the caption for the window.

4.6.3.4 Table

Communication profiles: CS, FF, GPE, ISA100, PB, PN

A menu of STYLE TABLE shall be rendered as a table. Each item shall be represented as one or multiple rows, e.g. ARRAYs. The submenus build a hierarchy, which shall be shown.

Communication profiles: HART

A menu of STYLE TABLE shall be rendered as a cascaded menu tree. Each item shall be represented as one or multiple rows, e.g. ARRAYs. The submenus build a hierarchy, which shall be shown. The label of the MENU-item (including BIT_ENUMERATED VARIABLES) shall be displayed. Selecting MENU item shall cause the contents to be displayed.

4.6.3.5 Page

A menu of STYLE PAGE within a WINDOW or DIALOG shall be rendered as a tab and spans the entire canvas width. When multiple pages (tabs) are rendered in the same window or dialog, each tab shall be individually selectable to enable the contents of a particular tab. The LABEL of the menu shall be used as the caption for the tab. There shall be an implicit ROWBREAK before and after the PAGE.

The EDD application shall interpret ROWBREAKs or other items between pages as horizontal separations.

COLUMNBREAKs and SEPARATORs between pages shall be ignored.

4.6.3.6 Group

A menu of STYLE GROUP takes the form of a group box. The LABEL of the menu shall be used as the caption for the group. In order to maintain a clear arrangement on the user interface, the following restricting rules apply.

- The nesting level of GROUP declarations is restricted to two. In other words, a GROUP may contain further GROUPs but these may not contain further GROUPs.
- When a second layer GROUP contains a GROUP, this GROUP shall be rendered as either a button or a hyperlink. The LABEL shall appear on the button or as the hyperlink text. When the button or hyperlink is pressed, a corresponding dialog shall be opened on top of the calling dialog or window.

Within a GROUP, usually logically related parameters and methods are grouped. The title of the GROUP indicates this relation, for example, “AnalogInput 1”. Often the same string is also used in the labels of the GROUP items, for example, “AnalogInput1_StaticRevision” or “Analog Input1_Blockmode”. In order to avoid this duplication of information, the EDD developer may assemble the items of a GROUP in a COLLECTION in order to override the original labels. If the parameters are to be referred to in a GROUP the parameters can be referenced using the COLLECTION; in other cases, especially without additional semantic context, the parameters can be referenced directly.

Figure 8 shows an EDD example with COLLECTION MEMBERS within a MENU of STYLE GROUP.

```
VARIABLE ail_strev
{
    LABEL "Analog Input 1: Static Revision";
    TYPE INTEGER(4);
}

VARIABLE ail_blobo
{
    LABEL "Analog Input 1: Block Mode";
    TYPE INTEGER(1);
}

MENU ail_group_double                                /* Leads to double display of */
{
    LABEL "Analog Input 1";
    STYLE GROUP;
    ITEMS
    {
        ail_strev,
        ail_blobo
    }
}

COLLECTION ail_groupcol
{
    MEMBERS
    {
        strev,ail_strev,"Static Revision";
        blobo,ail_blobo,"Block Mode";
    }
}

MENU ail_group_single                                /* Leads to single display of */
{
    LABEL "Analog Input 1";
    STYLE GROUP;
    ITEMS
    {
        ail_groupcol.strev,
        ail_groupcol.blobo
    }
}
```

Figure 8 – Usage of COLLECTION MEMBERS in MENUs of STYLE GROUP

4.6.4 Contained items

4.6.4.1 Overview

How a container renders contained items is described in 4.6.3.1. All containers render contained items in the same way. Contained items, such as methods, variables and others, have a LABEL attribute. In order not to disturb the layout of the EDD application, for example, with oversized buttons for methods, the EDD developer should not use very long strings for LABELs. For this reason, some EDD applications may truncate these labels. However, there shall be some methodology, such as a "tool tip", "pop- up window", "expanding window", etc., to display the entire string to the user.

4.6.4.2 Methods

A method should be rendered as a button or a hyperlink. The LABEL of the corresponding METHOD should appear on the button or as the hyperlink text. When the button or hyperlink is pressed, the corresponding METHOD should be executed.

4.6.4.3 Variables

4.6.4.3.1 General

The label, value and, if defined, the units of the variable should be displayed in a manner consistent with the definition of the corresponding VARIABLE.

The general handling of variables are as follows.

- HANDLING = READ or the menu item attribute is READ_ONLY: the value of the variable should not be editable.
- HANDLING = WRITE: the value should not be read from the device due to being referenced as a contained item, e.g. in a MENU.
- HANDLING = READ & WRITE: The value should be editable.
- CLASS = DYNAMIC: the value should be updated continuously with current read values from the device. If the value has been edited, continuous updates shall no longer occur so that the edited value is not overwritten with the value read from the device.

For string type VARIABLES the value should not be truncated or wrapped in multiple lines. Word wrapping could be confused with a true "new line" characters ('\n') in the string value.

When no VARIABLE value is available to be displayed, e.g. due to a change in VALIDITY or HANDLING, the EDD application shall indicate that the variable is in uninitialized state.

Table 20 shows how the EDD application should indicate the uninitialized state of VARIABLES.

Table 20 – Uninitialized state of VARIABLES on user interface

VARIABLE TYPE	Uninitialized state indication
Numeric values ENUMERATED String values Date, time and duration values	Blank with indication of "uncertain" state close to the value field
BIT_ENUMERATED	Indication of no bit set along with indication of "uncertain" state close to the value field
BOOLEAN	Indication of FALSE along with indication of "uncertain" state close to the value field

4.6.4.3.2 Variable of TYPE BIT_ENUMERATED

Multiple bits can be packaged in a single-bit enumerated variable. Each bit could have a distinct meaning. It is possible to display each bit separately and, in addition, to display the whole BIT_ENUMERATED variable. BIT_ENUMERATED variables may be displayed as checkboxes.

In case of showing a bit of a BIT_ENUMERATED variable, the variable reference should be extended with a mask in square brackets and the LABEL of the variable is not shown.

When the whole BIT_ENUMERATED VARIABLE is displayed, the display of the bits in the VARIABLE shall appear grouped together.

When a bit enumeration is of CLASS DIAGNOSTIC, then the bits displayed should be treated as a status indicator. A red indication should be presented when the bit is set and green when the bit is reset.

Communication profiles: HART

When the whole BIT_ENUMERATED VARIABLE is displayed, the display of the bits in the VARIABLE shall be displayed inside a group-box. The LABEL of the BIT_ENUMERATED VARIABLE shall be used as the label of the group box. EDD applications with limited display constraints that prevent the display of group boxes are not required to meet this requirement.

Figure 9 shows an EDD example for displaying single bits of a BIT_ENUMERATED variable.

```
VARIABLE var1
{
    LABEL "Var 1";
    TYPE BIT_ENUMERATED
    {
        { 0x1, "Bit 0"},
        { 0x2, "Bit 1"},
        { 0x4, "Bit 2"}
    }
}

MENU diagnostic_info
{
    LABEL "Diagnostic";
    ITEMS
    {
        var1[0x1],    // Bit 0
        var1[0x2],    // Bit 1
        var1[0x4]     // Bit 2
    }
}
```

Figure 9 – Displaying single bits of BIT_ENUMERATED

Figure 10 shows an EDD example that shows the differences between the full display of a BIT_ENUMERATED variable and how it may be displayed if all bits are explicitly addressed.

```

VARIABLE var1
{
    LABEL "Var 1";
    TYPE BIT_ENUMERATED
    {
        { 0x1, "Bit 0"},
        { 0x2, "Bit 1"},
        { 0x4, "Bit 2"}
    }
}

MENU var1_group
{
    LABEL var1.LABEL;
    STYLE GROUP;
    ITEMS
    {
        var1[0x1],    // Bit 0
        var1[0x2],    // Bit 1
        var1[0x4],    // Bit 2
    }
}

MENU diagnostic_info
{
    LABEL "Diagnostic";
    STYLE PAGE;
    ITEMS
    {
        var1,          // all bit should be shown
        COLUMNBREAK,
        var1[0x1],      // Bit 0
        var1[0x2],      // Bit 1
        var1[0x4],      // Bit 2
        COLUMNBREAK,
        var1_group
    }
}

```

Figure 10 – Displaying multiple bits of BIT_ENUMERATED

Figure 11 shows how the result of the Figure 10 EDD example in an EDD application may appear.

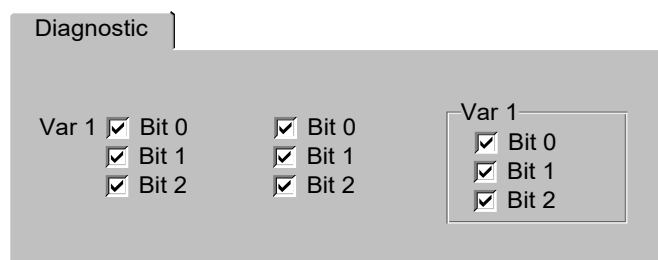


Figure 11 – Example of an EDD application for a variable of type BIT_ENUMERATED

4.6.4.3.3 Variable of TYPE INDEX

When a VARIABLE of type INDEX is presented to the user, the EDD application shall determine the value that shall be displayed by following rules:

- 1) If the description in the referenced ELEMENTS exist then this shall be displayed else;
- 2) If the LABEL of the referenced ELEMENTS exist then this shall be displayed else;
- 3) the LABEL of the ARRAY or LIST shall be displayed with the numeric value of the INDEX e.g. myarray[3].

If the INDEX value is out of range of the related array, a text should inform the user that the value is out of range.

If the variable is editable, it should be presented as a combo box.

4.6.4.3.4 Variable with HANDLING WRITE

According to the HANDLING attribute a VARIABLE with HANDLING WRITE is a "write-only" variable. This means that the variable is editable and may be written to the device, but it should not be read from the device.

For online session, EDD application shall display the VARIABLE value as uninitialized (see Table 20), until the VARIABLE is modified during the same edit session.

For offline session, EDD application shall display the VARIABLE value from the current cache.

Figure 12 shows an EDD example with a "write-only" variable.

```
//--- UI description

MENU    device_root_menu
{
    LABEL        "Device setup";
    ITEMS
    {
        menu_WriteOnly_online
    }
}

MENU    menu_WriteOnly_online
{
    LABEL        "ONLINE session";
    STYLE        DIALOG;
    ITEMS
    {
        Var1
    }
}

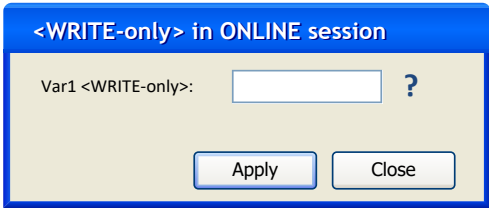
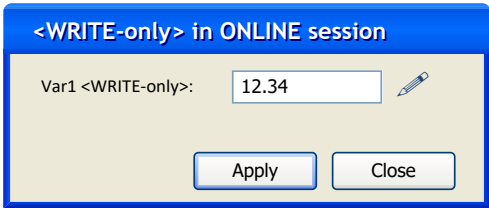
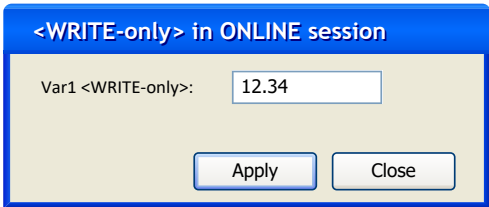
//--- Data description

VARIABLE Var1
{
    LABEL        "Var1 <WRITE-only>";
    CLASS        DEVICE;
    TYPE        FLOAT
    {
        DEFAULT_VALUE    11.22;
    }
    HANDLING    WRITE;
}
```

Figure 12 – EDD example with a "write-only" variable (HANDLING WRITE)

Table 21 shows the operating steps for a "write-only" variable in an online session.

Table 21 – Example of "write-only" variable in an online dialog

Step	User interface	Description
Step 1: The dialog is opened (initial state).		Since "Var1" is a "write-only" variable, the value cannot be read from device. "Var1" is displayed editable and blank with indication of "uncertain" state close to the value field .
Step 2: Edit "Var" = 12.34		The new value of "Var1" is displayed with indication of "changed" state close to the value field.
Step 3: Press "Apply" to commit the edited value		The new value of "Var1" is committed, i.e. transferred to the device. Then the committed value is displayed without indication of "uncertain" or "changed" state.

4.6.4.4 Collections and records

COLLECTIONs and RECORDs shall be displayed as MENU with STYLE GROUP. COLLECTIONs can contain COLLECTIONs. This is handled like nested groups (see 4.6.3.6). The rules about nesting groups apply also in this case.

4.6.4.5 Arrays and lists

Communication profiles: CS, FF, GPE, ISA100, PB, PN

Value arrays, reference arrays and LISTs shall be rendered as a scrollable area within the container. The item LABEL shall always be visible.

Communication profiles: FF

Each element of the array should be displayed as if each array element was specifically specified (in order) as menu items.

Communication profiles: HART

HART does not support arrays and lists references as contained items on a MENU.

Non-existent LIST elements referenced on a MENU should be treated as if their VALIDITY attribute value is FALSE.

4.6.4.6 Edit displays

An EDIT_DISPLAY shall be rendered in the container as either a button or a hyperlink. The LABEL of the corresponding EDIT_DISPLAY shall appear on the button or as the hyperlink text.

When the button or hyperlink is activated, the corresponding EDIT_DISPLAY shall be opened as a modal window. The content of an EDIT_DISPLAY contains DISPLAY_ITEMS and EDIT_ITEMS. The DISPLAY_ITEMS shall be displayed as read only items and EDIT_ITEMS shall be handled like ITEMS from a MENU of STYLE DIALOG.

4.6.4.7 Graphs

A graph should be displayed in a manner that is consistent with the definition of the corresponding GRAPH. Refer to the layout rules defined in 4.7 and IEC 61804-3 for information on the effect of the WIDTH attribute on the layout of the GRAPH. The EDD application should update the WAVEFORMs data when the user indicates that the waveform modification is complete (for example, the save button). The EDD application should provide a mechanism for the user to restore the original WAVEFORM without updating the data (for example, the cancel button).

4.6.4.8 Charts

A chart should be displayed in a manner that is consistent with the definition of the corresponding CHART. Refer to the layout rules defined in 4.7 and IEC 61804-3 for information on the effect of the WIDTH attribute on the layout of the CHART. For CHARTs that contain a time axis, there shall be an indication of what the time duration is (e.g. seconds, minutes, absolute time, etc.).

4.6.4.9 Images

Handheld applications may display an image referenced on a MENU as a hyperlink. Otherwise, EDD applications shall render images using the following rules:

- a) The images that are referenced by the MENU should be displayed.
- b) When INLINE is not specified, the EDD application shall display an IMAGE in original size. The EDD application shall also insert a ROWBREAK above the IMAGE, if items exist before the IMAGE and insert a ROWBREAK below the IMAGE, if items exist after the IMAGE.
- c) When INLINE is specified, the EDD application shall render the IMAGE depending on the LAYOUT_TYPE (see 4.7.2). For "equal-column-width layout" (see 4.7.2.2), the EDD application shall scale down an IMAGE if the IMAGE width is larger than the column width in which the IMAGE is contained. The EDD application shall keep original aspect ratio of IMAGES. For "optimized-column-width layout" (see 4.7.2.3), the EDD application shall not scale down an IMAGE, regardless whether INLINE is specified or not (see 4.7.5.10).
- d) The EDD application shall not scale up an IMAGE.
- e) If the IMAGE has the LINK attribute defined, this shall be indicated to the user.
- f) The EDD application shall display IMAGES having transparency against the background of its container. In this case, the color of the container applies as the background color of the image.

Communication profiles: HART

The size of an image shall be less than one hundred fifty thousand bytes (150 000) and the size of the sum of all images in a EDD shall not exceed five million bytes (5 000 000).

Images with a width of 210 pixels or less should be able to fit INLINE without any scaling applied. EDD applications with limited display constraints may not satisfy this recommendation.

4.6.4.10 Static text

The text that is referenced by the MENU should be displayed as a non-editable text. In a table view, long text may be truncated or wrapped in multiple lines on the menu. For all other views, the text will be wrapped in multiple lines, if the text is longer than the width of dialog, window, page, group, or column.

4.6.4.11 Grid

A grid should be displayed in a manner that is consistent with the definition of the corresponding GRID. The LABEL of the grid is shown at first and below the grid area. Refer to the layout rules defined in 4.7 and IEC 61804-3 for information on the effect of the WIDTH attribute on the layout of the GRID. Mechanisms should be provided to scroll the grid area, if the width or height is too small to show the complete data.

4.6.4.12 Plugin

A Plugin should be rendered as a button or a hyperlink. The LABEL of the corresponding PLUGIN should appear on the button or as the hyperlink text. When the button or hyperlink is pressed, the corresponding PLUGIN should be invoked.

4.7 Layout rules

4.7.1 Overview

Layout rules are rules that the EDD application shall follow for arranging the content of windows and dialogs on the display.

The basic elements of a layout are shown in Figure 13 and described in Table 22.

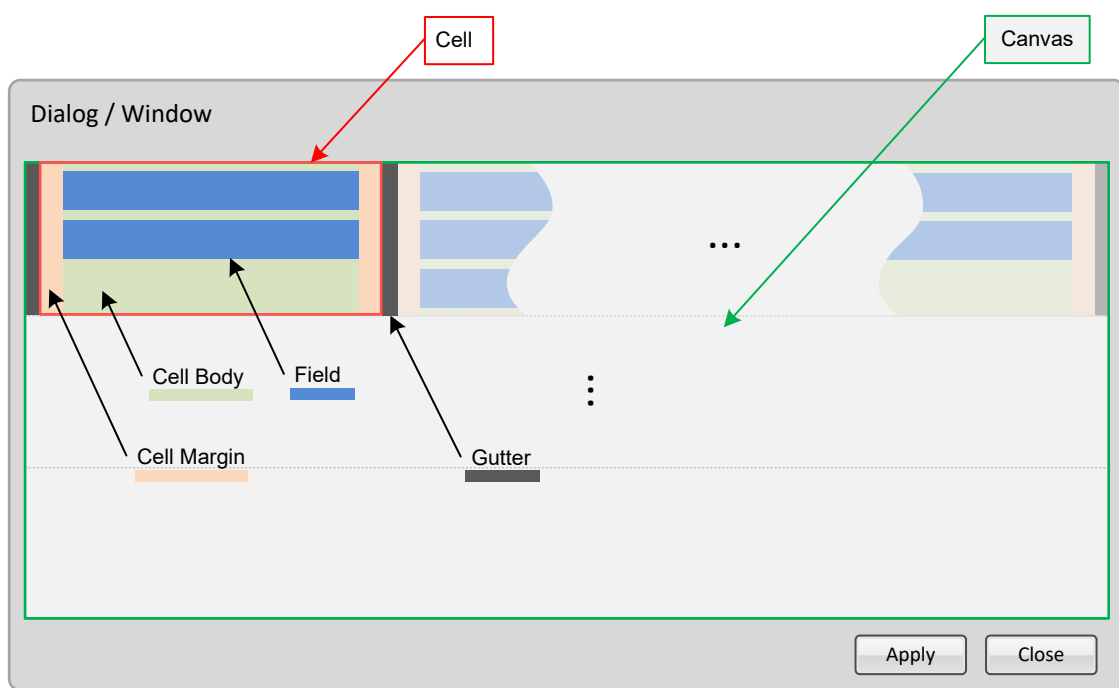


Figure 13 – Basic layout elements

Table 22 – Description of layout content

Layout element	Description and rules
Canvas	Area of WINDOWS, DIALOGs or PAGEs in which the EDD application shall arrange the content of the related MENU. A canvas is organized in rows and columns. The default display area for the canvas shall be wide enough to display at least three columns without horizontal scroll bar. The default display area of the canvas should be tall enough to display at least 15 simple fields in a single column without vertical scrolling. EDD applications with limited display constraints may not satisfy this recommendation. When two or more rows have the same number of columns, the column breaks in those rows shall be aligned. The organization of columns in the canvas shall be determined by the attributes of the construct LAYOUT_TYPE, COLUMNWIDTH_EQUAL or COLUMNWIDTH_OPTIMIZED
Cell	Area between row breaks and column breaks containing fields.
Field	Area that a menu item occupies.
Simple field	Area that would be occupied in a cell by a numeric VARIABLE, its label, its unit, and status indication (if displayed). All simple fields in a cell shall be displayed with All visible labels aligned Values aligned All visible units aligned All visible status aligned
Cell body	Area occupied by the collection of all fields in a cell. The organization of cell bodies in a row shall be determined by the attributes of the construct LAYOUT_TYPE, COLUMNWIDTH_EQUAL or COLUMNWIDTH_OPTIMIZED
Cell margin	The white space between the cell boundary and the cell body All cell margins in a row shall be equal
Gutter	The white space between any two cells and between a cell and the edge of the canvas in a given row All gutters in a row shall be equal

4.7.2 Controlling the layout by LAYOUT_TYPE attribute

4.7.2.1 Overview

The LAYOUT_TYPE attribute is part of the EDD identification information and specifies how the EDD application shall render all graphical user interfaces defined in the EDD.

4.7.2.2 Equal-column-width layout (COLUMNWIDTH_EQUAL)

When the LAYOUT_TYPE is defined as COLUMNWIDTH_EQUAL, each MENU with STYLE evaluated as WINDOW, DIALOG, PAGE, or GROUP shall be rendered using equal-width columns.

Figure 14 shows an example of a layout with equal column widths in each row.

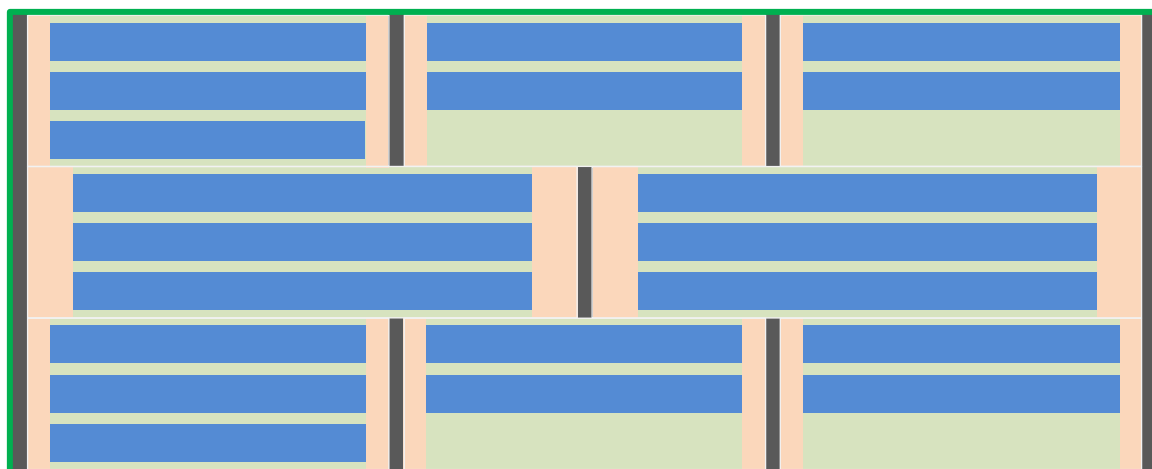


Figure 14 – Example of layout with equal column width

The width of each column in any row shall be the same as each other.

4.7.2.3 Optimized-column-width layout (COLUMNWIDTH_OPTIMIZED)

When the LAYOUT_TYPE is defined as COLUMNWIDTH_OPTIMIZED, each MENU with STYLE evaluated as WINDOW, DIALOG, PAGE, or GROUP shall be rendered using width-optimized columns.

Figure 15 shows an example of a layout with optimized column widths according to the content.

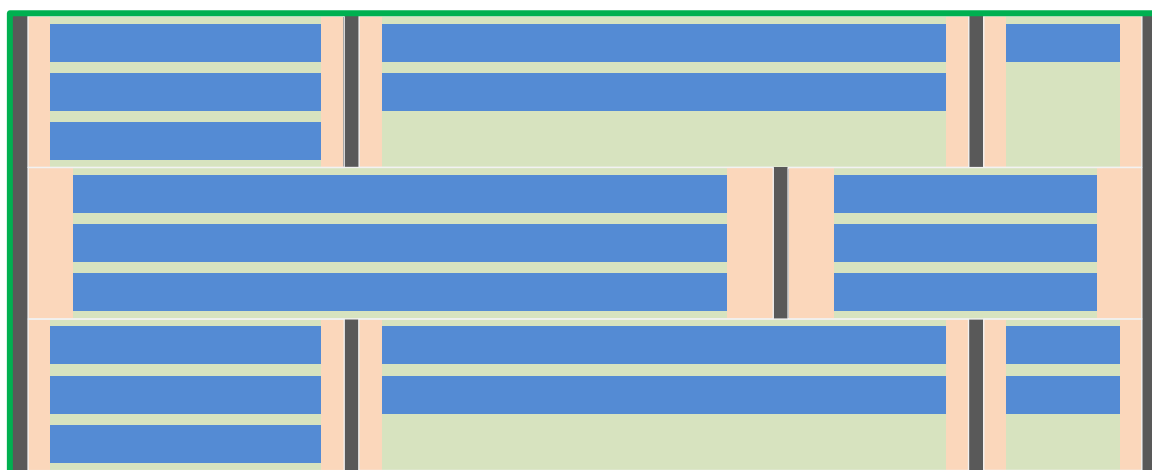


Figure 15 – Example of layout with optimized column width

The EDD application shall arrange the layout depending on the actual number of COLUMNBREAKs and the width of the containers. The width of each column shall be optimized according to the widest menu item in the column. All rows with the same number of columns shall be considered for the calculation.

In contrast to the equal-column-width layout, menu items will not be moved automatically to the next row when a menu item would span to a column higher than the 5th. To start a new layout row in an optimized-column-width layout, a ROWBREAK is required.

Figure 16 shows a layout cell containing a constant string (a), a read-only VARIABLE (b), a METHOD (c) rendered as button, and an editable VARIABLE (d). The labels of the VARIABLES are displayed to the left. The cell divided in separate areas for the label, the input field containing the value, the variable status and the unit.

Constant strings, menu items rendered as button, and the VARIABLE values shall be aligned and displayed in the input field area.

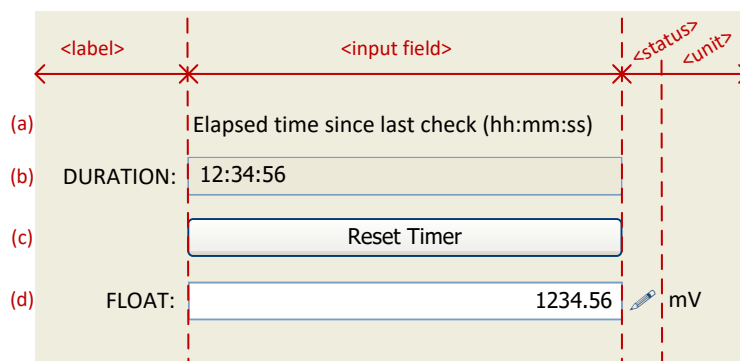


Figure 16 – Cell body in a layout with optimized column width (label to the left)

Figure 17 shows a layout cell containing the same menu items like in Figure 16, but the labels of the VARIABLES are displayed on top. Label area and input field area are merged.

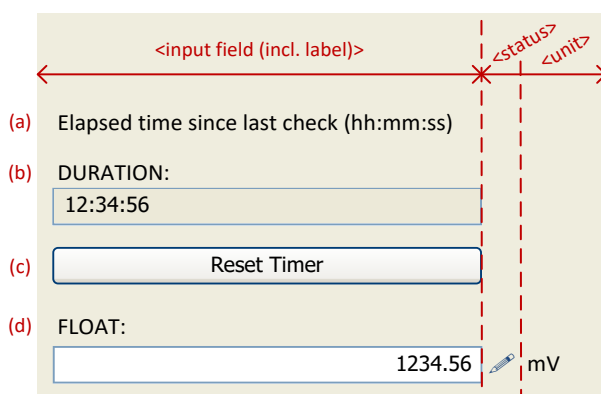


Figure 17 – Cell body in a layout with optimized column width (label on top)

Table 23 summarizes the minimum and maximum width limits for input fields according to the menu item type.

Table 23 – Minimum and maximum width for input fields spanning one column

Layout element	Minimum / maximum width for input fields (approx. pixel values)
Menu items rendered as button	Min. 80 pixels / no max. limit
Constant strings	No min. limit / max. 350 pixels
VARIABLES	Min. 100 pixels (may be wider depending on the UI control) / max. 350 pixels

The calculation of the optimized column width shall be made in the following steps:

- 1) The required width of each menu item shall be determined according to the following rules:
 - For menu items rendered as button or hyperlink, e.g. METHODS, the width shall be determined by the label.
 - For constant strings, the width shall be determined by the actual string length. For very long strings exceeding the max. width of an input field, automatic word wrapping shall be supported.
 - For VARIABLES, the width shall be calculated by totalizing the requested width for the label, for the value and unit and, if displayed, for the variable status. The width of the input field displaying the value depends on the variable's data type. For ENUMERATED or BIT_ENUMERATED variables, the width shall be determined by the widest enumeration element.
 Since in a layout showing the label on top, the label area and input field area are merged, the input field area should be determined considering both, the width of the label and the width of the input field. When the label is wider than the requested width of the input field, the input field should be expanded according to width of the label. Label and unit should not be truncated.
 - For IMAGES, the width shall be determined by the original width of the image and shall not be adjusted, regardless of whether or not the image referenced is marked with INLINE qualifier. If the IMAGE requires more width than available, the column shall be expanded.
 - For items rendered as GROUP, the width shall be determined according to the content considering the menu label and the margins in the group box. If the menu label is wider than the content, the width of the group box shall be determined according to the menu label.
- 2) Further, the minimum and maximum width limits for the input field as summarized in Table 23 shall be regarded. Note that in layouts showing the label on top, the input field may be wider than the maximum width limit.
- 3) The actual column width shall be evaluated as the width of the widest item in the column.
- 4) To provide a well-arranged layout, the items in each column shall be expanded to span the whole column by expanding the respective input fields (see Figure 16).
 For menu items not rendered with an input field, the full area of the menu item shall be expanded.
- 5) In a multi-row and -column layout, the number of columns may vary from row to row. The content of rows with a smaller number of columns shall be expanded, so that each row spans the same width. In this case, the input fields may be wider than the respective maximum width specified in Table 23.
- 6) After calculating the width of each column, the resulting width of the canvas shall be calculated by totalizing the width of each column. When the resulting width of the canvas is wider than the width of the screen, a horizontal scrollbar shall be provided.

When the user changes the size of the canvas by resizing the dialog or window, the following rules shall apply:

- When shrinking, the calculated optimized column width shall apply as the minimum width of each column.
- When expanding, the width gain shall be distributed evenly among all columns.
- The content of each column shall always span the whole column by adjusting the width of the respective input field.

When the required width changes due to a conditional expression, the resulting layout may grow, not shrink. If the resulting layout would require less width, the excess width shall be distributed evenly among all columns.

4.7.3 Layout rules for WIDTH and HEIGHT

4.7.3.1 General

VARIABLE, CHART, GRAPH and GRID have the attributes WIDTH and HEIGHT. The WIDTH attribute defines the number of columns of display. The HEIGHT attribute defines the vertical size of the display as a multiple of the minimum height for a simple field, see Table 24. The default value of WIDTH and HEIGHT is MEDIUM for CHART, GRAPH, GRID; XXX_SMALL for VARIABLE height, and width.

Equal-column-width layout:

The width of the EDD application should be split in up to a maximum of 5 columns depending on the number of COLUMNBREAKs and the width of the containers. Columns that would span to a column higher than the 5th column shall be moved to the next row through an implied ROWBREAK.

An implied ROWBREAK has the same behavior as an explicitly defined ROWBREAK.

Optimized-column-width layout:

The canvas should be divided depending on the number of COLUMNBREAKs and the width of the containers. Columns may span to a column higher than the 5th without being moved to the next row through an implied ROWBREAK.

The WIDTH and HEIGHT attribute define how many columns the item spans and the height in multiples of minimum height possible for a simple field e.g. variable, constant string, menu and method reference, see Table 24.

Table 24 – WIDTH and HEIGHT span and applicability

WIDTH and HEIGHT value	WIDTH	HEIGHT	Applicability
XXX_SMALL	1	1	VARIABLE
XX_SMALL	1	3	CHART GRAPH GRID VARIABLE
X_SMALL	1	5	CHART GRAPH GRID VARIABLE
SMALL	2	7	CHART GRAPH GRID VARIABLE
MEDIUM	3	8	CHART GRAPH GRID VARIABLE

WIDTH and HEIGHT value	WIDTH	HEIGHT	Applicability
LARGE	4	9	CHART GRAPH GRID VARIABLE
X_LARGE	5	11	CHART GRAPH GRID VARIABLE
XX_LARGE	5	13	CHART GRAPH GRID VARIABLE

4.7.3.2 VARIABLES spanning columns by WIDTH (optimized-column-width layout)

The WIDTH attribute of an item specifies the number of columns the item spans. Thereby the following rules shall apply:

- 1) When spanning normal, width consuming columns, the menu item shall be expanded to span all corresponding columns (see Figure 44).
- 2) When spanning empty or very narrow columns, the width increment of this menu item shall be divided evenly on all corresponding columns. The EDD application shall apply width increment steps considering the minimum width of input fields (see Table 23).
- 3) The empty or narrow columns shall be expanded accordingly (see Figure 19).
- 4) For VARIABLES spanning more than one column, only the input field shall be expanded.

Figure 19 shows a layout containing 2 rows, the 1st with 5 narrow columns and the 2nd with variables spanning a different number of columns by defining the WIDTH attribute. Figure 18 shows the corresponding source code.

```

MENU menu_Layout_Width_on_Variables
{
    LABEL    "WIDTH on Variables";
    STYLE    DIALOG;
    ITEMS
    {
        "1",    COLUMNBREAK,          // 5 "narrow" columns
        "2",    COLUMNBREAK,
        "3",    COLUMNBREAK,
        "4",    COLUMNBREAK,
        "5",
        ROWBREAK,
        var_ASCII,          // ASCII variable with no WIDTH → default
        var_ASCII_Width_X_SMALL, // ASCII variable with 'WIDTH X_SMALL'
        var_ASCII_Width_MEDIUM, // ASCII variable with 'WIDTH MEDIUM'
        var_ASCII_Width_LARGE,  // ASCII variable with 'WIDTH LARGE'
        var_ASCII_Width_X_LARGE // ASCII variable with 'WIDTH X_LARGE'
    }
}

//--- Exemplary VARIABLE definition
VARIABLE var_ASCII_Width_X_SMALL
{
    LABEL    "ASCII (X_SMALL = 1C)";
    CLASS    LOCAL;
    TYPE     ASCII(20);
}

```

Figure 18 – EDD source code for a layout with VARIABLES spanning columns

The width of the 1st column is determined by totalizing the width of

- the widest label, which is "ASCII (MEDIUM = 3C)",
- the value of the 1st variable "ASCII (Default = 1C)",
- no unit to be considered.

The width of column 2 to 5 is determined by the width increment steps due to spanning empty or narrow columns.

WIDTH on Variables

	1	2	3	4	5
ASCII (Default = 1C):	Test – Test – Test				
ASCII (X_SMALL = 1C):					
ASCII (SMALL = 2C):					
ASCII (MEDIUM = 3C):					
ASCII (LARGE = 4C):					
ASCII (X_LARGE = 5C):					

Increment step

OK Cancel

Figure 19 – Layout with VARIABLES spanning multiple columns

4.7.3.3 CHART and GRAPH spanning columns by WIDTH (optimized-column-width layout)

Depending on the CHART or GRAPH declaration in the EDD, the EDD application should define a minimum width regarding the width for structuring elements (e.g. y-axes, legends) and the characteristics of the UI control used to represent the CHART or GRAPH.

To minimize the need of a horizontal scrollbar, the EDD application should define a maximum width for a CHART or GRAPH spanning 5 columns considering the actual screen characteristics.

The resulting increment step as a minimum width gain per column is: $(\text{Max. width} - \text{min. width}) / 4$.

The WIDTH attribute of an item specifies the number of columns the CHART or GRAPH spans. Thereby the following rules shall apply:

- 1) When spanning empty or very narrow columns, the width gain of the CHART or GRAPH shall be divided evenly on all corresponding columns. The EDD application shall apply a number of the width increment steps according to the WIDTH attribute.
- 2) When spanning columns consuming more width than one increment step, the CHART or GRAPH shall be expanded to span all corresponding columns.
- 3) The maximum width may be exceeded to make all layout rows span the same width.

4.7.3.4 GRID spanning columns by WIDTH (optimized-column-width layout)

To minimize the need of a horizontal scrollbar, the EDD application should define a maximum width for a GRID spanning 5 columns considering the actual screen characteristics and the corresponding increment step as a minimum width gain per column.

The WIDTH attribute of an item specifies the number of columns the GRID spans. Thereby the following rules shall apply:

- 1) When spanning empty or very narrow columns, the width gain of the GRID shall be divided evenly on all corresponding columns. The EDD application shall apply a number of the width increment steps according to the WIDTH attribute.
- 2) When spanning columns consuming more width than one increment step, the GRID shall be expanded to span all corresponding columns.
- 3) The EDD application shall consider the actually required width for the GRID depending on the content (VECTORS) as the maximum width of the GRID. This width may be exceeded to make all layout rows span the same width.

4.7.4 Layout rules for COLUMNBREAK and ROWBREAK

4.7.4.1 Layout for protruding elements

The EDD application shall insert all the menu items in a single column one after the other until it encounters a COLUMNBREAK. The COLUMNBREAK will cause the next menu item to be inserted at the top of the next column in the same row. If there is a menu item occupying a region of space in the next column, then the menu item will be inserted in the row immediately below the occupying menu item. See Figure 20 for EDD source code example and Figure 21 for the corresponding layout.