



ISO/IEC JTC1/SC7
Software Engineering
Secretariat: CANADA (SCC)

ISO/IEC JTC1/SC7 N1543
July 14, 1996

DOC TYPE: CD Ballot

TITLE: CD 15475-2: Information Technology - Software Engineering Data Definition and Interchange - Transfer Format - Part 2: Syntax SYNTAX.1

SOURCE: WG11

PROJECT: 07.28.2.2

STATUS: First CD ballot

REFERENCES: Resolution 404, SC7 N1312, N1539

ACTION ID: ACT

DUE DATE: 1996-10-28. It would be most appreciated that a copy of the comments, if any, be send to the project 07.28 editor, Mr Jean Bérubé (jberube@ibm.net) before 1996-10-18.

DISTRIBUTION: P, O and L

MEDIUM: E-Mail - uuencoded compressed Word ; Diskette - compressed Word

NO. OF PAGES: 69 [201 kB]

DISK NUMBER: 8



| | |
|------------------------------------|--|
| ISO/IEC JTC1/SC7 CD 15475-2 | |
| Date 1996-07-14 | Reference number ISO/JTC 1/SC 7 N 1543 |
| Supersedes document SC 7 N 1312 | |

THIS DOCUMENT IS STILL UNDER STUDY AND SUBJECT TO CHANGE. IT SHOULD NOT BE USED FOR REFERENCE PURPOSES.

| | |
|---|--|
| ISO/JTC 1/SC 7 Committee Title Software Engineering Secretariat: Standards Council of Canada (SCC) | Circulated to P- and O-members, and to technical committees and organizations in liaison for: X discussion at Ottawa on 1996-10-21 X comment by 1996-10-21 X voting by (P-members only) 1996-10-28 Please return all votes and comments in electronic form directly to the SC 7 Secretariat by the due date indicated. |
|---|--|

ISO/IEC JTC1/SC7

Title: Information Technology - Software Engineering Data Definition and Interchange - Transfer Format
Part 2: Syntax SYNTAX.1

Project: 07.28.2.2

Introductory note: See Attachements

Medium: E-Mail - uuencoded compressed Word ; Diskette - compressed Word

No. of pages: 69



| Vote on Committee Draft ISO/IEC 15475-2 | |
|--|--|
| Date of circulation 1996-07-14 | Reference number ISO/JTC 1/SC 7 N 1543 |
| Closing date 1996-10-28 | |

| | |
|---|--|
| ISO/JTC 1/SC 7 Committee Title Software Engineering Secretariat: Standards Council of Canada (SCC) | Circulated to P-members of the committee for voting Please return all votes and comments in electronic form directly to the SC 7 Secretariat by the due date indicated. |
|---|--|

ISO/IEC JTC1/SC7

Title: Information Technology - Software Engineering Data Definition and Interchange - Transfer Format - Part 2: Syntax SYNTAX.1

Project: 07.28.2.2

Vote:

- APPROVAL OF THE DRAFT AS PRESENTED
- APPROVAL OF THE DRAFT WITH COMMENTS AS GIVEN ON THE ATTACHED
 - general:
 - technical:
 - editorial:
- DISAPPROVAL OF THE DRAFT FOR REASONS ON THE ATTACHED
 - Acceptance of these reasons and appropriate changes in the text will change our vote to approval
- ABSTENTION (FOR REASONS BELOW):

P-member voting:
National Body (Acronym)

Date:
YYCC-MM-DD

Submitted by:

Name

ISO IEC CD 15475-2
EIA/IS-109
SC7 WG11 N041R1

**Information Technology - Software Engineering Data
Definition and Interchange - Transfer Format
Part 2: Syntax SYNTAX.1**

| | | | |
|-------------------|---|---|-----------------|
| Author: | EIA CDIF | Source: | SC7 WG11 |
| Project: | 7.28.2.2 | Date: | 1996.05.31 |
| Action: | Circulation to JTC1 SC7 For CD Ballot, CD Review and Comments | Status: | Committee Draft |
| Reference: | SC7 N1220 | Resolution #343 Ottawa Plenary (94.06.17) | |
| | SC7 N1312 | CDIF SYNTAX.1 Working Draft | |
| | SC7 N1415 | Resolution #404 Brisbane Plenary (96.06.01) | |

Introduction

ISO/IEC JTC1 SC7 WG11, in its Project 7.28, has the objective of enabling the interchange of information between a variety of heterogeneous software engineering tools.

Many of the input documents for Project 7.28 are the result of developmental work performed by the CASE Data Interchange Format (CDIF) technical committee of the Electronics Industries Association (EIA). This is one of those documents, and is distributed as received from the CDIF editor.

At its Brisbane meeting, in June 95, SC7 approved the distribution of the committee drafts.

This document is intended to provide one part within the multipart standard resulting from Project 7.28. Documents will be reformatted to ISO Directives as part of the Committee Draft ballot and comment resolution.

Comments

Comments are requested within the timetable provided by the SC7 cover sheet (generally 100 days) to facilitate preparation for the next WG11 meeting. Comments should be sent by normal formal channels.

To facilitate planning and preparatory resolution proposals, WG11 would appreciate comments to be submitted to the WWW (www.cdif.org/comments.html) site identified for that purpose).

Comments are to be provided in electronic format, generally following the format used for the WWW page. If comments are also submitted on paper, they should use a similar format.

Please identify clearly the status of comments submitted, such as:

National Body Comments,
National Body Approved Expert Comments.

Jean Bérubé
Project Editor, 7.28

IDEgenic inc
87 Burton,
Greenfield Park, Québec,
Canada J4V 2Y6

Tél: 1 514 465 0245
Fax: 1 514 465 2398
jberube@ibm.net

711PRA-11
Explanatory Notes for SC7 Reviewers
of the
WG11-CDIF Documents
(Project 07.28)

- The “Overview” document (WG11 N047R1, SC7 N1540) should be read first since it describes all the other project 7.28 CD documents to be reviewed at this time (as well as several documents planned for the future). It also describes the overall architecture that applies to these documents.
- The “Framework” document (WG11 N046R1, SC7 N1541) should be read second, as it describes the entity-relationship-attribute modeling notation that is used in many of the documents.
- The “Data Modeling” document (WG11 N036, SC7 N1548) and the “Data Flow Modeling” (WG11 N035, SC7 N1549) document provide the substantive coverage of software engineering techniques within the first set of documents. These documents permit the structured representation of software designs prepared by the use of any of a number of well established software engineering techniques that produce what are commonly called data flow diagrams and/or entity-relationship- attribute diagrams. These two documents will be of the greatest interest to software engineering practitioners.
- The “Foundation”, “Common”, and “Data Definition” documents (WG11 N048R1, N049, N050 and respectively SC7 N1545, N1546, 1547) are “modeling infrastructure” documents that support the contents of the software engineering techniques document, Data Modeling and Data Flow Modeling. Elements in these three infrastructure documents are incorporated by reference into the technique documents as necessary.
- The Transfer Format documents, “General Rules”, “Syntax”, and “Encoding” (WG11 N040R1, N041R1, N042R1 and respectively SC7 N1542, N1543, N1544) describe how character sequences are formed in order to communicate software design representations. These may be read separately from the technique and infrastructure documents described above.
- Of special importance for this review is the “Foundation” document (WG11 N048R1, SC7 N1545). This document illustrates the format intended to be applied to all the documents in this set for subsequent balloting. WG11 specifically solicits reviewer comments related to the format of this document so that the reformatting effort for the other documents in this set need be done only once.

WG11 will greatly appreciate the national body review comments on these documents, and thanks in advance the reviewers for their efforts. Any early submissions of comments will be very helpful.

WG11 Foreword

In order to fully ensure that all comments received are processed efficiently and consistently, WG11 requests of all reviewers to please utilize the comments template that is provided with each document. Submitting comments in electronic form rather than on paper will be especially helpful.

If assistance with the usage of the comment forms is required, or if there are any other questions, please contact either Jean Berube, the Project 7.28 coordinating editor, or myself.

Sincerely,

Peter Eirich
WG11 Convenor

CD 15475-2

**Information Technology - Software Engineering Data
Definition and Interchange - Transfer Format**

Part 2: Syntax SYNTAX.

CDIF TITLE:

CDIF - Transfer Format - Syntax SYNTAX.1

EIA/IS-109 JANUARY, 1994

Revised 4/18/96 to incorporate ISO/IEC JTC1 SC7/WG11 comments submitted as input to the 1995 Brisbane meeting.

ELECTRONIC INDUSTRIES ASSOCIATION

ENGINEERING DEPARTMENT

NOTICE

EIA Engineering standards and publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating interchangeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for his particular need. Existence of such standards and publications shall not in any respect preclude any member or nonmember of EIA from manufacturing or selling products not conforming to such standards and publications, nor shall the existence of such standards and publications preclude their voluntary use by those other than EIA members, whether the standard is to be used either domestically or internationally.

Recommended standards and publications are adopted by EIA without regard to whether or not their adoption may involve patents on articles, materials or processes. By such action, EIA does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the standard or publication.

EIA INTERIM STANDARDS

EIA Interim Standards contain information deemed to be of technical value to the industry, and are published at the request of the originating Committee without necessarily following the rigorous public review and resolution of comments which is a procedural part of the development of an EIA Recommended Standard.

EIA Interim Standards should be reviewed on an annual basis by the formulating Committee and a decision made on whether to proceed to develop an EIA Recommended Standard on this subject. EIA Interim Standards must be canceled by the Committee and removed from the EIA Standards Catalog before the end of their third year of existence.

This EIA standard is considered to have international standardization implication, but the International Standardization Organization activity has not progressed to the point where a valid comparison between the EIA standard and the ISO work can be made.

COPYRIGHT 1994

Published by
ELECTRONIC INDUSTRIES ASSOCIATION
Engineering Department
2500 Wilson Blvd. Suite 203
Arlington, VA 22201

PRICE: Please refer to the current
Catalog of EIA & JEDEC STANDARDS & ENGINEERING PUBLICATIONS
or contact the EIA Electronic Information Group
(Phone: (703) 527-7001, Fax: (703) 527-0685)

All rights reserved
Printed in U.S.A.

ISBN 0-7908-0015-2

PREFACE

This standard will assist the vendors and users of CASE tools in developing mechanisms for interchanging information between CASE tools. This standard specifies an element of a family of related standards. When used together, these standards specify a mechanism for transferring information between CASE tools.

CDIF - CASE Data Interchange Format - Overview (ISO CD 15474-1: Information Technology - Software Engineering Data Definition and Interchange - Overview and Framework: Part 1 : Overview) and CDIF - Framework for Modeling and Extensibility (ISO CD 15474-2: Information Technology - Software Engineering Data Definition and Interchange - Overview and Framework: Part 2 : Framework for Modeling and Extensibility) should be read first when initially exploring CDIF. The first explains the overall CDIF Architecture and how the family of standards fits together. The second explains the scope, and modeling approach in CDIF. The CDIF Meta-meta-model and extensibility mechanism are also defined in this document.

EIA/IS-110 CDIF - Transfer Format - Encoding ENCODING.1 defines an encoding of SYNTAX.1. *EIA/IS-108 CDIF - Transfer Format - General Rules for Syntaxes and Encodings* describes how CDIF supports multiple syntaxes and encodings. This standard defines the CDIF Transfer Format syntax, SYNTAX.1.

This standard has been developed with the wide support and participation of vendors, users, academia and government involved in or familiar with the CASE industry, its products and the general requirements associated with interchanging information between these products.

The material contained in this publication has been copyrighted by the CASE Data Interchange Format (CDIF) Division of the Electronic Industries Association (EIA). Permission to copy this material in paper and electronic form has been granted to the international software engineering standards development organization (ISO/IEC JTC1/SC7) for the sole purpose of international standards development work. Organizations receiving copies of this document may not duplicate and distribute additional copies internally for any reason other than to facilitate their organization's review and comment back to ISO/IEC JTC1/SC7.

This page is intentionally blank

CONTENTS

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40

| | |
|---|-----------|
| 1. SCOPE | 1 |
| 2. INTRODUCTION | 3 |
| 2.1 Audience | 3 |
| 2.2 Structure of this Document | 3 |
| 2.3 Conventions | 4 |
| 2.3.1 Global Conventions..... | 4 |
| 2.3.2 BNF Conventions..... | 5 |
| 3. CONCEPTS AND DEFINITIONS | 7 |
| 3.1 Syntax Identifier | 7 |
| 3.2 Token Separation Rules | 7 |
| 4. SYNTAX SECTIONS AND STRUCTURES IN THE CDIF TRANSFER | 8 |
| 4.1 Introduction | 8 |
| 4.2 CDIF Transfer Components | 8 |
| 4.2.1 Introduction..... | 8 |
| 4.2.2 The Transfer Envelope | 8 |
| 4.2.3 The Transfer Contents | 8 |
| 4.3 Header Section | 9 |
| 4.3.1 Introduction..... | 9 |
| 4.3.2 Summary Clause | 10 |
| 4.4 Meta-model Section | 11 |
| 4.4.1 Introduction..... | 11 |
| 4.4.2 CDIF Subject Area Reference Clause..... | 12 |
| 4.4.3 Meta-model Extension Clause | 13 |
| 4.4.4 Meta-meta-entity Instance | 13 |
| 4.4.5 Meta-meta-attribute Instance | 14 |
| 4.4.6 Meta-meta-relationship Instance..... | 15 |
| 4.4.7 Enumerated Meta-attribute Extension..... | 16 |
| 4.5 Model Section | 16 |
| 4.5.1 Introduction..... | 16 |
| 4.5.2 Meta-entity Instance..... | 17 |
| 4.5.3 Meta-relationship Instance..... | 18 |
| 4.5.4 Meta-attribute Instance | 19 |
| 4.5.5 Meta-attribute Value | 19 |
| 4.5.5.1 Introduction | 19 |
| 4.5.5.2 Bitmap Value | 19 |
| 4.5.5.3 Boolean Value..... | 21 |
| 4.5.5.4 Date Value | 21 |
| 4.5.5.5 Enumerated Value..... | 21 |
| 4.5.5.6 Float Value | 21 |

| | | | | |
|----|------------|--|--------------------------|-----------|
| 1 | | 4.5.5.7 | Identifier Value | 22 |
| 2 | | 4.5.5.8 | Integer Value | 22 |
| 3 | | 4.5.5.9 | Integer List Value | 22 |
| 4 | | 4.5.5.10 | Point Value | 23 |
| 5 | | 4.5.5.11 | Point List Value | 24 |
| 6 | | 4.5.5.12 | String Value | 24 |
| 7 | | 4.5.5.13 | Text Value | 24 |
| 8 | | 4.5.5.14 | Time Value | 25 |
| 9 | 4.6 | Comments | | 25 |
| 10 | 4.7 | Syntax Terminal Symbols | | 26 |
| 11 | 5. | SYNTAX FORMAL GRAMMAR | | 28 |
| 12 | | APPENDIX A LL(1) EQUIVALENT OF SYNTAX.1 | | 35 |
| 13 | | APPENDIX B QUESTIONS AND ANSWERS | | 43 |
| 14 | | GLOSSARY | | 45 |
| 15 | | INDEX..... | | 53 |
| 16 | | RELATED EIA STANDARDS..... | INSIDE BACK COVER | |

FIGURES

| | | |
|---|---|---|
| 1 | | |
| 2 | Figure 1 Position in the CDIF Family of Standards | 1 |
| 3 | Figure 2 CDIF Transfer..... | 9 |

This page is intentionally blank

1. SCOPE

The CDIF Family of Standards is primarily designed to be used as a description of a mechanism for transferring information between CASE tools. It facilitates a successful transfer when the authors of the importing and exporting tools have nothing in common except an agreement to conform to CDIF. The language that is defined for the Transfer Format also has applicability as a general language for Import/Export from repositories. The CDIF Integrated Meta-model defined for CASE also has applicability as the basis of standard definitions for use in repositories.

The standards which form the complete family of CDIF Standards are documented in *EIA/IS-106 CDIF - CASE Data Interchange Format - Overview*. These standards cover the overall framework, the transfer format and the CDIF Integrated Meta-model.

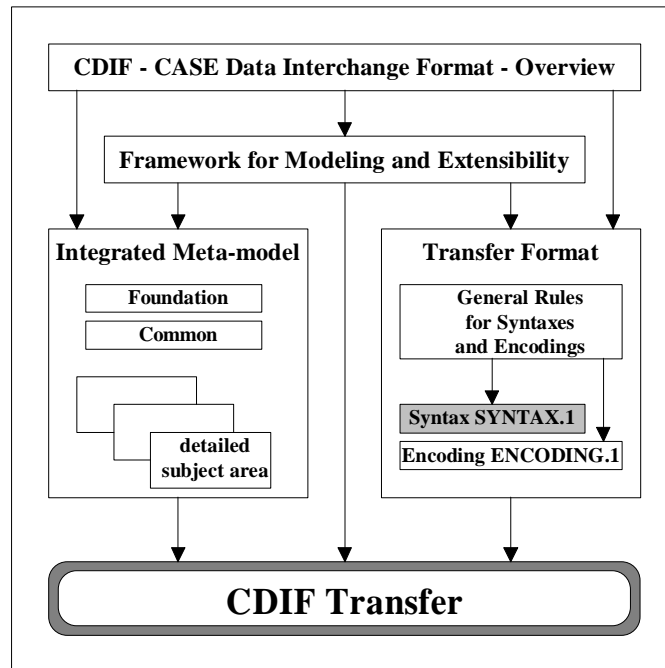


Figure 1
Position in the CDIF Family of Standards

The diagram in Figure 1 depicts the various standards that comprise the CDIF Family of Standards. The shaded box depicts this Standard and the role that it plays in the CDIF Family of Standards.

- 1 This document describes the standard CDIF Transfer Syntax. No encodings for SYNTAX.1 are
- 2 specified in this document. *EIA/IS-110 CDIF - Transfer Format - Encoding ENCODING.1*
- 3 specifies one standard encoding for this syntax.

2. INTRODUCTION

2.1 Audience

This document is intended to be used by anyone wishing to understand and/or use CDIF. This document provides a definition of a syntax for CDIF transfers. It is suitable for:

- those evaluating CDIF
- those who wish to understand the principles and concepts of a CDIF transfer
- those developing importers and exporters.

The document *EIA/IS-106 CDIF - CASE Data Interchange Format - Overview* and the document *EIA/IS-107 CDIF - Framework for Modeling and Extensibility* should be read first when initially exploring CDIF and before attempting to read other documents in the CDIF Family of Standards.

This document should be read in conjunction with *EIA/IS-108 CDIF - Transfer Format - General Rules for Syntaxes and Encodings*.

While there are no specific prerequisites for reading this document, it will be helpful for the reader to have familiarity with the following:

- Entity-Relationship-Attribute modeling
- CASE tools
- Information repositories
- Data dictionaries
- Multiple meta-layer modeling
- Formal grammars
- Transfer formats.

2.2 Structure of this Document

This document is organized into the following sections:

Section 1 'Scope'

This describes the CDIF Family of Standards and positions the current document within the set of documents.

1 **Section 2 'Introduction'**

2 This describes the intended audience, structure and conventions used in the document.

3 **Section 3 'Concepts and Definitions'**

4 This specifies the unique identifier for the syntax defined in this document and describes
5 the concept of token separation used in this document.

6 **Section 4 'Syntax Sections and Structures in the CDIF Transfer'**

7 This section describes, in detail, the exact syntax of each of the three major sections of a
8 transfer.

9 **Section 5 'Syntax Formal Grammar'**

10 This defines the grammar rules for the syntax in Backus Naur Form (BNF).

11 **Appendix A 'LL(1) Equivalent of SYNTAX.1 BNF'**

12 This contains a computer analyzed LL(1) grammar that is equivalent to the BNF in this
13 document.

14 **Appendix B 'Questions and Answers'**

15 This includes detailed responses to questions that have been raised by those reviewing this
16 document. It is intended that the inclusion of these will aid the reader in understanding
17 this document, and related issues concerning CDIF.

18 **Glossary**

19 The Glossary contains definitions of terms used within this Standard.

20 **Index**

21 A comprehensive index is provided.

22 **2.3 Conventions**

23 **2.3.1 Global Conventions**

24 The typographical and naming conventions defined below are used throughout the CDIF Family
25 of Standards.

- 26 ● Double quotes are used to introduce new terms (e.g., "model layer")
- 27 ● Bold type is used for emphasis (e.g., this is **important**)
- 28 ● All meta-objects and meta-meta-objects in CDIF (in meta-models and meta-meta-
29 meta-models) are named by concatenating all the words that name the meta-object or
30 meta-meta-object; the first letter of each word is upper-case, the rest are lower-
31 case (e.g., *MetaAttribute*, *AttributeDerivation*, *IsDrawnUsing*, *IsOptional*)

2.3.2 BNF Conventions

The grammar in this document is defined using an extended version of "Backus Naur Form" (BNF).

In BNF, each syntactic element of the language is defined by means of a "production rule". A "production rule" defines a syntactic element in terms of an expression, ordered left to right, that can be used to form an instance of the element. The expression can consist of characters, character strings, and other syntactic elements.

The version of BNF used in this document has the following conventions:

| <i>Symbol</i> | <i>Meaning</i> |
|---------------|--|
| < > | Angle braces delimit character strings that are the names of syntactic elements, i.e., non-terminal symbols. |
| ::= | This is the definition operator. It is used in a production rule to separate the element defined by the rule from its definition. The element being defined is to the left of the operator and the expression that defines the element is to the right of the operator. |
| [] | Square braces enclose optional elements in an expression. The portion of the expression within the braces may be explicitly specified or may be omitted. |
| { } | Curly braces group elements in an expression, so that the group can be treated as a single syntactic element. |
| | This is the alternative operator. The vertical bar indicates that the portion of the expression to the right of the bar is an alternative to the portion to the left of the bar. If the vertical bar appears at a position where it is not enclosed in curly or square braces, it specifies a complete alternative for the element defined by the production rule. If the vertical bar appears in a portion of the expression enclosed in curly or square braces, it specifies alternatives for the contents of the innermost pair of such braces. |
| ... | The ellipsis indicates that the element to which it applies in a formula may be repeated any number of times (but note that optionality is covered by the square braces convention). If the ellipsis appears immediately to the right of a closing curly or square brace, "}" or "]", then it applies to the portion of the expression enclosed between that closing brace and the corresponding opening brace, "{" or "[". If an ellipsis appears immediately to the right of any other element, then it applies only to that element. |

| | | |
|----|--------------------------|---|
| 1 | !! | This is the comment operator. This operator introduces normal English |
| 2 | | text. It is used as a comment in the BNF and when the definition of a |
| 3 | | syntactic element is not expressed further in BNF. |
| 4 | <TokenName> | Anything emboldened and surrounded by angle braces is, a token , a |
| 5 | | BNF non-terminal symbol that is not further defined in the syntax. The |
| 6 | | detailed representation of a token is specified in the encoding. |
| 7 | Terminal | Anything emboldened, and not surrounded by angle braces, is a literal |
| 8 | | value (i.e., it represents the character string itself). |
| 9 | x .. y | This is the range operator. It refers to any symbol in the range x to y. |
| 10 | | For example, 0..255 refers to all integers 0 to 255 inclusive. |

11 The examples in this document use the representation of the terminal symbols defined in the clear
12 text encoding ENCODING.1.

3. CONCEPTS AND DEFINITIONS

3.1 Syntax Identifier

The syntax defined in this version of this standard shall have a CDIF Standardized Syntax Identifier of "SYNTAX.1", and a Version of "02.00.00". This is used to define the standardized syntax used in a CDIF Transfer (see *EIA/IS-108 CDIF - Transfer Format - General Rules for Syntaxes and Encodings*).

The syntax of the CDIF Syntax ID and the Syntax Version is as follows:

```
<SyntaxID> ::=  
    "SYNTAX.1"
```

```
<SyntaxVersion> ::=  
    "02.00.00"
```

3.2 Token Separation Rules

The "tokens" (or terminal symbols) are the primitive level elements in the syntax. Tokens are normally separated by one or more "whitespace" characters, as defined in the specific encoding. After token identification, these whitespace characters are ignored by importers.

Certain terminal symbols act as token separators; these terminal symbols do not need to be preceded or followed by whitespace characters. All other terminal symbols must be followed by one of these token separators or by one or more whitespace characters. The terminal symbols that act as token separators are listed in Section 4.7 Syntax Terminal Symbols.

4. SYNTAX SECTIONS AND STRUCTURES IN THE CDIF TRANSFER

4.1 Introduction

This section describes, in detail, the exact syntax of each of the major sections of a transfer, together with the syntax of each of the component structures.

Examples of transfers given in this section use the CDIF Standard Encoding as defined in *EIA/IS-110 CDIF - Transfer Format - Encoding ENCODING.1*. Two complete examples of a transfer are contained in that document.

4.2 CDIF Transfer Components

4.2.1 Introduction

The general syntax of a CDIF Transfer is as follows:

```
<CDIFTransfer> ::=
    <TransferEnvelope>
    <TransferContents>
```

Figure 2 illustrates the complete structure of a CDIF transfer. A transfer is typically contained in a file, but other means of transfer such as pipes, mailboxes, shared memory, or any other mechanism that can be interpreted as a byte stream may be used.

4.2.2 The Transfer Envelope

The Transfer Envelope consists of the CDIF Signature, the Syntax Identifier and the Encoding Identifier. As the Transfer Envelope syntax is the same for any CDIF transfer, it is defined in *EIA/IS-108 CDIF - Transfer Format - General Rules for Syntaxes and Encodings*. The syntax identifier for SYNTAX.1 is specified in 3.1 Syntax Identifier of this document.

4.2.3 The Transfer Contents

As the first level of the grammar of the Transfer Contents is the same for any CDIF transfer, it is defined in *EIA/IS-108 CDIF - Transfer Format - General Rules for Syntaxes and Encodings*, but is reproduced here for clarity:

```
<TransferContents> ::=
    <HeaderSectionClause> <MetaModelSectionClause>
    [ <ModelSectionClause> ]
```

The further levels are detailed in the next section.

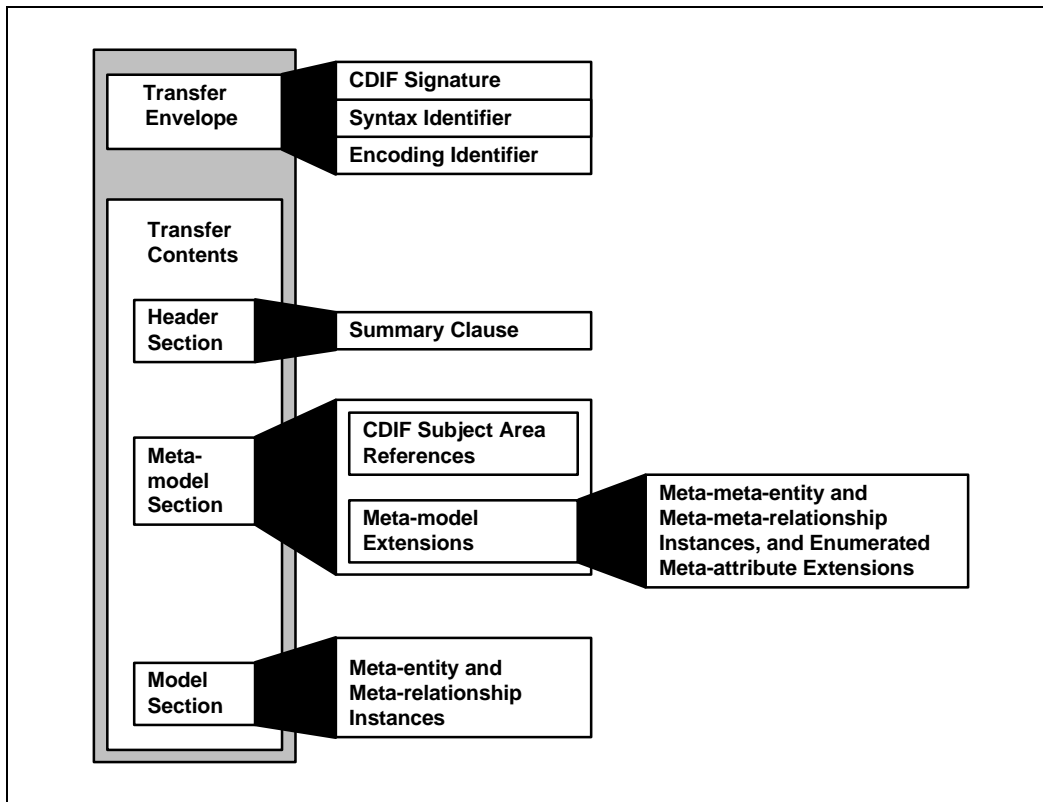


Figure 2
CDIF Transfer

4.3 Header Section

4.3.1 Introduction

The Header Section defines information that applies to the whole transfer.

The syntax of the Header Section clause is:

```
<HeaderSectionClause> ::=  
    <OpenScope> <HeaderKeyword>  
    <SummaryClause>  
    <CloseScope>
```

that is, an introductory keyword followed by a Summary clause, all enclosed within scope delimiters.

1 An example of the Header Section clause is:

```
2     ( :HEADER  
3         <SummaryClause>  
4     )
```

6 **4.3.2 Summary Clause**

7 The Summary clause specifies summary information about the transfer, in the form of a number of
8 items. Each item is introduced by an identifier. This standard defines meanings for certain
9 summary section identifiers. Other identifiers may be used, to introduce other items of summary
10 information, but their meanings are not defined in this standard.

11 The syntax of the Summary clause is:

```
12     <SummaryClause> ::=  
13         <OpenScope> <SummaryKeyword>  
14         [ <IdentifierValuePair> ] ...  
15         <CloseScope>  
  
16     <IdentifierValuePair> ::=  
17         <OpenScope>  
18         <SummaryIdentifier> <StringValue>  
19         <CloseScope>  
  
20     <SummaryIdentifier> ::=  
21         <Identifier>  
  
22     <StringValue> ::= <String>
```

23 The identifiers must conform to the rules for data type Identifier.

24 The defined summary section identifiers and their definitions are:

- 25 • ExporterName - The name of the exporting software product
- 26 • ExporterVersion - The version of the exporting software product
- 27 • ExportDate - The date of commencement of the export, in the form
28 YYYY/MM/DD
- 29 • ExportTime - The time of commencement of the export, in the form
30 HH:MM:SS
- 31 • PublisherName - The name of the person (or function) who initiated the export

1 An example of a Summary clause is:

```
2      (: SUMMARY
3         (ExporterName           "AutoPal ")
4         (ExporterVersion       "3.2 ")
5         (ExportDate            "1991/01/12")
6         (ExportTime            "13:00:00")
7         (PublisherName         "Andrew Shilling")
8         (HardwarePlatform      "Apple MacIntosh")
9         (VendorCompanyName     "UltraCASE")
10      )
```

12 4.4 Meta-model Section

13 4.4.1 Introduction

14 The Meta-model Section of the transfer consists of references to standardized Subject Areas,
15 followed by extensions to the Meta-model. The Meta-model for the transfer, known as the
16 "Working Meta-model", is defined by the set of meta-meta-entity and meta-meta-relationship
17 instances that are used in any of the referenced Subject Areas, plus those added by extensions.

18 The Working Meta-model may be extended in the following ways:

19 1. Adding instances to the Working Meta-model of meta-meta-entities to define additional
20 *MetaEntities*, *MetaRelationships*, *MetaAttributes* and *SubjectAreas*.

21 2. Adding instances to the Working Meta-model of meta-meta-relationships to define
22 additional associations between instances of meta-meta-entities in the Working Meta-
23 model. The following are the meta-meta-relationships currently defined in *EIA/IS-107*
24 *CDIF - Framework for Modeling and Extensibility*:

- 25 • *HasSource* to identify the source *MetaEntity* of a *MetaRelationship*
- 26 • *HasDestination* to identify the destination *MetaEntity* of a
27 *MetaRelationship*
- 28 • *IsLocalMetaAttributeOf* to associate a *MetaAttribute* with a specific *MetaEntity*
29 or *MetaRelationship*
- 30 • *HasSubtype* to define that one *MetaEntity* or *MetaRelationship* is a
31 subtype of another
- 32 • *IsUsedIn* to define that a *MetaEntity*, *MetaRelationship* or
33 *MetaAttribute* is used in a specific *SubjectArea*.

34 3. Extending domains of *MetaAttributes* with Enumerated data type to include additional
35 legal values.

1 As a conveyor of meta-models, this section contains instances of meta-meta-model concepts. The
2 mapping between these concepts, as defined in *EIA/IS-107 CDIF - Framework for Modeling and*
3 *Extensibility*, and this syntax is straight forward and therefore clarification is included only where
4 necessary.

5 The syntax of the Meta-model Section clause is:

```
6     <MetaModelSectionClause> ::=  
7         <OpenScope> <MetaModelKeyword>  
8         <CDIFSubjectAreaReferenceClause>...  
9         [ <MetaModelExtensionClause> ]...  
10        <CloseScope>
```

11 An example of the Meta-model Section clause is:

```
12     (:META-MODEL  
13         <CDIFSubjectAreaReferenceClause>...  
14         [ <MetaModelExtensionClause> ]...  
15     )
```

17 4.4.2 CDIF Subject Area Reference Clause

18 The syntax of the CDIF Subject Area Reference clause is:

```
19     <CDIFSubjectAreaReferenceClause> ::=  
20         <OpenScope>  
21         <SubjectAreaReferenceKeyword> <SubjectAreaName>  
22         <OpenScope>  
23         <VersionNumberKeyword> <SubjectAreaVersionNumber>  
24         <CloseScope>  
25         <CloseScope>
```

```
26     <SubjectAreaName> ::=  
27         <MetaObjectName>
```

```
28     <SubjectAreaVersionNumber> ::=  
29         <String>
```

30 <SubjectAreaName> must be the value of the *Name* meta-meta-attribute of the relevant
31 *SubjectArea* instance. <SubjectAreaVersionNumber> must be the value of the *VersionNumber*
32 meta-meta-attribute of the relevant *SubjectArea* instance. A reference to at least one CDIF
33 Subject Area must be included in the Meta-model Section.

34 An example of the CDIF Subject Area Reference clause is:

```
35     (:SUBJECTAREAREFERENCE Foundation  
36         (:VERSIONNUMBER "01.00")  
37     )
```

4.4.3 Meta-model Extension Clause

The syntax of the Meta-model Extension clause is:

```
<MetaModelExtensionClause> ::=  
    <MetaMetaEntityInstance>  
    | <MetaMetaRelationshipInstance>  
    | <EnumeratedMetaAttributeExtension>
```

Note that although the syntax allows Meta-meta-entity Instance, Meta-meta-relationship Instance and Enumerated Meta-attribute Extension clauses to be placed in arbitrary order, no forward references are allowed in a transfer.

4.4.4 Meta-meta-entity Instance

Instances of meta-meta-entities are specified by the name of the meta-meta-entity (e.g., *MetaEntity*, *MetaRelationship*, *MetaAttribute* or *SubjectArea*), followed by a unique CDIF Meta-identifier for the instance, followed by the names and values of the other meta-meta-attributes for the meta-meta-entity.

The syntax of the Meta-meta-entity Instance clause is:

```
<MetaMetaEntityInstance> ::=  
    <OpenScope>  
    <MetaMetaEntityName>  
    <CDIFMetaIdentifier>  
    [ <MetaMetaAttributeInstance> ] ...  
    <CloseScope>  
  
<MetaMetaEntityName> ::=  
    <MetaMetaObjectName>  
  
<CDIFMetaIdentifier> ::=  
    <Identifier>
```

<MetaMetaEntityName> must be the name of a meta-meta-entity, as defined in *EIA/IS-107 CDIF - Framework for Modeling and Extensibility*, e.g., *MetaEntity*, *MetaRelationship*, *MetaAttribute* or *SubjectArea*.

CDIF Meta-identifiers beginning with a numeric character are reserved for use in the CDIF Integrated Meta-model.

A meta-meta-attribute Instance clause is used to represent each of the meta-meta-attributes (other than the *CDIFMetaIdentifier* meta-meta-attribute) of the meta-meta-entity.

<MetaMetaAttributeInstance> is defined in the next section.

1 Several examples of the Meta-meta-entity Instance clause are:

- ```
2 (1) (SubjectArea NEW00001 [<MetaMetaAttributeInstance>] ...)
3 (2) (MetaEntity ME001
4 [<MetaMetaAttributeInstance>] ...
5)
6 (3) (MetaAttribute MA001
7 [<MetaMetaAttributeInstance>] ...
8)
9 (4) (MetaRelationship MR001
10 [<MetaMetaAttributeInstance>] ...
11)
```

#### 13 4.4.5 Meta-meta-attribute Instance

14 Instances of meta-meta-attributes (other than the *CDIFMetaIdentifier* meta-meta-attribute) are  
15 specified by a sequence consisting of the name of the meta-meta-attribute followed by its value.

16 The syntax of the Meta-meta-attribute Instance clause is:

```
17 <MetaMetaAttributeInstance> ::=
18 <OpenScope>
19 <MetaMetaAttributeName> <MetaMetaAttributeValue>
20 <CloseScope>
21
22 <MetaMetaAttributeName> ::=
23 <MetaMetaObjectName>
24
25 <MetaMetaAttributeValue> ::=
26 <MetaAttributeValue>
```

25 <MetaMetaAttributeName> must be the name of a meta-meta-attribute associated with the meta-  
26 meta-entity, as defined in *EIA/IS-107 CDIF - Framework for Modeling and Extensibility*.

27 <MetaAttributeValue> is defined in Section 4.5.5 Meta-attribute Value.

28 Several examples of the Meta-meta-attribute Instance clause are:

- ```
29 (1) (Name *Function*)
30 (2) (Description #[This defines a Business Function which is associated with
31     a particular Process or Processes in a Data Flow Model]#)
32 (3) (Aliases "Business Function,Activity")
33 (4) (IsOptional -True-)
```

4.4.6 Meta-meta-relationship Instance

Instances of meta-meta-relationships are specified by a sequence consisting of the name of the meta-meta-relationship and the CDIF meta-identifiers for the source and destination meta-meta-entities. *CollectableMetaObject.IsUsedIn.SubjectArea*, *MetaAttribute.IsLocalMetaAttributeOf.AttributableMetaObject*, *AttributableMetaObject.HasSubtype.AttributableMetaObject*, *MetaRelationship.HasSource.MetaEntity*, and *MetaRelationship.HasDestination.MetaEntity* are full names of meta-meta-relationships.

There shall be no forward references to meta-meta-entities defined later in the Meta-model Section.

The syntax of the Meta-meta-relationship Instance clause is:

```
<MetaMetaRelationshipInstance> ::=
    <OpenScope>
    <FullMetaMetaRelationshipName>
    <SourceMetaMetaEntityCDIFMetaIdentifier>
    <DestinationMetaMetaEntityCDIFMetaIdentifier>
    <CloseScope>

<FullMetaMetaRelationshipName> ::=
    <SourceMetaMetaEntityName> <Dot> <MetaMetaRelationshipName>
    <Dot> <DestinationMetaMetaEntityName>

<SourceMetaMetaEntityName> ::=
    <MetaMetaObjectName>

<MetaMetaRelationshipName> ::=
    <MetaMetaObjectName>

<DestinationMetaMetaEntityName> ::=
    <MetaMetaObjectName>

<SourceMetaMetaEntityCDIFMetaIdentifier> ::=
    <CDIFMetaIdentifier>

<DestinationMetaMetaEntityCDIFMetaIdentifier> ::=
    <CDIFMetaIdentifier>

<CDIFMetaIdentifier> ::=
    <Identifier>
```

<FullMetaMetaRelationshipName> must be the full name of a meta-meta-relationship, as defined in *EIA/IS-107 CDIF - Framework for Modeling and Extensibility*.

1 Several examples of the Meta-meta-relationship Instance clause are:

- 2 (1) (MetaAttribute.IsLocalMetaAttributeOf.AttributableMetaObject
3 MYID002 XYZ001)
4 !! This adds a Meta-meta-relationship instance to make the *MetaAttribute*
5 with *CDIFMetaIdentifier* MYID002 a local meta-attribute of the
6 *MetaEntity* or *MetaRelationship* with *CDIFMetaIdentifier* XYZ001.
7 (2) (CollectableMetaObject.IsUsedIn.SubjectArea
8 MATT006 SUBJ001)
9

10 4.4.7 Enumerated Meta-attribute Extension

11 The syntax of the Enumerated Meta-attribute Extension clause is:

```
12 <EnumeratedMetaAttributeExtension> ::=
13     <OpenScope>
14     <ExtendMetaAttributeKeyword> <CDIFMetaIdentifier>
15     <OpenScope>
16     <EnumeratedIdentifierValue>
17     [<EnumeratedSeparator> <EnumeratedIdentifierValue>]...
18     <CloseScope>
19     <CloseScope>
20
21 <CDIFMetaIdentifier> ::=
22     <Identifier>
23
24 <EnumeratedIdentifierValue> ::=
25     <EnumeratedValue>
```

24 *CDIFMetaIdentifier* must identify a *MetaAttribute* with *DataType* Enumerated defined in the
25 standardized model or earlier in the meta-model extensions. The Enumerated Identifier Values
26 are appended to those values that have already been defined for the *MetaAttribute*.

27 An example of the Enumerated Meta-attribute Extension clause is:

```
28 (:EXTENDMETA-ATTRIBUTE MA001 (<Encrypted>,<Nonencrypted>))
```

30 4.5 Model Section

31 4.5.1 Introduction

32 The Model Section contains references to attributable meta-objects (object types) and actual
33 model data. The object types referenced here are instances of *MetaEntities* and
34 *MetaRelationships* that were defined in the Meta-model Section as part of the CDIF Subject Area
35 references or in the Meta-model Extension clauses. The model data are in the form of meta-entity
36 instances, meta-relationship instances and meta-attribute instances.

1 As a conveyor of models, this section contains instances of meta-model concepts. The mapping
2 between these concepts, as defined in the CDIF Integrated Meta-model, and this syntax is
3 straightforward and therefore clarification is included only where necessary.

4 All meta-entity and meta-relationship instances in the Model section are uniquely identified by the
5 *CDIFIdentifier* meta-attribute. The exporter is responsible for ensuring the uniqueness of these
6 identifiers. An importer does not have to retain them, having completed the importation process
7 (successfully or unsuccessfully).

8 The syntax of the Model Section clause is:

```
9     <ModelSectionClause> ::=  
10         <OpenScope>  
11         <ModelKeyword> <ObjectClause>...  
12         <CloseScope>  
  
13     <ObjectClause> ::=  
14         <MetaEntityInstance> | <MetaRelationshipInstance>
```

15 An example of the Model Section clause is:

```
16     (:MODEL <ObjectClause> ... )
```

18 4.5.2 Meta-entity Instance

19 Instances of *MetaEntities* are specified by a sequence consisting of the name of the *MetaEntity* (as
20 defined in the meta-model) and a unique identifier for the meta-entity instance, followed by the
21 names and values of any meta-attributes of the meta-entity (as defined in the meta-model).

22 The syntax of the Meta-entity Instance clause is:

```
23     <MetaEntityInstance> ::=  
24         <OpenScope>  
25         <MetaEntityName> <CDIFIdentifier>  
26         [ <MetaAttributeInstance> ] ...  
27         <CloseScope>  
  
28     <MetaEntityName> ::=  
29         <MetaObjectName>  
  
30     <CDIFIdentifier> ::=  
31         <Identifier>
```

32 An example of the Meta-entity Instance clause is:

```
33     (DataModel MOD01 [ <MetaAttributeInstance> ...`      )
```

4.5.3 Meta-relationship Instance

Instances of *MetaRelationships* are specified by a sequence consisting of the name of the *MetaRelationship* (as defined in the meta-model), a unique identifier for the instance, a source *MetaEntity* identifier, a destination *MetaEntity* identifier, and the names and values of any meta-attributes of the meta-relationship (as defined in the meta-model).

There shall be no forward references to meta-entities defined later in the Model Section.

The syntax of the Meta-relationship Instance clause is:

```
<MetaRelationshipInstance> ::=
    <OpenScope>
    <FullMetaRelationshipName>
    <MetaRelationshipCDIFIdentifier>
    <SourceMetaEntityCDIFIdentifier>
    <DestinationMetaEntityCDIFIdentifier>
    [ <MetaAttributeInstance> ] ...
    <CloseScope>

<FullMetaRelationshipName> ::=
    <SourceMetaEntityName> <Dot> <MetaRelationshipName> <Dot>
    <DestinationMetaEntityName>

<SourceMetaEntityName> ::=
    <MetaObjectName>

<MetaRelationshipName> ::=
    <MetaObjectName>

<DestinationMetaEntityName> ::=
    <MetaObjectName>

<MetaRelationshipCDIFIdentifier> ::=
    <CDIFIdentifier>

<SourceMetaEntityCDIFIdentifier> ::=
    <CDIFIdentifier>

<DestinationMetaEntityCDIFIdentifier> ::=
    <CDIFIdentifier>

<CDIFIdentifier> ::=
    <Identifier>
```

<SourceMetaEntityName>, <MetaRelationshipName> and <DestinationMetaEntityName> must be the names used in the definition of the meta-relationship.

1 An example of the Meta-relationship Instance clause is:

```
2 (DataModel.IsCollectionOf.DataModelObject R001 MOD01 ENT02)
```

3 4 **4.5.4 Meta-attribute Instance**

5 Instances of *MetaAttributes* (other than the *CDIFIdentifier* meta-attribute) are specified by a
6 sequence consisting of the name of the *MetaAttribute* (as defined in the meta-model), followed by
7 its value.

8 The syntax of the Meta-attribute Instance clause is:

```
9 <MetaAttributeInstance> ::=  
10     <OpenScope>  
11     <MetaAttributeName> <MetaAttributeValue>  
12     <CloseScope>  
  
13 <MetaAttributeName> ::=  
14     <MetaObjectName>
```

15 An example of the Meta-attribute Instance clause is:

```
16 (Name <MetaAttributeValue>)
```

17 **4.5.5 Meta-attribute Value**

18 **4.5.5.1 Introduction**

19 The syntax of the Meta-attribute Value clause is:

```
20 <MetaAttributeValue> ::=  
21     <BitmapValue> | <BooleanValue> | <DateValue>  
22     | <EnumeratedValue> | <FloatValue> | <IdentifierValue>  
23     | <IntegerValue> | <IntegerListValue> | <PointValue>  
24     | <PointListValue> | <StringValue> | <TextValue>  
25     | <TimeValue>
```

26 **4.5.5.2 Bitmap Value**

27 A Bitmap value is defined as a list of pixel values, preceded by the height and width dimensions of
28 the bitmap. A pixel value is a red/green/blue triple representing the intensity of the respective
29 colors. Pixel values in the bitmap are ordered left-to-right and top-to-bottom.

1 The syntax of the Bitmap Value clause is:

```
2     <BitmapValue> ::=
3         <BitmapKeyword> <Height> <Width>
4         <OpenScope>
5         <Bitmap>
6         <CloseScope>
7
7     <Height> ::=
8         <HeightKeyword> <PositiveInteger>
9
9     <Width> ::=
10        <WidthKeyword> <PositiveInteger>
11
11    <Bitmap> ::=
12        <PixelValue> [ <ListSeparator> <PixelValue> ] ...
13
13    <PixelValue> ::=
14        <OpenScope>
15        <PixelRedIntensity> <PixelSeparator>
16        <PixelGreenIntensity> <PixelSeparator>
17        <PixelBlueIntensity>
18        <CloseScope>
19
19    <PixelRedIntensity> ::=
20        <PixelIntensity>
21
21    <PixelGreenIntensity> ::=
22        <PixelIntensity>
23
23    <PixelBlueIntensity> ::=
24        <PixelIntensity>
```

25 The following are examples of valid and invalid bitmap values:

- ```
26 (1) :BITMAP :HEIGHT 2 :WIDTH 2
27 ((120,50,35),(130,80,70),
28 (100,28,231),(111,255,0)) - Valid
29 (2) :BITMAP :HEIGHT 1 :WIDTH 2
30 ((120,50,35),(130,80,70)) - Valid
31 (3) :BITMAP :HEIGHT 2 :WIDTH 2
32 ((255,255,255),(255,255,255),
33 (255,255,255),(255,255,255)) - Valid
34 (4) :BITMAP ((120,50,35),(130,80,70),
35 (100,28,231),(111,255,0)) - Invalid: missing height and width
36 (5) :HEIGHT 1 :WIDTH 2
37 ((120,50,35),(130,80,70)) - Invalid: missing bitmap keyword
```

### 4.5.5.3 Boolean Value

The syntax of the Boolean Value clause is:

```
<BooleanValue> ::=
 <TrueValue> | <FalseValue>
```

The following are examples of the Boolean Value clause:

- (1) -TRUE-
- (2) -FALSE-

### 4.5.5.4 Date Value

The syntax of the Date Value clause is:

```
<DateValue> ::=
 <DateKeyword> <Date> <DateClassValue>

<DateClassValue> ::=
 <Absolute> | <RelativePositive> | <RelativeNegative>
```

The following are examples of valid and invalid date values:

- (1) :DATE 1940/12/07 Absolute - Valid
- (2) :DATE 1920/07/20 - Invalid: <DateClassValue> missing
- (3) :DATE 0002/10/11 RelativePositive - Valid
- (4) 1940/12/07 Absolute - Invalid: date keyword missing

### 4.5.5.5 Enumerated Value

<EnumeratedValue> is a terminal symbol in the syntax. Its representation is specified in the encoding.

An example of the Enumerated Value clause is:

```
<Red>
```

### 4.5.5.6 Float Value

A Float value is a decimal number with up to 16 decimal digits of precision within the range of  $10^{-1023}$  to  $10^{1023}$ .

<FloatValue> is a terminal symbol in the syntax. Its representation is specified in the encoding.

1 Several examples of valid Float values are:

- 2 (1) #f4E9
- 3 (2) #f123.45E2
- 4 (3) #f-123.45E15
- 5 (4) #f+0.23E5
- 6 (5) #f0.23E-3
- 7 (6) #f0.23E+3

#### 8 9 **4.5.5.7 Identifier Value**

10 <IdentifierValue> is a terminal symbol in the syntax. Its representation is specified in the  
11 encoding.

12 An example of the Identifier Value clause is:

13 \*johnBrownsBody\*

#### 14 15 **4.5.5.8 Integer Value**

16 Integer values are expressed as decimal values, binary values, hexadecimal values or octal values.

17 The syntax of the Integer Value clause is:

18 <IntegerValue> ::=  
19                   <DecimalIntegerValue> | <BinaryValue> | <HexadecimalValue>  
20                   | <OctalValue>

21 The following are examples of the Integer Values clause:

- 22 (1) #d12345               -     positive decimal
- 23 (2) #d-12345             -     negative decimal
- 24 (3) #d+12345             -     positive decimal
- 25 (4) #b10101              -     positive binary
- 26 (5) #b-10101             -     negative binary
- 27 (6) #o31727              -     positive octal
- 28 (7) #o-31727             -     negative octal
- 29 (8) #h1e2cf              -     positive hexadecimal
- 30 (9) #hffffffff           -     positive hexadecimal
- 31 (10) #h-ffffffff         -     negative hexadecimal

#### 32 33 **4.5.5.9 Integer List Value**

34 An Integer List is defined as a sequence of integers.

1 The syntax of the Integer List Value clause is:

```
2 <IntegerListValue> ::=
3 <IntegerListKeyword> <OpenScope>
4 <IntegerValue> [<ListSeparator> <IntegerValue>] ...
5 <CloseScope>
```

6 The following are valid and invalid examples of the Integer List Value clause:

- ```
7 (1)    :INTEGERLIST (#d10,#d20,#d11,#d15,#d26,#d32)    - Valid
8 (2)    :INTEGERLIST (#b10101,#h32fe,#o37712)           - Valid
9 (3)    :INTEGERLIST (#d10,#d2.4)                       - Invalid: 2.4 is not an integer value
10 (4)    (#d10,#d20,#d11,#d15, #d32)                   - Invalid: missing integer list keyword
```

12 4.5.5.10 Point Value

13 A Point Value is defined by three integer values representing x, y and z coordinates.

14 The syntax of the Point Value clause is:

```
15     <PointValue> ::=
16         <PointKeyword> <Point>
17
18     <Point> ::= <OpenScope>
19         <XValue> <PointSeparator>
20         <YValue> <PointSeparator>
21         <ZValue>
22         <CloseScope>
23
24     <XValue>    ::=
25         <Integer>
26
27     <YValue>    ::=
28         <Integer>
29
30     <ZValue>    ::=
31         <Integer>
```

28 The following are valid and invalid examples of the Point Value clause:

- ```
29 (1) :POINT(0 0 0) - Valid
30 (2) :POINT(1 303 292) - Valid
31 (3) :POINT(20 30) - Invalid: three coordinates must be supplied
32 (4) :POINT(qq eo 50) - Invalid: must be integer values
```

### 4.5.5.11 Point List Value

A Point List is defined as a sequence of points.

The syntax of the Point List Value clause is:

```
<PointListValue> ::=
 <PointListKeyword> <OpenScope>
 <Point> [<ListSeparator> <Point>] ...
 <CloseScope>

<Point> ::= <OpenScope>
 <XValue> <PointSeparator>
 <YValue> <PointSeparator>
 <ZValue>
 <CloseScope>
```

The following are examples of the Point List Value clause:

```
(1) :POINTLIST ((0 0 0),(1 1 1),(3 3 3)) - Valid
(2) :POINTLIST ((1 303 292),(321 22 33)) - Valid
(3) :POINTLIST 12) - Invalid: does not have an <OpenScope>
(4) ((0 0 0),(1 1 1)) - Invalid: point list keyword missing
```

### 4.5.5.12 String Value

A String Value is defined as a character string.

The syntax of the String Value clause is:

```
<StringValue> ::= <String>
```

An example of the String Value clause is:

```
"This is a string"
```

### 4.5.5.13 Text Value

The text data type is represented as a group of characters, not necessarily printable nor of any pre-determined maximum length. It is specified as a sequence of one or more text strings; the text value itself is the concatenation of the contents of the individual text strings.

The syntax of the Text Value clause is:

```
<TextValue> ::= <TextString> [<ListSeparator> <TextString>] ...
```

1 The specific encoding being used may constrain the maximum length of an individual text string.

2 An example of the Text Value clause is<sup>1</sup>:

```
3 #[Program SumIntegers(Input,Output);
4 var
5 total,input_integer : Integer;
6 begin
7 while not EOF(Input) do
8 begin
9 ReadLn(input_integer);
10 total:=total+input_integer
11 end;
12 WriteLn('Total =',total);
13 end.]#
```

#### 15 4.5.5.14 Time Value

16 The syntax of the Time Value clause is:

```
17 <TimeValue> ::=
18 <TimeKeyword> <Time> <TimeClassValue>

19 <TimeClassValue> ::=
20 <AbsoluteUTC> | <AbsoluteLocal> | <RelativePositive> |
21 <RelativeNegative>
```

22 The following are examples of valid and invalid time values:

- |    |     |                                     |                                     |
|----|-----|-------------------------------------|-------------------------------------|
| 23 | (1) | :TIME 07:20:23 AbsoluteUTC          | - Valid                             |
| 24 | (2) | :TIME 29:25:19 AbsoluteLocal        | - Invalid: illegal hour value 29    |
| 25 | (3) | :TIME 03:59:59                      | - Invalid: <TimeClassValue> missing |
| 26 | (4) | :TIME 00:00:00.250 RelativePositive | - Valid                             |
| 27 | (5) | 07:20:23 AbsoluteUTC                | - Invalid: time keyword missing     |

## 28 4.6 Comments

Comments may appear anywhere in the syntax between any two terminal symbols. Comments consist of a sequence of printable characters. Embedded whitespace characters are allowed.

<Comment> is a terminal symbol in the syntax. Its representation is specified in the encoding.

---

<sup>1</sup>This example of the Text Value clause has been laid out with whitespace characters, such as <CR>, transformed for presentation.

The specific encoding being used may constrain the maximum length of a Comment.

The following are examples of valid and invalid comments<sup>2</sup>:

- (1)    #| this is  
          a multi-line  
          comment  
          |#                               - Valid
- (2)    ( 1 2 #| buckle  
                  my shoe|# 3 4)       - Valid (in line comment)
- (3)    #| hello ... <eof>           - Invalid (unbalanced delimiters)

## 4.7 Syntax Terminal Symbols

The following are the terminal symbols in this syntax. Their representation is dependent on the specific encoding used. ENCODING.1 has been used for the examples in this document.

<Absolute>  
<AbsoluteUTC>  
<AbsoluteLocal>  
<BinaryValue>  
<BitmapKeyword>  
<CloseScope>  
<Comment>  
<Date>  
<DateKeyword>  
<DecimalIntegerValue>  
<Dot>  
<EnumeratedSeparator>  
<EnumeratedValue>  
<ExtendMetaAttributeKeyword>  
<FalseValue>  
<FloatValue>  
<HeaderKeyword>  
<HeightKeyword>  
<HexadecimalValue>  
<Identifier>  
<IdentifierValue>  
<Integer>  
<IntegerListKeyword>  
<ListSeparator>  
<MetaModelKeyword>  
<MetaMetaObjectName>  
<MetaObjectName>  
<ModelKeyword>

---

<sup>2</sup>These examples of the Comment clause have been laid out with whitespace characters, such as <CR>, transformed for presentation. <eof> is used to represent the end of the file.

<OctalValue>  
<OpenScope>  
<PixelIntensity>  
<PixelSeparator>  
<PointKeyword>  
<PointListKeyword>  
<PointSeparator>  
<PositiveInteger>  
<RelativeNegative>  
<RelativePositive>  
<String>  
<SubjectAreaReferenceKeyword>  
<SummaryKeyword>  
<TextString>  
<Time>  
<TimeKeyword>  
<TrueValue>  
<VersionNumberKeyword>  
<WidthKeyword>

The following terminal symbols are used as keywords in the grammar of this syntax.

<BitmapKeyword>  
<DateKeyword>  
<ExtendMetaAttributeKeyword>  
<HeaderKeyword>  
<HeightKeyword>  
<IntegerListKeyword>  
<MetaModelKeyword>  
<ModelKeyword>  
<PointKeyword>  
<PointListKeyword>  
<SubjectAreaReferenceKeyword>  
<SummaryKeyword>  
<TimeKeyword>  
<VersionNumberKeyword>  
<WidthKeyword>

The following terminal symbols act as token separators in this syntax:

<CloseScope>  
<Dot>  
<EnumeratedSeparator>  
<ListSeparator>  
<OpenScope>  
<PixelSeparator>  
<PointSeparator>

## 5. SYNTAX FORMAL GRAMMAR

This section contains a collection of the syntax presented in Section 4 Syntax Structures in the CDIF Transfer. This BNF alone is not a complete definition of SYNTAX.1. There are additional definitions and restrictions expressed in Section 3 Concepts and Definitions and in the text in Section 4 Syntax Structures in the CDIF Transfer.

```
<Bitmap> ::=
 <PixelValue> [<ListSeparator> <PixelValue>] ...

<BitmapValue> ::=
 <BitmapKeyword> <Height> <Width>
 <OpenScope>
 <Bitmap>
 <CloseScope>

<BooleanValue> ::=
 <TrueValue> | <FalseValue>

<CDIFIdentifier> ::=
 <Identifier>

<CDIFMetaIdentifier> ::=
 <Identifier>

<CDIFSubjectAreaReferenceClause> ::=
 <OpenScope>
 <SubjectAreaReferenceKeyword> <SubjectAreaName>
 <OpenScope>
 <VersionNumberKeyword> <SubjectAreaVersionNumber>
 <CloseScope>
 <CloseScope>

<CDIFTransfer> ::=
 <TransferEnvelope>
 <TransferContents>

<DateClassValue> ::=
 <Absolute> | <RelativePositive> | <RelativeNegative>

<DateValue> ::=
 <DateKeyword> <Date> <DateClassValue>

<DestinationMetaEntityCDIFIdentifier> ::=
 <CDIFIdentifier>

<DestinationMetaEntityName> ::=
 <MetaObjectName>
```

```
1 <DestinationMetaMetaEntityCDIFMetaIdentifier> ::=
2 <CDIFMetaIdentifier>

3 <DestinationMetaMetaEntityName> ::=
4 <MetaMetaObjectName>

5 <EnumeratedIdentifierValue> ::=
6 <EnumeratedValue>

7 <EnumeratedMetaAttributeExtension> ::=
8 <OpenScope>
9 <ExtendMetaAttributeKeyword> <CDIFMetaIdentifier>
10 <OpenScope>
11 <EnumeratedIdentifierValue>
12 [<EnumeratedSeparator> <EnumeratedIdentifierValue>] ...
13 <CloseScope>
14 <CloseScope>

15 <FullMetaMetaRelationshipName> ::=
16 <SourceMetaMetaEntityName> <Dot> <MetaMetaRelationshipName>
17 <Dot> <DestinationMetaMetaEntityName>

18 <FullMetaRelationshipName> ::=
19 <SourceMetaEntityName> <Dot> <MetaRelationshipName> <Dot>
20 <DestinationMetaEntityName>

21 <HeaderSectionClause> ::=
22 <OpenScope> <HeaderKeyword>
23 <SummaryClause>
24 <CloseScope>

25 <Height> ::=
26 <HeightKeyword> <PositiveInteger>

27 <IdentifierValuePair> ::=
28 <OpenScope>
29 <SummaryIdentifier> <StringValue>
30 <CloseScope>

31 <IntegerListValue> ::=
32 <IntegerListKeyword> <OpenScope>
33 <IntegerValue> [<ListSeparator> <IntegerValue>] ...
34 <CloseScope>

35 <IntegerValue> ::=
36 <DecimalIntegerValue> | <BinaryValue> | <HexadecimalValue>
37 | <OctalValue>

38 <MetaAttributeInstance> ::=
39 <OpenScope>
40 <MetaAttributeName> <MetaAttributeValue>
41 <CloseScope>
```

```
1 <MetaAttributeName> ::=
2 <MetaObjectName>

3 <MetaAttributeValue> ::=
4 <BitmapValue> | <BooleanValue> | <DateValue>
5 | <EnumeratedValue> | <FloatValue> | <IdentifierValue>
6 | <IntegerValue> | <IntegerListValue> | <PointValue>
7 | <PointListValue> | <StringValue> | <TextValue>
8 | <TimeValue>

9 <MetaEntityInstance> ::=
10 <OpenScope>
11 <MetaEntityName> <CDIFIdentifier>
12 [<MetaAttributeInstance>] ...
13 <CloseScope>

14 <MetaEntityName> ::=
15 <MetaObjectName>

16 <MetaMetaAttributeInstance> ::=
17 <OpenScope>
18 <MetaMetaAttributeName> <MetaMetaAttributeValue>
19 <CloseScope>

20 <MetaMetaAttributeName> ::=
21 <MetaMetaObjectName>

22 <MetaMetaAttributeValue> ::=
23 <MetaAttributeValue>

24 <MetaMetaEntityInstance> ::=
25 <OpenScope>
26 <MetaMetaEntityName>
27 <CDIFMetaIdentifier>
28 [<MetaMetaAttributeInstance>] ...
29 <CloseScope>

30 <MetaMetaEntityName> ::=
31 <MetaMetaObjectName>

32 <MetaMetaRelationshipInstance> ::=
33 <OpenScope>
34 <FullMetaMetaRelationshipName>
35 <SourceMetaMetaEntityCDIFMetaIdentifier>
36 <DestinationMetaMetaEntityCDIFMetaIdentifier>
37 <CloseScope>

38 <MetaMetaRelationshipName> ::=
39 <MetaMetaObjectName>
```

```
1 <MetaModelExtensionClause> ::=
2 <MetaMetaEntityInstance>
3 | <MetaMetaRelationshipInstance>
4 | <EnumeratedMetaAttributeExtension>
5
6 <MetaModelSectionClause> ::=
7 <OpenScope> <MetaModelKeyword>
8 <CDIFSubjectAreaReferenceClause>...
9 [<MetaModelExtensionClause>]...
10 <CloseScope>
11
12 <MetaRelationshipCDIFIdentifier> ::=
13 <CDIFIdentifier>
14
15 <MetaRelationshipInstance> ::=
16 <OpenScope>
17 <FullMetaRelationshipName>
18 <MetaRelationshipCDIFIdentifier>
19 <SourceMetaEntityCDIFIdentifier>
20 <DestinationMetaEntityCDIFIdentifier>
21 [<MetaAttributeInstance>] ...
22 <CloseScope>
23
24 <MetaRelationshipName> ::=
25 <MetaObjectName>
26
27 <ModelSectionClause> ::=
28 <OpenScope>
29 <ModelKeyword> <ObjectClause>...
30 <CloseScope>
31
32 <ObjectClause> ::=
33 <MetaEntityInstance> | <MetaRelationshipInstance>
34
35 <PixelBlueIntensity> ::=
36 <PixelIntensity>
37
38 <PixelGreenIntensity> ::=
39 <PixelIntensity>
40
41 <PixelRedIntensity> ::=
42 <PixelIntensity>
43
44 <PixelValue> ::=
45 <OpenScope>
46 <PixelRedIntensity> <PixelSeparator>
47 <PixelGreenIntensity> <PixelSeparator>
48 <PixelBlueIntensity>
49 <CloseScope>
```

```
1 <Point> ::= <OpenScope>
2 <XValue> <PointSeparator>
3 <YValue> <PointSeparator>
4 <ZValue>
5 <CloseScope>

6 <PointListValue> ::=
7 <PointListKeyword> <OpenScope>
8 <Point> [<ListSeparator> <Point>] ...
9 <CloseScope>

10 <PointValue> ::=
11 <PointKeyword> <Point>

12 <SourceMetaEntityCDIFIdentifier> ::=
13 <CDIFIdentifier>

14 <SourceMetaEntityName> ::=
15 <MetaObjectName>

16 <SourceMetaMetaEntityCDIFMetaIdentifier> ::=
17 <CDIFMetaIdentifier>

18 <SourceMetaMetaEntityName> ::=
19 <MetaMetaObjectName>

20 <StringValue> ::= <String>

21 <SubjectAreaName> ::=
22 <MetaObjectName>

23 <SubjectAreaVersionNumber> ::=
24 <String>

25 <SummaryClause> ::=
26 <OpenScope> <SummaryKeyword>
27 [<IdentifierValuePair>] ...
28 <CloseScope>

29 <SummaryIdentifier> ::=
30 <Identifier>

31 <TextValue> ::= <TextString> [<ListSeparator> <TextString>] ...

32 <TimeClassValue> ::=
33 <AbsoluteUTC> | <AbsoluteLocal> | <RelativePositive> |
34 <RelativeNegative>

35 <TimeValue> ::=
36 <TimeKeyword> <Time> <TimeClassValue>
```

```
1 <TransferContents> ::=
2 <HeaderSectionClause> <MetaModelSectionClause>
3 [<ModelSectionClause>]
4
4 <Width> ::=
5 <WidthKeyword> <PositiveInteger>
6
6 <XValue> ::=
7 <Integer>
8
8 <YValue> ::=
9 <Integer>
10
10 <ZValue> ::=
11 <Integer>
```



1  
2 **APPENDIX A**  
**<sup>3</sup>LL(1) EQUIVALENT OF SYNTAX.1**

3 This grammar is describes the same syntax as the BNF in Section 5 Syntax Formal Grammar. This grammar is  
4 strong LL(1) and has been computer analyzed.

5  
6 1 <CDIFTransfer> ::= <TransferEnvelope> <TransferContents>  
7  
8  
9 ! This section found in General Rules for Syntaxes and Encodings  
10  
11  
12 2 <TransferEnvelope> ::=  
13 <CDIFSignature> , <SyntaxIdentifier> , <EncodingIdentifier>  
14  
15  
16 3 <CDIFSignature> ::= CDIF  
17  
18 4 <SyntaxIdentifier> ::=  
19 SYNTAX <TransferEnvelopeSpace> <SyntaxId>  
20 <TransferEnvelopeSpace> <SyntaxVersion>  
21  
22 5 <EncodingIdentifier> ::=  
23 ENCODING <TransferEnvelopeSpace> <EncodingId>  
24 <TransferEnvelopeSpace> <EncodingVersion>  
25  
26  
27 6 <EncodingId> ::= <TransferEnvelopeString>  
28  
29 7 <EncodingVersion> ::= <TransferEnvelopeString>  
30  
31 8 <SyntaxId> ::= <TransferEnvelopeString>  
32  
33 9 <SyntaxVersion> ::= <TransferEnvelopeString>  
34  
35  
36 10 <TransferEnvelopeString> ::=  
37 " <EnvelopePrintableCharacter> <TransferEnvelopeStringTail>  
38  
39 11 <TransferEnvelopeStringTail> ::=  
40 "  
41 12 | <EnvelopePrintableCharacter> <TransferEnvelopeStringTail>

---

<sup>3</sup>This appendix is not a formal part of the attached EIA standard but is included for purposes of information only.

```
1
2
3 ! This section found in SYNTAX.1
4
5
6 13 <TransferContents> ::=
7 <HeaderSectionClause> <MetaModelSectionClause>
8 <ModelSectionClauseOptional>
9
10
11
12 14 <HeaderSectionClause> ::=
13 <OpenScope> <HeaderKeyword> <SummaryClause> <CloseScope>
14
15
16 15 <SummaryClause> ::=
17 <OpenScope> <SummaryKeyword>
18 <IdentifierValuePairOptionalList> <CloseScope>
19
20 16 <IdentifierValuePairOptionalList> ::=
21 <IdentifierValuePair> <IdentifierValuePairOptionalList>
22 17 | ... EMPTY PRODUCTION ...
23
24 18 <IdentifierValuePair> ::=
25 <OpenScope> <SummaryIdentifier> <StringValue> <CloseScope>
26
27 19 <SummaryIdentifier> ::= <Identifier>
28
29
30
31 20 <MetaModelSectionClause> ::=
32 <OpenScope> <MetaModelKeyword> <MetaModelClauseList>
33 <CloseScope>
34
35
36 21 <MetaModelClauseList> ::=
37 <CDIFSubjectAreaReferenceClause>
38 <MetaModelClauseOptionalList>
39
40 22 <MetaModelClauseOptionalList> ::=
41 <OpenScope> <MetaModelClauseOptionalListTail>
42 23 | ... EMPTY PRODUCTION ...
43
44 24 <MetaModelClauseOptionalListTail> ::=
45 <CDIFSubjectAreaReferenceClauseTail>
46 <MetaModelClauseOptionalList>
47 25 | <MetaModelExtensionClauseTail>
48 <MetaModelExtensionClauseOptionalList>
49
50
```

```
1 26 <CDIFSubjectAreaReferenceClause> ::=
2 <OpenScope> <SubjectAreaReferenceKeyword> <SubjectAreaName>
3 <OpenScope> <VersionNumberKeyword>
4 <SubjectAreaVersionNumber> <CloseScope> <CloseScope>
5
6 27 <SubjectAreaName> ::= <MetaObjectName>
7
8 28 <SubjectAreaVersionNumber> ::= <String>
9
10 29 <CDIFSubjectAreaReferenceClauseTail> ::=
11 <SubjectAreaReferenceKeyword> <SubjectAreaName> <OpenScope>
12 <VersionNumberKeyword> <SubjectAreaVersionNumber>
13 <CloseScope> <CloseScope>
14
15
16 30 <MetaModelExtensionClauseOptionalList> ::=
17 <MetaModelExtensionClause>
18 <MetaModelExtensionClauseOptionalList>
19 31 | ... EMPTY PRODUCTION ...
20
21 32 <MetaModelExtensionClause> ::=
22 <OpenScope> <MetaModelExtensionClauseTail>
23
24 33 <MetaModelExtensionClauseTail> ::=
25 <EnumeratedMetaAttributeExtensionTail>
26 34 | <MetaMetaObjectInstanceTail>
27
28
29 35 <EnumeratedMetaAttributeExtensionTail> ::=
30 <ExtendMetaAttributeKeyword> <CDIFMetaIdentifier>
31 <OpenScope> <EnumeratedIdentifierValueList> <CloseScope>
32 <CloseScope>
33
34 36 <EnumeratedIdentifierValueList> ::=
35 <EnumeratedIdentifierValue>
36 <EnumeratedIdentifierValueListTail>
37
38 37 <EnumeratedIdentifierValueListTail> ::=
39 <EnumeratedSeparator> <EnumeratedIdentifierValueList>
40 38 | ... EMPTY PRODUCTION ...
41
42 39 <EnumeratedIdentifierValue> ::= <EnumeratedValue>
43
44
45 40 <MetaMetaObjectInstanceTail> ::=
46 <MetaMetaEntityName> <MetaMetaObjectInstanceTailTail>
47 <CloseScope>
48
49 41 <MetaMetaObjectInstanceTailTail> ::=
50 <Dot> <MetaMetaRelationshipInstanceTail>
51 42 | <MetaMetaEntityInstanceTail>
52
53
```

```
1 43 <MetaMetaEntityInstanceTail> ::=
2 <CDIFMetaIdentifier> <MetaMetaAttributeInstanceOptionalList>
3
4 44 <MetaMetaAttributeInstanceOptionalList> ::=
5 <MetaMetaAttributeInstance>
6 <MetaMetaAttributeInstanceOptionalList>
7 45 | ... EMPTY PRODUCTION ...
8
9 46 <MetaMetaAttributeInstance> ::=
10 <OpenScope> <MetaMetaAttributeName> <MetaMetaAttributeValue>
11 <CloseScope>
12
13 47 <MetaMetaAttributeName> ::= <MetaMetaObjectName>
14
15 48 <MetaMetaAttributeValue> ::= <MetaAttributeValue>
16
17 49 <MetaMetaRelationshipInstanceTail> ::=
18 <MetaMetaRelationshipName> <Dot>
19 <DestinationMetaMetaEntityName>
20 <SourceMetaMetaEntityCDIFMetaIdentifier>
21 <DestinationMetaMetaEntityCDIFMetaIdentifier>
22
23 50 <MetaMetaRelationshipName> ::= <MetaMetaObjectName>
24
25 51 <DestinationMetaEntityName> ::= <MetaObjectName>
26
27
28
29 52 <ModelSectionClauseOptional> ::=
30 <OpenScope> <ModelKeyword> <ObjectClauseList> <CloseScope>
31 53 | ... EMPTY PRODUCTION ...
32
33
34 54 <ObjectClauseList> ::= <ObjectClause> <ObjectClauseListTail>
35
36 55 <ObjectClauseListTail> ::=
37 <ObjectClause> <ObjectClauseListTail>
38 56 | ... EMPTY PRODUCTION ...
39
40 57 <ObjectClause> ::=
41 <OpenScope> <MetaEntityName> <MetaObjectInstanceTail>
42 <MetaAttributeInstanceOptionalList> <CloseScope>
43
44 58 <MetaObjectInstanceTail> ::=
45 <MetaEntityInstanceTail>
46 59 | <MetaRelationshipInstanceTail>
47
48
49 60 <MetaEntityInstanceTail> ::= <CDIFIdentifier>
50
51 61 <MetaEntityName> ::= <MetaObjectName>
52
53
```

```
1 62 <MetaRelationshipInstanceTail> ::=
2 <Dot> <MetaRelationshipName> <Dot>
3 <DestinationMetaEntityName>
4 <MetaRelationshipCDIFIdentifier>
5 <SourceMetaEntityCDIFIdentifier>
6 <DestinationMetaEntityCDIFIdentifier>
7
8 63 <MetaRelationshipName> ::= <MetaObjectName>
9
10 64 <MetaAttributeInstanceOptionalList> ::=
11 <MetaAttributeInstance> <MetaAttributeInstanceOptionalList>
12 65 | ... EMPTY PRODUCTION ...
13
14 66 <MetaAttributeInstance> ::=
15 <OpenScope> <MetaAttributeName> <MetaAttributeValue>
16 <CloseScope>
17
18 67 <MetaAttributeName> ::= <MetaObjectName>
19
20
21
22 68 <MetaAttributeValue> ::=
23 <BitmapValue>
24 69 | <BooleanValue>
25 70 | <DateValue>
26 71 | <EnumeratedValue>
27 72 | <FloatValue>
28 73 | <IdentifierValue>
29 74 | <IntegerListValue>
30 75 | <IntegerValue>
31 76 | <PointListValue>
32 77 | <PointValue>
33 78 | <StringValue>
34 79 | <TextValue>
35 80 | <TimeValue>
36
37
38 81 <IntegerListValue> ::=
39 <IntegerListKeyword> <OpenScope> <IntegerValueList>
40 <CloseScope>
41
42 82 <IntegerValueList> ::= <IntegerValue> <IntegerValueListTail>
43
44 83 <IntegerValueListTail> ::=
45 <ListSeparator> <IntegerValue> <IntegerValueListTail>
46 84 | ... EMPTY PRODUCTION ...
47
48
49 85 <PointValue> ::= <PointKeyword> <Point>
50
51 86 <Point> ::=
52 <OpenScope> <XValue> <PointSeparator> <YValue>
53 <PointSeparator> <ZValue> <CloseScope>
```

```
1
2
3 87 <PointListValue> ::=
4 <PointListKeyword> <OpenScope> <PointList> <CloseScope>
5
6 88 <PointList> ::= <Point> <PointListTail>
7
8 89 <PointListTail> ::=
9 <ListSeparator> <Point> <PointListTail>
10 90 | ... EMPTY PRODUCTION ...
11
12
13 91 <BitmapValue> ::=
14 <BitmapKeyword> <Height> <Width> <OpenScope> <Bitmap>
15 <CloseScope>
16
17 92 <Height> ::= <HeightKeyword> <PositiveInteger>
18
19 93 <Width> ::= <WidthKeyword> <PositiveInteger>
20
21 94 <Bitmap> ::= <PixelValue> <BitmapTail>
22
23 95 <BitmapTail> ::=
24 <ListSeparator> <PixelValue> <BitmapTail>
25 96 | ... EMPTY PRODUCTION ...
26
27 97 <PixelValue> ::=
28 <OpenScope> <PixelRedIntensity> <PixelSeparator>
29 <PixelGreenIntensity> <PixelSeparator>
30 <PixelBlueIntensity> <CloseScope>
31
32 98 <PixelRedIntensity> ::= <PixelIntensity>
33
34 99 <PixelGreenIntensity> ::= <PixelIntensity>
35
36 100 <PixelBlueIntensity> ::= <PixelIntensity>
37
38
39 101 <BooleanValue> ::=
40 <FalseValue>
41 102 | <TrueValue>
42
43
44 103 <DateValue> ::= <DateKeyword> <Date> <DateClassValue>
45
46 104 <DateClassValue> ::=
47 <Absolute>
48 105 | <RelativeNegative>
49 106 | <RelativePositive>
50
51
```

```
1 107 <IntegerValue> ::=
2 <BinaryValue>
3 108 | <DecimalIntegerValue>
4 109 | <HexadecimalValue>
5 110 | <OctalValue>
6
7
8 111 <StringValue> ::= <String>
9
10
11 112 <TextValue> ::= <TextString> <TextValueTail>
12
13 113 <TextValueTail> ::=
14 <ListSeparator> <TextString> <TextValueTail>
15 114 | ... EMPTY PRODUCTION ...
16
17
18 115 <TimeValue> ::= <Time> <TimeClassValue>
19
20 116 <TimeClassValue> ::=
21 <AbsoluteUTC>
22 117 | <AbsoluteLocal>
23 118 | <RelativeNegative>
24 119 | <RelativePositive>
25
26
27
28 120 <CDIFIdentifier> ::= <Identifier>
29
30 121 <CDIFMetaIdentifier> ::= <Identifier>
31
32
33 122 <DestinationMetaEntityCDIFIdentifier> ::= <CDIFIdentifier>
34
35 123 <DestinationMetaMetaEntityName> ::= <MetaMetaObjectName>
36
37 124 <DestinationMetaMetaEntityCDIFMetaIdentifier> ::= <CDIFMetaIdentifier>
38
39
40 125 <MetaMetaEntityName> ::= <MetaMetaObjectName>
41
42
43 126 <MetaRelationshipCDIFIdentifier> ::= <CDIFIdentifier>
44
45
46 127 <SourceMetaEntityCDIFIdentifier> ::= <CDIFIdentifier>
47
48 128 <SourceMetaMetaEntityCDIFMetaIdentifier> ::= <CDIFMetaIdentifier>
49
50
51 129 <XValue> ::= <Integer>
52
53 130 <YValue> ::= <Integer>
```

1  
2 131 <ZValue> ::= <Integer>  
3  
4

1

This page is intentionally blank



1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23

## APPENDIX B

### <sup>4</sup>QUESTIONS AND ANSWERS

The following questions and answers are presented in order to help the reader's understanding of the standard. They have been derived from discussions held within the CDIF Technical Committee and presentations to outside observers.

**Question 1:**

**How do I distinguish between extending an existing *MetaEntity* and adding a new *MetaEntity*?**

**Answer:**

New *MetaEntities* are introduced by using the Meta-meta-entity Instance clause to add the new *Meta-Entity*. If the name of the *MetaEntity* already exists in any of the referenced Subject Areas (or as a result of a previous Meta-model Extension clause), then this is a semantic error. *MetaEntities* are extended by using the Meta-meta-relationship Instance clause to associate additional *MetaAttributes* or *MetaRelationships* with the *MetaEntity*.

**Question 2:**

**Why are forward references not allowed in a transfer?**

**Answer:**

Avoiding the use of forward references makes life easier for the importer.

**Question 3:**

**How do I add a value to list of valid values for a meta-attribute?**

**Answer:**

You use an enumerated attribute instance extension clause. Note that this will not add a new instance of a meta-attribute. This will extend the domain of an attribute.

---

<sup>4</sup>This appendix is not a formal part of the attached EIA standard but is included for purposes of information only.

1 **Question 4:**

2 **What are CDIFIdentifiers and CDIFMetaIdentifiers used for?**

3 **Answer:**

4 The CDIFMetaIdentifiers are local to a transfer and are used to reference meta-entities and  
5 subject areas in meta-relationship definitions. The CDIFIdentifiers are local to a transfer and are  
6 used to reference entities in relationship definitions. The CDIFIdentifiers and  
7 CDIFMetaIdentifiers of one transfer do not have any given association with those of another  
8 transfer.

9 **Question 5:**

10 **Can CDIFIdentifiers be used for incremental transfers?**

11 **Answer:**

12 No. CDIFIdentifiers are unique only within the context of a single transfer.

13 **Question 6:**

14 **What do we do with meta-relationships?**

15 **Answer:**

16 Relationships between meta-entities are added using Meta-meta-relationship Instance clauses.  
17 The meta-relationships are CollectableMetaObject.IsUsedIn.SubjectArea,  
18 MetaAttribute.IsLocalMetaAttributeOf.AttributableMetaObject,  
19 AttributableMetaObject.HasSubtype.AttributableMetaObject,  
20 MetaRelationship.HasSource.MetaEntity, and MetaRelationship.HasDestination.MetaEntity.

21 **Question 7:**

22 **How do I relate a MetaEntity to a SubjectArea?**

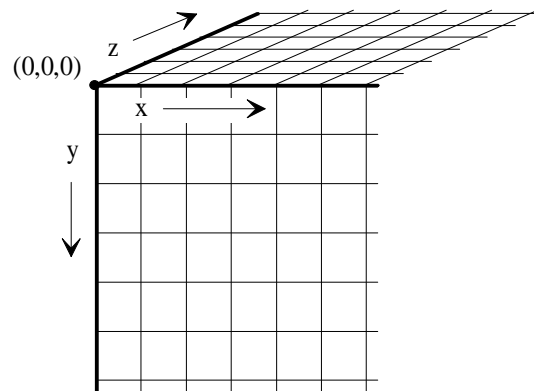
23 **Answer:**

24 Use the Meta-meta-relationship Instance clause to create an instance of the meta-meta-  
25 relationship CollectableMetaObject.IsUsedIn.SubjectArea.

## GLOSSARY<sup>1</sup>

|                                 |                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>ATTRIBUTE</b>                | A single-valued characteristic of an entity or relationship.                                                                                                                                                                                                                                                                                                                                                      |
| <b>CDIF</b>                     | See <i>CDIF Family of Standards</i> .                                                                                                                                                                                                                                                                                                                                                                             |
| <b>CDIF CLEAR TEXT ENCODING</b> | A human readable encoding for the CDIF syntax, SYNTAX.1.                                                                                                                                                                                                                                                                                                                                                          |
| <b>CDIF COORDINATE FRAME</b>    | The system of graphics addressing used by CDIF. Graphics space is defined as a coordinate system that is a three dimensional rectilinear grid. The intersections of grid lines define addresses (CDIF Points). The intersection of any two grid lines defines a coordinate plane. Each grid cell in a coordinate plane is a CDIF Pixel. All grid coordinates are integers and all grid lines are infinitely thin. |

The figure below describes the CDIF Coordinate Frame.



---

<sup>1</sup>This glossary is not a formal part of the attached EIA standard, but is included for purposes of information only.

**CDIF FAMILY OF STANDARDS**

The CDIF Family of Standards is composed of a set of standards that, when used together, provide a standard definition for the interchange of information between CASE tools.

**CDIF IDENTIFIER**

An attribute that uniquely identifies an object in the Model Section of a transfer.

**CDIF INTEGRATED META-MODEL**

The description of the set of concepts and notations used to define a model. The CDIF Integrated Meta-model defines an Entity-Relationship-Attribute model that is used to construct and define models used in systems development.

**CDIF METAIDENTIFIER**

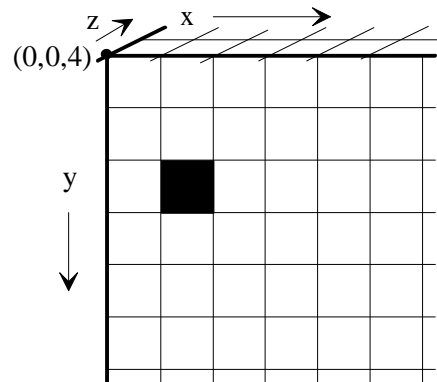
A meta-meta-attribute that uniquely identifies a meta-object in the Meta-model Section of a transfer.

**CDIF META-META-MODEL**

The description of the set of concepts and notations used to define a meta-model. Specifically, the CDIF Meta-meta-model defines an Entity-Relationship-Attribute model that is used to construct and define both meta-models and the CDIF Meta-meta-model itself.

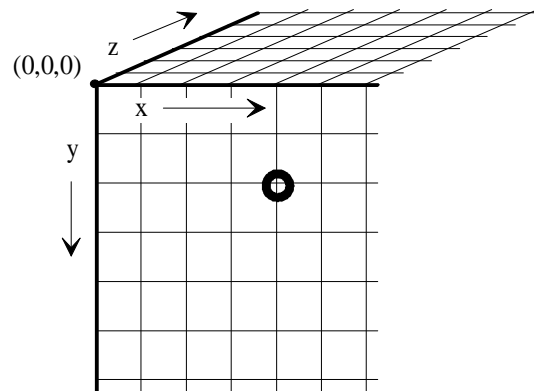
**CDIF PIXEL**

A grid cell on the coordinate plane in the CDIF Coordinate Frame. The shaded area in the figure below is a CDIF Pixel in the  $(x,y,4)$  coordinate plane.



**CDIF POINT**

The location of a point in 3-dimensional space as specified by the  $x,y,z$  coordinates. The circle in the figure below encloses a point located at  $(4,2,0)$ .



**CDIF TRANSFER**

A combination of a particular syntax, a particular encoding of that syntax, and a meta-model. In other words, a complete definition of the format and contents of a transfer.

|                                          |                                                                                                                                                                                                                               |
|------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>CDIF TRANSFER FORMAT</b>              | A combination of a particular syntax and a particular encoding of that syntax which together provides a complete definition of the transfer format.                                                                           |
| <b>CDIF TRANSFER SYNTAX AND ENCODING</b> | A standard vehicle format supported by CDIF. The combination of SYNTAX.1 and ENCODING.1 forms the initial CDIF Transfer Syntax and Encoding.                                                                                  |
| <b>CHARACTER SET</b>                     | A character set is a collection of characters used in an Encoding to represent terminal symbols. The character set used is significant in the encoding of text and string meta-attributes for a CDIF Transfer.                |
| <b>CLEAR TEXT</b>                        | A form of encoding where the resulting physical file is human-readable.                                                                                                                                                       |
| <b>COORDINATE PLANE</b>                  | See <i>CDIF Coordinate Frame</i> .                                                                                                                                                                                            |
| <b>ENCODING</b>                          | An encoding defines how the elements of a syntax are physically represented using an identified character set. Details of representation of the various terminal symbols and data types in the syntax's grammar are provided. |
| <b>ENCODING.1</b>                        | The CDIF Family of Standards supports multiple transfer formats, each composed of a syntax and an encoding. ENCODING.1 is the primary encoding defined within the CDIF Family of Standards.                                   |
| <b>ENTITY</b>                            | An object (i.e., thing, event or concept) that occurs in a model (i.e., transfer).                                                                                                                                            |
| <b>INFORMATION CONTENT</b>               | The Information Content is the set of meta-model and model instances found in a CDIF Transfer.                                                                                                                                |

**INSTANCE**

An individual occurrence of a type (e.g., the diagram *MyDiagram* is an instance of the *Diagram* type).

Usually, an instance is an object which occurs one meta-level beneath its (type's) definition. This is not always true, however (e.g., within the CDIF presentation subject areas where icon type definitions and their instances are both contained within the same meta-level).

**INTEGRATED META-MODEL**

See *CDIF Integrated Meta-model* .

**META**

According to Webster's Ninth New Collegiate Dictionary: *more comprehensive: transcending - used with the name of a discipline to designate a new but related discipline designed to deal critically with the original one.*

Meta- is used in CDIF generally as a prefix to a concept to imply definition information about the concept. Specifically, used to designate the location of an object in the three model layers.

**META-ATTRIBUTE**

A definition of a characteristic of a meta-entity or meta-relationship. Instances of a meta-attribute occur in a model as data values.

**META-ENTITY**

A definition of a type of data object that occurs in CDIF models. Specifically, a meta-entity represents a set of zero or more meta-attributes, stored together to represent a thing, event or concept that has instances in a model.

**META-LAYERS**

See *Model Layers*.

|                               |                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>META-META-ATTRIBUTE</b>    | A definition of a characteristic of a meta-meta-entity or meta-meta-relationship. Instances of a meta-meta-attribute occur in a meta-model as meta-data values.                                                                                                                                                                                                                                                                       |
| <b>META-META-ENTITY</b>       | A definition of the behavior and structure of meta-entities, meta-relationships, meta-attributes, or subject areas (i.e., a definition of the meta-object definitions used to describe information in models).                                                                                                                                                                                                                        |
| <b>META-META-MODEL</b>        | See <i>CDIF Meta-meta-model</i> .                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>META-META-RELATIONSHIP</b> | A definition of a type of data object that occurs in CDIF meta-models. Specifically, a meta-meta-relationship represents the definition of a relationship between instances of meta-meta-entities.                                                                                                                                                                                                                                    |
| <b>META-MODEL</b>             | <p>A meta-model contains detailed definitions of the meta-entities, meta-relationships and meta-attributes whose instances represent an actual CDIF transfer.</p> <p>The CDIF Integrated Meta-model (as defined in the set of standards that comprise the CDIF Family of Standards) is a definition of all of the types of information that can be transferred in a CDIF Transfer without using the CDIF extensibility mechanism.</p> |
| <b>META-OBJECT</b>            | A meta-object is a generic term for meta-entities and meta-relationships                                                                                                                                                                                                                                                                                                                                                              |
| <b>META-RELATIONSHIP</b>      | A definition of a type of data object that occurs in CDIF models. Specifically, a meta-relationship represents the definition of a relationship between meta-entities that has instances in a model. A meta-relationship may also define a set of zero or more meta-attributes, stored together to represent characteristics of a relationship between meta-entities.                                                                 |

|                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>MODEL</b>               | <p>A specific collection of software engineering data. This is called a model because it usually represents a model of a software system under development.</p> <p>A collection of instances of meta-objects.</p>                                                                                                                                                                                                                                                                                                                                                   |
| <b>MODEL LAYERS</b>        | <p>The different layers of definition (or abstraction) used in defining the CDIF Family of Standards. The four model layers in CDIF are: user data, model, meta-model, meta-meta-model.</p> <p>Any given model layer provides an accurate and complete definition of all the instances that may occur one layer below the given layer. For example, the meta-meta-model provides a set of definitions that are used to construct and understand the meta-model; the meta-model provides a set of definitions that are used to construct and understand a model.</p> |
| <b>NON-TERMINAL SYMBOL</b> | <p>A part of the hierarchical definition of a syntax that is further decomposed in the hierarchy.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>PRODUCTION RULE</b>     | <p>The definition of a syntactic element in the form of an expression consisting of characters, character strings, and other syntactic elements.</p>                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>RELATIONSHIP</b>        | <p>A real-world association among one or more entities. Where the association is between an entity and itself, the relationship is said to be recursive.</p>                                                                                                                                                                                                                                                                                                                                                                                                        |

|                           |                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>SUBJECT AREA</b>       | A related collection of meta-object instance definitions. Subject areas are used to define scoped areas of interest. Subject areas overlap to ensure the integration of the overall Meta-model, but a tool need only use those subject areas relevant to the data to be exported/imported.                                                                                                 |
| <b>SYNTAX</b>             | A syntax is a definition of the format of information in a CDIF transfer. The definition is in the form of a grammar. It is specified with regard to ordering and repetition, down to the level of terminals, but does not specify the representation of any of the terminal objects in the grammar. Those decisions are defined in the encoding of the syntax.                            |
| <b>SYNTAX.1</b>           | The CDIF Family of Standards supports multiple transfer formats, each composed of a syntax and an encoding. SYNTAX.1 is the primary syntax defined within the CDIF Family of Standards.                                                                                                                                                                                                    |
| <b>TERMINAL SYMBOL</b>    | A part of the hierarchical definition of a syntax that is <b>not</b> further decomposed in the hierarchy.                                                                                                                                                                                                                                                                                  |
| <b>TRANSFER</b>           | See <i>CDIF Transfer</i> .                                                                                                                                                                                                                                                                                                                                                                 |
| <b>TRANSFER FORMAT</b>    | See <i>CDIF Transfer Format</i> .                                                                                                                                                                                                                                                                                                                                                          |
| <b>WORKING META-MODEL</b> | The working meta-model is the definition of the specific meta-objects that may be instantiated in the model section of a CDIF Transfer. The working meta-model comprises the meta-objects in the CDIF Integrated Meta-model that are used by the subject areas referenced in the meta-model section of the transfer, and the meta-objects defined as extensions in the meta-model section. |

## INDEX

1

### 2 Special Symbols

- 3 **!!** 6
- 4 **...** 5
- 5 **::=** 5
- 6 **<>** 5
- 7 **[]** 5
- 8 **{ }** 5
- 9 **|** 5

### 10 A

- 11 Alternative operator 5
- 12 Angle braces 5
- 13 Attribute
- 14 glossary entry 45

### 15 B

- 16 Backus Naur Form (See BNF)
- 17 BNF 5
- 18 (See also Formal grammar)
- 19 conventions 5-6
- 20 Bold type 4, 6
- 21 Braces
- 22 angle 5
- 23 curly 5
- 24 square 5
- 25 Byte stream 8

### 26 C

- 27 CDIF
- 28 glossary entry 45
- 29 CDIF Clear Text Encoding
- 30 glossary entry 45
- 31 CDIF Coordinate Frame
- 32 glossary entry 45
- 33 CDIF Family of Standards 1
- 34 glossary entry 46
- 35 CDIF Identifier 17
- 36 glossary entry 46
- 37 CDIF Integrated Meta-model

- 38 glossary entry 46
- 39 CDIF Meta-Meta-Model
- 40 glossary entry 46
- 41 CDIF MetaIdentifier
- 42 glossary entry 46
- 43 CDIF Pixel
- 44 glossary entry 47
- 45 CDIF Point
- 46 glossary entry 47
- 47 CDIF Signature 8
- 48 CDIF Transfer
- 49 glossary entry 47
- 50 CDIF Transfer Format
- 51 glossary entry 48
- 52 CDIF Transfer Syntax and Encoding
- 53 glossary entry 48
- 54 Character Set
- 55 glossary entry 48
- 56 Clear Text
- 57 glossary entry 48
- 58 Comment operator 6
- 59 Conventions 4
- 60 BNF 5-6
- 61 Coordinate Plane
- 62 glossary entry 48
- 63 Curly braces 5

### 64 D

- 65 Data type
- 66 identifier 10
- 67 DateClassValue 21
- 68 Definition operator 5
- 69 Document
- 70 structure 3

### 71 E

- 72 Element
- 73 syntactic 5
- 74 Ellipsis 5
- 75 Encoding
- 76 glossary entry 48
- 77 identifier 8
- 78 ENCODING.1
- 79 glossary entry 48

|    |                                             |                |                                        |            |
|----|---------------------------------------------|----------------|----------------------------------------|------------|
| 1  | Entity                                      | 49             | TimeKeyword                            | 27         |
| 2  | glossary entry                              | 48             | VersionNumberKeyword                   | 27         |
| 3  | Enumerated meta-attribute                   | 51             | WidthKeyword                           | 27         |
| 4  | extension                                   | 16             | MetaAttributeInstance                  | 19, 29     |
| 5  | Expression                                  | 5              | MetaAttributeName                      | 19, 30     |
| 6  | <b>F</b>                                    |                | MetaAttributeValue                     | 19, 30     |
| 7  | File                                        | 8              | MetaEntityInstance                     | 17, 30     |
| 8  | Formal grammar                              |                | MetaEntityName                         | 17, 30     |
| 9  | MetaModelExtensionClause                    | 31             | MetaMetaAttributeInstance              | 14, 30     |
| 10 | Formal grammar                              | 28-33          | MetaMetaAttributeName                  | 14, 30     |
| 11 | (See also BNF)                              |                | MetaMetaAttributeValue                 | 14, 30     |
| 12 | Bitmap                                      | 20, 28         | MetaMetaEntityInstance                 | 13, 30     |
| 13 | BitmapValue                                 | 20, 28         | MetaMetaEntityName                     | 13, 30     |
| 14 | BooleanValue                                | 21, 28         | MetaMetaRelationshipInstance           | 15, 30     |
| 15 | CDIFIdentifier                              | 17, 18, 28     | MetaMetaRelationshipName               | 15, 30     |
| 16 | CDIFMetaIdentifier                          | 13, 15, 16, 28 | MetaModelExtensionClause               | 13         |
| 17 | CDIFSubjectAreaReferenceClause              | 12, 28         | MetaModelSectionClause                 | 12, 31     |
| 18 | CDIFTransfer                                | 8, 28          | MetaRelationshipCDIFIdentifier         | 18, 31     |
| 19 | DateClassValue                              | 28             | MetaRelationshipInstance               | 18, 31     |
| 20 | DateValue                                   | 21, 28         | MetaRelationshipName                   | 18, 31     |
| 21 | DestinationMetaEntityCDIFIdentifier         | 18, 28         | ModelKeyword                           | 26         |
| 22 | DestinationMetaEntityName                   | 18, 28         | ModelSectionClause                     | 17, 31     |
| 23 | DestinationMetaMetaEntityCDIFMetaIdentifier | 15, 29         | ObjectClause                           | 31         |
| 24 | DestinationMetaMetaEntityName               | 15, 29         | PixelBlueIntensity                     | 20, 31     |
| 25 | EnumeratedIdentifierValue                   | 16, 29         | PixelGreenIntensity                    | 20, 31     |
| 26 | EnumeratedMetaAttributeExtension            | 16, 29         | PixelRedIntensity                      | 20, 31     |
| 27 | FullMetaMetaRelationship                    | 29             | PixelValue                             | 20, 31     |
| 28 | FullMetaMetaRelationshipName                | 15             | Point                                  | 23, 24, 32 |
| 29 | FullMetaRelationshipName                    | 18, 29         | PointListKeyword                       | 27         |
| 30 | HeaderSectionClause                         | 9, 29          | PointListValue                         | 24, 32     |
| 31 | Height                                      | 20, 29         | PointValue                             | 23, 32     |
| 32 | IdentifierValuePair                         | 10, 29         | separator                              |            |
| 33 | IntegerListValue                            | 23, 29         | CloseScope                             | 27         |
| 34 | IntegerValue                                | 22, 29         | Dot                                    | 27         |
| 35 | keyword                                     |                | EnumeratedSeparator                    | 27         |
| 36 | BitmapKeyword                               | 27             | ListSeparator                          | 27         |
| 37 | DateKeyword                                 | 27             | OpenScope                              | 27         |
| 38 | ExtendMetaAttributeKeyword                  | 27             | PixelSeparator                         | 27         |
| 39 | HeaderKeyword                               | 27             | PointSeparator                         | 27         |
| 40 | HeightKeyword                               | 27             | SourceMetaEntityCDIFIdentifier         | 18, 32     |
| 41 | IntegerListKeyword                          | 27             | SourceMetaEntityName                   | 18, 32     |
| 42 | MetaModelKeyword                            | 27             | SourceMetaMetaEntityCDIFMetaIdentifier | 15, 32     |
| 43 | ModelKeyword                                | 27             | SourceMetaMetaEntityName               | 15, 32     |
| 44 | PointKeyword                                | 27             | StringValue                            | 10, 24, 32 |
| 45 | PointListKeyword                            | 27             | SubjectAreaName                        | 12, 32     |
| 46 | SubjectAreaReferenceKeyword                 | 27             | SubjectAreaVersionNumber               | 12, 32     |
| 47 | SummaryKeyword                              | 27             | SummaryClause                          | 10, 32     |
| 48 |                                             |                | SummaryIdentifier                      | 10, 32     |
|    |                                             |                | SyntaxID                               | 7          |
|    |                                             |                | SyntaxVersion                          | 7          |

---

|    |                                |    |                       |
|----|--------------------------------|----|-----------------------|
| 1  | TextValue 24, 32               | 52 | YValue 23, 33         |
| 2  | TimeClassValue 25, 32          | 53 | ZValue 23, 33         |
| 3  | TimeValue 25, 32               | 54 | Formal grammar        |
| 4  | token                          | 55 | token                 |
| 5  | Absolute 26                    | 56 | OctalValue 27         |
| 6  | AbsoluteLocal 26               |    |                       |
| 7  | AbsoluteUTC 26                 | 57 | <b>G</b>              |
| 8  | BinaryValue 26                 |    |                       |
| 9  | BitmapKeyword 26               | 58 | Grammar 5             |
| 10 | CloseScope 26                  |    |                       |
| 11 | Comment 26                     | 59 | <b>H</b>              |
| 12 | Date 26                        |    |                       |
| 13 | DateKeyword 26                 | 60 | Header Section 9      |
| 14 | DecimalIntegerValue 26         |    |                       |
| 15 | Dot 26                         | 61 | <b>I</b>              |
| 16 | EnumeratedSeparator 26         |    |                       |
| 17 | EnumeratedValue 26             | 62 | Identifier            |
| 18 | ExtendMetaAttributeKeyword 26  | 63 | data type 10          |
| 19 | FalseValue 26                  | 64 | Information Content   |
| 20 | FloatValue 26                  | 65 | glossary entry 48     |
| 21 | HeaderKeyword 26               | 66 | Instance              |
| 22 | HeightKeyword 26               | 67 | glossary entry 49     |
| 23 | HexaDecimalValue 26            | 68 | Integrated Meta-model |
| 24 | Identifier 26                  | 69 | glossary entry 49     |
| 25 | IdentifierValue 26             |    |                       |
| 26 | Integer 26                     | 70 | <b>J</b>              |
| 27 | IntegerListKeyword 26          |    |                       |
| 28 | ListSeparator 26               | 71 | <b>K</b>              |
| 29 | MetaMetaObjectName 26          |    |                       |
| 30 | MetaModelKeyword 26            | 72 | Keyword 27            |
| 31 | MetaObjectName 26              |    |                       |
| 32 | OpenScope 27                   | 73 | <b>L</b>              |
| 33 | PixelIntensity 27              |    |                       |
| 34 | PointKeyword 27                |    |                       |
| 35 | PointSeparator 27              |    |                       |
| 36 | PositiveInteger 27             |    |                       |
| 37 | RelativeNegative 27            |    |                       |
| 38 | RelativePositive 27            |    |                       |
| 39 | String 27                      |    |                       |
| 40 | SubjectAreaReferenceKeyword 27 |    |                       |
| 41 | SummaryKeyword 27              | 74 | <b>M</b>              |
| 42 | TextString 27                  |    |                       |
| 43 | Time 27                        | 75 | Mailbox 8             |
| 44 | TimeKeyword 27                 | 76 | Meta-                 |
| 45 | TrueValue 27                   | 77 | glossary entry 49     |
| 46 | VersionNumberKeyword 27        | 78 | Meta-attribute        |
| 47 | WidthKeyword 27                | 79 | enumerated            |
| 48 | token;PixelSeparator 27        | 80 | extension 16          |
| 49 | TransferContents 8, 33         | 81 | glossary entry 49     |
| 50 | Width 20, 33                   | 82 | instance              |
| 51 | XValue 23, 33                  |    |                       |

|    |                        |    |                          |
|----|------------------------|----|--------------------------|
| 1  | clause 19              | 49 | glossary entry 51        |
| 2  | value                  | 50 | Notational conventions 5 |
| 3  | clause 19-25           |    |                          |
| 4  | Meta-entity            | 51 | <b>O</b>                 |
| 5  | glossary entry 49      |    |                          |
| 6  | instance               |    |                          |
| 7  | clause 17-18           | 52 | ObjectClause 17          |
| 8  | Meta-language 5        |    |                          |
| 9  | Meta-layers            | 53 | <b>P</b>                 |
| 10 | glossary entry 49      |    |                          |
| 11 | Meta-meta-attribute    | 54 | Pipe 8                   |
| 12 | glossary entry 50      | 55 | Prerequisites 3          |
| 13 | instance               | 56 | Production Rule          |
| 14 | clause 14-15           | 57 | glossary entry 51        |
| 15 | Meta-meta-entity       | 58 | Production rule 5        |
| 16 | glossary entry 50      |    |                          |
| 17 | instance               |    |                          |
| 18 | clause 13-14           | 59 | <b>Q</b>                 |
| 19 | Meta-meta-model        |    |                          |
| 20 | glossary entry 50      | 60 | Quotes 4                 |
| 21 | Meta-meta-object       |    |                          |
| 22 | name 4                 | 61 | <b>R</b>                 |
| 23 | Meta-meta-relationship |    |                          |
| 24 | glossary entry 50      | 62 | Range 6                  |
| 25 | instance               | 63 | Relationship             |
| 26 | clause 15-16           | 64 | glossary entry 51        |
| 27 | Meta-model             | 65 | Rule                     |
| 28 | extension              | 66 | production 5             |
| 29 | clause 13              |    |                          |
| 30 | glossary entry 50      | 67 | <b>S</b>                 |
| 31 | section 11-16          |    |                          |
| 32 | Meta-object            | 68 | Shared memory 8          |
| 33 | glossary entry 50      | 69 | Square braces 5          |
| 34 | name 4                 | 70 | Subject Area             |
| 35 | Meta-relationship      | 71 | glossary entry 52        |
| 36 | glossary entry 50      | 72 | Subject area             |
| 37 | instance               | 73 | reference                |
| 38 | clause 18-19           | 74 | clause 12-13             |
| 39 | Model                  | 75 | Summary clause 9, 10-11  |
| 40 | glossary entry 51      | 76 | Syntactic element 5      |
| 41 | section 16-25          | 77 | Syntax                   |
| 42 | Model layers           | 78 | glossary entry 52        |
| 43 | glossary entry 51      | 79 | identifier 7, 8          |
| 44 | <b>N</b>               | 80 | version 7                |
|    |                        | 81 | version 7                |
| 45 | Name                   | 82 | SYNTAX.1                 |
| 46 | meta-meta-object 4     | 83 | glossary entry 52        |
| 47 | meta-object 4          |    |                          |
| 48 | Non-terminal symbol    |    |                          |

1 **T**

- 2 Terminal Symbol
- 3   glossary entry 52
- 4 Terminal symbol 6
- 5 Token separation rules 7
- 6 Transfer
- 7   contents 8-26
- 8   description 8
- 9   Envelope 8
- 10   glossary entry 52
- 11 Transfer Format
- 12   glossary entry 52
- 13 Transfer syntax
- 14   definition 8-27

15 **U**

16 **V**

- 17 Vertical bar 5

18 **W**

- 19 Whitespace 7
- 20 Working Meta-model
- 21   glossary entry 52

22 **X**

23 **Y**

24 **Z**

25

This page is intentionally blank

## **RELATED EIA STANDARDS**

1

*EIA/IS-106 CDIF - CASE Data Interchange Format - Overview*

2

*EIA/IS-107 CDIF - Framework for Modeling and Extensibility*

3

*EIA/IS-108 CDIF - Transfer Format - General Rules for Syntaxes and Encodings*

4

*EIA/IS-110 CDIF - Transfer Format - Encoding ENCODING.1*

5