5i’

FASCICLE X.7

Recommendations Z.301 to Z.341

MAN-MACHINE LANGUAGE (MML)

Blanc

MONTAGE: PAGE 2 = PAGE BLANCHE

SECTION 1

GENERAL PRINCIPLES

Recommendation Z.301

INTRODUCTION TO THE CCITT MAN-MACHINE LANGUAGE

1 Field of application

The man-machine language (CCITT MML) can be used to facilitate operation and maintenance functions of Stored Program Control SPC systems of different types. Depending upon national requirements, CCITT MML can also be used to facilitate installation and acceptance testing of such systems.

In many cases, SPC systems will be supported by auxiliary systems, e.g., in operation and maintenance centres and/or centres for other purposes such as sales, subscribers' complaints, etc., to carry out functions in cooperation with the SPC system. Different types of communication may be required for this cooperation. To clarify where the CCITT MML is intended to be used, a configuration is shown in Figure 1/Z.301 which illustrates the case of three separate systems. Local and remote man-machine terminals may be used. The configuration of systems in a network may vary, but this does not alter the principles governing the field of application of the MML.

The CCITT MML is intended to handle the functions required at the interface marked 1 while other methods may be required for the interface marked 2. Interface 2 is not considered. Since interface 1 is the interface of interest, it should be stressed that no assumptions are made concerning the physical location of any supporting software or whether, indeed, that software is entirely resident in any one place rather than distributed.

Although telephone signalling and switching has been considered the primary application area for the MML, these Recommendations accommodate the extension of the MML into other areas such as data switching, ISDN operations and maintenance, and software development environments.

In the Recommendations of this Part, the term *man* | is used in the sense of *user* , and the terms *machine* | and *system* | are used interchangeably.

2 Man-machine communication model

Man-machine communication, the means of exchanging information between users and systems, can be represented by a layered model in which each layer defines features that support such communication. In their entirety, these features offer users an appropriate man-machine interface. The model is shown in Figure 2/Z.301 where higher layers are based upon features offered by the lower layers. The man-machine interface, for any given system, represented by the highest layer of the model, is based on the repertoire of inputs, outputs, special actions and man-machine interaction mechanisms, including dialogue procedures made available by the layers below.

These features are, in turn, supported by the lower layers in which the semantics associated with each MML function (actions, objects, information entities and their interrelationships) and the MML syntax are defined. The lowest layer of the model is identified in the set of system functions to be controlled and in the capabilities available in the man-machine terminals connected to the system.

Figure 1/Z.301, (M), p.

Figure 2/Z.301 (comme tableau) [T1.301], p.

3 Organization of the MML Recommendations

The Recommendations on man-machine language are grouped in five sections:

1 General principles

2 Basic syntax and dialogue procedures

3 Extended MML for visual display terminals

4 Specification of the man-machine interface

5 Glossary of terms.

Section 1 | ives an introduction to man-machine communication by the CCITT MML and contains information of a general nature. *Section 2* deals with syntax and dialogue procedures for terminals where no advantage is taken or can be taken of enhanced input and output facilities which are usually available on visual display terminals (VDTs). *Section 3* describes capabilities of VDTs and kinds of dialogue elements suitable for conveying the syntax of any application, including the syntax specified in Section 1, which can be applied to the operation and maintenance of SPC systems. As terminal

technology advances and the theory of the man-machine interface evolves, greatly improved interfaces are possible. On the other hand, basic terminals will remain in use. Therefore this section provides a framework that accommodates interfaces possible on more sophisticated terminals and at the same time ensures that syntactic details presented at both sophisticated and basic terminals in a given application are consistent. *Section 4* identifies operation, maintenance, installation and acceptance testing functions to be controlled by the MML. A methodology is defined by which the semantics relating to MML functions may be generated and by which the inputs, outputs and special actions may be specified; specific Recommendations on Subscriber Administration, Routing Administration, Traffic Measurements Administration, and Network Management Administration are included. *Section 5* contains a summary of the terms used in Sections 1 to 4 together with short definitions to aid the reader seeking an explanation of a term.

4 Organization of Section 1

Section 1 consists of two Recommendations:

Z.301 Introduction to the CCITT man-machine language

Z.302 The meta-language for describing MML syntax and dialogue procedures.

Recommendation Z.302 | nables the reader to interpret the diagrams used to specify MML syntax and dialogue procedures in Sections 2 and 3.

5 Basis of MML

The MML contains features which are sufficient to ensure that all relevant functions for the operation, maintenance, installation and acceptance testing of SPC systems can be performed.

The basic attributes of the language are summarized in the following:

a) The MML provides a consistent interface which is easy to learn and easy to use by novices as well as by experts, making possible the input of commands and the interpretation of outputs in a way convenient to all users.

b) The MML is flexible, allowing system design to be optimized according to the tasks to be performed. It offers a variety of input/output features including direct input, menus and forms.

c) The MML is adaptable to different kinds of personnel and to different national languages and organizational requirements.

d) The MML is structured to allow graceful incorporation of new technology.

The MML should be sufficiently flexible to meet Administrations' requirements for the organization of their operation and maintenance staff and for the security of their SPC systems; it should not restrict their selection of terminal types. The MML covers the man-machine interface including those functions that are initiated by the system and those that are initiated by the user. It should be implemented in such a way that errors in commands or control actions will not cause the system to stop, unduly alter the system configuration or take up undue resources.

6 Input/output

As indicated in Figure 1/Z.301, the interface being recommended is that between the user and an I/O device or devices. These devices must at least be capable of handling the code of the characters of the CCITT International Alphabet No. 5 both for input and for visual textual output to the user. Input will normally be from a keyboard device, but for bulk input of data and/or commands, some temporary storage medium such as paper tape, cassette, disc, etc., could be used. For output, a variety of device types is possible, including paper tape punches, teletypewriters, line printers, visual display terminals, etc.

7 Extensibility and sub-setting

The MML has an open-ended structure such that the addition of any new function or requirement will have no influence on the existing ones.

The language structure is such that sub-sets can be created. Sub-setting may be for various purposes, e.g., staff sub-sets, in which selection is done to meet the needs of certain sections of staff; application sub-sets, in which selection is made for convenience of application, etc.

Recommendation Z.302

THE META-LANGUAGE FOR DESCRIBING MML SYNTAX AND DIALOGUE PROCEDURES

1 Introduction

Syntax diagrams are a method of defining language syntax non-terminal symbol boxes connected by flowlines. An annotation symbol is used to insert comments. The syntax of a language can be defined by a series of syntax diagrams, each diagram defining a particular non-terminal symbol. In the MML Recommendations, syntax diagrams are used to assist in specifying the syntax of the MML input, MML output and the user-system dialogue procedures. A path through a syntax diagram defines an MML input, an MML output or a man-machine dialogue structure.

The sequence of symbols in a path through syntax diagrams does not always imply a corresponding order in time or in place. The order in time is only significant in dialogue procedures for changes in the direction of the information flow, i.e. from input to output or from output to input. For output on printers it represents an order in place (from left to right and from top to bottom). However, for output on VDTs, the order in place only applies to positions within a screen window (see Recommendation Z.322).

The following describes the use of syntax diagrams and states a set of rules for their use.

2 Terminology

2.1 Terminal symbols are those characters or strings of characters which actually appear in the input or output. To avoid possible misunderstanding, format effectors are represented by a crossed mnemonic of the intended format effector.

The syntax diagrams used in MML are based on those used to describe the programming language PASCAL [1].

2.2 A non-terminal symbol does not immediately appear in MML input or MML output; it represents, within a syntax diagram, another syntax diagram by name. Hence it is an abbreviated symbol for a more complex construct (consisting of a set of terminal and/or non-terminal symbols) used in several places.

2.3 Annotation symbols (see § 3.7) are used to insert references to descriptive or explanatory notes. For example, they may be used to indicate mutually exclusive paths through a diagram.

3 Rules

3.1 Every symbol box (terminal or non-terminal) and consequently each diagram must have one, and one only, entry and one, and one only, exit flowline.

3.2 Each diagram must fit on a single page. Thus there is no off-page connector symbol.

3.3 Flowlines are always unidirectional. The preferred direction for flowlines which select alternatives is down. The preferred direction for flowlines which connect symbols is left-to-right. The preferred direction for flowlines which indicate repetitions (loops) is counterclockwise.

3.4 An arrowhead is required wherever any two flowlines come together, and wherever a flowline enters a symbol box. Additional arrowheads may be inserted wherever it is felt that this will improve the clarity of the diagram.

3.5 Terminal symbols are surrounded by boxes with rounded corners. The width of the box is proportional to the number of characters contained in the box. For short terminal symbols, the box may become a circle. Symbols representing system input are surrounded by a single solid line and those representing system output by a double solid line:

— for terminal input symbols see Figure 1a)/Z.302 and Figure 1b)/Z.302,

— for terminal output symbols see Figure 1c)/Z.302 and Figure 1d)/Z.302.

3.6 Non-terminal symbols are surrounded by rectangular boxes. The name of the non-terminal symbol must be written in lower case characters. Every non-terminal symbol must have an associated syntax diagram except where the symbol is annotated ‘‘Not further expanded in diagram form’’. The non-terminal symbol used to name a particular syntax diagram must appear at the upper left corner of the diagram. Symbols representing system input are surrounded by a single solid line, those representing system output by a double solid line and symbols representing a combination of input and output by an outer solid and an inner dashed line:

a) for the non-terminal input symbol see Figure 1e)/Z.302,

b) for the non-terminal output symbol see Figure 1f)/Z.302,

c) for the non-terminal input/output symbol used in dialogue procedures see Figure 1g)/Z.302.

3.7 An annotation is denoted by the following symbol:

Figure 34050, (M), p.

where n is a number referring to a descriptive or explanatory note. The text of the note must be located at the foot of the diagram.

Figure 1/Z.302, (M), p.

Reference

[1] JENSEN (K.), WIRTH (N.): PASCAL, User Manual and Report, *Springer Verlag* , New York, 1975.

MONTAGE: PAGE 8 = PAGE BLANCHE

SECTION 2

BASIC SYNTAX AND DIALOGUE PROCEDURES

Recommendation Z.311

INTRODUCTION TO SYNTAX AND DIALOGUE PROCEDURES

1 Scope of the Section

Section 2 deals with syntax and dialogue procedures for terminals where no advantage is taken or can be taken of enhanced input and output facilities which are usually available on VDTs. This basic MML is therefore compatible with the use of VDTs used in the manner of teletypewriters, hard copy printers, etc., at the man-machine interface.

2 Organization of Section 2

Section 2 consists of the following Recommendations:

Z.311 Introduction to syntax and dialogue procedures

Z.312 Basic format layout

Z.313 (spare)

Z.314 The character set and basic elements

Z.315 Input (command) language syntax specification

Z.316 Output language syntax specification

Z.317 Man-machine dialogue procedures.

Recommendation Z.317 | escribes the operational procedures for a dialogue between user and system. For aspects of input syntax, reference is made to *Recommendation Z.315* and for aspects of output syntax, reference is made to *Recommendation Z.316* . Recommendation Z.316 also deals with output outside dialogue. The specification of basic syntax elements for input and output, together with the characters to be used, is contained in *Recommendation Z.314* . Formats to be used on teletypewriters and hard copy printers are described in *Recommendation Z.312* .

Recommendation Z.312

BASIC FORMAT LAYOUT

1 General

To facilitate filing and retrieval of recorded information in MML, it is recommended that this information should be recorded on pages with an identification header at the top of each page. The top and bottom lines of a page should not be used but should be left blank.

It is further recommended that the layout for printing information in MML should be based on a maximum of 72 characters per line and 66 lines per page, as this format can be accommodated on both the A4 and the 11-inch standard size paper and can be printed by standard teletypewriters.

Where a number of characters per line in excess of 72 is required, a second format is recommended. This accommodates 120 characters per line and would be used, for example, on line printers.

In order to save paper or where paging to facilitate filing of output is not required, paging may be suppressed by suppressing the generation of all superfluous line feeds.

To distinguish between the formats recommended, they are further indicated as format F1 for the paper sizes A4 and A5L and format F2 for the paper size A4L. In the recommended formats specified below, International Standard ISO/2784 [1] has been taken into account.

2 Recommended formats for presenting information in MML

2.1 *Format F1*

According to this format, which is based on the A4 and the 11-inch standard size paper, the maximum number of characters per line is 72. The number of lines per page may be 66, using the full 11-inch and A4 paper sizes or 33, using half the paper size (5.5 inch or A5L).

Information presented in this format can also be displayed on most of the VDTs available on the market. However, the number of lines which can be displayed at the same time on these devices is, in general, not more than 20 to 25 lines.

2.2 *Format F2*

This format allows a maximum of 120 characters printed on a line and has 66 lines per page. It can be accommodated on paper having a width equal to the A4L standard.

Reference

[1] International Organization for Standardization: *Continuous forms used for information processing. Sizes and Sprocket feed holes* , ISO 2784-1974.

Recommendation Z.314

THE CHARACTER SET AND BASIC ELEMENTS

1 General

The character set and the basic elements used in the syntax are essential components of MML inputs, MML outputs and the man-machine dialogue procedures.

2 The character set

The character set to be used for the CCITT MML is a sub-set of the CCITT International Alphabet No. 5 which has been established jointly by the CCITT and the International Organization for Standardization.

To allow for possible implementation of the CCITT MML using national languages, the sub-set is taken from the basic code table given in Recommendation T.50 [1] The code positions reserved in this table for national use are not contained in the basic character set of the CCITT MML, but may be used in these national implementations.

According to Recommendation T.50 [1] transmission control characters and information separators are intended to control or to facilitate transmission of information over telecommunication networks. Hence these control characters are not used in the MML. This will avoid interference with data transmission procedures when information in the MML is transmitted via a data transmission network.

It is furthermore recommended when information is printed or displayed that devices are used which print or display different graphic symbols for the digit zero and the capital letter O.

The characters selected for use in the CCITT MML are given in Table 1/Z.314.

Tableau 1/Z.314, (M), (comme figure), p.

General remarks — The characters proper to the open positions are considered as outside the MML. They are implementation dependent and, together with the characters named in the table but not included in the MML, may be used in accordance with the rules given in Recommendation T.50 [1]. The position of a character in the table can be indicated by its column and row number, e.g., Pos. 3/1 gives the position of the digit 1 in the table. The table gives also the binary codes allocated to the table positions according to Recommendation T.50 [1]. The bits are identified by $b_{7, 6}$, bit, and b_1 is the lowest order, or least significant bit.

3 Summary of use of characters

The use of characters in the character set, except for letters, digits, and characters used solely as graphic characters and format effectors, is described in Table 2/Z.314. CCITT International Alphabet No. 5 code is indicated by position number (see Table 1/Z.314).

3.1 *Letter*

A letter is one of the characters listed in Table 1/Z.314, columns 4, 5, 6 and 7. However, positions 5/15 and 7/15 are excluded. The characters reserved for national use may be used as letters or as graphic characters.

3.2 *Digit*

A digit is one of the characters listed in Table 1/Z.314, column 3, positions 0 to 9.

3.3 *Graphic characters*

Graphic characters are a collection of characters one or more of which may be used to improve readability. Graphic characters which have other syntactic uses are listed in Table 2/Z.314. The $ (position 2/4 in Table 1/Z.314) is the only character used solely as a graphic character.

3.4 *Format effector*

The format effectors used in MML are the characters FE1 to FE5 and SP (space) as defined in Table 1/Z.314. The character BACK SPACE (FE0 in Recommendation T.50 [1]) is not regarded as a format effector in the MML.

4 Basic elements used in the syntax

Syntax diagrams of the basic elements used in the syntax are given in § 5 in sub-paragraphs with numbers corresponding to those in § 4.

4.1 *Identifier*

An identifier is a string of one or more characters which begins with a letter and, if applicable, subsequently contains only digits and/or letters e.g., U, UPDATE, UPD8.

4.2 *Symbolic name*

A symbolic name is a string of one or more characters used for the purpose of representing an entity which cannot be adequately represented by numerals or identifiers. The string contains at least one letter and/or at least one of the graphic characters + (plus sign), ## (number sign), % (percent sign) plus any number of digits, including none. The characters may appear in any order. For example a time duration of 6 hours may be represented by the symbolic name 06H, a 10 percent threshold value by 10%, a signalling system such as CCITT No. 6 by SS##6.

4.3 *Decimal numeral*

A decimal numeral is a character combination, consisting of a digit or digits and an optional . (full stop), preceded by the special character combination D’ (D apostrophe). If the numeric default base for an information unit (see Recommendation Z.315) is decimal, then the D’ is optional.

4.4 *Nondecimal numerals*

A nondecimal numeral is a character combination preceded by a special character combination indicating the type of numeral.

4.4.1 H’ (H apostrophe) is used to indicate a hexadecimal numeral, the following characters thus being any of: digits 0 to 9 or letters A, B, C, D, E, F.

4.4.2 O’ (letter O apostrophe) is used to indicate an octal numeral, the following characters thus being any of: digits 0, 1, 2, 3, 4, 5, 6, 7.

4.4.3 B’ (B apostrophe) is used to indicate a binary numeral, the following characters thus being digit(s) 0 and/or digit(s) 1.

4.4.4 K’ (K apostrophe) is used to indicate a keyed numeral, the following characters thus being any of: digits 0-9, * (asterisk), ## (number sign), or letters A, B, C, D.

4.4.5 When the default base for an information unit (see Recommendation Z.315) is one of the nondecimal numerals e.g., hexadecimal, the corresponding character combination, i.e. H’ in this example, is optional.

H.T. [1T1.314]

TABLE 2/Z.314

Summary of use of characters

CCITT International Alphabet No. 5 (Recommendation T.50) [1]			Man-machine language use
Character or character string	Position number	Name	
CAN	1/8	Cancel	Used as a deletion character.
!	2/1	exclamation mark	An indicator used in dialogue procedures (continuation character in input language).
’’	2/2	quotation mark	A text string delimiter and a graphic character.
##	2/3	number sign	A character which may be used in symbolic names and keyed numerals and as a graphic character.
%	2/5	percent sign	A character which may be used in symbolic names and as a graphic character.
&	2/6	ampersand	A separator for information grouping and a graphic character.
’	2/7	apostrophe	A separator used when indication of type of numeral is required. The character is placed between a letter indicating the type of numeral and the numeral itself. Also used as a graphic character.
(	2/8	left parenthesis	Used for delimiting arithmetical expressions, and conditions in a selection argument. Also a graphic character.
)	2/9	right parenthesis	Used for delimiting arithmetical expressions, and conditions in a selection argument. Also a graphic character.
*	2/10	asterisk	Used for keyed numerals, as an arithmetic operator and as a graphic character.
+	2/11	plus sign	A character which may be used in symbolic names, as an arithmetic operator and as a graphical character.
++	2/11 2/11	plus sign, plus sign	A separator used for separating the increment from a group of consecutive parameter values.
,	2/12	comma	A separator used to separate parameters (if more than one) within a block of parameters.
—	2/13	hyphen	A separator used to separate information units or to separate identifiers and/or index numbers within compound parameter names. Also used as an arithmetic operator and as a graphic character.
			A separator used for subdividing a number into an integer part and a fraction part and as a graphic character.
/	2/15	solidus	

Used as an arithmetic operator and as graphic character. }			
: A separator used to separate blocks of parameters from each other and from the command code, an indicator used in the parameter block request indication and a separator used in output. }	3/10	colon	{
; An indicator used to terminate a command (execution character). }	3/11	semicolon	{
< An indicator used as a ready indicator for the system to output that it is ready to receive information, and a relational operator used in a selection argument. }	3/12	less than sign	{
A separator used to separate the parameter name and the parameter value of a parameter. Also a relational operator used in a selection argument. }	3/13	equal sign	{

Tableau 2/Z.314 [1T1.314], p.6

H.T. [2T1.314]

TABLE 2/Z.314 *(cont.)*

{ CCITT International Alphabet No. 5 (Recommendation T.50) [1] } Character or character string	Man-machine language use Position number	Name	
> A separator to terminate the destination identifier and a relational operator used in a selection argument. }	3/14	greater than sign	{
<= A relational operator used in a selection argument. }	3/12 3/13	less than or equal sign	{
<> less than or greater than sign } A relational operator used in a selection argument. }	3/12 3/14 {	{	
>= A relational operator used in a selection argument. }	3/14 3/13	greater than or equal sign	{
? An indicator used for prompting or help. }	3/15	question mark	{
&& Separator used for information grouping. }	2/6 2/6	ampersand, ampersand	{
&— Separator used for information grouping. }	2/6 2/13	ampersand, hyphen	{
&&— Separator used for information grouping. }	2/6 2/6 2/13	ampersand, ampersand, hyphen	{
/*	2/15 2/10	solidus, asterisk	Used to open a comment.
*/	2/10 2/15	asterisk, solidus	Used to close a comment.

Tableau 2/Z.314 [2T1.314], p.7

4.5 *Text string*

A text string allows input of a literal text, including any delimiters which would have syntactical meanings when input outside a text string. It consists of a string of zero or more characters enclosed by a ‘‘ (quotation mark) at the beginning and end. The string may contain any of the characters belonging to the character set defined in § 2 except correction characters (see Recommendation Z.315). If ‘‘ (quotation mark) is required within a string, it is represented by ‘‘’’ (double quotation marks). Text strings in output need not be delimited by quotation marks. Text strings for use in extended MML (Recommendations Z.321-Z.323) need not be delimited by quotation marks.

4.6 *Arithmetical expression*

An arithmetical expression is a combination of certain basic elements and arithmetic operators delimited by parentheses.

4.7 *Ancillary facilities*

Additional facilities have been provided when using MML commands as follows.

4.7.1 *Comment facility*

A comment is defined as a character string enclosed between the separators /* (solidus asterisk) and */ (asterisk solidus), where the character string may contain any characters except the sequence */ (asterisk solidus) and correction characters (see Recommendation Z.315). The character string, including the delimiters, has neither MML syntactical nor semantical significance. However, if it occurs in a text string, it is regarded as being part of the text string. A comment may be inserted only before and/or after a separator, indicator, arithmetic delimiter [((left parenthesis),) (right parenthesis)], arithmetic operator [+ (plus sign), - (hyphen), / (solidus), * (asterisk)], identifier or information unit [excluding the ' (apostrophe) between the type of numeral and the numeral itself and the . (full stop) between the integer and fractional part of a number].

4.7.2 *Escape syntax*

In some systems it is not possible to use characters with syntactical meaning [e.g., ; (semi colon), - (hyphen)] or correction characters as data. In such systems an escape indication may be used in order to introduce the following character as data.

A specific escape indication is not proposed due to the diverse nature of terminals.

No syntax diagram is given.

4.7.3 *Format effector*

A format effector (see § 3.4) is used to format input and output in a suitable manner. Format effectors have no significance in a command and may appear anywhere in input.

No syntax diagram is given.

4.8 *Separator*

A separator is a character or a string of characters used to separate items of information in the input or output and it may, in addition, have structural, semantic or other significance.

No syntax diagram is given.

4.9 *Indicator*

An indicator is a character used to indicate a state or make a request.

No syntax diagram is given.

5 Definition of the basic elements used in the syntax in diagrams

All these elements may be used in both input and output but for simplicity only the input elements are shown in the diagrams. The output elements are identical to the input elements.

5.1 *Identifier*

Figure CCITT-26400, (M), p.

5.2 *Symbolic name*

Figure CCITT-21391, (M), p.

5.3 *Index number*

Figure T1000690-87, (N), p.

5.4 *Decimal numeral*

Figure CCITT-29381, (M), p.

5.5 *Nondecimal numerals*

5.5.1 *Hexadecimal numeral*

Figure CCITT-26421, (M), p.

5.5.2 *Octal numeral*

Figure CCITT-29841, (M), p.

5.5.3 *Binary numeral*

Figure CCITT-29851, (M), p.

5.5.4 *Keyed numeral*

Figure CCITT-26431, (M), p.

5.6 *Text string*

Figure T1000701-88, (N), p.

5.7 *Arithmetical expression*

Figure CCITT-29401, (M), p.

5.8 *Ancillary facilities*

5.8.1 *Comment*

Figure CCITT-34210, (M), p.

Reference

[1] CCITT Recommendation *International Alphabet No. 5* , Rec. T.50.

Recommendation Z.315

INPUT (COMMAND) LANGUAGE SYNTAX SPECIFICATION

1 General

The following text describes the elements of the input language. Syntax diagrams of the input language are given in § 4 in sub-paragraphs with numbers corresponding to those in § 2. Where input elements are used in output, reference to these elements is made in the output language description Recommendation Z.316. Procedural aspects are taken into account in Recommendation Z.317. It should be noted that certain areas of the syntax allow options to be taken which could result in a syntax clash. The taking of such options must be chosen to suit the particular system involved.

2 Command structure

2.1 *Command*

A command begins with the command code, which defines the function to be performed by the system. If further information is required a command code can be followed by a parameter part from which it is separated by a : (colon). The parameter part consists of one or more blocks of parameters (see §§ 2.3 and 2.9.1). A command is always completed by an execution character (see Recommendation Z.317).

2.2 *Command code*

The command code is composed of up to three identifiers separated by a - (hyphen) (e.g., functional area - object type - action). Where command codes are in the form of single mnemonic abbreviations, it is recommended that they consist of the same number of characters.

2.3 *Block of parameters*

A block of parameters contains information necessary to execute the function specified in the command code. The information in a block of parameters is expressed in the form of a number of parameters specific to the command. If more than one parameter is included, they shall be separated by a , (comma). All parameters in any one block shall be of the same kind i.e. either position defined parameters or parameter name defined parameters.

2.4 *Parameters*

A parameter identifies and contains a piece of information and may be either position defined or parameter name defined. Non-relevant parameters may be omitted in accordance with §§ 2.4.1 and 2.4.2.

2.4.1 *Position defined parameter*

A position defined parameter consists of a parameter value which may be preceded by a parameter name from which it is separated by an = (equal sign). Parameters must be given in a predetermined order within the parameter block. Where a parameter value is not to be given, the parameter is omitted leaving the appropriate separator or the appropriate indicator used to terminate a command. This indicates the parameter's position in the block of parameters. Parameter omission can imply that the default value is meant. The default value can also be indicated by giving a parameter value assigned for this purpose.

2.4.2 *Parameter name defined parameter*

A parameter name defined parameter consists of a parameter name followed by a parameter value from which it is separated by an = (equal sign). These parameters may be given in an arbitrary order within the parameter block. Where a parameter value is not to be given, the parameter name and separator = (equal sign) and the separator , (comma) following the parameter are also omitted. This omission can imply that the default value is meant. The default value can also be indicated by giving a parameter value assigned for this purpose. Where a parameter value implies the parameter name the latter and the separator = (equal sign) can be omitted.

2.5 *Parameter name*

A parameter name unambiguously indicates the kind and structure of the subsequent parameter value and thereby defines the parameter value and how it shall be interpreted. It is an identifier. It is either a simple parameter name or a compound parameter name. The simple parameter name indicates a single parameter value and a compound name indicates a parameter value in a list or table of similar parameter types.

2.5.1 *Simple parameter name*

A simple parameter name consists of one identifier.

2.5.2 *Compound parameter name*

A compound parameter name consists of one or more identifiers and/or index number all separated by a separator - (hyphen).

2.5.2.1 *Index number*

An index number is one or more digits.

2.6 *Parameter value*

A parameter value contains the information required to specify the appropriate object(s) or value(s) and consists of one or more information units. In the case where no information grouping (see § 2.9) is applied a parameter value reduces to a parameter argument. Refer to § 2.10 for data base query aspects.

2.7 *Parameter argument*

A parameter argument contains the information required to specify the appropriate object or value. It is the form of a parameter value when no information grouping is applied (see § 2.9). A parameter argument consists of a simple or a compound parameter argument.

2.7.1 *Simple parameter argument*

A simple parameter argument consists of one information unit.

2.7.2 *Compound parameter argument*

A compound parameter argument consists of two or more information units separated by a - (hyphen).

2.8 *Information unit*

An information unit constitutes the smallest unit of information in the language from a syntactical point of view. An information unit can be a numeral, an identifier, a symbolic name, a text string or an arithmetical expression. A numeral always has a default base (e.g., hexadecimal) which can be overwritten, if required, by introducing the desired base as specified in Recommendation Z.314. However, the default base for a keyed numeral cannot be overwritten by another base.

2.9 *Information grouping*

Information grouping is used to improve the speed and ease of input activities. It is performed by grouping sets of information of the same type within the same command.

2.9.1 *Grouping of blocks of parameters*

If several blocks of parameters are to be included in one command they shall be separated by a : (colon).

2.9.2 *Grouping of parameter arguments*

Input of more than one parameter argument within one parameter of a command can be achieved by grouping parameter arguments.

2.9.2.1 *Grouping of simple parameter arguments*

It is possible to indicate several simple parameter arguments within the same parameter value separated by an & (ampersand). *Example 1:* | 5&9 means the simple parameter arguments 5 and 9.

The interpretation of the separators && (ampersand ampersand) and &&- (ampersand ampersand hyphen) is not exclusive. Other interpretations exist. One alternative would imply that no specific increment is inherent in the syntax. That is, the relationship of the values between the upper and lower values in the sequence is a semantic relationship dependent upon the function for which the sequence is being specified.

In the case of a sequence of consecutive (implicit increment value = 1) simple parameter arguments, it is possible to indicate the arguments by writing the lower and upper simple parameter arguments separated by an && (ampersand ampersand) 6, 7, 8 and 9.

An explicit increment value can be specified following the upper parameter argument separated by ++ (plus plus). *Example 3:* | 5&&9++2 means the simple parameter arguments 5, 7 and 9.

Other combinations of the above possibilities may also be used when required. *Example 4:* | 5&&7&9 means the simple parameter arguments 5, 6, 7 and 9. *Example 5:* | 5&&9++2&10 means the simple parameter arguments 5, 7, 9 and 10.

2.9.2.2 *Grouping of compound parameter arguments*

It is possible to indicate several compound parameter arguments within the same parameter value separated by an & (ampersand). *Example 1:* | 5-1&6-3 means the two compound parameter arguments 5-1 and 6-3.

If a group of compound parameter arguments differs only in the last information unit, the first compound parameter argument is completely specified, whereas all subsequent compound parameter arguments are represented only by their last information units, separated by an &- (ampersand hyphen). *Example 2:* | 7-1&-3 means the two compound parameter arguments 7-1 and 7-3.

If a group of compound parameter arguments differs only in the last information unit and constitutes a consecutive sequence (implicit increment value = 1), it is possible to indicate the arguments by writing the lower and upper information units separated by an &&- (ampersand ampersand hyphen) parameter arguments 7-1, 7-2 and 7-3. *Example 4:* | 7-1&-3&&-5 means the four compound parameter arguments 7-1, 7-3, 7-4 and 7-5.

An explicit increment value can be specified following the upper information unit separated by ++ (plus plus).

Any combination of the above possibilities may also be applied when required. *Example 5:* | 5-1&&-3&8-2&-5&-6 means the six compound parameter arguments 5-1, 5-2, 5-3, 8-2, 8-5 and 8-6. *Example 6:* | 5-1&&-7++2&8-1&-3 means the six compound parameter arguments 5-1, 5-3, 5-5, 5-7, 8-1 and 8-3.

2.10 *Data base queries*

Data base queries are expressed in terms of projection and selection information. Projection information can be represented by a parameter. Its name identifies the projection function. Its group of parameter argument(s) identifies the appropriate field(s) of the data records to be displayed. Selection information can be represented by a parameter where the name identifies the selection function and the value identifies a (group of) selection argument(s). A selection argument comprises one or more conditions that should all be satisfied. A condition is specified by an identifier and a (group of) parameter argument(s) separated by a relational operator. The identifier specifies the name of the field and of the record to be selected. Omission of the selection information implies that the query is not conditional.

The names ‘‘projection’’ and ‘‘selection’’ are chosen for the example only. Other names such as ‘‘select’’ and ‘‘where’’ may apply.

Examples:

query-dbx: projection = field a,

selection = (field c = 0);

This command requests the records that satisfy the selection criterion field c = 0 of data set x; however, only field a of the selected records needs to be displayed.

query-dbx: projection = field a & field b,

selection = (field b > 5, field c = 1);

This command requests the records that satisfy both the selection criteria field b > 5 and field c = 1 of data set y. The resulting display need only show the fields a and b of the selected records.

query-dbz: projection = field a & field b & field d,

selection = (field d < = 7, field e = 0) & (field b = P);

This command requests from data set z the records that satisfy both the criteria field d< = 7 and field e = 0. It also requests the records that satisfy the criterion field b = P. The display of all selected records need only show fields a, b, and d.

Warning

The use of characters , (comma) and & (ampersand) in CCITT-MML corresponds to the operators AND and OR in predicate logic. A general assumption is made that predicate logic is not used by normal operating personnel. Confusion can also be avoided by realizing the functions of the various separation characters in CCITT-MML. The comma is used as a separator of parameters within a block, where all parameters together play a role in executing the command. The ampersand serves as a separator in information grouping, and is used to input one command ‘‘value1&value2’’, as an alternative to inputting two commands, one for ‘‘value1’’, and one for ‘‘value2’’.

Restriction

To avoid meaningless expressions, the parameter argument if used in combination with a non-symmetrical relational operator in syntax diagram 4.10.1.1 (condition) should be restricted to numerals. However, identifiers and symbolic names are allowed if they represent members of an ordered set.

3 Corrections and delete command

Corrections can be made by the deletion and resubmission of input.

Specific characters are not proposed because of the diverse nature of Input/Output terminal devices available.

3.1 *Delete last character*

The facility may be used to delete successive input characters back to the last system output (see § 3.2).

3.2 *Delete to last system output*

This facility deletes all input characters after the last system output, being either the ready indication or prompting output (see Recommendation Z.317).

3.3 *Delete command*

The delete command request is conveyed by the CAN character (cancel). The use of this character causes the system to respond with an acknowledgement that presents input after the last command executed is cancelled. The system should respond with a new ready indication to indicate that it is waiting for a new command code (see Recommendation Z.317).

4 Definition of the input (command) language structure in syntax diagrams

4.1 *Command*

Figure CCITT-21342, (M), p.

4.2 *Command code*

Figure CCITT-29351, (M), p.

4.3 *Block of parameters*

Figure CCITT-21352, (M), p.

4.4 *Parameters*

4.4.1 *Position defined parameter*

Figure CCITT-29360, (M), p.

4.4.2 *Parameter name defined parameter*

Figure CCITT-29370, (M), p.

4.5 *Parameter name*

Figure T1000710-87, (N), p.

4.5.1 *Simple parameter name*

Figure CCITT-26370, (M), p.

4.5.2 *Compound parameter name*

Figure T1000720-87, (N), p.

4.6 *Parameter value*

Figure T1000730-87, (N), p.

4.7 *Parameter argument*

Figure CCITT-34170, (M), p.

4.7.1 *Simple parameter argument*

Figure CCITT-26370, (M), p.

4.7.2 *Compound parameter argument*

Figure CCITT-34200, (M), p.

4.8 *Information unit*

Figure CCITT-26390, (M), p.

4.9 *Information grouping*

4.9.1 *Group of blocks of parameters*

See syntax diagram § 4.1.

4.9.2 *Group of parameter arguments*

Figure CCITT-34170, (M), p.

4.9.2.1 *Group of simple parameter arguments*

Figure CCITT-69470, (M), p.

4.9.2.2 *Group of compound parameter arguments*

Figure CCITT-69480, (M), p.

4.10 *Data base queries*

4.10.1 *Selection argument*

Figure T1000740-87, (N), p.

4.10.1.1 *Condition*

Figure T1000750-87, (N), p.

4.10.1.2 *Symmetrical relational operator*

Figure T1000760-87, (N), p.

4.10.1.3 *Non-symmetrical relational operator*

Figure T1000770-87, (N), p.

4.10.2 *Group of selection arguments*

Figure T1000780-87, (N), p.

Recommendation Z.316

OUTPUT LANGUAGE SYNTAX SPECIFICATION

1 General

Syntax diagrams of the output language are given in § 3 in sub-paragraphs having numbers corresponding to those in § 2. Where input elements are used in output, a reference is made to the input language description Recommendation Z.315. Procedural aspects utilizing output other than output outside dialogue are taken into account in Recommendation Z.317.

2 Output structure

2.1 *Output outside dialogue*

The output described is output outside dialogue. This output is either a spontaneous output indicating a certain event, e.g., an alarm situation, or it is a delayed response to an interactive operating sequence (see Recommendation Z.317). An example of such a delayed response is a traffic measurement result.

2.2 *Header*

The header is given in output outside dialogue. It is also used in the dialogue procedure (see Recommendation Z.317). The main purpose of the header is to mark the output outside dialogue or the record of the dialogue for identification and information. The header can also be used for special purposes for an operation and maintenance centre. Recommended contents are information related to source identification, date and time. More information not related to the input or output function can be added to the header as additional header information.

The header is introduced by format effectors and/or graphic characters selected from a layout option.

2.2.1 *Layout option*

A layout option is a combination of format effectors and graphic characters used to bound elements of the output in a clear and readable form.

2.2.1.1 *Graphic characters*

Graphic characters are used to improve readability of output.

2.2.1.2 *Format effector*

A format effector is used to format output in a suitable manner. Certain format effectors are specifically incorporated in the output definition given in § 3, but where the format effector element is shown any of the format effectors specified for MML can be used. No syntax diagram is shown.

2.2.2 *Source identifier*

A source identifier indicates the physical area in which an output was generated.

2.2.3 *Calendar date*

The output of the date in the header is based on the International Standard (ISO 2014) [1] for the writing of calendar dates in all-numeric form. The calendar date shall be written in the following order: year, month, day. The calendar date shall consist of a two decimal digit or four decimal digit year, a two decimal digit month, and a two decimal digit day of the month. The allowable characters between year and month and between month and day are hyphen or space.

Examples:

The 4th October 1979 shall be written in one of the following ways:

a) 19791004;

b) 1979-10-04;

c) 1979 10 04;

d) 791004;

e) 79-10-04;

f) 79 10 04.

The calendar date in input should preferably have a layout similar to that in output.

2.2.4 *Time of day*

The output of the time in the header is based on the International Standard (ISO 3307) [2]. However, in MML the output of a decimal fraction of hours, minutes, or seconds is not utilized in the header.

Time representations are based upon the 24-hour timekeeping system. The sequencing of time elements shall be from high order to low order (left to right): hours, minutes, seconds. The hour shall be represented by a two-digit decimal number ranging from 00 up to and including 23. The minute shall be represented by a two-digit decimal number ranging from 00 up to and including 59. The second shall be represented by a two digit decimal number ranging from 00 up to and including 59.

Examples:

Hours, minutes	1225	or	12:25
Hours, minutes, seconds	122501	or	12:25:01

2.2.5 *Additional header information*

Additional header information is general information which has no relation to the function of the output, e.g.:

— sequence number,

— processor number,

— output device,

— day of the week.

2.3 *Alarm statement*

The alarm statement may give information of a general class such as the degree of alarm or the source of alarm.

2.3.1 *Variable text*

Variable text is a set of information units which contains information unique to the event which caused the output.

2.4 *Additional information*

Additional information is general information related to the output, e.g.:

— type of output e.g., maintenance, statistics. This is not the same as identification of output, (see § 2.6),

— output recipient identification.

2.5 *Command reference*

A command reference supplies a command sequence number when needed in output outside dialogue as a reference to a previous input. In addition to the command sequence number it may also include clarifying text. It also may appear in dialogue procedures (see Recommendation Z.317).

2.5.1 *Clarifying text*

Clarifying text is a set of information units used to make the purpose and contents of the output more clear to the reader. Several clarifying texts could appear in an output.

2.6 *Identification of output*

Identification of output provides a unique identity for an output in a system's repertoire of outputs. Therefore, it could be used as a reference to the explanation of the output in a manual.

2.7 *Text block*

A text block is any combination of clarifying texts, variable texts, parameter name defined parameters and/or tables which gives information wherever it is needed or requested. For VDT applications this may be a displayed form.

2.8 *Table*

A table is an ordered presentation of interrelated information.

Clarifying text within a table can be used as labels to each column contained within the table. Where a table name or additional information associated with the table is required the clarifying text appearing at the beginning of the table in the syntax diagram of § 3.8 could be used.

When parameter name defined parameters are used to label columns each parameter should be complete, i.e. contain a parameter value (see Recommendation Z.315).

2.8.1 *New line*

New line is a character combination necessary to reset an output device to the beginning of a new line. It is recognized that the character combination is device dependent but can contain the characters CR (carriage return) and LF (line feed). No syntax diagram is shown.

2.9 *End of output*

An end of output is an indication that an output is finished.

2.10 *Comments in output*

The purpose of a comment in output is as for clarifying text (see § 2.5.1) with the exception that the syntax is as for comment in input so that it may be discarded during a subsequent re-input. No syntax diagram is shown.

3 Definition of the output language syntax in diagrams

3.1 *Output outside dialogue*

Figure CCITT-29412, (M), p.

3.2 *Header*

Figure CCITT-26312, (M), p.

3.2.1 *Layout option*

Figure CCITT-29471, (M), p.

3.2.1.1 *Graphic character*

Figure CCITT-29501, (M), p.

3.2.2 *Source identifier*

Figure CCITT-34220, (M), p.

3.2.3 *Calendar date*

Figure CCITT-69490, (M), p.

3.2.4 *Time of day*

Figure CCITT-69500, (M), p.

3.2.5 *Additional header information*

Figure CCITT-34250, (M), p.

3.3 *Alarm statement*

Figure CCITT-34261, (M), p.

3.3.1 *Variable text*

Figure CCITT-34270, (M), p.

3.4 *Additional information*

Figure CCITT-34250, (M), p.

3.5 *Command reference*

Figure CCITT-34281, (M), p.

3.5.1 *Clarifying text*

Figure CCITT-34270, (M), p.

3.6 *Identification of output*

Figure CCITT-34291, (M), p.

3.7 *Text block*

Figure T1000790-87, (N), p.

3.8 *Table*

Figure CCITT-19111, (M), p.

3.9 *End of output*

Figure CCITT-19044, (M), p.

References

[1] *Writing of Calendar Dates in All-Numeric Form* , ISO Standard 2014-1976.

[2] *Information Interchange — Representation of Time of the Day* , ISO Standard 3307-1975.

Recommendation Z.317

MAN-MACHINE DIALOGUE PROCEDURES

1 General

Man-machine communication comprises two types of information interchange, namely *dialogue* and *output outside dialogue* ; they occur sequentially and in no particular order. Output outside dialogue is fully defined in Recommendation Z.316.

Dialogue is that part of man-machine communication initiated and, normally, terminated by the user. It is accomplished by means of the dialogue procedures described in this Recommendation. In the text, the terms ‘‘dialogue’’ and ‘‘dialogue procedure’’ are used interchangeably.

The text in § 2 describes the dialogue procedure, the syntax diagrams of which are given in § 3 in sub-divisions having numbers corresponding to those used in § 2.

A systematic analysis of possible errors made by users is not considered. Diagrams mainly refer to correctly given commands and only obvious error situations are considered. It is recognized that the diagrams are not exhaustive and some of them might be modified when error recovery procedures have been completely considered.

2 Definition of the dialogue procedure

2.1 *Overview of the* | fBdialogue procedure

A dialogue is opened by a procedure prologue. The procedure prologue contains the various preparations which must be performed before commands can be initiated. It may include a header from the system. Following the procedure prologue a destination prologue can precede one or more interactive operating sequences. The dialogue can be terminated by a procedure epilogue.

2.2 procedure prologue

The procedure prologue may consist of three parts given in the following order:

— the request, which is an action to activate the man-machine terminal and the system;

— the identification of the user. The identification of the user is optional. Identification may be bypassed under special conditions, for example system initialization. In situations where no identification procedure is used, then it must be possible to allow access only for certain periods per day, e.g., office hours;

— a header, which is given from the system and contains the exchange identification, information relating to date and time, etc. Headers can be optional for a system or within a system for certain terminals.

The procedure prologue is intended to be executed only once at the beginning of a dialogue. The procedure prologue is followed by a ready indication inviting a destination prologue or an interactive operating sequence.

The request, the identification of the user and the header are defined in the following paragraphs.

2.2.1 request

The request is a manual action to activate the terminal and the system or to cause an interrupt. The composition of the request is highly dependent on the type of terminal and implementation.

The request can consist of keying the break key or actuating a control switch, power on, etc. and/or keying a sequence of characters on the keyboard.

2.2.2 identification procedure

The identification procedure is used to identify the user to the system. The identification procedure may involve the use of identity cards which provide secure access to the system.

After a user has been identified to the system, different authorization levels may be applied that restrict access to groups of commands depending on security or functional classification.

The identification procedure (see figure 3.2.2/Z.317) is flexible, with many options, but the following guidelines apply:

— if an identity card is used, it should always be preceded or followed by a password;

— for security reasons, it might be required to suppress all response from the system to the identification procedures;

— after a number of consecutive attempts some appropriate action is needed. For example: generate an alarm, or temporarily block access to the system from that terminal.

2.2.2.1 ready indication

The ready indication indicates that the direction of the dialogue has changed and that the system is waiting for information to be given at the terminal. The ready indication is defined as the character < (less than sign) optionally preceded by the appropriate format effectors. The < (less than sign) character is not necessarily required in extended MML (Recommendations Z.321-Z.323), as the information that the terminal is ready for input can be given by cursor position, or additional information contained somewhere in the menu or form.

2.2.3 header

The header (see Recommendation Z.316) is output by the system at the end of the procedure prologue.

2.3 destination prologue

The destination prologue consists of a destination identifier terminated by the separator > (greater than sign) so as to distinguish it from a command.

The destination identifier indicates the physical area where the command is to be mainly processed, e.g., exchange identification, processor number. It consists of one or more information units separated by - (hyphen). The destination could also be defined by a parameter in the command.

The destination identifier may be followed by a header to indicate that a selected destination is allowed, available and ready or alternatively by a rejection output to indicate the converse.

2.4 **procedure epilogue**

The procedure epilogue is used to terminate the dialogue procedure. The composition of the procedure epilogue is highly dependent on the type of terminal and implementation. The procedure epilogue can consist of actuating a control switch, power off, etc. and/or keying a sequence of characters on the keyboard and/or the output of end of dialogue from the system.

2.5 **interactive operating sequence**

The interactive operating sequence may consist of a single command entry sequence terminated by an optional end statement or of a series of command entry sequences or special actions. The latter occurs when, as a result of partial execution of a function, the system requests the supply of further information in the form of special actions or further commands for which human judgement and/or decision is required.

2.5.1 **command entry sequence**

A command entry sequence contains a single command code, together with an alternating sequence of one or more parameter blocks and an appropriate number of executions.

Any interactive operating sequence may be stopped prematurely by the user with the entry of a particular command entry sequence. The latter could consist of a certain command which is independent of any interactive operating sequence, e.g., EXIT, etc.

2.5.2 *Manual response*

Special actions can include manual responses, such as the actuation of keys on terminals or switchframes and the replacement of equipment.

2.5.3 *Interaction request output*

The system generates an interaction request output in order to obtain further actions.

2.5.4 **end statement**

An end statement is an indication that an operating sequence has finished.

2.6 *Direct parameter input*

Only one method of inputting parameters is dealt with in direct parameter input. For other methods refer to Recommendations Z.321 to Z.323.

Direct parameter input consists of an optional parameter block entry sequence preceded by the separator : (colon). The none or more parameter blocks are to be terminated by the execution character ; (semicolon) or by the continuation character ! (exclamation mark) to initiate the required functions which will result in a response output.

If terminated by an execution character and responded by an acceptance or rejection output, the system concludes the direct parameter input. If terminated by a continuation character and responded by an acceptance or rejection output, the system is required to return a parameter block request indication that functions as an indication to proceed with the input of the next block or blocks of parameters. If responded by a request output the system is required to return a parameter block request indication that functions as an invitation for entering either an updated part of the current block of parameters (e.g., a parameter that was erroneously input) or an expansion of the current block of parameters, dependent on the contents of the request output. Following the parameter block request indication, the command entry sequence can be abandoned by invoking the delete command function.

The parameters are input in accordance with the parameter block entry sequence.

2.6.1 *Parameter block entry sequence*

The parameter block entry sequence is used to input a block of parameters. All parameters are entered according to the input syntax. The entry of the parameters may be done directly without help from the system as described in Recommendation Z.315, or assistance from the system may be requested by calling the prompting facility. Prompting helps in providing a correct input by the system giving guidance on the next input requirement.

The output given by the prompting facility can be either of the following:

a) Guidance output followed by a ? (question mark). The guidance may apply to the complete block of parameters, to that part of the block of parameters that is still to be input or to the single parameter next to be input. Moreover it may contain an indication that the input supplied is sufficient and that an execution order may be given. Guidance can be requested anywhere in the parameter block entry sequence.

b) Parameter name output followed by an = (equal sign). The parameter name applies to the parameter value next to be input.

It is the objective of the parameter name output or guidance output to assist the user in giving correct input required by the system for the current command. In both cases the system may verify input received — if possible — and prompt with enough information to enable input to continue.

What kind of prompting output is given is dependent on the prompting facilities supported by the system involved and — if more than one facility is supported — on the place of the request for prompting.

These recommendations address prompting on request of the user. Unsolicited system directed prompting is also possible but is not covered by these recommendations.

Following ‘‘parameter name output’’, a default value for the parameter cannot be implied by simply omitting the value. A specific ‘‘default indicator’’ must be given. If, however, a further ? (question mark) is input, the system will give guidance output, and default by omission may then be possible.

2.6.2 *Parameter block request indication*

The parameter block request indication consists of a : (colon) optionally preceded by the appropriate format effectors and/or the appropriate command code.

2.7 *Response output*

Response output covers all types of output conveying information about the state of an input. Types of response output are acceptance output, rejection output and request output.

A list of categories of each type of response output is given below. Each category is identified by means of the status of the requested action or by means of the error introduced by the user. The title of each category is not meant to be interpreted as the text to be associated with each response output. Additional categories may be created, e.g., by dividing into several parts any one of the categories listed below.

2.7.1 *Acceptance output*

Acceptance output is an indication that an input to the system is syntactically correct and complete and that the appropriate system actions will be initiated, or have already been carried out. In the latter case, this indication may take the form of the result of the actual action.

Category of acceptance output Description

COMMAND EXECUTED The input command was correct and the requested action(s) was successfully performed. The execution of some commands may produce a result to be output immediately after the command has been input. In this case, the result itself may act as the acceptance output.

COMMAND ACCEPTED The input command was correct and the requested action(s) was accepted. This action(s) is either in progress or has been scheduled to be performed. Subsequent outputs related to this requested action may follow later.

2.7.2 *Rejection output*

Rejection output is an indication by the system that the input received is not valid and will not be acted upon, nor can correction be applied, e.g., when the system determines that the user is not authorized to request the action required by the command.

Category of rejection output Description

UNACCEPTABLE COMMAND The command form is valid but the requested action conflicts with the current system or equipment status, e.g., an attempt to restore an in-service unit.

NO SYSTEM RESOURCES The requested action cannot be executed now due to unavailable system resources such as system overload, excessive queue lengths, busy programs, etc. The command may be entered again later.

TRANSMISSION ERROR A transmission error occurred in the input and the system will not accept the command.

SYSTEM ACCESS UNAVAILABLE Input/output access to the system is currently unavailable.

GENERAL ERROR Any rejection that cannot be placed in one of the more specific rejection output categories.

INVALID PASSWORD The input password is unknown to the system or has been input from an improper terminal.

ILLEGAL COMMAND The input command cannot be requested under the current password or from the terminal from which it has been requested.

INVALID SEQUENCE In an interactive operating sequence a command has been entered in the wrong sequence.

UNKNOWN COMMAND CODE The input command is not recognized by the system.

TIME OUT ERROR ##1 The next input character has not been received in time for processing and the command has been aborted.

INVALID COMMAND

CODE SEPARATOR

–v'1P' –v'8p' The command code contains an invalid separator.

INVALID COMMAND

CODE IDENTIFIER

–v'1P' –v'8p' The command code contains an invalid identifier.

2.7.3 *Request output*

Request output is an output message which requests further input action, e.g., to correct an erroneous parameter.

Category of request output Description

INVALID SEPARATOR The wrong input character has been used as a separator.

INVALID INDICATOR The wrong input character has been used as an indicator.

INVALID PARAMETER NAME A parameter name not associated with this command has been input.

EXTRA PARAMETERS Too many parameters have been entered or a parameter has been entered in a command not requiring parameters.

MISSING PARAMETER One or more parameters required by the command have not been entered.

INCONSISTENT PARAMETER The set of parameters in a command does not form a valid set, or the parameters received at an intermediate point are not a valid subset.

MISSING DATA One or more information units of a parameter argument have been omitted.

INCONSISTENT DATA One or more parameter arguments are inconsistent with arguments associated with other parameters, or with the presence (absence) of other parameters in the command, or with data already in the system, although each could be individually valid.

INVALID INFORMATION GROUPING The type of information grouping used in the input of the parameter value is not valid.

RANGE ERROR The value(s) assigned to a parameter is out of the range of the allowed values.

INVALID INFORMATION UNIT The information unit(s) introduced to specify the value(s) of a parameter does not match with the syntactic element requested for the information unit(s).

2.7.4 *Miscellaneous output*

A category of output that does not belong to one of the types above is that given when the dialogue is closed on the initiative of the system.

Category of output Description

TIME OUT ERROR ##2 The next input after the completion of a command has not been received in time and the dialogue has been aborted.

3 **Definition of the dialogue procedure syntax in diagrams**

Recommendations Z.315 and Z.316 describe the input and output syntactic elements used, but not defined, in this Recommendation.

3.1 *Dialogue procedure*

Figure CCITT-34311, (M), p.

3.2 *Procedure prologue*

Figure CCITT-29530, (M), p.

3.2.1 *Request*

Figure CCITT-26491, (M), p.

3.2.2 *Identification procedure*

Figure T1000800-87, (N), p.

3.2.2.1 *Ready indication*

Figure T1000810-87, (N), p.

3.2.2.2 *Identification or password invitation*

Figure CCITT-34330, (M), p.

3.2.2.3 *Password*

Figure CCITT-34341, (M), p.

3.3 *Destination prologue*

Figure CCITT-34360, (M), p.

3.3.1 *Destination identifier*

Figure CCITT-29590, (M), p.

3.4 *Procedure epilogue*

Figure CCITT-57210, (M), p.

3.4.1 *End of dialogue*

Figure CCITT-19043, (M), p.

3.5 *Interactive operating sequence*

Figure CCITT-29611, (M), p.

3.5.1 *Command entry sequence*

Figure CCITT-29641, (M), p.

3.5.2 *Manual response*

Figure CCITT-29621, (M), p.

3.5.3 *Interaction request output*

Figure CCITT-29631, (M), p.

3.5.4 *End statement*

Figure CCITT-19043, (M), p.

3.6 *Direct parameter input*

Figure CCITT-57190, (MC), p.

3.6.1 *Parameter block entry sequence*

Figure CCITT-29671, (M), p.

3.6.2 *Parameter block request indication*

Figure CCITT-34441, (M), p.

3.6.3 *Guidance output*

Figure CCITT-29602, (M), p.

3.7 *Response output*

3.7.1 *Acceptance output*

Figure CCITT-34460, (M), p.

3.7.2 *Rejection output*

Figure CCITT-34470, (M), p.

3.7.3 *Request output*

Figure CCITT-34480, (M), p.

4 Input/output management

4.1 *General*

The question of input/output management is highly hardware and system dependent. Input/output management strategies should be provided to:

— solve any conflict of output outside dialogue directed to an input/output (I/O) device involved in a dialogue procedure;

— solve any conflict of more than one output outside dialogue competing for the same I/O device;

— permit the user to perform a dialogue at any time.

4.2 *Priorities of output*

The priority of an output outside dialogue will determine the behaviour of the output in relation to a dialogue procedure and in relation to other outputs. System crash messages and those outputs that occur after a dangerous situation, implying an immediate recovery procedure such as system reload, are not governed by the following input/output management procedures but may be output at any time.

The priority of an output outside dialogue is the property of the output and dictates the sequence of the output. When several outputs are competing for the use of the same I/O device, the output with the highest priority is output first. Outputs of the same priority are output on a first come first served basis. From an input/output management point of view there shall be two classes of priority for output outside dialogue: high, low.

Lengthy outputs shall be divided into convenient units. Interruptions of output shall only occur at the end of an output unit. A suitable dimension for a unit of output shall be sufficient to allow the output of a meaningful message.

4.3 *Output to a device not in a dialogue procedure*

An output outside dialogue directed to an I/O device not involved in a dialogue procedure is always output, unless another output is in progress on that I/O device, in which case the current output must be completed first. These outputs may be interrupted by input (see § 4.5).

Optionally a system may choose to output the current output only up to the end of the current unit of output before outputting a waiting high priority output.

4.4 *Output to a device in a dialogue procedure*

High priority outputs, which are outputs outside dialogue, are allowed either to be announced or to interrupt the dialogue between interactive operating sequences When a high priority output is announced by means of a message waiting indication, an acceptance input can be given which will cause the waiting output to take place (see § 4.4.1 for an extended syntax diagram for output interrupting input).

Low priority outputs, which are outputs outside dialogue, are not allowed to be announced or to interrupt the dialogue and should be delayed until the end of the dialogue.

4.4.1 *Interruption in dialogue due to input/output management*

Figure CCITT-34491, (M), p.

4.5 *Input interrupting output*

A facility is provided to allow the interruption of an output occurring at an I/O device. However, a request, rejection or acceptance output (where it is not used as the result of the actual action) cannot be interrupted. The output may be interrupted by means of a request as defined in § 2.2.1. When the above request has been made the dialogue with the system can be started/continued.

The interrupted output may be managed by giving an instruction to resume, cancel or restart it. Alternatively, the interrupted output may be managed according to the property of the message itself, assigned at the time of message design.

When the interrupt request is given, the interrupt shall be carried out after the current unit of output.

Interruption in other places is not excluded.

5 Time-out control inside dialogue

Two particular time-outs are identified within a dialogue. The time-outs are provided to prevent lockout of outputs and/or to prove the presence of the user. The latter is used when the system has functions for procedure prologue and epilogue. In this case, two time-outs may be provided where the first one is used within any input. The second time-out is set after completion of the procedure prologue, the destination prologue, and the command entry sequence. Both time-outs are cancelled by the receipt of any input.

When the first time-out elapses, it is suggested that cancellation of the actual input should occur. When the second time-out elapses, it is suggested that the epilogue procedure should take place. Any output can take place when the first time-out has elapsed.

ANNEX A

(to Recommendation Z.317)

Use of SDL to describe MML dialogue procedures

A.1 *Introduction*

The specification and Description Language (SDL) described in the Z.100 series Recommendations can be used to describe MML dialogue procedures. This annex provides SDL examples of MML dialogue procedures from Recommendation Z.317.

A.2 *SDL description of dialogue procedures*

The SDL diagrams in figures A-1/Z.317 to A-3/Z.317 cover the main procedural aspects described in § 3 of Recommendation Z.317, excluiding the ‘‘Parameter entry sequence’’. Also, other aspects, such as I/O management and timing recommended in §§ 4 and 5 of Recommendation Z.317 have not been dealt with in the SDL diagrams.

The SDL diagrams have been developed with the aim of describing the MML interface. The SDL elements are:

SDL element	*Purpose*
INPUT	What the operator keys in
OUTPUT	System response
DECISION	A system decision point
ALTERNATIVE	Shows different implementation possibilities

The SDL diagrams correspond to the following figures in Recommendation Z.317:

Figure A-1/Z.317 Procedure prologue (§ 3.2)

Request (§ 3.2.1)

Identification procedure (§ 3.2.2)

Figure A-2/Z.317 Destination prologue (§ 3.3)

Procedure epilogue (§ 3.4)

Figure A-3/Z.317 Interactive operating sequence (§ 3.5)

Command entry sequence (§ 3.5.1)

Direct parameter input (§ 3.6)

Figure A-1/Z.317, (feuillet 1 sur 2), (N), p.81

Figure A-1/Z.317, (feuillet 2 sur 2), (N), p.82

Figure A-2/Z.317, (N), p.83

Figure A-3/Z.317, (feuillet 1 sur 2), (N), p.84

Figure A-3/Z.317, (feuillet 2 sur 2), (N), p.85