INTERNATIONAL TELECOMMUNICATION UNION

CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

RED BOOK

VOLUME VI – FASCICLE VI.13

MAN-MACHINE LANGUAGE (MML)

RECOMMENDATIONS Z.301-Z.341

VIIITH PLENARY ASSEMBLY
MALAGA-TORREMOLINOS, 8-19 OCTOBER 1984

Geneva 1985

INTERNATIONAL TELECOMMUNICATION UNION

CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

RED BOOK

VOLUME VI – FASCICLE VI.13

MAN-MACHINE LANGUAGE (MML)

RECOMMENDATIONS Z.301-Z.341

VIIITH PLENARY ASSEMBLY
MALAGA-TORREMOLINOS, 8-19 OCTOBER 1984

Geneva 1985

ISBN 92-61-02261-8

CONTENTS OF THE CCITT BOOK
APPLICABLE AFTER THE EIGHTH PLENARY ASSEMBLY (1984)

RED BOOK

Volume I – Minutes and reports of the Plenary Assembly.

Opinions and Resolutions.

Recommendations on:

- the organization and working procedures of the CCITT (Series A);
- means of expression (Series B);
- general telecommunication statistics (Series C).

List of Study Groups and Questions under study.

Volume II – *(5 fascicles, sold separately)*

FASCICLE II.1 – General tariff principles – Charging and accounting in international telecommunications services. Series D Recommendations (Study Group III).

FASCICLE II.2 – International telephone service – Operation. Recommendations E.100-E.323 (Study Group II).

FASCICLE II.3 – International telephone service – Network management – Traffic engineering. Recommendations E.401-E.600 (Study Group II).

FASCICLE II.4 – Telegraph Services – Operations and Quality of Service. Recommendations F.1-F.150 (Study Group I).

FASCICLE II.5 – Telematic Services – Operations and Quality of Service. Recommendations F.160-F.350 (Study Group I).

Volume III – *(5 fascicles, sold separately)*

FASCICLE III.1 – General characteristics of international telephone connections and circuits. Recommendations G.101-G.181 Study Groups XV, XVI and CMBD).

FASCICLE III.2 – International analogue carrier systems. Transmission media – characteristics. Recommendations G.211-G.652 (Study Group XV and CMBD).

FASCICLE III.3 – Digital networks – transmission systems and multiplexing equipments. Recommendations G.700-G.956 (Study Groups XV and XVIII).

FASCICLE III.4 – Line transmission of non telephone signals. Transmission of sound-programme and television signals. Series H, J Recommendations (Study Group XV).

FASCICLE III.5 – Integrated Services Digital Network (ISDN). Series I Recommendations (Study Group XVIII).

Volume IV – *(4 fascicles, sold separately)*

FASCICLE IV.1 – Maintenance; general principles, international transmission systems, international telephone circuits. Recommendations M.10-M.762 (Study Group IV).

FASCICLE IV.2 – Maintenance; international voice frequency telegraphy and fascimile, international leased circuits. Recommendations M.800-M.1375 (Study Group IV).

FASCICLE IV.3 – Maintenance; international sound programme and television transmission circuits. Series N Recommendations (Study Group IV).

FASCICLE IV.4 – Specifications of measuring equipment. Series 0 Recommendations (Study Group IV).

Volume V – Telephone transmission quality. Series P Recommendations (Study Group XII).

Volume VI – *(13 fascicles, sold separately)*

FASCICLE VI.1 – General Recommendations on telephone switching and signalling. Interface with the maritime mobile service and the land mobile services. Recommendations Q.1-Q.118 *bis* (Study Group XI).

FASCICLE VI.2 – Specifications of Signalling Systems Nos. 4 and 5. Recommendations Q.120-Q.180 (Study Group XI).

FASCICLE VI.3 – Specifications of Signalling System No. 6. Recommendations Q.251-Q.300 (Study Group XI).

FASCICLE VI.4 – Specifications of Signalling Systems R1 and R2. Recommendations Q.310-Q.490 (Study Group XI).

FASCICLE VI.5 – Digital transit exchanges in integrated digital networks and mixed analogue-digital networks. Digital local and combined exchanges. Recommendations Q.501-Q.517 (Study Group XI).

FASCICLE VI.6 – Interworking of signalling systems. Recommendations Q.601-Q.685 (Study Group XI).

FASCICLE VI.7 – Specifications of Signalling System No. 7. Recommendations Q.701-Q.714 (Study Group XI).

FASCICLE VI.8 – Specifications of Signalling System No. 7. Recommendations Q.721-Q.795 (Study Group XI).

FASCICLE VI.9 – Digital access signalling system. Recommendations Q.920-Q.931 (Study Group XI).

FASCICLE VI.10 – Functional Specification and Description Language (SDL). Recommendations Z.101-Z.104 (Study Group XI).

FASCICLE VI.11 – Functional Specification and Description Language (SDL), annexes to Recommendations Z.101-Z.104 (Study Group XI).

FASCICLE VI.12 – CCITT High Level Language (CHILL). Recommendation Z.200 (Study Group XI).

FASCICLE VI.13 – Man-Machine Language (MML). Recommendations Z.301-Z.341 (Study Group XI).

Volume VII – *(3 fascicles, sold separately)*

FASCICLE VII.1 – Telegraph transmission. Series R Recommendations (Study Group IX). Telegraph services terminal equipment. Series S Recommendations (Study Group IX).

FASCICLE VII.2 – Telegraph switching. Series U Recommendations (Study Group IX).

FASCICLE VII.3 – Terminal equipment and protocols for telematic services. Series T Recommendations (Study Group VIII).

Volume VIII – *(7 fascicles, sold separately)*

FASCICLE VIII.1 – Data communication over the telephone network. Series V Recommendations (Study Group XVII).

FASCICLE VIII.2 – Data communication networks: services and facilities. Recommendations X.1-X.15 (Study Group VII).

FASCICLE VIII.3 – Data communication networks: interfaces. Recommendations X.20-X.32 (Study Group VII).

FASCICLE VIII.4 – Data communication networks: transmission, signalling and switching, network aspects, maintenance and administrative arrangements. Recommendations X.40-X.181 (Study Group VII).

FASCICLE VIII.5 – Data communication networks: Open Systems Interconnection (OSI), system description techniques. Recommendations X.200-X.250 (Study Group VII).

FASCICLE VIII.6 – Data communication networks: interworking between networks, mobile data transmission systems. Recommendations X.300-X.353 (Study Group VII).

FASCICLE VIII.7 – Data communication networks: message handling systems. Recommendations X.400-X.430 (Study Group VII).

Volume IX – Protection against interference. Series K Recommendations (Study Group V). Construction, installation and protection of cable, and other elements of outside plant. Series L Recommendations (Study Group VI).

Volume X – *(2 fascicles, sold separately)*

FASCICLE X.1 – Terms and definitions.

FASCICLE X.2 – Index of the Red Book.

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

CONTENTS OF FASCICLE VI.13 OF THE RED BOOK

Part I – Recommendations Z.301 to Z.341

Man-machine language (MML)

Rec. No.		Page
SECTION 1 –	*General principles*	
Z.301	Introduction to the CCITT man-machine language	3
Z.302	The meta-language for describing MML syntax and dialogue procedures	6
SECTION 2 –	*Basic syntax and dialogue procedures*	
Z.311	Introduction to syntax and dialogue procedures	9
Z.312	Basic format layout	9
Z.314	The character set and basic elements	10
Z.315	Input (command) language syntax specification	17
Z.316	Output language syntax specification	22
Z.317	Man-machine dialogue procedures	30
SECTION 3 –	*Extended MML for visual display terminals*	
Z.321	Introduction to the extended MML for visual display terminals	43
Z.322	Capabilities of visual display terminals	44
Z.323	Man-machine interaction	50
SECTION 4 –	*Specification of the man-machine interface*	
Z.331	Introduction to the specification of the man-machine interface	71
Z.332	Methodology for the specification of the man-machine interface – General working procedure	75
Z.333	Methodology for the specification of the man-machine interface – Tools and methods	82

Rec. No. Page

SECTION 5 – *Glossary of terms*

Z.341 Glossary of terms 153

Part II – Supplements to Recommendations Z.301 to Z.341

Supplement No. 1 MML publicity file 173

Supplement No. 2 MML implementation record 174

PRELIMINARY NOTES

1 The Questions entrusted to each Study Group for the Study Period 1985-1988 can be found in Contribution No. 1 to that Study Group.

2 In this Fascicle, the expression "Administration" is used for shortness to indicate both a telecommunication Administration and a recognized private operating agency.

PART I

Recommendations Z.301 to Z.341

MAN-MACHINE LANGUAGE (MML)

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

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 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.

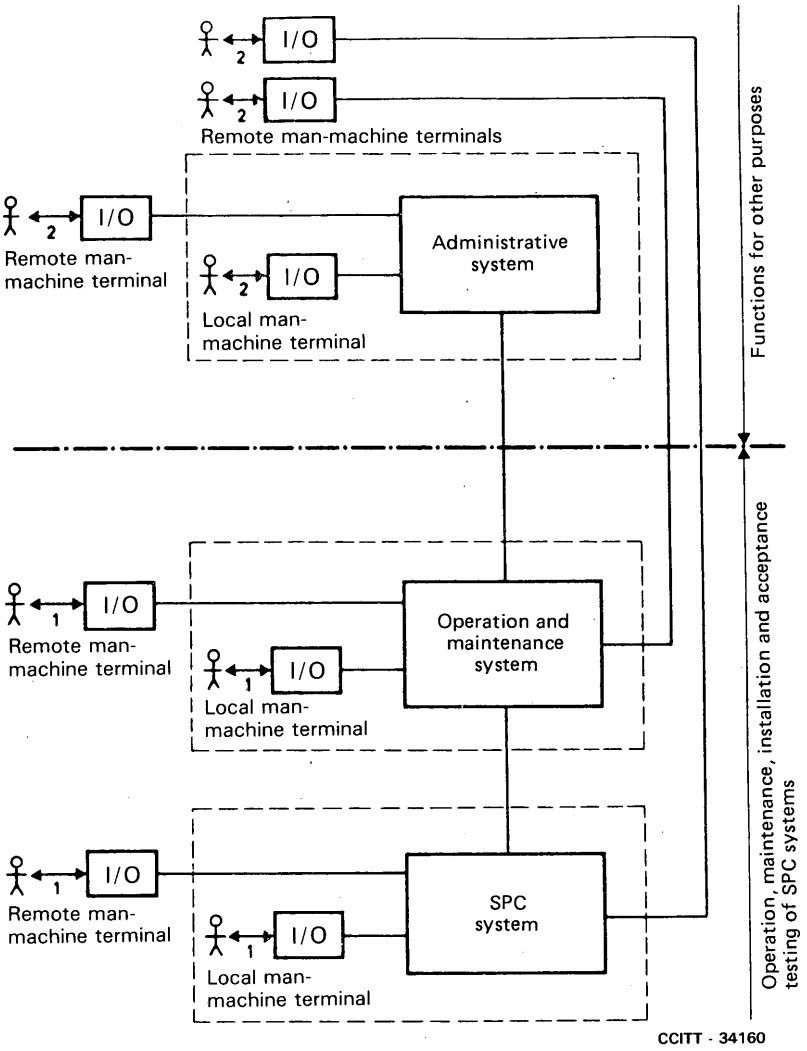
Although telephone signalling and switching has been considered the primary application area for the MML, these Recommendations accommodate the extension of the MML into other areas such as data switching, ISDN operations and maintenance, and software development environments.

In the Recommendations of this Part, the terms *machine* and *system* are used interchangeably, as are *man* and *user.*

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.



1 Man-machine terminals where CCITT MML is recommended to be used.
2 Man-machine terminals where the use of CCITT MML is not considered.
I/O Input/output.

FIGURE 1/Z.301

Field of application of CCITT MML

Man-machine interface	
Inputs, outputs, special actions	Man-machine interaction mechanisms, including dialogue procedures
MML function semantics	MML syntax
System functions	Terminal capabilities

FIGURE 2/Z.301

Man-machine communication model

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 gives 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; some examples of MML function semantics 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 enables 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 language 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[1]. A syntax diagram consists of terminal and 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.

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.

---

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

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.

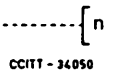
3.6 Non-terminal symbols are surrounded by rectangular boxes. The name of the non-terminal symbol must be written in lower case characters. Every non-terminal symbol must have an associated syntax diagram except where the symbol is annotated "Not further expanded in diagram form". The non-terminal symbol 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:



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.

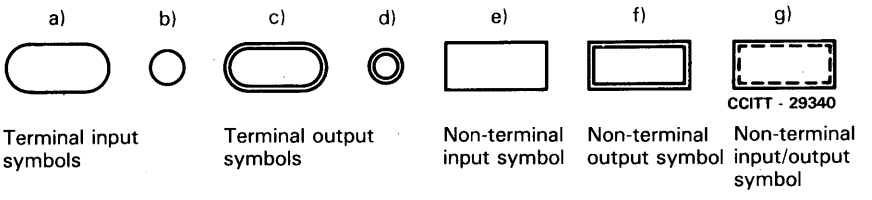


FIGURE 1/Z.302

Terminal and non-terminal symbols to be used for the CCITT man-machine language

Reference

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

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

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 describes 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 V.3 [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 V.3 [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.

TABLE 1/Z.314

Character set to be used for the CCITT man-machine language

				b_7	0	0	0	0	1	1	1	1
				b_6	0	0	1	1	0	0	1	1
				b_5	0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1	Pos.	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL		SP	0	ⓐ	P	ⓐ	p
0	0	0	1	1		DC_1	!	1	A	Q	a	q
0	0	1	0	2		DC_2	"	2	B	R	b	r
0	0	1	1	3		DC_3	#	3	C	S	c	s
0	1	0	0	4		DC_4	$	4	D	T	d	t
0	1	0	1	5			%	5	E	U	e	u
0	1	1	0	6			&	6	F	V	f	v
0	1	1	1	7	BEL		'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT (FE1)	EM	)	9	I	Y	i	y
1	0	1	0	10	LF (FE2)	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT (FE3)	ESC	+	;	K	ⓐ	k	ⓐ
1	1	0	0	12	FF (FE4)		,	<	L	ⓐ	l	ⓐ
1	1	0	1	13	CR (FE5)		-	=	M	ⓐ	m	ⓐ
1	1	1	0	14	SO		.	>	N	ⓐ	n	ⓐ
1	1	1	1	15	SI		/	?	O	_	o	DEL

CCITT - 26622

ⓐ These positions are reserved for national use.

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 V.3 [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 V.3 [1]. The bits are identified by b_7, b_6, ... b_1, where b_7 is the highest order, or most significant 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 V.3 [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.

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).

TABLE 2/Z.314

Summary of use of characters

CCITT International Alphabet No. 5 (Recommendation V.3) [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	per cent 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 as a graphic character.
)	2/9	right parenthesis	Used for delimiting arithmetical expressions and as 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 graphic 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. Also used as an arithmetic operator and as a graphic character.
.	2/14	full stop	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 a graphic character.
:	3/10	colon	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.
;	3/11	semicolon	An indicator used to terminate a command (execution character).
<	3/12	less than sign	An indicator used as a ready indicator for the system to output that it is ready to receive information.
=	3/13	equal sign	A separator used to separate the parameter name and the parameter value of a parameter.
>	3/14	greater than sign	A separator to terminate the destination identifier. Also a graphic character.
?	3/15	question mark	A indicator used for prompting or help.
&&	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	Separator used for information grouping.
/*	2/15, 2/10	solidus asterisk	Used to open a comment.
*/	2/10, 2/15	asterisk solidus	Used to close a comment.

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*

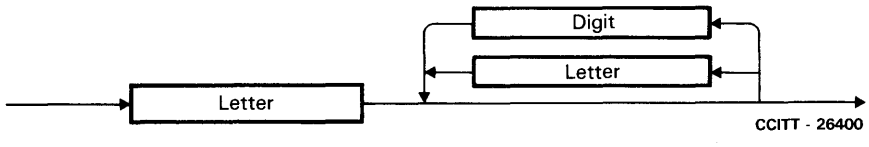
An indicator is a character used to indicate a state or make a request.

No syntax diagram is given.

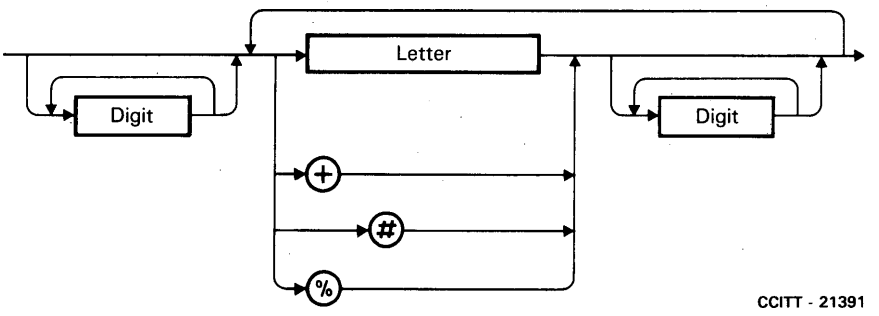
5 Definition of the basic elements used in the syntax in diagrams

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.

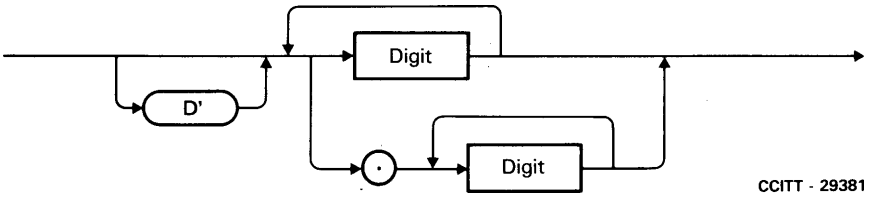
5.1 *Identifier*



5.2 *Symbolic name*

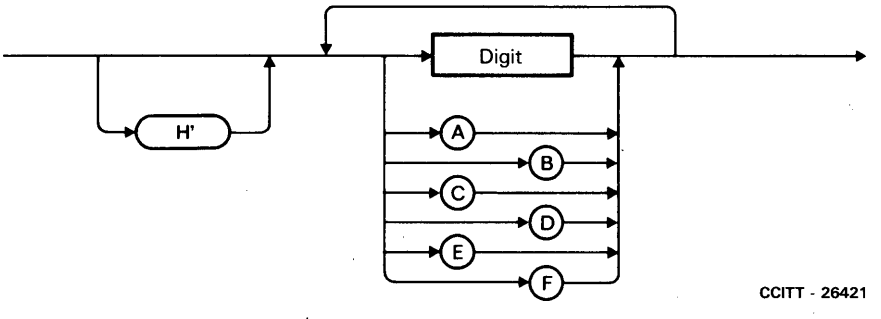


5.3 *Decimal numeral*

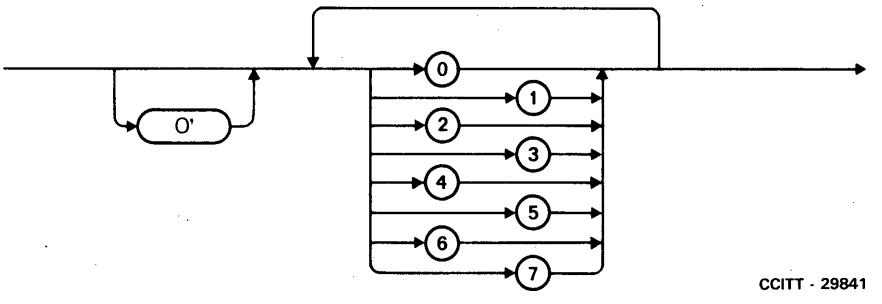


5.4 *Nondecimal numerals*

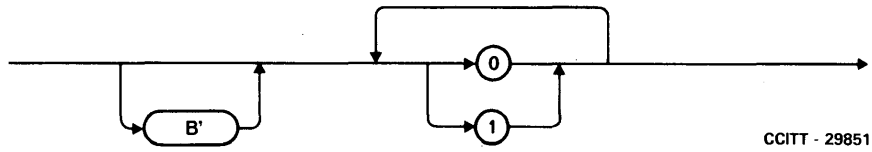
5.4.1 *Hexadecimal numeral*



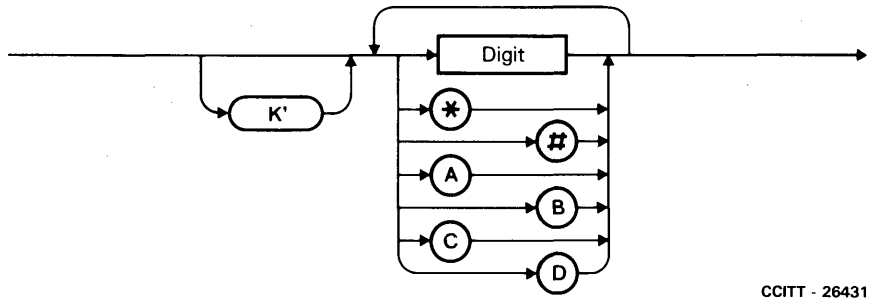
5.4.2 *Octal numeral*



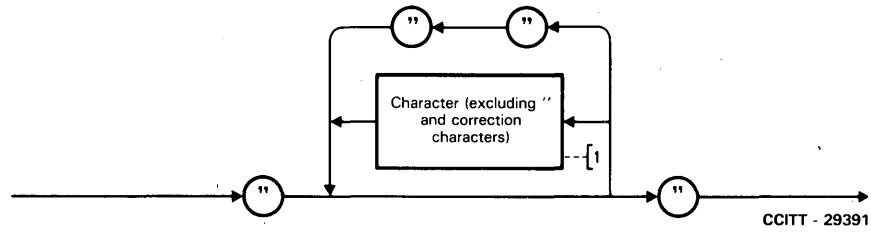
5.4.3 *Binary numeral*



5.4.4 *Keyed numeral*

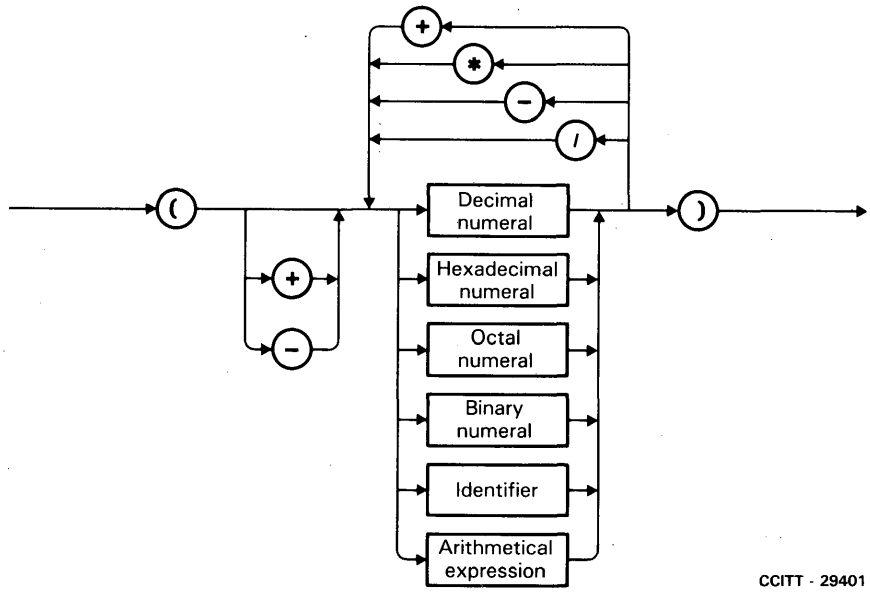


5.5 *Text string*



1) Not further expanded in diagram form.

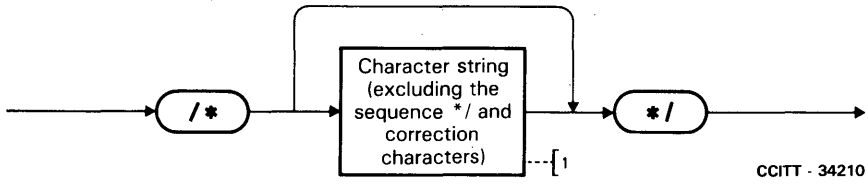
5.6 *Arithmetical expression*



Note – The deepest level of the arithmetical expression has to satisfy the diagram in a form where the box "arithmetical expression" is omitted.

5.7 *Ancillary facilities*

5.7.1 *Comment*



1) Not further expanded in diagram form.

Reference

[1] CCITT Recommendation *International Alphabet No. 5*, Vol. VIII, Fascicle VIII.1, Rec. V.3.

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.

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.

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.

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)[1]. *Example 2:* 5&&9 means the simple parameter arguments 5, 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) [1]. *Example 3:* 7-1&&-3 means the three compound 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.

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).

---

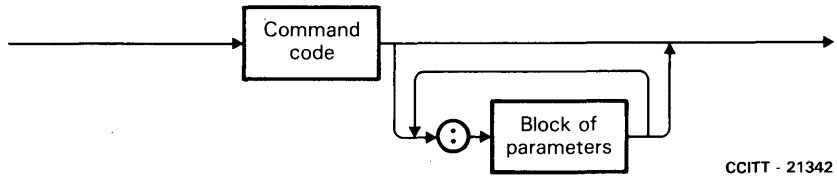
[1] The interpretation of the separators && (ampersand ampersand) and &&- (ampersand ampersand hyphen) is not exclusive. Other interpretations exist. One alternative would imply that no specific increment is inherent in the syntax. That is, the relationship of the values between the upper and lower values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

3.3 *Delete command*

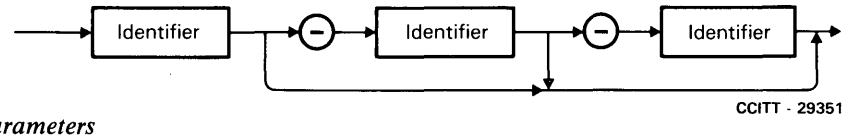
The delete command request is conveyed by the CAN character (cancel). The use of this character causes the system to respond with an acknowledgement that present input after the last command executed is cancelled. The system should respond with a new ready indication to indicate that it is waiting for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

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

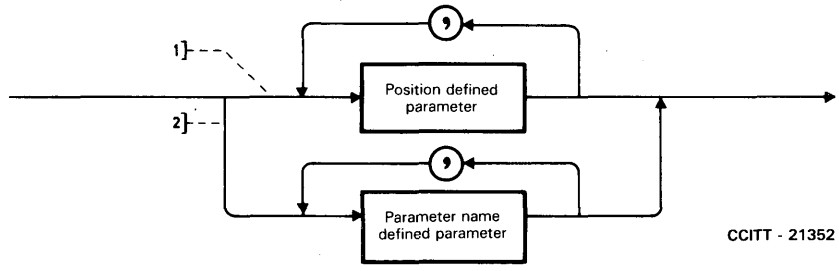
4.1 *Command*



4.2 *Command code*



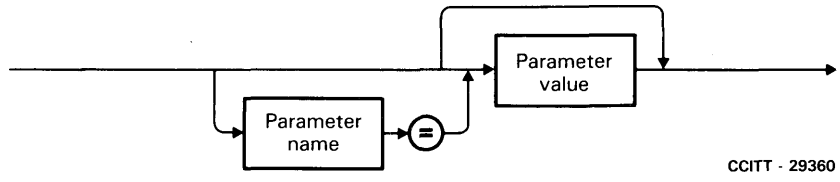
4.3 *Block of parameters*



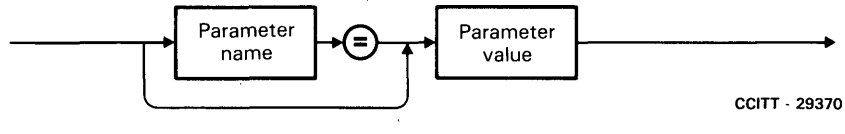
1) Upper main branch valid only for block of position defined parameters.
2) Lower main branch valid only for block of parameter name defined parameters.

4.4 *Parameters*

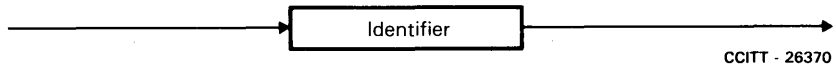
4.4.1 *Position defined parameter*



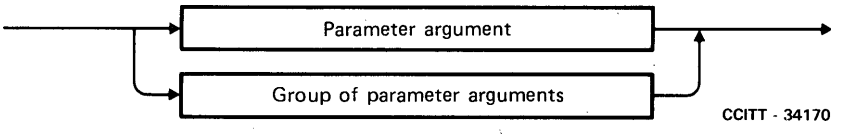
4.4.2 *Parameter name defined parameter*



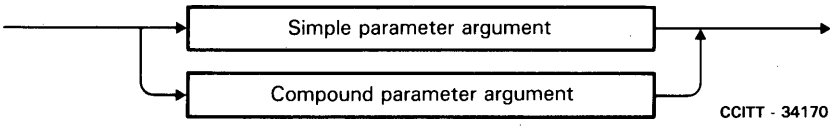
4.5 *Parameter name*



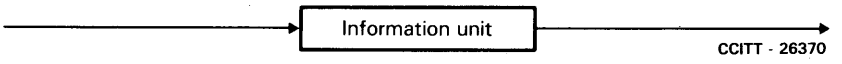
4.6 *Parameter value*



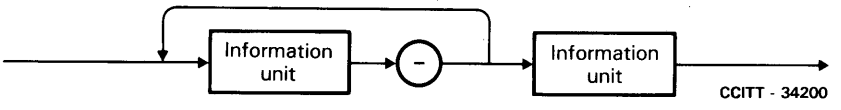
4.7 *Parameter argument*



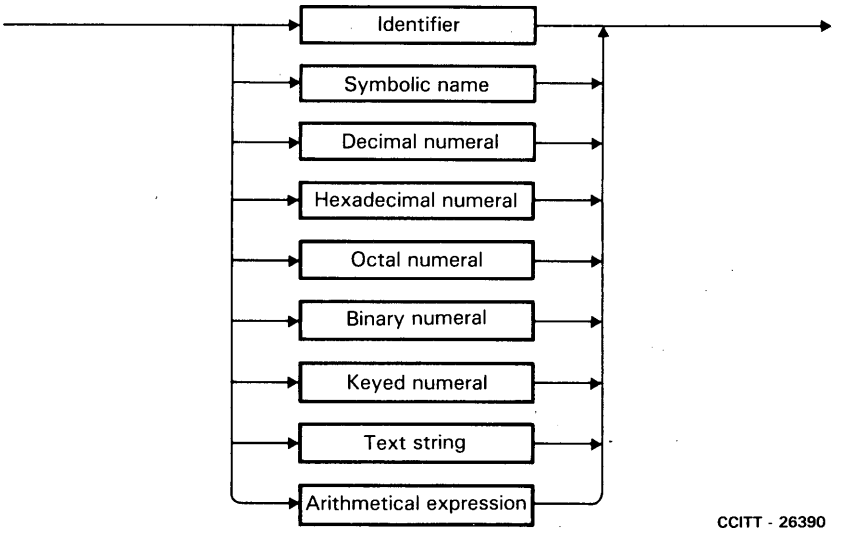
4.7.1 *Simple parameter argument*



4.7.2 *Compound parameter argument*



4.8 *Information unit*

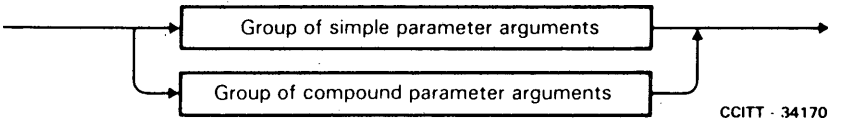


4.9 *Information grouping*

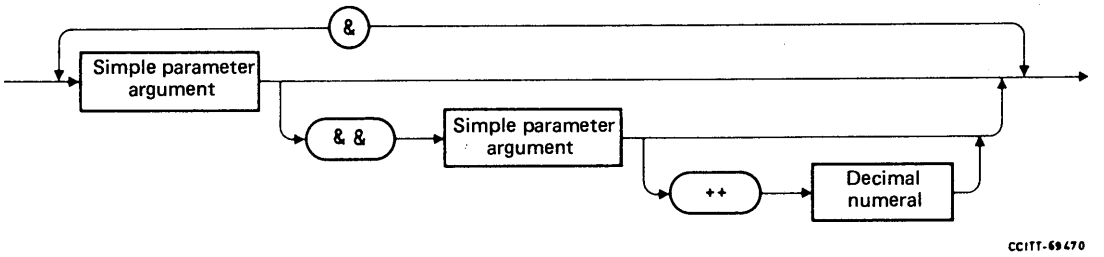
4.9.1 *Group of blocks of parameters*

See syntax diagram § 4.1.

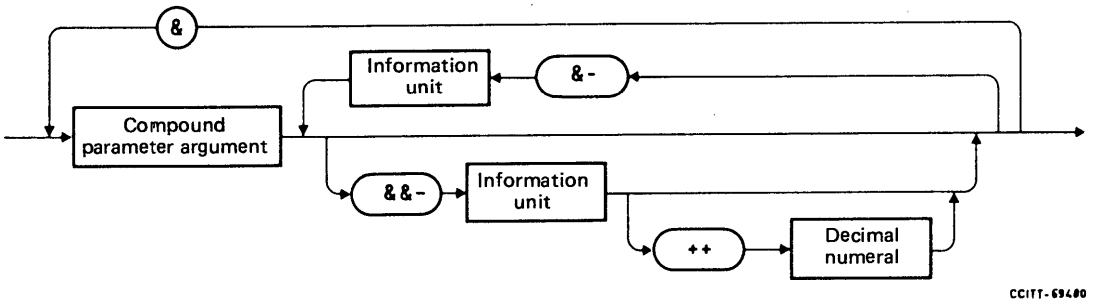
4.9.2 *Group of parameter arguments*



4.9.2.1 *Group of simple parameter arguments*



4.9.2.2 *Group of compound parameter arguments*



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.

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*

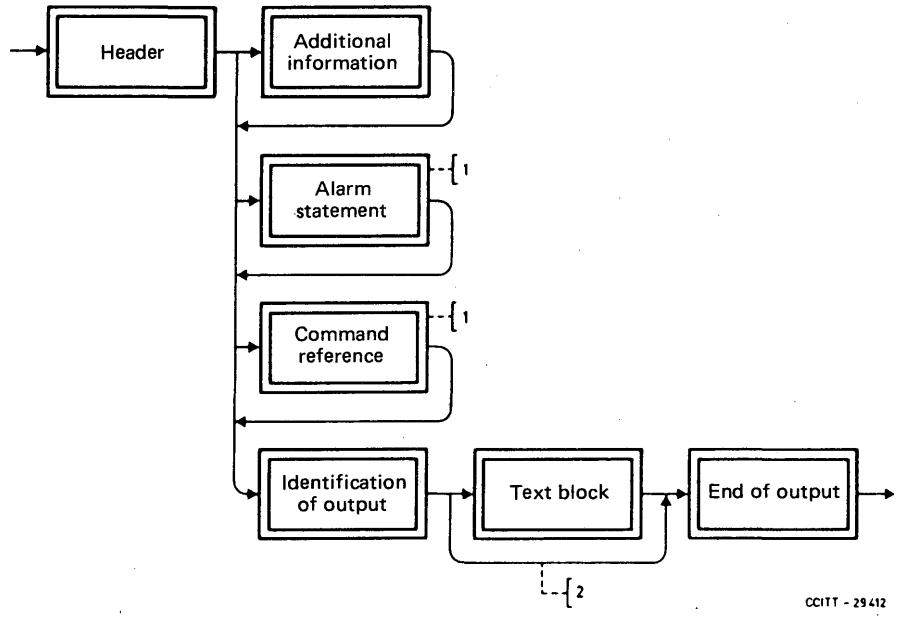
An end of output is an indication that an output is finished.

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

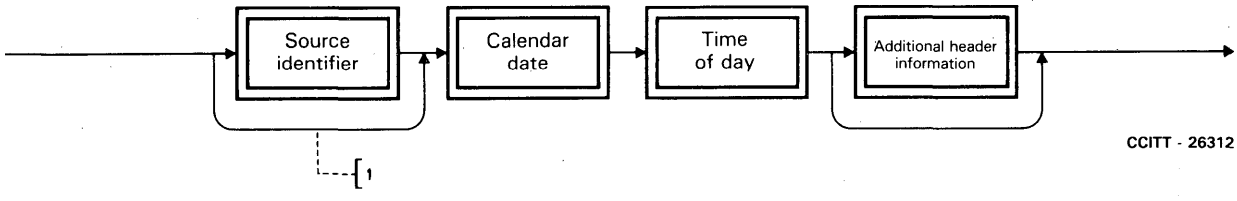
3.1 *Output outside dialogue*



1) Command reference and alarm statement could appear in the same output, e.g. if a control system unit is taken out of service by means of a command.

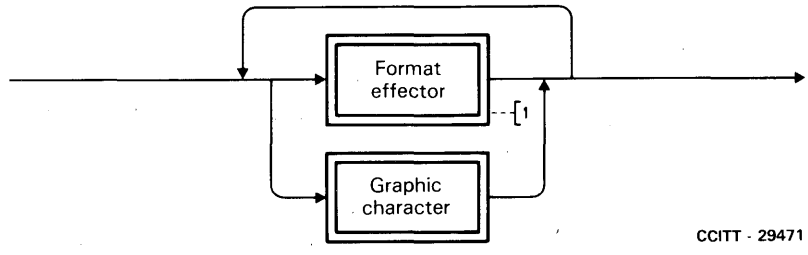
2) This by-pass can be taken only when the identification of output contains sufficient information.

3.2 *Header*



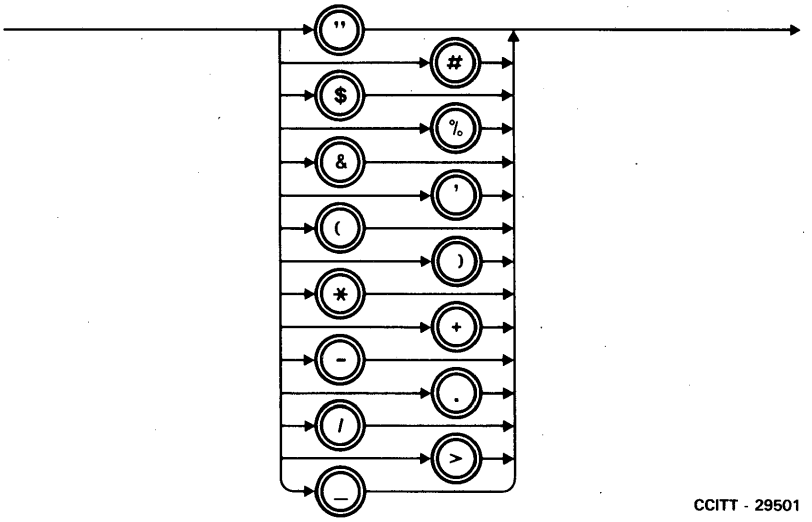
1) Source identifier may be omitted where there is only one source producing outputs.

3.2.1 *Layout option*

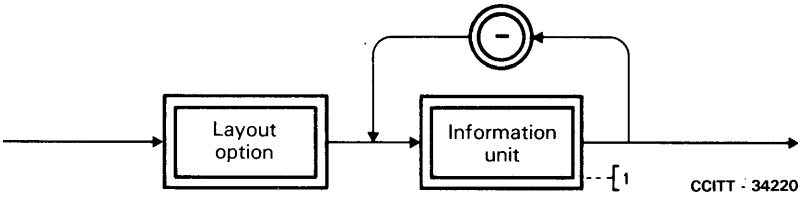


1) Not further expanded in diagram form.

3.2.1.1 *Graphic character*

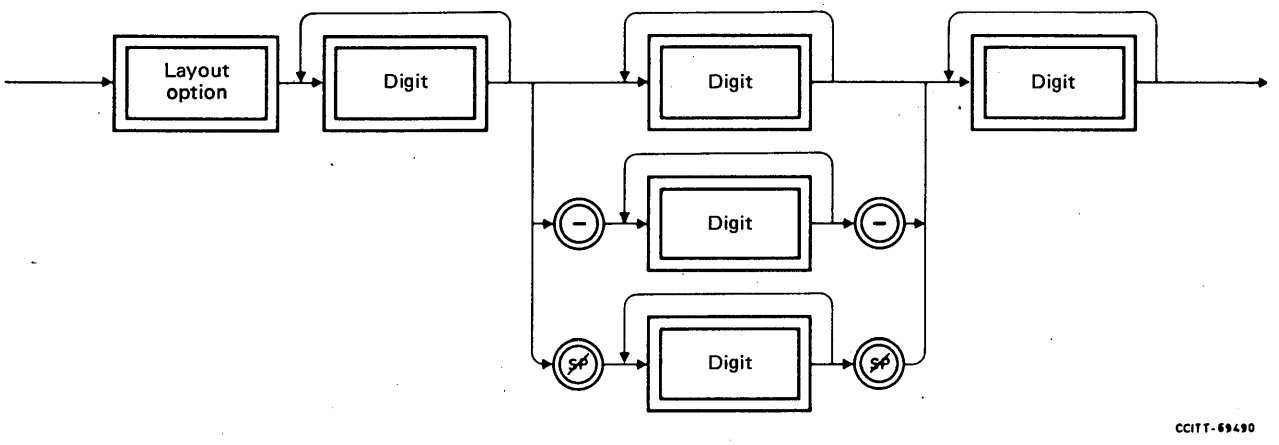


3.2.2 *Source identifier*

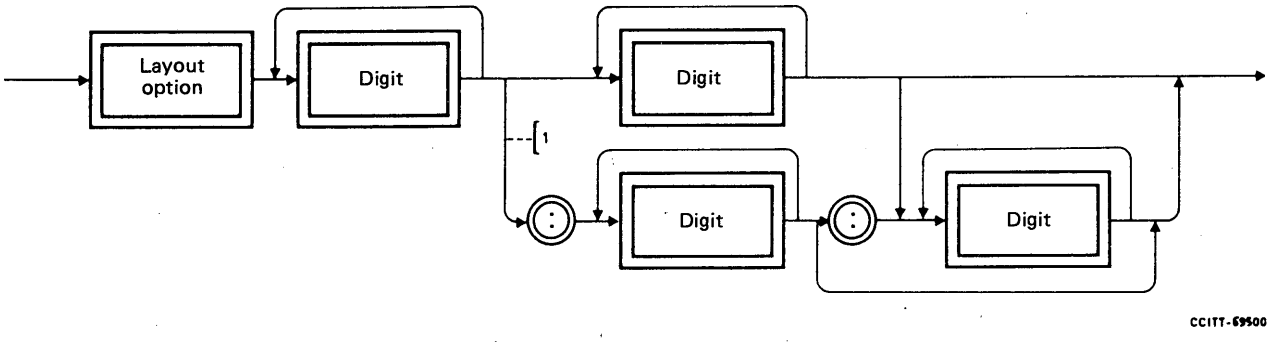


1) See Recommendation Z.315.

3.2.3 *Calendar date*



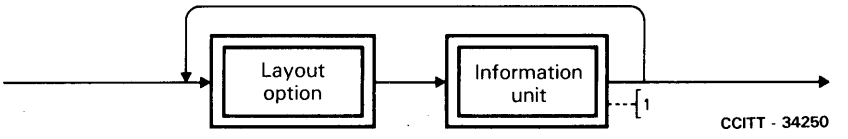
3.2.4 *Time of day*



1) a) If required to facilitate visual human understanding of output, a : (colon) may be used to separate hours, minutes and seconds (refer to [2]).

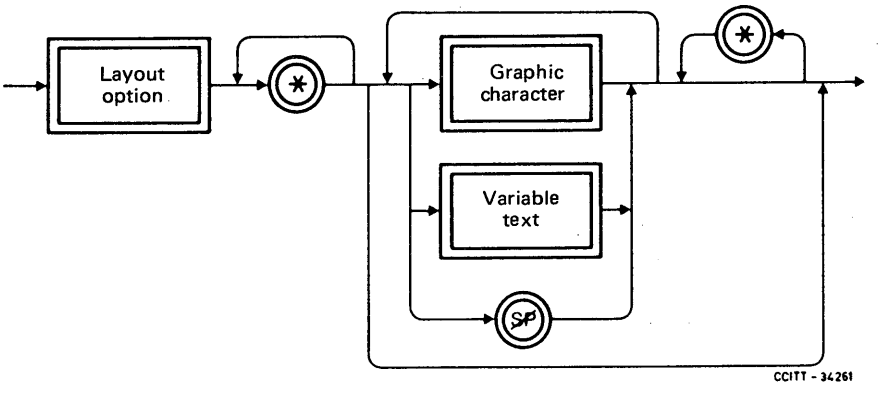
b) This use of the : (colon) is not allowed in input since the character is used as a separator between blocks of parameters.

3.2.5 *Additional header information*

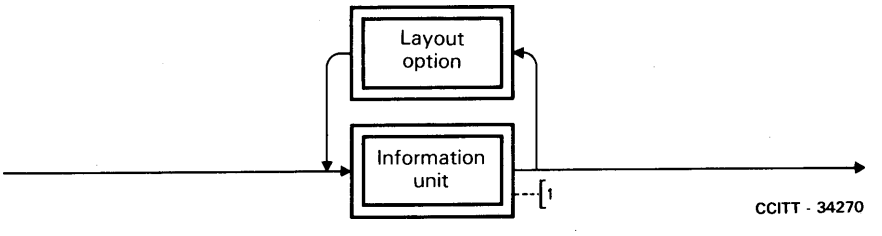


1) See Recommendation Z.315.

3.3 *Alarm statement*

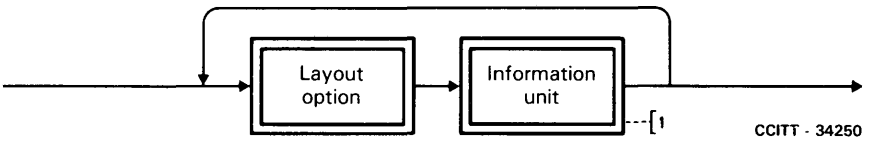


3.3.1 *Variable text*



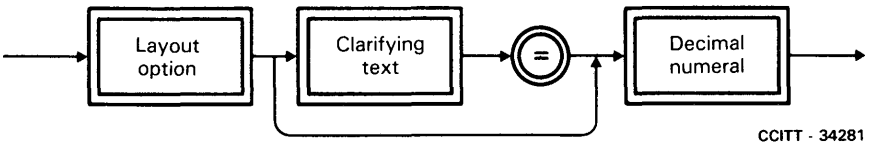
1) See Recommendation Z.315.

3.4 *Additional information*

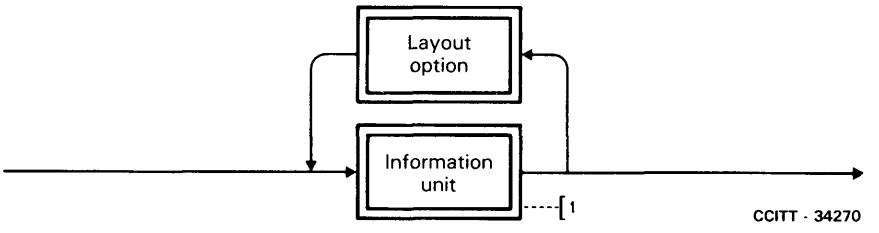


1) See Recommendation Z.315.

3.5 *Command reference*

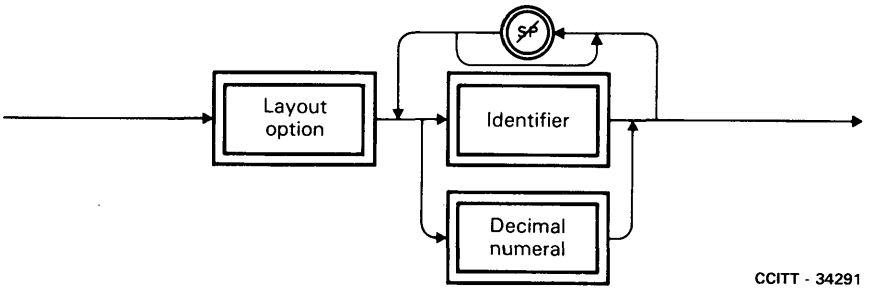


3.5.1 *Clarifying text*

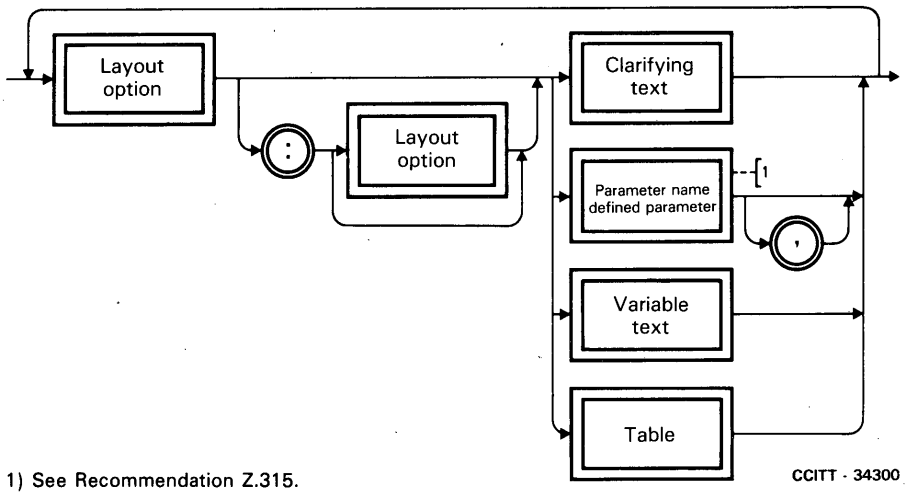


1) See Recommendation Z.315.

3.6 *Identification of output*

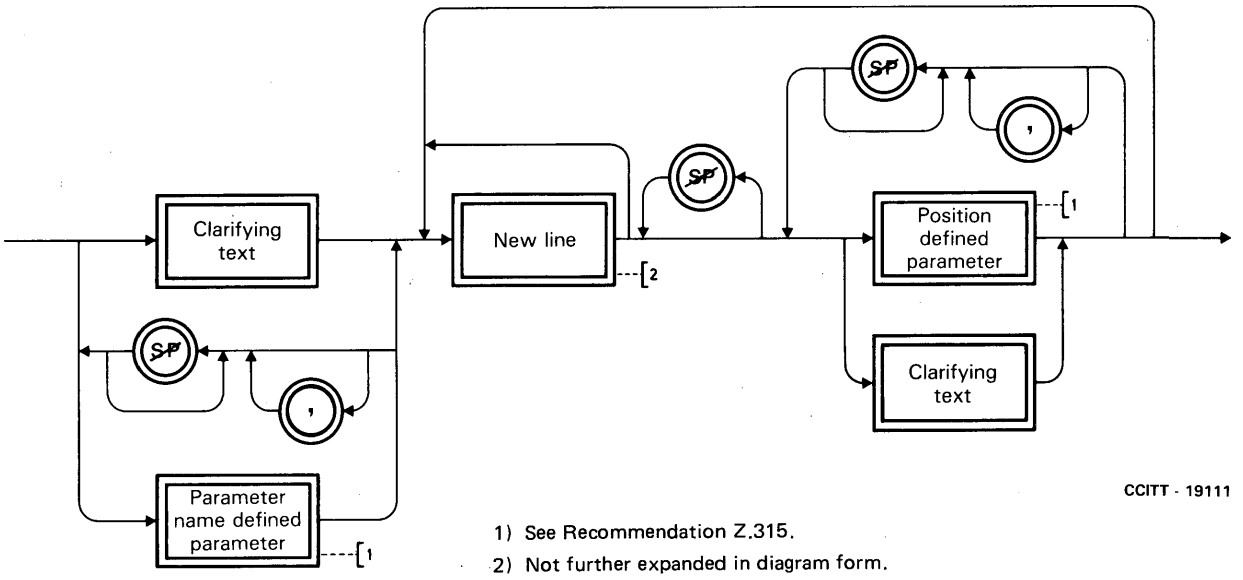


3.7 *Text block*



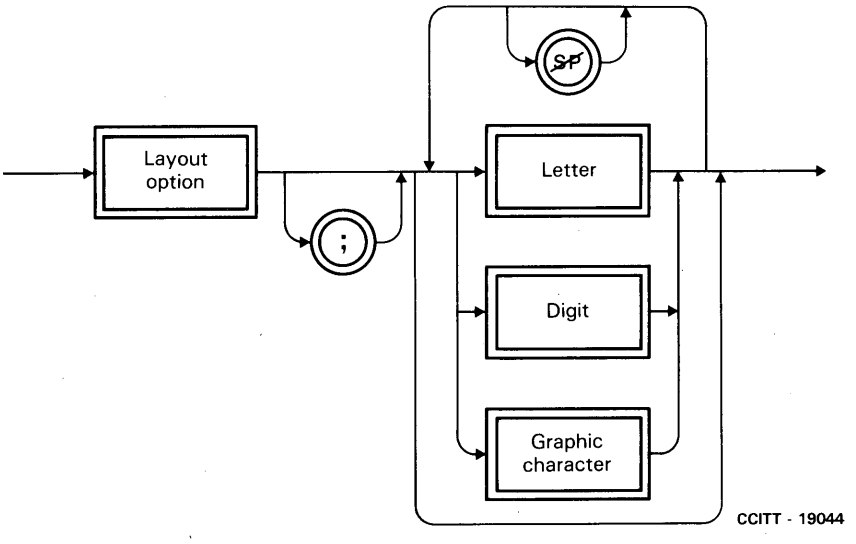
1) See Recommendation Z.315.

3.8 *Table*



1) See Recommendation Z.315.
2) Not further expanded in diagram form.

3.9 *End of output*



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.

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* dialogue 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;
- 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 for identification and authorization of a user. The identification can provide access to groups of commands which can have different security or functional classifications e.g. traffic measurement function. The identification invitation may request the user to identify himself by means of a password or an identity card. The password must be input following a ready indication.

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.

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 response output.

If terminated by an execution character and responded by acceptance or rejection output, the system concludes the direct parameter input. If terminated by a continuation character and responded by 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 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 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	The command code contains an invalid separator.
INVALID COMMAND CODE IDENTIFIER	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*

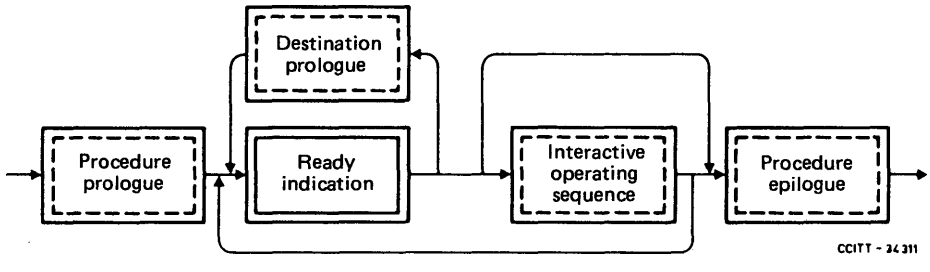
A category of output that does not belong to one of the types above is that given when the dialogue is closed on the initiative of the system.

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.

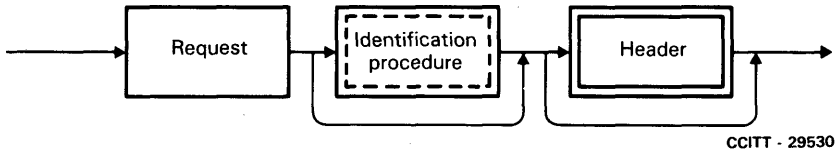
3 Definition of the dialogue procedure syntax in diagrams

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

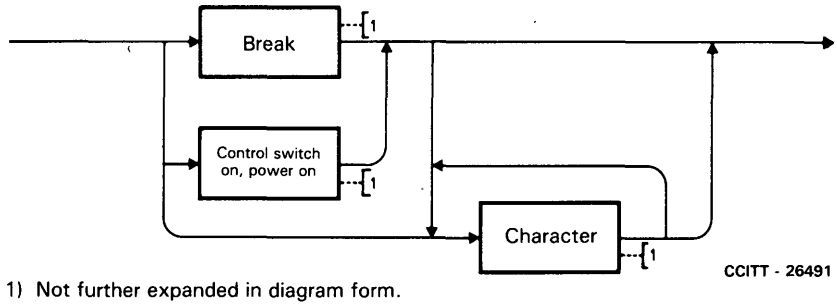
3.1 *Dialogue procedure*



3.2 *Procedure prologue*

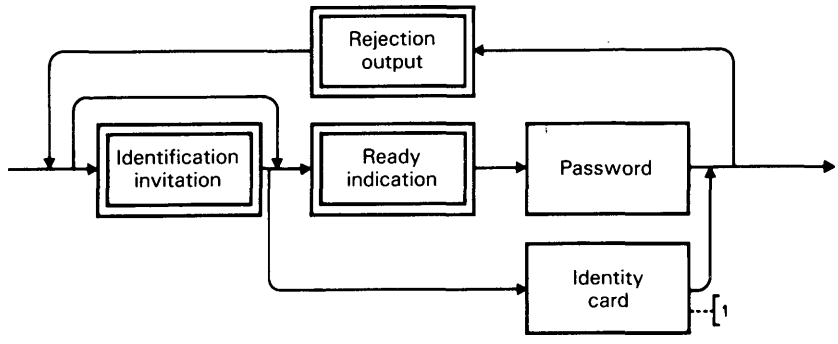


3.2.1 *Request*



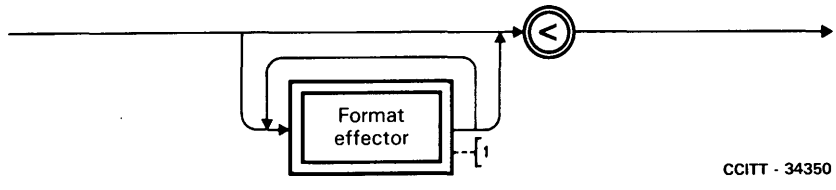
1) Not further expanded in diagram form.

3.2.2 *Identification procedure*



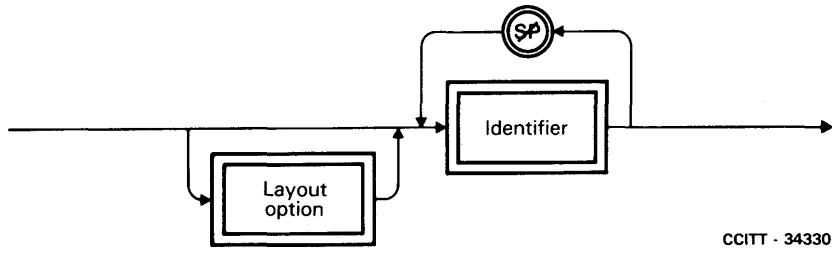
1) Not further expanded in diagram form.

3.2.2.1 *Ready indication*

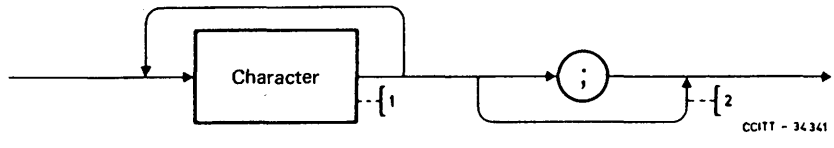


1) Not further expanded in diagram form.

3.2.2.2 *Identification invitation*



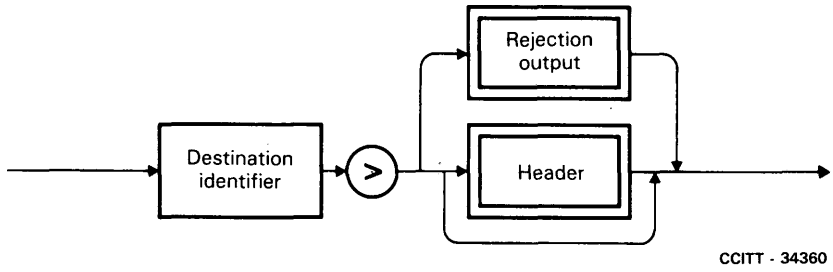
3.2.2.3 *Password*



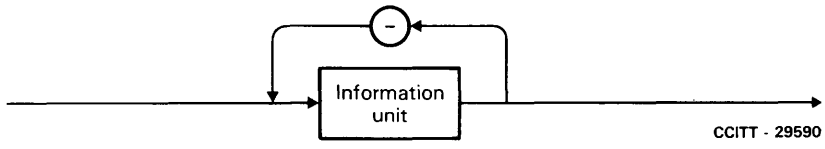
1) Not further expanded in diagram form.

2) If an explicit MML indicator is used to terminate the input, it is recommended to be the ; (semicolon). On the other hand the bypass reflects that other mechanisms to terminate the input are available, e.g. an implicit length of a password.

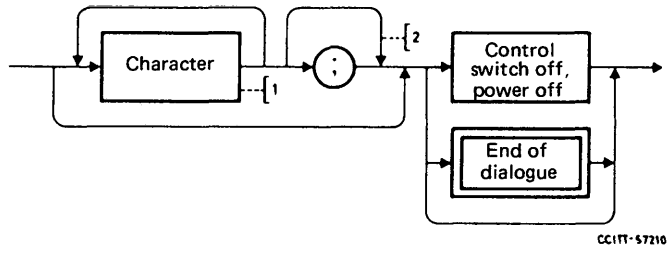
3.3 *Destination prologue*



3.3.1 *Destination identifier*



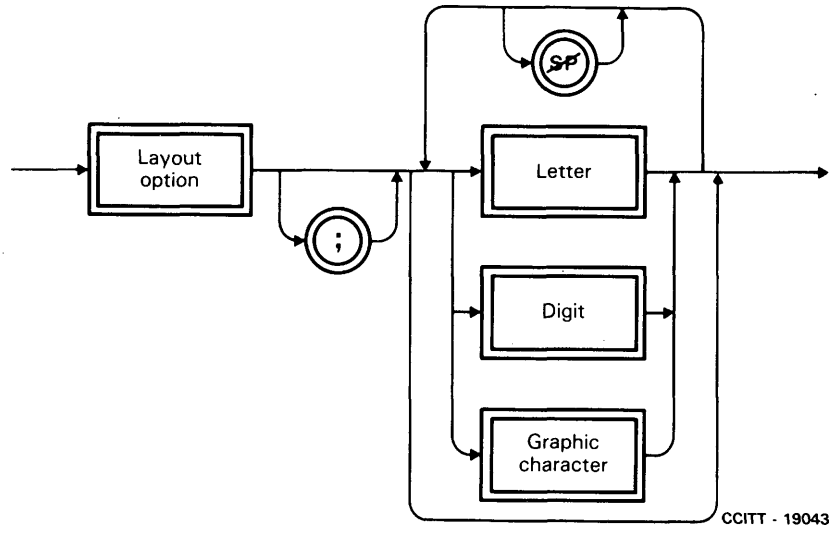
3.4 *Procedure epilogue*



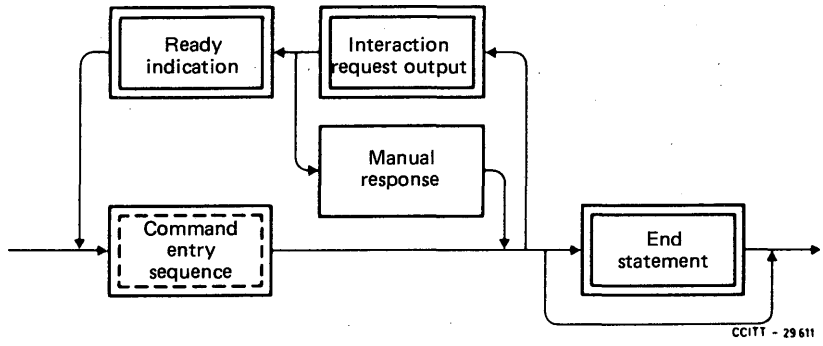
1) Not further expanded in diagram form.

2) If an explicit MML indicator is used to terminate the input it is recommended to be the ; (semicolon). On the other hand the bypass reflects that other mechanisms to terminate the input are available, e.g. a unique set of characters such as "OFF", "BYE".

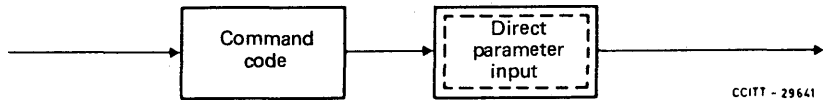
3.4.1 *End of dialogue*



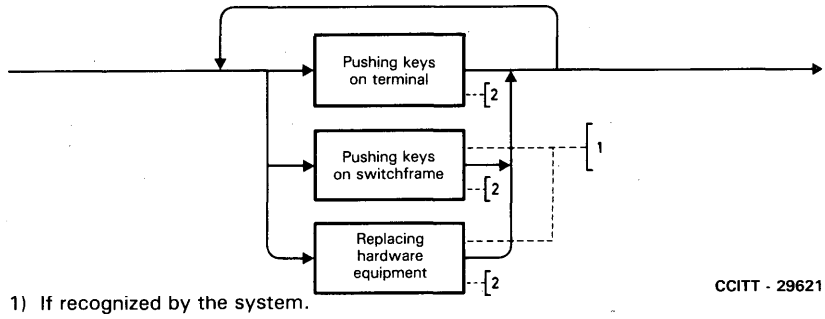
3.5. *Interactive operating sequence*



3.5.1 *Command entry sequence*

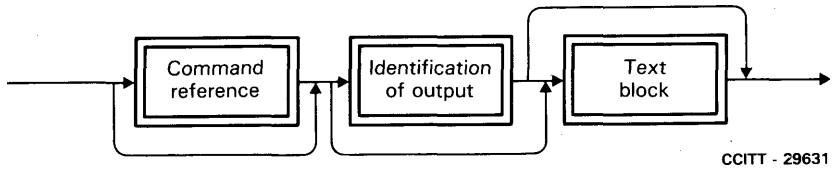


3.5.2 *Manual response*

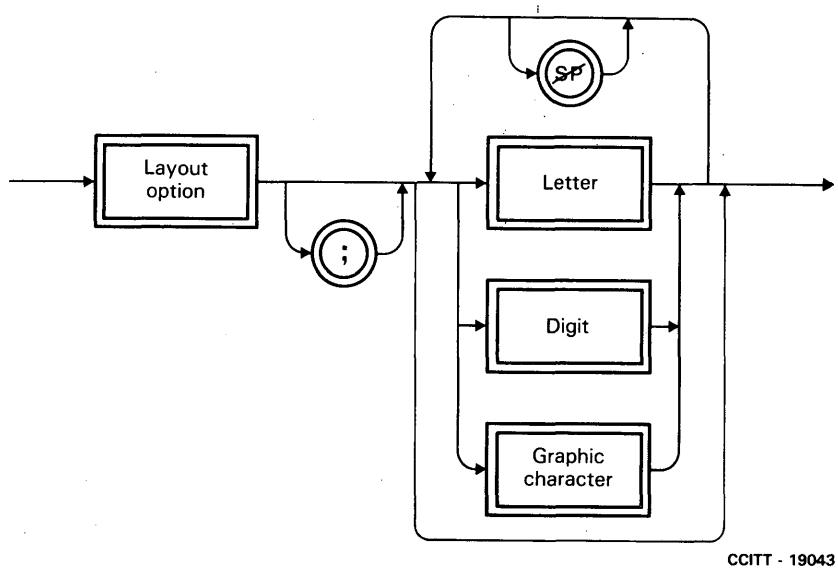


1) If recognized by the system.
2) Not further expanded in diagram form.

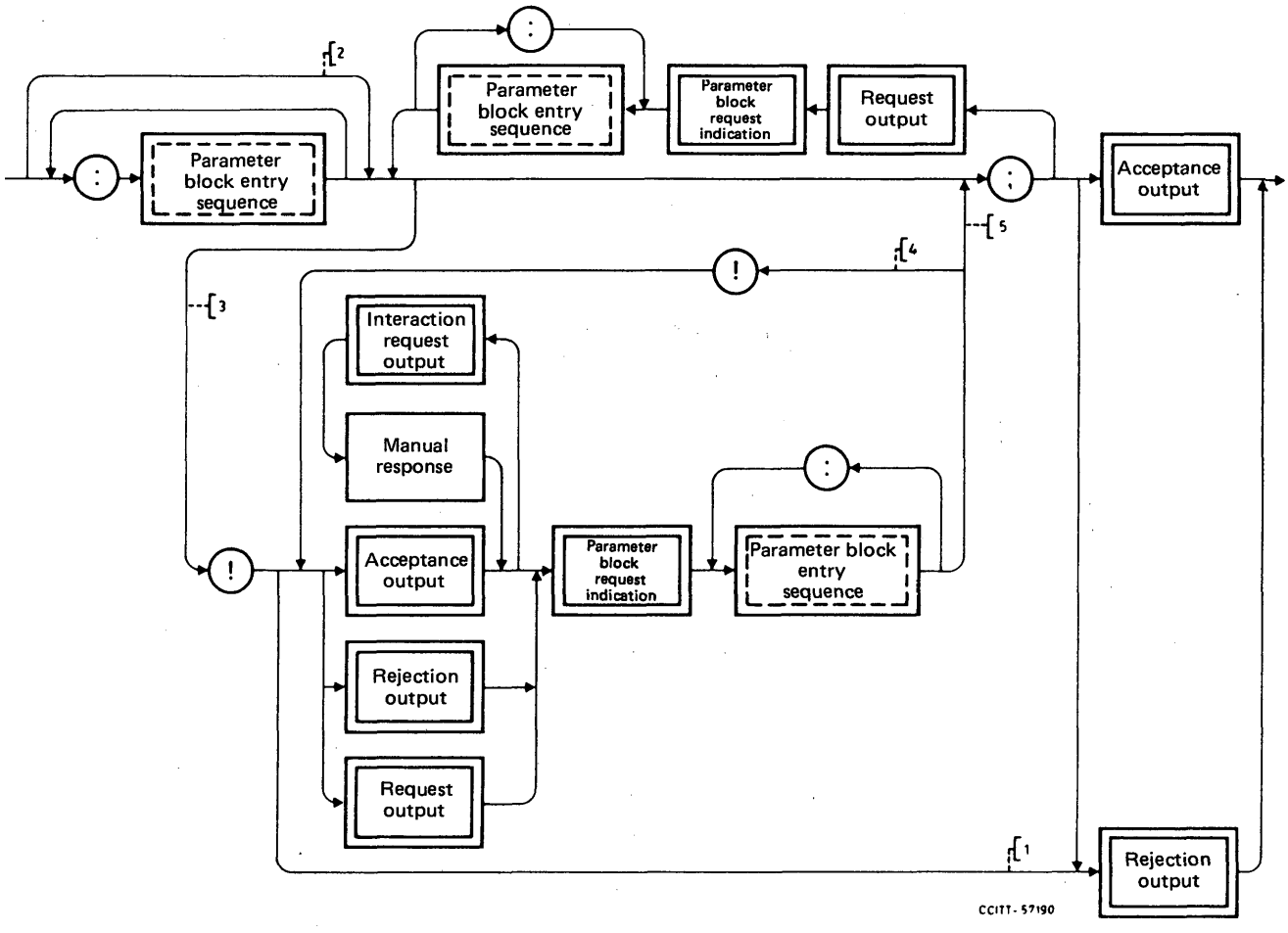
3.5.3 *Interaction request output*



3.5.4 *End statement*

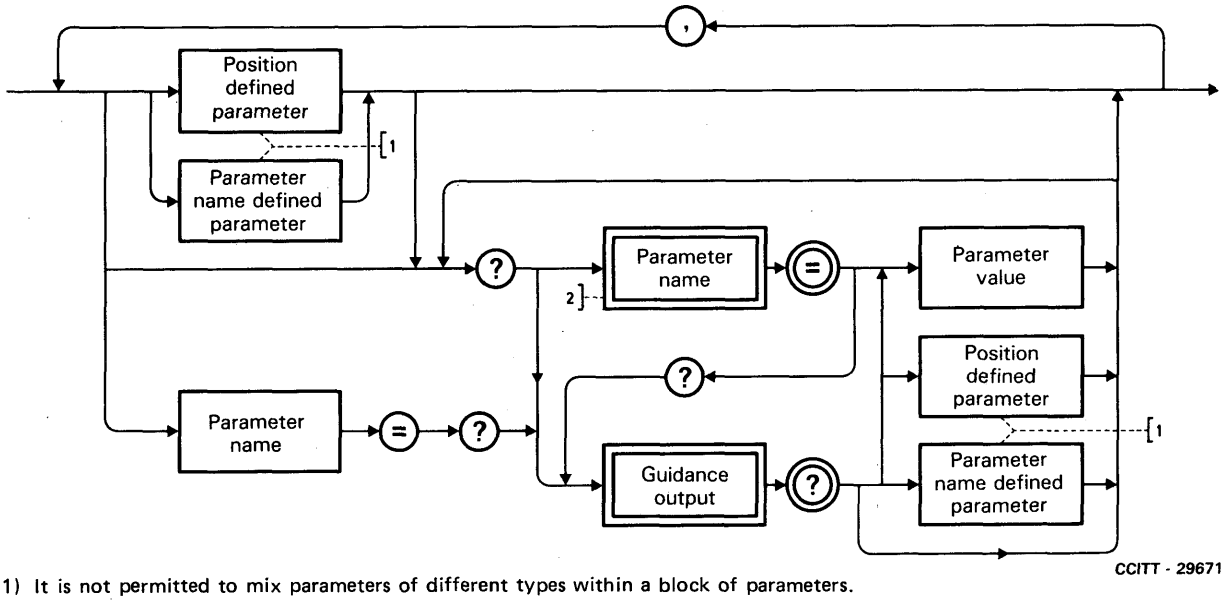


3.6 *Direct parameter input*



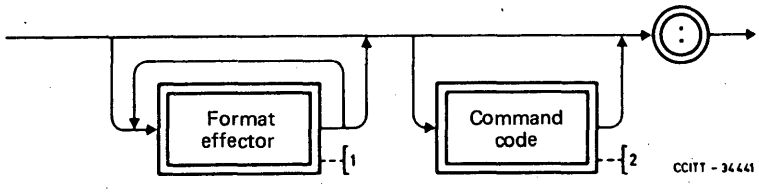
1) Only if command code is not valid.
2) Command without parameters or with default parameters only.
3) First command of a continuation series.
4) Subsequent command of a continuation series.
5) Last command of a continuation series.

3.6.1 *Parameter block entry sequence*



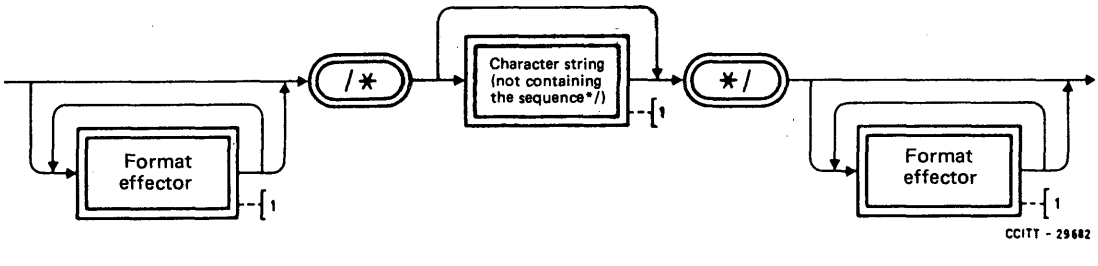
1) It is not permitted to mix parameters of different types within a block of parameters.
2) See Recommendation Z.315.

3.6.2 *Parameter block request indication*



1) Not further expanded in diagram form.
2) See Recommendation Z.315.

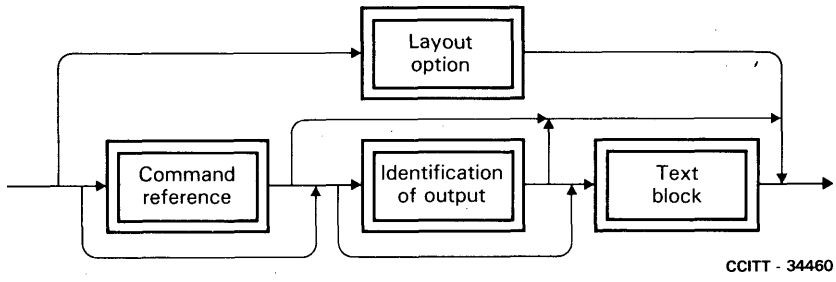
3.6.3 *Guidance output*



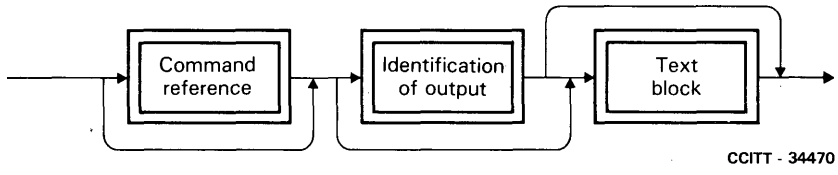
1) Not further expanded in diagram form.

3.7 *Response output*

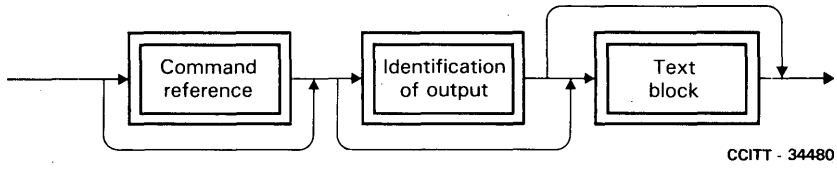
3.7.1 *Acceptance output*



3.7.2 *Rejection output*



3.7.3 *Request output*



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).

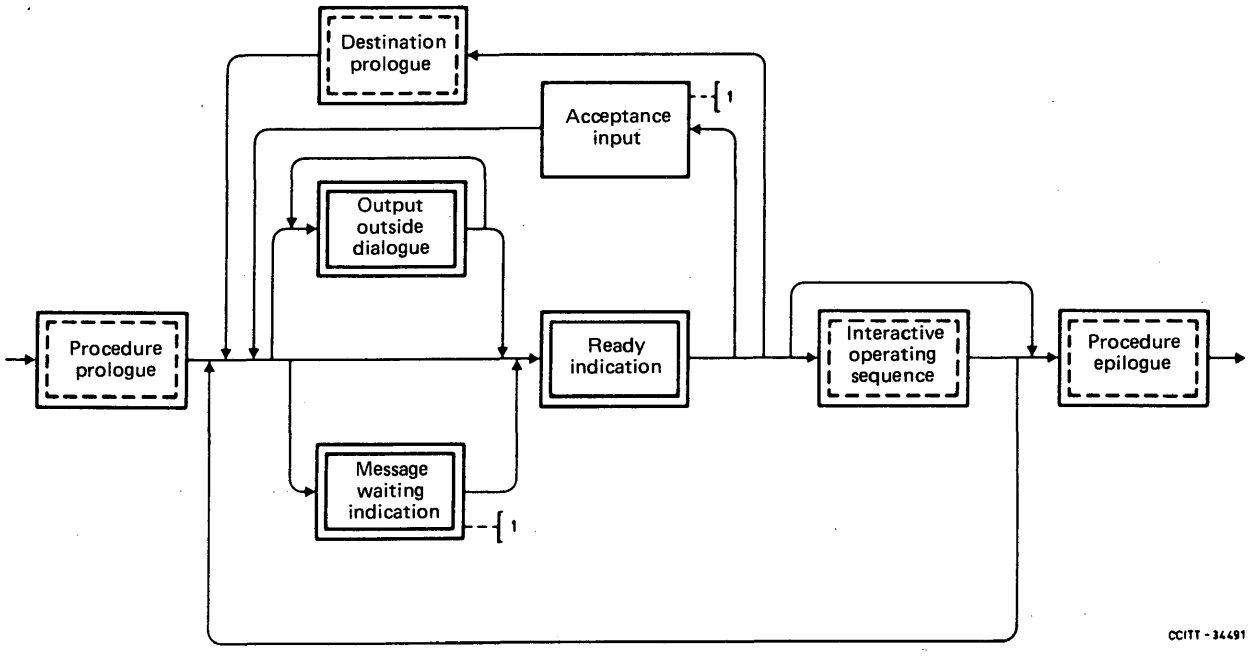
Optionally a system may choose to output the current output only up to the end of the current unit of output before outputting a waiting high priority output.

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[1]. 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*



1) Not further expanded in diagram form.

1) Interruption in other places is not excluded.

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.

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.

SECTION 3

EXTENDED MML FOR VISUAL DISPLAY TERMINALS

Recommendation Z.321

INTRODUCTION TO THE EXTENDED MML FOR VISUAL DISPLAY TERMINALS

1 Scope of the Section

This Section deals with man-machine interfaces that take advantage of the input and output facilities usually available on visual display terminals (VDTs). The procedures described are not necessarily confined to this type of terminal; they can also be applied to printer-oriented terminals, such as teletypewriters, within the limits imposed by the facilities available at those terminals, e.g. information entry through menu selection.

By maintaining consistency with Recommendations Z.311-Z.317, these Recommendations facilitate a transition from a man-machine interface using basic syntax and dialogue procedures as described in Section 1 to one based on VDTs.

Diagrams and examples are used to clarify and illustrate the concepts explained in the text. The diagrams do not include exceptional cases and do not specify all possibilities available with the extended MML; those not shown diagrammatically, but which are allowed in the text, are subjects for further study and are not excluded from the extended MML. Similarly, the examples shown are not intended to imply a particular system implementation.

The Recommendations cover aspects of VDTs that users see and use, e.g. data entry, data display, interactive control, user assistance, etc. Specific terminal characteristics are avoided wherever possible.

2 Organization of Section 3

Section 2 consists of the following Recommendations:

Z.321 Introduction to the extended MML for visual display terminals.

Z.322 Capabilities of visual display terminals.

Z.323 Man-machine interaction.

Recommendation Z.322 describes many of the capabilities currently available in VDTs. *Recommendation Z.323* focusses on actual man-machine interactions (i.e. *how* the capabilities are used) by addressing various aspects such as dialogue elements, monologue outputs, user assistance and interactive control.

3 Human factors

3.1 *The human factor view of the man-machine interface*

Human factor science characterizes the man-machine interface as any part of a system that the user comes in contact with – either physically, perceptually or conceptually. The user's conceptual model of a system is the knowledge that organizes how the system works and how it can be used to accomplish tasks. The conceptual model forms an integral part of the user interface.

3.2 *The need for human factors considerations*

The aim of human factors is to satisfy the largest possible proportion of potential users rather than to tailor the system to one user, particularly one with a detailed and sophisticated knowledge of the system. Therefore a proper man-machine interface takes account of the user's needs as well as system requirements. Poor quality will show up as a high proportion of input errors, loss of user confidence and motivation and high training costs. A high quality man-machine interface is based on a truly representative user model.

Recognized human factors literature has been used in the formulation of Recommendations Z.322 and Z.323. Where appropriate, human factor aspects have been incorporated into the texts.

Recommendation Z.322

CAPABILITIES OF VISUAL DISPLAY TERMINALS

1 Introduction

This Recommendation describes some of the capabilities which are important to the user and which are commonly available on interfaces based on VDTs. It is not an exhaustive list of capabilities. The use of additional capabilities, not covered in these Recommendations, is not precluded. Not all of the capabilities described need be present on a given system. Graphics capabilities are for future study and are therefore not considered in detail in these Recommendations.

System implementation of these capabilities may vary, depending for example on the degree of intelligence in the terminal itself and the allocation of responsibility for the man-machine interface among system components.

Items covered are treated from the point of view of the importance of their characteristics for designing the man-machine interface. Therefore, in general, human factors are dealt with individually for each item.

2 Screen

2.1 *Character definition*

Under study in Question 10/II.

2.2 *Character repertoire*

For further study.

2.3 *Cursor*

The cursor is important in the operation of an alphanumeric display because it directs the user's attention to that position on the screen appropriate to the task at hand, e.g. where the next character will appear. The cursor also allows the user to conveniently specify the location on the screen where an entry or change is desired by the user.

A set of general qualities required of a cursor include:

a) easily found by the user at any character position in the display;

b) easily tracked as it is moved through the display;

c) does not interfere with the reading of the symbol that it marks;

d) should not be so distracting as to impair the search for unrelated information displayed elsewhere on the screen;

e) should be of a form that is unique and reserved for that purpose only;

f) should be stable in respect to the position to which it is addressed until it is readdressed elsewhere as a result of user or system action.

2.4 *Screen partitioning*

A screen of a visual display terminal is physically partitioned by the following definitions.

2.4.1 *Visible display*

The visible display is the entire physical screen of a VDT (see Figure 1/Z.322).

2.4.2 *Border area*

The border area is that part of a visible display which is physically unavailable for displaying or entering data (see Figure 1/Z.322).

2.4.3 *Display area*

The display area is that part of a visible display which is available for displaying or entering data (see Figure 1/Z.322).

2.4.4 *Window*

A window is a part of the display area (sometimes the entire display area) used for entering and/or displaying functionally related data.

The limits of a window must be obvious to the user. There may be one or several windows within the display area (see Figure 1/Z.322).

The most important attributes are:

- its position in the display area may vary in time;
- its size may vary over time;
- one window may overlap another, either hiding some information, or share the same information between the windows;
- scrolling may occur within a window independently of the rest of the display area;
- each individual window may be associated with an independent activity.

2.4.5 *Field*

A field is a part of a window (sometimes the entire window), which is used for entering and/or displaying data.

The most important attributes, which may vary in time, are:

a) its position within the window;

b) its size: height and width;

c) its type:
   - for entering information (input field): accessible in writing by user and system (e.g. default value);
   - for displaying information (output field): inaccessible in writing by user.

The limits of an input field must be obvious to the user. There may be one or several fields within one window (see Figure 1/Z.322).

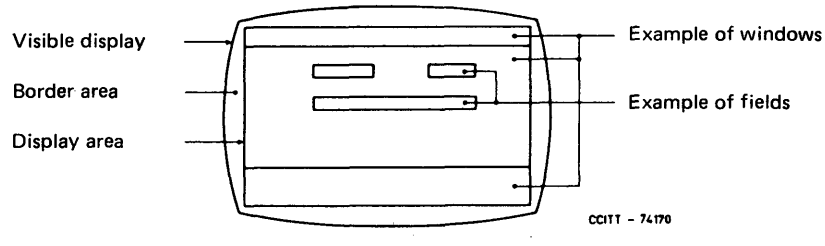


FIGURE 1/Z.322

Screen partitioning

2.5 *Display capacity*

The display capacity of a VDT is described by:

- display area size;
- display area format;
- display memory.

2.5.1 *Display area size*

The display area size is specified in terms of height and width dimensions.

It is smaller than or equal to the visible display.

2.5.2 *Display area format*

The display area format is specified in terms of the number of lines and the number of columns of characters.

A visual display terminal with a given display area size may use varying display area formats depending on the selected character size.

2.5.3 *Display memory*

The display memory contains information, some of which is displayed somewhere in the display area (see Figure 2/Z.322).

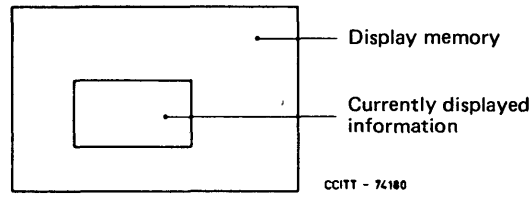


FIGURE 2/Z.322

Conceptual view of display memory

2.5.4 *Display memory recall functions*

The display memory recall functions are control functions (see § 7) used for recalling information from the display memory not currently displayed in the display area.

When these functions are used within a window, the window remains fixed in the display area.

The specific functions which may be accessed are:

a) in which direction, up or down, to move in order to display information not currently displayed[1];

b) how the movement is to be made:
   - stepwise; an increment may be a window or a line;
   - continuous (smooth).

Different terminals may employ different techniques in order to move (vertically or horizontally) information not currently displayed:

- windowing,
- scrolling.

The difference between them is best explained by giving an example. In the example, function keys are used to access the functions and the movement is made vertically and stepwise by line. What differs in the two techniques is what happens when for example an "up-action" is desired. Figure 3/Z.322 gives a view of a window in the display area with information inside before and after pressing the window-up key.

---

[1] The application of horizontal movement and combinations of horizontal and vertical movement needs further study.

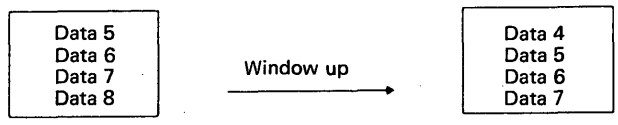


FIGURE 3/Z.322

Example of windowing

Figure 4/Z.322 gives a view of a window in the display area with information inside before and after pressing the scroll-up key.

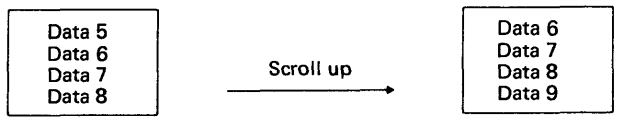


FIGURE 4/Z.322

Example of scrolling

The "windowing up" is physically identical as "scrolling down". The difference is just the direction of the arrow on the key performing the movement.

2.6 *Physical characteristics*

Under study in Question 10/II.

2.7 *Video attributes*

Video attributes are used to emphasize certain important information, e.g. a title, a message, a chosen item, in order to attract the attention of the user. Video attributes work on the characters of the information shown within an entire window, a part of a window, an entire field or within just a part of the field.

The following video attributes may be provided singularly or in combination:

2.7.1 *Luminance*

Under study in Question 10/II.

Information can be displayed in different levels of luminance.

2.7.2 *Colour*

Information can be displayed in different colours.

2.7.3 *Flashing*

Information can be displayed alternately as normal characters and as spaces in the prevailing background colour.

2.7.4 *Underline*

Information can be displayed with underlined characters. However, this type of video attribute might make it difficult to observe the cursor on terminals where the underline character is used as the cursor.

2.7.5 *Size*

Information can be displayed in different character sizes.

2.7.6 *Font*

Information can be displayed in different fonts, e.g. italics, bold.

2.7.7 *Inverse video*

Information can be displayed by inverting the image of the characters, such as going from light characters on a dark background to dark characters on a light background.

2.7.8 *Concealment*

Information can be displayed as space characters, e.g. secret parts of a password.

3 Other output devices

For further study.

4 Keyboard

4.1 *Key characteristics*

Under study in Question 10/II.

4.2 *Keyboard repertoire*

For further study.

4.3 *Physical characteristics*

Under study in Question 10/II.

5 Other input devices

For further study.

6 Transmission characteristics

There are two fundamental transmission mechanisms commonly employed and referred to as "character mode" and "block mode".

If a terminal uses character mode transmission, each and every character input at the keyboard is sent to the controlling processor one at a time. Thus, as is the case with the syntax of Recommendation Z.315, if certain regular keys have special meanings ascribed to them e.g. ; or !, then they can act as specific triggers to the controlling software which then performs some process on the preceding information in accordance with the given syntax rules.

If the same terminal uses block mode transmission, all of the regular typewriter keys and some of the special purpose keys only have an effect local to the terminal, i.e. the information input goes into the "memory" of the terminal and onto the screen normally, but not to the controlling processor. The implication is, obviously, that special actions assigned to these keys do not get processed until an explicit "send" action is made. A "send" action by the user is only required when information is to be moved from the terminal to the host processor.

The important point for the purposes of these Recommendations is that the use of a "send" key is not explicitly shown at any time. It is recommended that systems that employ "block mode" transmission either convey very explicit instruction on when a "send" action is required of the user or are designed to be able to accept and respond intelligently to incomplete input, i.e. "send" can be used by the user at any point without a fundamental disruption of the dialogue. As far as possible this will shield the user from the effects of the transmission mode employed.

7 Control functions

Control functions are those functions related to the man-machine interface that are applied by the user independently while in a dialogue with the system functions. Control functions have no direct impact on the system functions. Control functions are subdivided into cursor control functions and interface control functions.

7.1 *Cursor control functions*

A cursor is generally used as an indicator of the position where an action will take place, such as a character being written on the screen – either by the system or by the user. Cursor control functions do not directly affect the overall system state, but assist users in selecting data entry fields, editing fields, etc.

Examples include:

a) *Home position of the cursor*

Here "home" means a position in the display area to which the cursor can be consistently moved, from any position, by a single keystroke. The actual position in the display area which represents "home" may vary according to the activity being performed and the current layout of the display area.

b) *Movement control of the cursor*

Assuming that the VDT used supports direct cursor addressing, the following types of cursor movement are possible:

i) by the system, and

ii) by the user via cursor control functions. General cursor control functions independent of dialogue are:

- one line up;
- one line down;
- one place left;
- one place right.

Ideally, cursor movement should be easy to accomplish by means of a single, dedicated key for each function. Shifted characters should be avoided. If a cursor positioning control key is used, it should repeat when held down. Cursor movement may also be controlled by other input devices, e.g. light pen, trackball, mouse or joystick.

When cursor positioning is incremental by discrete steps, the step size of cursor movement should be consistent in both right and left directions and both up and down directions. However, the cursor may bypass inaccessible fields.

When character size is variable on the display, incremental cursor positioning should have a variable step size corresponding to the currently selected character size.

7.2 *Interface control functions*

Functions of this class are used to force specific actions relating to the interface. They are invoked by various means, including pressing dedicated control keys.

Examples of man-machine interface control functions include, but are not limited to:

- send (other words for the same function are "transmit" and "enter") [see § 6];
- editing control functions (insert character, insert line, replace character, etc.);
- capitals lock (the condition where letters are input as capitals only);
- display memory recall functions [see § 2.5.4];
- select different font [see § 2.7.6];
- select different character size [see § 2.7.5].

MAN-MACHINE INTERACTION

1 Introduction

This Recommendation describes *how* interactions should take place between the user and the system from a logical viewpoint. It describes how an effective man-machine interface should appear to the user when utilizing the capabilities of VDTs as described in Recommendation Z.322. This Recommendation supersedes Recommendations Z.311-Z.317 for interfaces based on VDTs, referencing parts of them where appropriate. Specific human factor guidelines are included within the appropriate divisions of the text.

The capabilities of VDTs, e.g. multiple windows, inverse video, etc., when used consistently can lead to a more effective man-machine interface. Additional dialogue procedures are possible and often preferable with VDTs, e.g. using different windows for different functions. Likewise, the transient nature of information presented on a screen may affect the selection of information display and the manner of presentation. The terminal capabilities available must be considered in conjunction with guidelines presented in this Recommendation in order to produce the most effective interface.

Many advances in the state of the art of man-machine interface design are incorporated in Recommendation Z.323. However, the use of graphics capabilities have not yet been considered in any detail in these Recommendations and must be studied further. The needs of the user moving between different systems or different types of terminals are best facilitated by ensuring that capabilities are used consistently and that user assistance is an integral part of interface design. Interfaces designed according to the principles outlined in this Recommendation will tend to be more user friendly, effective interfaces.

2 Common aspects

2.1 *Data display*

Data display is the presentation of information by the system to the user. During a dialogue the number, dimension and position of fields and windows in the display area may be changed. Not all fields and windows need necessarily display information at any one time.

Visual display terminals facilitate information entry through menu selection and form filling. Since presentation of more information at one time might cause confusion, care must be taken to label information clearly, to keep displays simple, to highlight information consistently and in moderation, and to maintain a consistent information layout as far as possible.

2.1.1 *General guidelines*

The layout of output is dependent on what type of data is presented. There are three basic types, combinations of which are possible:

- textual data;
- numeric data;
- tabular data.

a) *Guidelines for textual data*:

- text should be written using upper and lower case letters;
- abbreviations should not be used if confusion might be caused;
- plain text should be used rather than codes.

b) *Guidelines for numeric data*:

- strings of more than five numeric characters may be presented in groups of two to four;
- standardized forms should be used.

c) *Guidelines for tabular data*:

- in case of lengthy columns, spacing between about every five items improves readability;
- items which are related to each other should be placed close together;
- figures arranged in columns are easier to compare than figures arranged in a row;
- integers should be right justified;
- numerical entries with decimals should be justified with respect to a fixed decimal point position;
- text and labels should be left justified;
- if any text continues on another line, it should begin in the same column as the text above.

2.1.2 *Accessible and inaccessible parts of the display area*

VDTs provide the capability to characterize some fields of the screen as accessible for writing by the system only, some other fields accessible for the system and the user.

The fields used for the display of headers, of parameter identities, of output, of delimiters, etc., should be accessible for writing only by the system (output fields). The fields used for the input of parameters should be accessible both to the system and to the user (input fields). The system can highlight these fields, for example, by underlining to distinguish the field or a default value, if appropriate. The user can access the field to input the desired value(s), to edit the previous input value(s), or to edit the offered default value.

The user may attempt to write into a field reserved for the system. This should not be allowed, an indication should be sent to the user and the input characters should be ignored. The type of this indication depends on the terminal facilities and may be an audible or visible signal. However, the terminal shall immediately recover from this situation so that the user can proceed.

2.1.3 *Highlighting*

Highlighting is used to emphasize visually a portion of a display area to make it stand out from adjacent portions, i.e. to call the viewer's attention to it. It should be used consistently and in moderation. In particular, care should be taken not to confuse or otherwise overload the user by highlighting.

There are a number of areas where highlighting may be applied, such as:

- defaults in forms;
- optional information entry in forms;
- indication of system irregularities and their urgency, etc.

There are a number of possible highlighting techniques, such as:

- different levels of luminance;
- colour;
- flashing;
- underlining;
- different character sizes or fonts;
- small or capital letters (lower or upper case);
- pointing with arrows, asterisk, etc.;
- inverse video;
- combinations of the above.

Some guidelines that should be followed in all applications of highlighting are:

a) when colour screens are used:

- in order to reduce problems for colour-blind users and to facilitate a transition between colour and monochromatic terminals within the same system, colour should normally be used in combination with some other means of distinction. Note also that some colours may have psychological associations, perhaps depending on the cultural tradition of a nation, e.g. red for danger, green for proceed;
- be consistent in the use of colour;

- the number of colours with specific meanings should be limited. Associating meanings with too many colours may confuse the user;
- colour combinations should be chosen such that there is sufficient contrast in hue and density wherever two colours meet. This is particularly true in the case where a text is displayed over a colour background;
- colour combinations should be chosen with care, as many combinations can be displeasing to the eye;

b) use only one level of luminance in addition to the normal level when highlighting. Variations in room lighting, specific VDTs and user perceptions make it unlikely that more than two levels will be universally distinguished;

c) when using more than one highlighting technique, do not highlight more than 30% of the display. If everything is highlighted, even differently, then nothing is highlighted;

d) since flashing attracts much attention, its use should be restricted to special applications, e.g. alarms. Once the user acknowledges the perception of the flashing information, the flashing should be stopped;

e) if the user needs to read text from a flashing area, the flashing should be slow in order to make the text readable. An alternative would be flashing pointers, pointing to the text area of importance;

f) in one system, or at least in each job area, highlighting facilities should be consistently applied.

2.1.4 *Information layout*

A user should always be able to recognize at first sight:

- where parameter input is desired in a form;
- where system response is expected;
- where the system status is displayed;
- where user guidance is expected, if requested;
- where menus are displayed.

Therefore, the information layout, when determined by the system, should follow common rules in such a way that information of certain categories will be displayed in certain portions of the display area.

The information layout should be consistent in any one system. Information, which is not necessary in certain job areas, may be omitted.

The following basic windows of a display area can be distinguished:

- the *system status window* indicates the global status of the subsystems, e.g. by highlighting whether the subsystems are working regularly or are affected by some fault. In some applications this window may be omitted.
- the *dialogue window* (see also § 3.5.1) is used for display of information output by the system and input by the user during a dialogue. This window normally is the most essential portion of the screen for a man-machine dialogue. Therefore, it normally occupies the largest part of the display area.
- in addition, other windows are possible, e.g. function key label window.

2.2 *Input editing*

Editing mechanisms can be used to correct erroneous input during data entry or to change previously entered input in order to resubmit it.

Several possibilities of editing can be distinguished, including the following:

- delete last character or last n characters;
- delete or overwrite last field;
- delete or overwrite arbitrary fields;
- insert characters.

Editing mechanisms may be dependent on the facilities of a terminal, such as function keys.

2.3 *Response time*

In a system operating normally, response output (see Recommendation Z.317) to a command should be presented to the user within a psychologically acceptable time limit, normally taken to be of the order of two seconds after input. For any given type of command, this time limit should be as uniform as possible in order to meet the expectations of the user.

Depending on the nature of the command, two types of response output can be distinguished:

a) that which conveys the results of the execution of the command;

b) that which concerns the acceptance only of the command, results being communicated to the user by output outside dialogue.

Response output concerning user errors should be given to the user as soon as possible. Although a fixed rule cannot be defined, the following guidelines can be given:

- syntactical errors must be discovered very early by the system; the response time should be within the psychologically acceptable time limit;
- semantic errors can sometimes be discovered early, sometimes late, depending on the type of command and on the nature of the error; normally the feedback should be given to the user as soon as the error is detected;
- semantic errors in pre-scheduled jobs should be indicated to the user either immediately after the command input, if this is possible, or at the time the result is expected.

2.4 *Directives*

The presentation of system output in the form of guidance output, menus, form output, waiting system reports, next page, etc., can be controlled by means of input statements called directives. It is possible to qualify the effect of directives either by the use of context or by the use of additional parameters.

Directives are used to direct the system to present information rather than to execute a command; they can also be used in the interaction between the user and the system prior to command execution.

Directives can be given to the system by a word, e.g. HELP, by a special character, e.g. "?" (question mark) or by a dedicated function key.

Directives can never cause any change in the state of the system. This distinction from commands is made to encourage users to make full use of such facilities without fear of altering the system unintentionally.

2.5 *Help facilities*

Help facilities comprise the availability of help output such as guidance output in direct information entry or clarifying text in menu item selection and form filling. Help output can assist the user in deciding to input a certain command or string of commands to perform a certain job.

Help output is displayed at the user's request by the use of directives.

Examples of different types of information that could be obtained in a help output are:

- how to obtain more specific help. A single help output at the highest level of simplicity might be displayed when the user enters a directive without any parameters, and the precise nature of help required is unclear from the context;
- general principles of dialogue procedure;
- what jobs can be performed;
- a description of either classes of commands or a single command in detail. The user must specifically request that such output be displayed, either from the highest level of help output or via the parameter on the help directive;
- how the job is performed without actually executing it;
- what the user has done so far;
- what kind of entry the system expects from the user, e.g. possible commands, range of a parameter value, example of a correct parameter entry;
- the meaning and consequences of forms, commands, menu items, etc., which are displayed on the screen;

- the syntax or a short explanation of a specific command or job;
- a short description of a specific parameter, e.g. its default value or the permitted range of values.

In order to make the help facilities as effective as possible, the following guidelines can be given:

- help should be available in a consistent manner throughout the system;
- there should be various levels of help available so that both novices and experienced users can quickly obtain the required information;
- unnecessary codes and abbreviations in help messages and in explanations should be avoided;
- where multiple pages of help are available, it should be possible to have any page displayed without having to display intervening pages;
- help messages should preferably not overwrite data, error information or user commands and vice versa. In cases where this is unavoidable, a simple mechanism should be provided to retrieve the original information;
- the type and level of detail of help information provided should be consistent with the anticipated needs of the user at any particular stage of a dialogue. For instance, a "help" request made prior to inputting anything at a terminal could result in a high level introduction to the man-machine interface facilities, whereas a "help" request made instead of inputting a parameter value could result in detailed information on what possible values that parameter could have, and perhaps what each value means;
- in a structured help system, it should be possible for a user to request directly the exact level of detail required without having to step through intermediate higher level information.

2.6 *Defaults*

In some applications the normal and most frequently used input can be predicted by the system. Default values which can be considered critical in the sense that they may create situations dangerous to the system integrity should be avoided.

2.6.1 *Use of default values during data entry*

To make the user's work easier, input of the most frequently used parameter values may be prepared by the system. If this offer does not match the user's desire, the possibility to overwrite the default must be open.

An offered default can be accepted by the user, either by active selection such as pressing a dedicated function key or by passive selection, i.e. without taking specific action.

The overwriting or deletion of defaults can be done by editing mechanisms as described in § 2.2.

2.6.2 *Display of defaults during data entry*

The main reason for using defaults is to simplify the user's information entry to the system.

To achieve this, defaults should be offered by the system and may be highlighted as described in § 2.1.3, so that it is obvious to the user which data entry area he has filled himself and which has been filled by the system. The highlighting technique should be consistent in a system or at least in a certain job area.

2.7 *Input error handling*

2.7.1 *Input error information*

In the event of erroneous input, some form of input error information, normally in the form of request output (see Recommendation Z.317), must be presented to the user.

Ideally, input error information would contain:

- where the error was detected;
- what kind of error was made;
- how to recover from it or at least how to find a way to recover from it.

In some cases it may be difficult to supply the user with all this information.

In many cases the input error information may be self-contained, in other cases reference may be made to other sources of information.

The length and detail of the message should be proportional to the nature of the error; the user should not have to look at a long explanation for a simple error.

Coded messages and intimidating jargon such as "syntax error" should be avoided. Messages should be polite and should not patronize or insult the intelligence of the user.

When an error is detected and error information is displayed, the field containing the error may be highlighted.

2.7.2 *Location of error information*

Error information should always appear in a consistent manner on the screen. This should be common within one system or at least within one job area.

2.7.3 *Multiple errors*

Multiple independent errors in one data entry should, if possible, be reported together at one and the same time.

Incidences of conflicting combinations of parameters or parameter values should be treated by the error information as a single subject.

2.7.4 *Correction of errors*

Following detection of an error situation, the user should be provided with mechanisms to correct the erroneous input. Such mechanisms could include:

- the system placing the cursor on the erroneous field and requesting input;
- the user addressing the field, e.g. by name, number or lightpen, or cursor control keys or joystick to get to the field(s) which needs to be changed.

The erroneous information should remain on the screen until it is corrected.

3 Dialogue procedure

3.1 *General*

In the general description of the dialogue procedure, aspects of error correction and of help request are not included. These topics are treated in the detailed descriptions of the specific dialogue elements. For examples of dialogue procedures, see Annex A.

3.1.1 *Structure*

The dialogue procedure is depicted in Figure 1/Z.323.

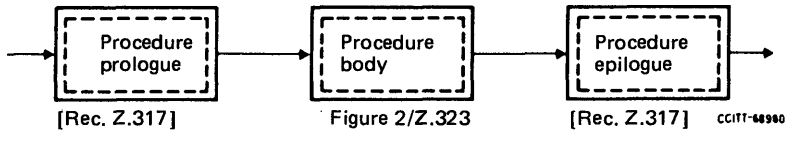


FIGURE 1/Z.323

Dialogue procedure

The dialogue is divided into three main parts:

- prologue;
- body;
- epilogue.

For the procedure prologue and the procedure epilogue, refer to Recommendation Z.317. The procedure body is depicted in Figure 2/Z.323.

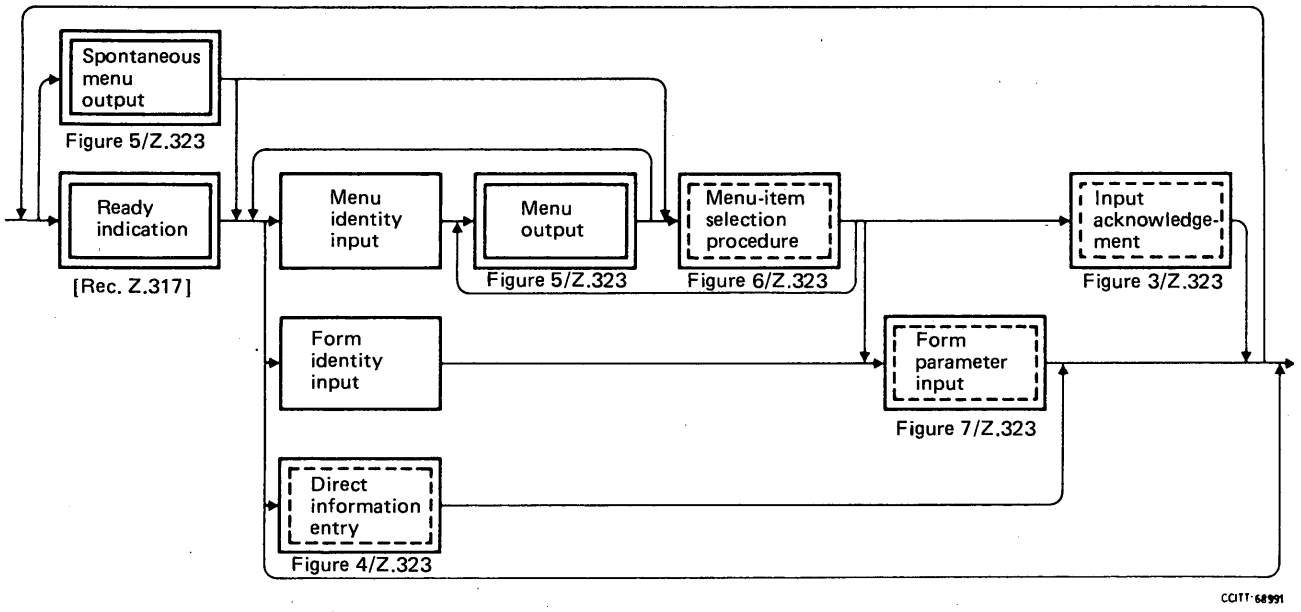


FIGURE 2/Z.323

Procedure body

3.1.2 *Dialogue elements*

In the CCITT MML, three different dialogue elements can be distinguished with respect to the method of entering information into the system via a man-machine terminal:

- direct information entry;
- information entry through menu-item selection;
- information entry through form filling.

Information entry can be accomplished exclusively by one of the dialogue elements or – if a system supports more than one dialogue element – by a combination of elements, e.g.:

- menu-item selection and direct information entry;
- menu-item selection and form filling.

3.1.3 *Selection of dialogue elements*

Choosing the right dialogue element depends very much on the nature of the job to be performed and the experience of the user. Often there are many different job areas that the user could deal with during his session at the terminal and the best method, for an inexperienced user, when selecting a job area, and then a specific job in this area, may be to use menu selection(s).

The experienced user would probably prefer a more direct method to reach a specific job, but will also use menu-item selection(s) when performing jobs that are infrequently used. Therefore the availability of both dialogue elements is attractive.

For maintenance staff who gain access to a system via the public switched telephone network with a simple portable terminal, it may not be possible to use every dialogue element due to restrictions imposed by the terminal characteristics.

Directives may be used for selecting dialogue elements. They may be either abbreviated menu or form identities, or function keys. The abbreviated menu or form identities need to be uniquely distinguishable from command codes, e.g. an abbreviated form identity could consist of a command code terminated by a question mark.

If direct information entry is available besides other dialogue elements, then direct information entry should always be possible after output of a ready indication or a menu. This may or may not require the use of a directive.

It should be possible to enter an allowed command or destination identifier even if a displayed menu does not contain it.

3.1.4 *Start and end of information entry*

The system invites the start of information entry by the output of:

- a spontaneous menu (one that is automatically given) and/or
- a ready indication.

The spontaneous menu given may be different depending on the authority of the user or the terminal involved. Any menu can always be requested by the use of a directive.

Completion of information entry always results in an Input Acknowledgement as is shown in Figure 3/Z.323 or in an appropriate error treatment.

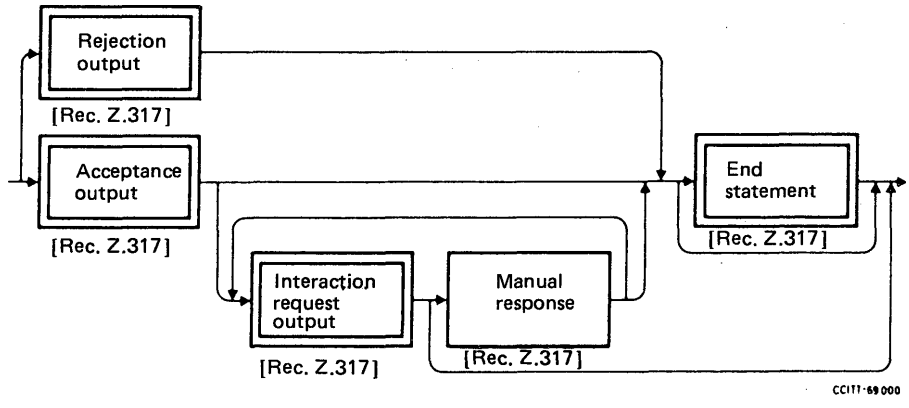


FIGURE 3/Z.323

Input acknowledgement

As in Recommendation Z.317, an Acceptance Output may be followed by an Interaction Request Output.

3.1.5 *End of input indication*

In all dialogue elements the user may need to mark the end of input in order to have the information interpreted by the system. This can be done by some special indicators (see Recommendation Z.314) which contain an implicit end of input indication or by special function keys, e.g. "send". If more than one dialogue element is provided in a system, the end of input indication should be consistently used for all dialogue elements.

3.2 *Direct information entry*

Direct information entry can apply to any area of application of the CCITT MML.

The direct information entry, recommended for operation and maintenance, installation and acceptance testing of SPC systems, consists of two sub-elements:

- destination prologue;
- interactive operating sequence.

See Figure 4/Z.323.

For both sub-elements, refer to Recommendation Z.317.

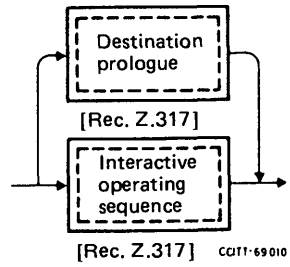


FIGURE 4/Z.323

Direct information entry

3.2.1 *Information entry*

Direct information entry may comprise:

- destination identifier to enable the destination of the information entered subsequently to be changed;
- command code to identify the type of activity to be executed;
- parameter values necessary to allow execution of a requested action;
- manual response as a part of an entering procedure requiring hardware manipulation such as operating switches, replacing equipment, etc.

These aspects are specified in Recommendations Z.315 and Z.317.

3.2.2 *Execution of a command*

A request to execute a command will eventually lead to acceptance or rejection output, refer to Recommendation Z.317.

3.2.3 *User assistance*

3.2.3.1 *Help facilities*

A request for help may result in guidance output.

3.2.3.2 *Guidance output*

Guidance output is in general related to a command and contains information such as:

- the complete block of parameters to be input for a specific command;
- that part of the block of parameters that is still to be input;
- the parameter next to be input;
- the fact that the complete parameter block has been entered and a request to execute a command can be given.

3.2.3.3 *Error correction aspects*

Input error information can be contained in guidance output or in request output (refer to Recommendation Z.317 and to § 2.7).

3.3 *Information entry through menu-item selection*

The essential advantage of menu-item selection as a way of interaction is that it can remove memory load from the user. The items available are laid out for inspection, and the way in which each item may be selected is obvious.

The task of performing any transaction using a menu is thus reduced to:

- scanning the items;
- finding the required item (if already known by the user), or deciding which item to choose (if not already known by the user);
- selecting an item.

The use of menus is particularly appropriate for applications where there will be many casual users or where there may be frequent interruptions to work at the terminal, and for activities which are infrequently performed.

Menus may be used as a means to arrive at a command code, to select a new destination or to assemble and to execute a command with all its relevant parameters. The system outputs a list of items (the menu output), from which the user can select the appropriate item. In a menu selection procedure, a selection of items from subsequent menu outputs may be needed.

3.3.1 *Display of the menu output*

The menu output (see Figure 5/Z.323) may contain several types of information:

- menu identity;
- menu items;
- additional information.

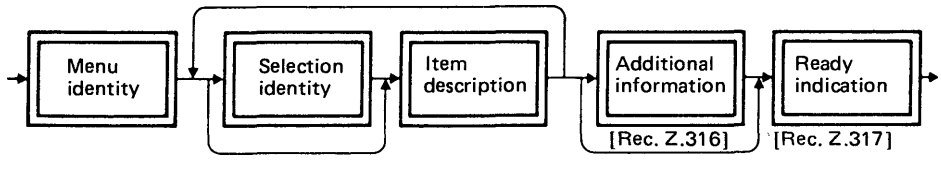


FIGURE 5/Z.323

Menu output

The information can be displayed in fields and/or given by highlighting techniques.

The *menu identity* is displayed in a field at the head of the menu. It identifies the menu, preferably in a concise meaningful manner to allow an easy recognition of the nature of the menu.

A *menu item* is displayed in a field that contains a brief description of the item and an optional selection identity. By inputting such an identity, a choice can be made. The selection identity should be displayed at the left side of this field.

The *additional information* is intended to present more information to the user in order to aid the selection of an item from the menu, e.g. the text "Enter choice".

The menu layout in the window should be consistent throughout all the menus in a given system. Only one menu should be presented at a time, always displayed in its entirety.

3.3.2 *Item selection*

Refer to Figures 6/Z.323 and 2/Z.323.

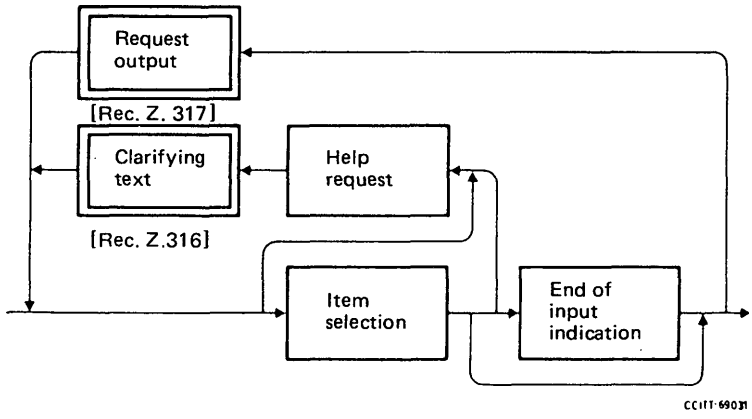


FIGURE 6/Z.323

Menu item selection procedure

The selection of an item can be done in two basic ways:

a) inputting the selection identity;

b) pointing at the item by using techniques such as cursor positioning, lightpen, touch screen, function key, etc.

Selection of more than one item from one menu is not permitted.

When using a hierarchy of menus, it may be helpful for the user to be able to return to the previous menu.

When the user notifies the system that he has made his selection, the system confirms the input by a new menu, a form output, or an input acknowledgement.

3.3.3 *User assistance*

3.3.3.1 *Help facilities*

During selection the user can ask for help at any moment. Besides, for general help information the user may ask for specific help information by inputting a specific help request.

The system reacts to the user with a clarifying text.

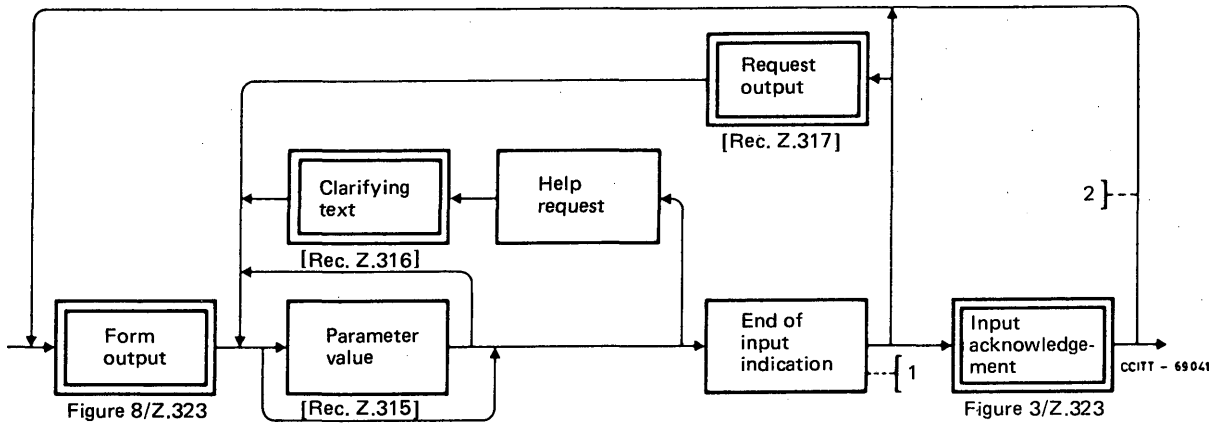
3.3.3.2 *Error correction aspects*

The system can ask the user to correct his selection if this is not valid. The response is given in the form of request output (refer to § 2.7).

3.4 *Information entry through form filling*

Form filling is a useful method of information entry when flexibility is needed, for example when optional as well as mandatory items of data are required by a command.

When this data entry procedure is to be used, the system first outputs a list of the parameters of the required command (the form output). After the form has been filled by the user with the necessary parameter values, the assembled data can be entered in the system for execution of the command. Refer to Figure 7/Z.323.



1) The end of input indication can be used to indicate whether execution, continuation or next page of form is required.

2) Further study is required to address proper provision of user timing control at this point.

FIGURE 7/Z.323

Form parameter input

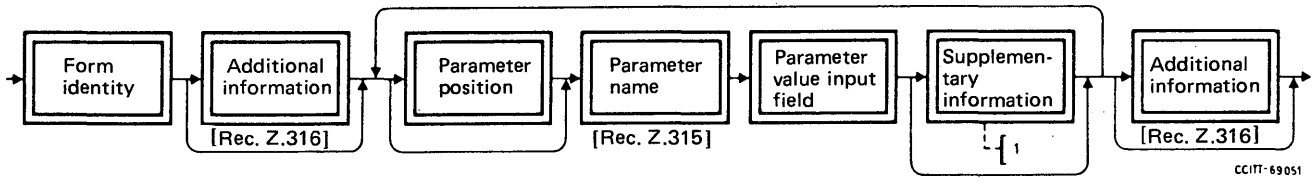
3.4.1 *Display of the form output*

The form output (see Figure 8/Z.323) may contain several types of information:

a) form identity;

b) per parameter:
   - parameter identity,
   - parameter value input field,
   - supplementary information;

c) additional information.



1) Supplementary information may be provided by highlighting the parameter value input field.

FIGURE 8/Z.323

Form output

The above information can be displayed in fields and/or given by highlighting techniques.

The *form identity* is displayed in a field at the head of the form. It identifies the command, preferably in a concise meaningful manner, to allow easy recognition of the nature of the form, and an optional identity for command reference.

The *parameter identity* is displayed in a field and contains the parameter name and an optional parameter position which could be used as a reference in a request output. The parameter position should be displayed at the left side of this field.

The *parameter value input field* is an accessible field. Initially this field is either empty and should be filled in by the user, or the system may display in this field the default value which can be overwritten by the user.

The *supplementary information* provides an explanation to the user, if required, to aid input of the parameter value. It may give information such as:

- whether the parameter is optional;
- in which form the value should be entered, e.g. in alphanumeric form.

The *additional information* presents general information to the user with respect to the whole form, e.g. guidance on how to submit the form to the system after finishing the input of parameter values.

The information applying to a particular parameter (parameter identity, parameter value and supplementary information) should clearly be associated with that parameter, i.e. co-located. The position of the fields in the displayed form should be consistent over the form. In any one area of application, they should be consistent from one form to another.

If punctuation is used for delimiting fields, the punctuation from the appropriate direct information entry technique should be used.

3.4.2 *Information entry*

The user can fill in the form by inputting the wanted values in the accessible fields. A displayed value, e.g. a default value in an accessible field can be changed by inputting a new value at the same position. Optional values can be omitted by not entering any data in that specific accessible field.

The syntax diagram of the parameter value is given in Recommendation Z.315.

3.4.3 *Execution of a command specified by a form*

The user should notify the system that he wants to enter the information contained in the form into the system. The user should be given the capability to request the system to continue to operate with the same form and/or subsequent forms. Mechanisms to control this capability are left for further study.

3.4.4 *User assistance*

3.4.4.1 *Help facilities*

During inputting parameter values the user can ask for help at any moment. Besides general help information he can ask for specific help information by entering a specific help request.

3.4.4.2 *Error correction aspects*

The system can request correction if one or more parameters do not pass the validity check and give some clarification on what kind of information is required. The cursor and/or highlighting can be used to indicate which value should be corrected.

The user can correct the parameter value by changing the value and re-entering the form contents to the system (refer to § 2.7).

3.5 *Dialogue window aspects*

3.5.1 *Dialogue window*

The dialogue window may be used to accomplish several tasks including:

a) output of optional session header information such as date and time, source identifier, user, etc.;
b) output of menus and forms;
c) direct information entry;
d) information entry through menu-item selection;
e) information entry through form filling;
f) output of response outputs;
g) output outside dialogue;
h) output of graphics information;
i) guidance output.

3.5.2 *Dialogue consequences outside the dialogue window*

In general, the effects of a dialogue cause changes in the dialogue window. However, command results may also influence the other windows, e.g. the status window. Also, the function key label window, if used, may be influenced by the dialogue, e.g. if function keys are used to select items from a menu.

4 Monologue output

A monologue output is any output from the system which occurs outside a dialogue. This includes output outside dialogue as described in Recommendation Z.317, system status and alarm information, function key labelling, date and time, etc. Usually, each type of monologue output occurs in an appropriate window on the screen. The occurrence of a monologue output may be accompanied by an audio signal or highlighting in order to stimulate user action, e.g. on alarms. In general, it is not helpful to output information on a VDT which is not immediately useful to the user.

4.1 *Output outside dialogue*

Output outside dialogue is a spontaneous output indicating a certain event, e.g. an alarm situation, or an output in response to a previously entered command, e.g. traffic measurement result. Output outside dialogue should not normally disrupt a dialogue in progress. There are several possible means of achieving this, e.g. message waiting indicators.

4.2 *System information*

System information is information related to the status of the system and may contain items such as:

- system status indicators;
- alarm indicators;
- message waiting indicator.

4.3 *Function key labels*

Function key labels may be displayed in the display area to inform the user as to what functions may be accessed via programmable function keys. They may be displayed as characters or symbols and with various highlighting techniques. It should be obvious to which function key each function key label is associated.

Consistency should be applied when assigning labels to function keys so that frequently occurring labels appear in the same position in the display area.

ANNEX A

(to Recommendation Z.323)

Examples of dialogue procedures

A.1 *General*

In § 3 of the body of this Recommendation (Dialogue procedure), a number of dialogue elements have been described and Figure 2/Z.323 showing how various inputs and outputs are related has been introduced.

The purpose of this annex is to clarify how the various elements interact. This is done by showing in a number of examples how the interaction between the user and the system appear to the user. Since the main part of this interaction takes place within the dialogue window, other windows have not been mentioned explicitly.

It is important to bear in mind that the examples are intended only to illustrate some of the possibilities described in the dialogue procedure in § 3 of the body of this Recommendation and that they are not to be considered as Recommendations.

Several alternatives exist for subdividing the dialogue window. For example the dialogue window may be divided into a session header window, a work window, an input window, and an output window. The last two windows can be combined. However, to better illustrate the concepts of input and output as represented in these Recommendations, separate input and output windows are used in the following examples.

The relative position of windows in the examples is shown in Figure A-1/Z.323. The relative sizes of the windows in this Figure are not significant, nor are the lines used to delimit the windows. The actual method of best distinguishing windows from each other, if necessary, is terminal dependent.

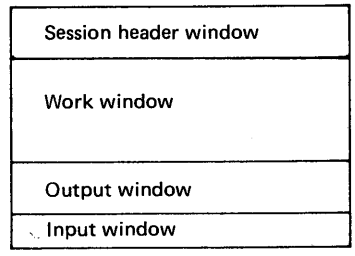


FIGURE A-1/Z.323

Dialogue window layout

The *session header window* (not shown in the examples) would be used to identify the dialogue in terms of date and time, source identifier, user, etc. It will normally occupy a fixed portion of the dialogue window.

The *work window* is used to display menus and forms.

The *output window* is used for response output and output outside dialogue.

The *input window* is used for the display of direct input and/or item selection. It can also be used for the display of entered directives. If applicable, the ready indicator is displayed in the input window.

The last three of these windows are used concurrently in the examples, but one window may disappear in favour of another. The window size depends on the dialogue element used and may consequently vary during a dialogue session.

Each example contains:

- a series of information layout figures, accompanied by some clarifying text, showing how the contents of the dialogue window (see Figure A-1/Z.323) change;
- a dialogue procedure body syntax diagram where the path of the selected interaction is indicated by bold connecting lines.

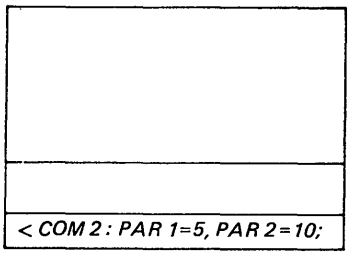
In order to aid comprehension of the information layout figures, the following assumptions are made:

- the user enters an end of input indication (not shown) between a figure showing input to the next one;
- the system clears the input window between two subsequent figures.

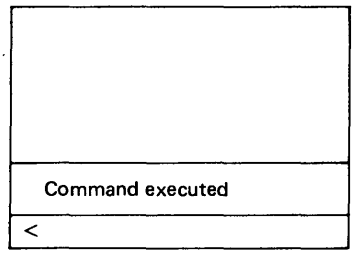
It should be noted that help requests and input error handling are not treated in the examples, i.e. it is presumed that all commands and directives are entered correctly. Each figure shows both the output from the system and the following input made by the user. The user input is written in italics in order to distinguish it from the system output. The sizes of the windows are not significant.

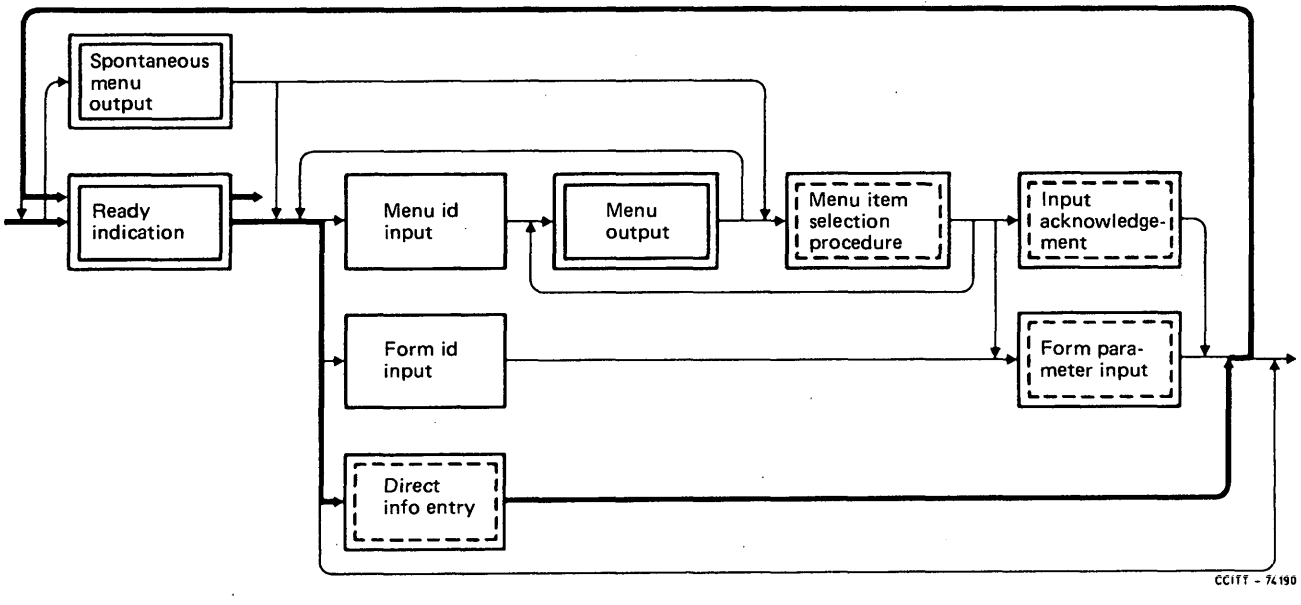
A.2 *Example 1*

1. The user knows the command code as well as the parameters and enters the entire command by direct information entry.

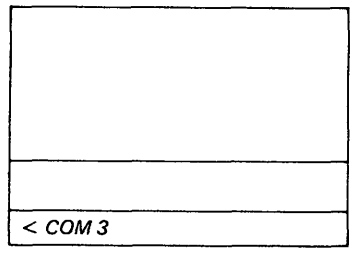


2. An acceptance output is displayed and the system is ready for the next input.

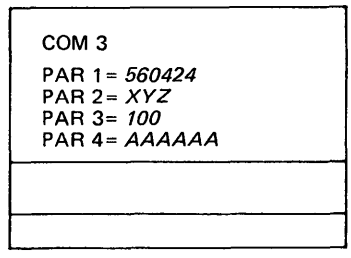




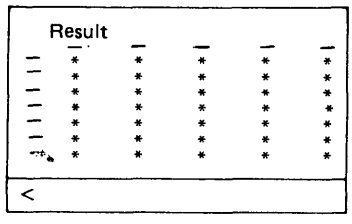
1. The user knows the command code but not the parameters. He enters a directive in the form of the command code.

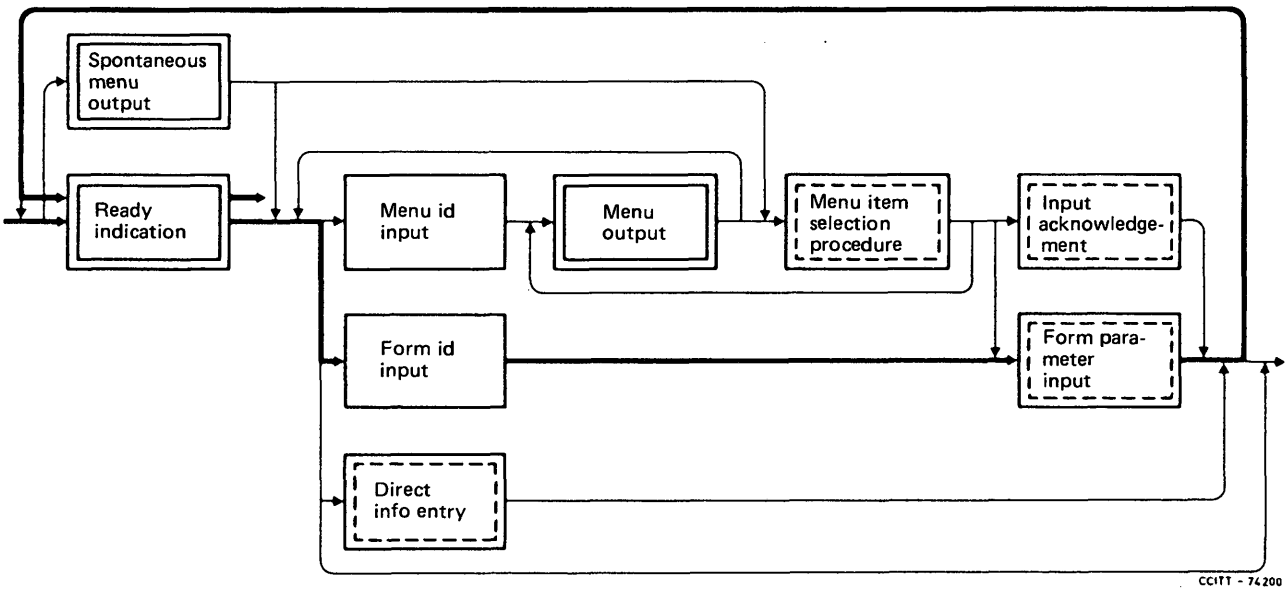


2. A form output is displayed. The form is filled and entered. Note that the ready indication is not displayed during form filling.



3. An acceptance output is displayed in the form of a result and the system is ready for the next input. Note that the output in this example is so spacious that the output window has increased at the expense of the work window.





A.4 *Example 3*

1. A spontaneous menu output is automatically displayed. The menu items refer to other menus on a lower and more specific level. The user chooses the appropriate menu and enters the associated selection identity.

Menu
1. Menu 1
2. Menu 2
3. Menu 3
4. Menu 4

< *1*

2. A new menu output is displayed. In this case the menu items represent command codes. The user selects the wanted command code by entering the associated selection identity.

Menu 1
1. COM 1
2. COM 2
3. COM 3

< *1*

3. A form output is displayed. The form is filled and entered by the user.

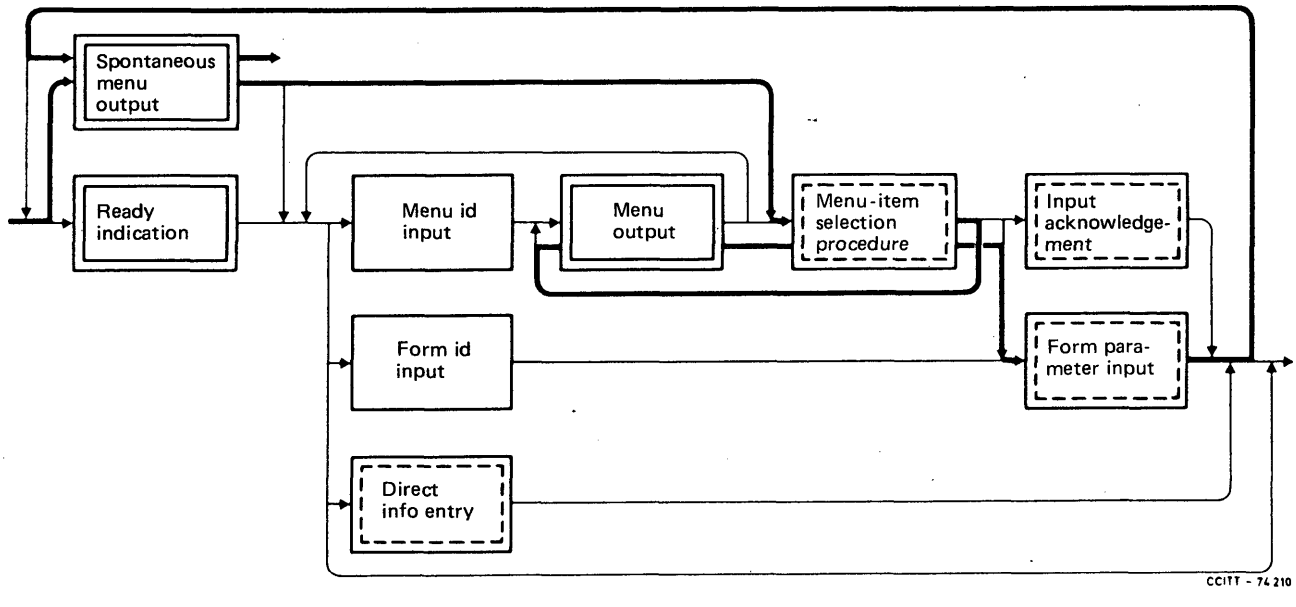
COM 1
PAR 1= *1234* PAR 2 =*GIGA*
PAR 3= *9999* PAR 4 =*500*
PAR 5= *ABCDE*

4. An acceptance output is displayed together with the spontaneous menu. The system is ready for the next input.

Menu
1. Menu 1
2. Menu 2
3. Menu 3
4. Menu 4

Command executed

<



A.5 *Example 4*

1. The user enters a directive in the form of a menu identity in order to shortcut to a certain menu.

< *MENU 3*

2. A menu output containing items referring to other menus is displayed and a selection identity is entered.

Menu 3

1. Menu 31
2. Menu 32
3. Menu 33

< *3*

3. The selected menu is displayed. The items in the menu represent command codes. The user recognizes the command code and then remembers the parameters. The entire command is entered directly.

Menu 33

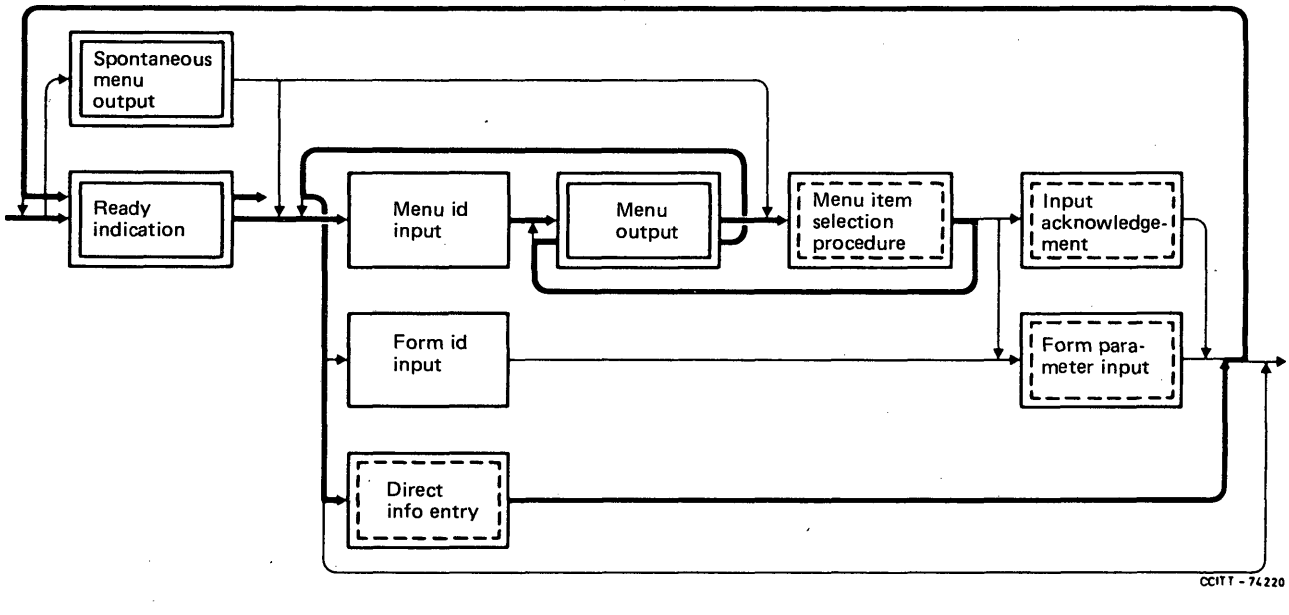
1. COM 1
2. COM 2
3. COM 3
4. COM 4

< *COM 2: PAR 1=5, PAR 2=10;*

4. An acceptance output is displayed and the system is ready for the next input.

Command executed

<



A.6 *Example 5*

1. A spontaneous menu output is automatically displayed. The user already knows the command code and enters it.

Menu
1. Menu 1 2. Menu 2 3. Menu 3 4. Menu 4
< *COM 4*

2. This command requires two forms to be filled in. The first form output is displayed. The user fills in the parameters and enters the form.

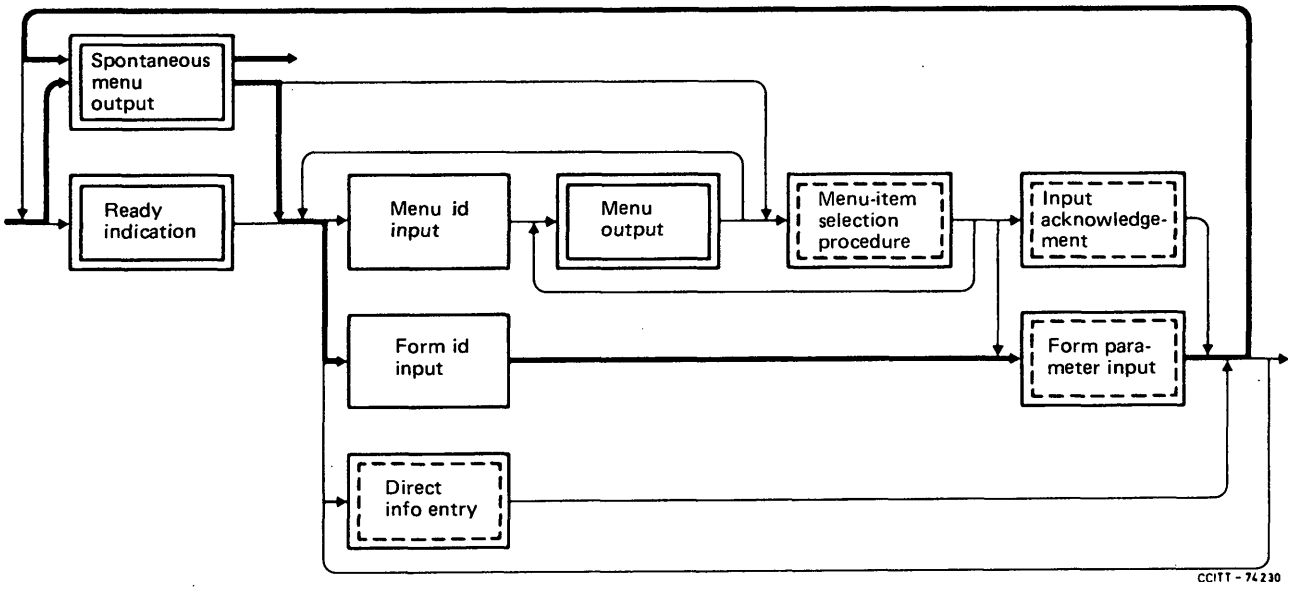
COM 4
PAR 1 = *6543* PAR 2 = *GHIJK* PAR 3 = *333* PAR 4 = *XXXXXXX*

3. The second form output is displayed and the user fills in the rest of the parameters and enters the form.

COM 4
PAR 5= *AEFE* PAR 6= *LES* PAR 7= *DIDIT*

4. An acceptance output is displayed. The system is ready for the next input.

Menu
1. Menu 1 2. Menu 2 3. Menu 3 4. Menu 4
Command executed
<



PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

SECTION 4

SPECIFICATION OF THE MAN-MACHINE INTERFACE

Recommendation Z.331

INTRODUCTION TO THE SPECIFICATION OF THE MAN-MACHINE INTERFACE

1 Scope of the Section

The man-machine interface comprises the set of inputs, outputs and special actions, together with the man-machine interaction mechanisms, including dialogue procedures. Those elements are combined to manipulate the varied telecommunications functions which cover the management of SPC telecommunications systems. Consideration of these functions has been an essential prerequisite for the development of the CCITT MML Recommendations.

As stated in Recommendation Z.301, the CCITT MML can be used to facilitate operation, maintenance, installation, and acceptance testing of SPC systems. With the tendency of Administrations to centralize operations and maintenance jobs, many of the SPC systems functions may be controlled at terminals associated with operation and maintenance systems as well as at terminals associated with SPC systems. These terminals can be either local or remote relative to the system.

In order to help Administrations aiming to achieve uniformity among various systems, the MML Recommendations include not only the syntax of the language and dialogue procedures, but also the semantics relevant to the man-machine interface. Section 4 provides the means for deriving such semantics.

2 Organization of Section 4

Section 4 consists of the following Recommendations:

Z.331 Introduction to the specification of the man-machine interface

Z.332 Methodology for the specification of the man-machine interface – general working procedure

Z.333 Methodology for the specification of the man-machine interface – tools and methods.

Recommendation Z.331 lists the operation, maintenance, installation and acceptance testing functions to be controlled by means of the MML.

Recommendation Z.332 presents the first part, the general working procedure of a methodology by which the man-machine interface can be generated for a particular functional area or subarea.

Recommendation Z.333 presents the second part, the tools and methods, of a methodology by which the man-machine interface can be generated for a particular functional area or subarea.

3 Functions to be controlled by means of the MML

The functions to be controlled by means of the MML are subdivided into four main areas: operation, maintenance, installation and acceptance testing. They are listed below. Based on the relationships existing among them, in each main area functions are grouped into functional areas and sometimes functional sub-areas. Due to the potentially different organization needs and system design philosophies, it is recognized that not all functions apply to every system.

The list of functions is based on a previous Recommendation [1] which was jointly produced with other Study Groups. This list is not complete and it is expected to continue to evolve. The Methodology for the Specification of the Man-Machine Interface has been applied to the traffic measurement administration and the maintenance of circuits between exchanges. This work was done in cooperation with the Study Groups involved in the study of these functional areas and the results are appended as examples to the description of the methodology (Recommendations Z.332 and Z.333).

3.1 *Operation functions*

3.1.1 *Subscriber administration*

- taking subscribers' lines into/out of service[1];
- allocating changes and remove classes of subscriber service;
- changing a subscriber's number;
- blocking and unblocking a subscriber's line;
- interrogating a subscriber's classes of service;
- interrogating blocked subscriber lines;
- reading subscriber's charging information;
- retrieving charging information;
- tracing malicious call;
- putting a subscriber on subscriber charging observation, etc.

3.1.2 *Routing administration*

3.1.2.1 *Changing data relating to a group of circuits*

- changing signalling system parameters;
- inserting a new group of circuits;
- changing the search order in a bidirectional group of circuits;
- adding a new circuit;
- changing the group affiliation of a circuit;
- changing the position of a circuit from one inlet or outlet to another inlet or outlet in the switching matrix.

3.1.2.2 *Changing routing and analysis data*

- changing the alternative routing parameters;
- changing the traffic routing for network management purposes;
- changing the analysis data (start of selection, number of digits to be sent, etc.).

3.1.3 *Traffic administration*

3.1.3.1 *Traffic measurements administration*[2] (see Recommendation E.502)

- performing measurements of traffic parameters;
- scheduling traffic measurements: execution and results output;
- managing measurements' data;
- retrieving measurements' data.

---

[1] Subscribers' lines includes lines on PABX.

[2] These functions have been derived by the application of the methodology.

3.1.3.2 *Traffic analysis administration* (see Recommendation E.502)

- inputting measured data;
- inputting the identification and capacity information of the measurement object;
- managing traffic data records;
- managing the output of reports;
- managing analysis description data;
- supervising the control of the time-delay of the various analysis operations.

3.1.4 *Tariff and charging administration*

- changing the tariff for a certain traffic destination;
- changing parameters for a charging rate;
- changing time for switching between day and night rate;
- reading accounting statistics (accounting between operating companies);
- changing the parameters involved in the accounting methods for traffic between different operating companies.

3.1.5 *System control operation*

- setting and reading of a calendar;
- administering man-machine terminal assignments;
- administering files;
- administering man-machine terminal capabilities;
- administering user authority;
- administering the system (hardware/software) configuration.

3.1.6 *Network management*

- changing routing information;
- changing criteria for initiating network management actions.

3.2 *Maintenance functions*

3.2.1 *Maintenance of subscribers' lines*

- testing one subscriber's line and associated equipment;
- testing a group of subscribers' lines and associated equipment;
- measuring one subscriber's line and associated equipment;
- measuring a group of subscribers' lines and associated equipment;
- blocking or unblocking a subscriber's line for maintenance purposes;
- observing or supervising of subscribers' lines and equipment.

3.2.2 *Maintenance of circuits between exchanges and associated equipment*[3] (see Recommendation M.250)

- testing/measuring one circuit or a group of circuits and associated equipment;
- observing and supervising circuits and associated equipment;
- control the status of a circuit or a group of circuits and associated equipment;
- analysing maintenance data;
- administering and controlling maintenance reports.

3.2.3 *Switching network maintenance*

- making test calls;
- initiating a call trace;
- holding faulty connections;
- testing and measuring peripheral equipment (relay sets, signalling receivers and senders, etc.);

---

[3] These functions have been derived by the application of the methodology.

- testing and measuring switch units;
- reducing service for low priority subscribers;
- setting up a connection via a specific path through the network;
- supervising and measuring the quality of service of the switching network;
- localizing faults in the speech path network;
- providing access for traffic observation for maintenance purposes;
- reporting alarms;
- recording switch unit status.

3.2.4 *Control system maintenance*

- reporting system status;
- reporting alarms;
- localizing faults;
- testing on a functional basis after repair;
- initiating periodic testing operations;
- changing system configuration for maintenance purposes;
- checking consistency of data;
- initiating restart;
- setting traps for programme fault tracing;
- changing memory contents;
- memory dumping for maintenance purposes;
- controlling overload parameters;
- changing the criteria for the recognition of degradation of service;
- reducing service for low-priority subscribers.

3.3 *Installation functions*[4)]

3.3.1 *SPC system installation*

3.3.1.1 *SPC system hardware installation*

Installing:

- network blocks;
- trunks;
- signalling equipment;
- test equipment;
- blocks of subscriber-circuits;
- interface equipment;
- control equipment;
- memory equipment;
- input/output devices.

3.3.1.2 *SPC system software installation*

Installing:

- operational packages;
- test programmes;
- statistics programmes;
- programmes patches;
- signalling systems programmes;
- services, facilities programmes;
- system data.

4) Installation also covers the extensions or reductions of the system after it is placed into service.

3.4 *Acceptance testing functions*

Acceptance testing functions include any additional functions beyond those presented above to aid the administrations when testing a system to check its compliance with the Administrations' specifications.

Reference

[1] CCITT Recommendation *List of functions*, Vol. VI, Yellow Book, Fascicle VI.7, Rec. Z.318, 1980.

Recommendation Z.332

METHODOLOGY FOR THE SPECIFICATION OF THE MAN-MACHINE INTERFACE

GENERAL WORKING PROCEDURE

1 Introduction

Recommendation Z.331 provides a summary of the functions which are to be controlled by means of MML. Each functional area in this list is to be specified in detail to allow the generation of function-related semantics.

The use of such semantics in conjunction with the features provided by the Recommendations in Sections 2 and 3 allows the specification of the man-machine interface.

In order to produce a detailed specification, a formal method of working that provides a common approach is necessary. This Recommendation provides a methodology for such purposes.

In order to assign properly the responsibility for the application of the methodology, its application can be viewed as a two-stage process.

The first stage involves the generation of function-related semantics. This stage is aimed primarily at those experts working in CCITT Study Groups who are responsible for developing Recommendations associated with functions to be controlled by MML. However, it is recognized that the repertoire of such functions considered in CCITT Recommendations cannot cover the requirements of all Administrations or of all SPC systems. Therefore this stage is also aimed at Administrations, private operating agencies and scientific/industrial organizations who may find it necessary to specify functions peculiar to their individual needs.

The second stage of the application of the methodology involves the derivation of the actual man-machine interface using the semantics and the relevant features of Sections 2 and 3. This stage is the responsibility of Administrations, private operating agencies, and scientific/industrial organizations.

2 Orientation of the methodology: Administration centred and system centred

The methodolgy for the specification of the man-machine interface must be based on a common understanding of the concept of function.

Three different classes of system functions may be defined as follows:

1) *Class A functions or man-machine language (MML) functions*

   Those system functions which provide the MML user with the means of control of system functions by MML. The word "control" is assumed to include all types of inputs and outputs.

   Any Class A function can be subdivided into a general part which relates to e.g. the syntax check, information transmission control, etc., and an application part which relates to the job in hand.

   Example: Create a traffic measurement.

2) *Class B functions*

   Those system functions which can be controlled at least partially by the MML user by means of MML functions.

   Example: Performing measurements of traffic parameters.

3) *Class C functions*

Those system functions which are not at all controlled by the MML user in a given system during operation. Class C functions are not referred to in the following methodology.

The relationship between the concepts of "job" and the different types of functions is shown in Figure 1/Z.332.

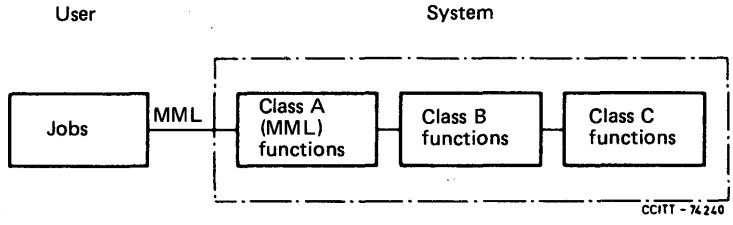


FIGURE 1/Z.332

This definition of MML function embodies the concept of both system actions and human actions performed on objects. The methodology presented in the following sections is based on the understanding of this concept.

To clarify the concept of "job" as applicable to operations and maintenance the following definition is provided.

Job: A discrete administrative activity within a telecommunications business which is designated as a part of the overall plan for running the business and characterized by man-machine communication and/or manual actions.

It is recognized that in the future the degree of automation of operation and maintenance jobs in the telecommunication network will increase as the application of auxiliary systems is broadened. Consequently, it is expected that all or part of a certain Class B function implemented in one system may appear as a Class C function in another system. The result is that the number and type of Class A functions supporting the same set of operation and maintenance jobs may differ from one system compared to another system.

3 General working procedure

The general working procedure consists of five phases:

1) the identification of Administration needs;
2) the identification, in sufficient detail, of the MML functions, i.e. those needed for control of the system by the user;
3) the identification of the information structure associated with each MML function;
4) the specification of the actual man-machine interface;
5) the verification and validation of phases 2, 3 and 4.

A more formal representation of this general working procedure is presented in Figures 2/Z.332, 3/Z.332 and 4/Z.332. The representation is made by means of Functional Block Interaction Diagrams as defined in the Z.100 series Recommendations on the Specification and Description Language (SDL). Figure 2/Z.332 represents the procedure at a high level showing its basic factors. Figure 3/Z.332 describes, at a lower degree of detail, the five phases presented above in terms of the information which should be produced and considered in each phase and their relationships. Figure 4/Z.332 describes, in the same terms, the two sub-phases into which phase 2 is further decomposed. As a drawing convention, information which is used primarily to support the activities performed in the various phases is indicated in the upper part of the Functional Block symbol.

Each phase is more fully described in the following paragraphs regarding its purpose, input and output products, relevant methods and tools, and CCITT Study Group responsibilities. Additionally, two examples of the application of this methodology are presented in Annexes A and B to Recommendations Z.332 and Z.333.

To achieve greater commonality among the various functional areas when performing phases 1, 2 and 3, harmonization of the terminology used is essential. A glossary of terms that may be useful in a number of functional areas has been provided in Recommendation Z.333.

This glossary is expected to evolve as MML function semantics activity continues. In addition, a glossary of terms specific to each functional area should also be provided as indicated below.

It should be emphasized that terminology harmonization refers to those phases of the methodology described herein which are the responsibility of the CCITT. It is not the intention of this recommendation, through its glossary or annexed examples, to recommend specific terminology for use at the actual man-machine interface. The present intent is rather that manufacturers and Administrations utilize the *concepts*, as here defined, that this terminology represents. They will select their own terminology to represent these concepts as applicable to their needs in specifying the actual interface. A common understanding of the definitions of these concepts will improve the coherence of the set of CCITT Recommendations in MML function semantics, as well as facilitate discussion concerning the capabilities of different systems with respect to the same as well as different functional areas.

The output of each phase is to be listed in a series of documents based on the terminology of Figures 3/Z.332 and 4/Z.332.

Phases	*Name*
1	Document A – List of System Independent Class B Functions and List of Jobs
2.1	Document B – Function Models
2.2	Document C – List of MML Functions
3	Document D – Information Structure of each MML Function
4	Document E – Specification of the man-machine interface
5	Document F – Verification and Validation Results
1-5	Document G – Glossary of terms.

The application of the methodology to a specific functional area may vary. Documents A-G may be produced for the functional area as a whole or the functional area may be divided into sub-areas and each treated separately. The primary rationale for the approach selected should be the coherence and maintainability of the total set of documents prepared for the functional area. If the second approach is selected, its details, including an unambiguous description of the main area and the identified sub-areas, should be documented also.

3.1 *Phase 1: identification of needs*

Purpose

To identify the various Administration needs in order to prepare a list of jobs to be performed by means of man-machine communications and to prepare an agreed list of system independent functions which are expected to be controlled by means of the MML (Class B functions). Terminology harmonization is essential.

Input

Inputs to the process of identifying Class B functions arise from three sources. First, CCITT Study Groups can provide operations and maintenance models and lists of Class B functions which are embodied in those models.

Second, Administrations can provide information on the jobs by which their systems are operated and maintained. Some indication as to the relative importance or frequency might be helpful in the process of specifying the man-machine interface.

The third input is the current version of Recommendation Z.331.

Output

List of System Independent Class B Functions and List of Jobs (Document A).

These functions and jobs could be performed at terminals associated with operations and maintenance systems or SPC systems. A certain set of these functions and jobs might be able to be performed only at terminals associated with operations and maintenance systems or only at terminals associated with SPC systems.

Tools and methods

It will be necessary to take into account the following:

- directives from other Study Group experts;
- guidelines, as described in Recommendation Z.333;
- terminology harmonization guidelines, as described in Recommendation Z.333.

Use of SDL is also recommended.

Responsibility

Function experts of relevant Study Groups supported by MML experts.

3.2 *Phase 2: MML function identification*

Purpose

To identify, using harmonized terminology, MML functions related to Class B functions. This phase is an iterative procedure involving the application of several tools to identify the list of MML functions, i.e. those functions that are described in sufficient detail to allow the derivation of the man-machine interface. A diagrammatical representation of this phase is shown in Figure 4/Z.332.

Input

List of system independent Class B functions and a list of jobs, both obtained as output of phase 1.

Output

- List of MML functions.
- Other information (whenever applicable).

} Document C

3.2.1 *Sub-phase 2.1: modelling*

Purpose

To represent, using harmonized terminology, the various functions of those parts of telecommunications systems controlled by MML by means of models.

Input

List of system independent Class B functions.

Output

- Description of Class B functions by means of models.
- Other information (whenever applicable).

} Document B

Tools and methods

- At present informal modelling is available and there exists a need to identify and develop a formal method of modelling. SDL could be used for parts of the modelling work.
- Terminology harmonization guidelines, as described in Recommendation Z.333.

Responsibility

Function experts of relevant Study Groups supported by MML experts.

3.2.2 *Sub-phase 2.2: MML function decomposition*

Purpose

To identify, using harmonized terminology, each MML function considering both the model and the defined list of jobs.

Input

- List of jobs.
- List of system independent Class B functions.

Output

- List of MML functions.
- Other information (whenever applicable).

} Document C

Tools and methods

- The use of SDL is applicable. In order to represent or derive the MML functions, the MML function decomposition method should be applied
- Terminology harmonization guidelines, as described in Recommendation Z.333.

Responsibility

Function experts of relevant Study Groups supported by MML experts.

3.3 *Phase 3: information structure identification*

Purpose

To identify, using harmonized terminology, the information structure of each MML function in order to provide a clear picture of the associated semantics (action, objects, information entities and their interrelationships). Separate diagrams for the structure of information related to input functions and to those outputs whose significance is such that benefits would be gained by their standardization should be provided.

The content of information structure diagrams should be limited to information related to such semantics. Other information, such as information related to possible parameter values, if desired, may be listed separately or as footnotes.

A one-to-one correspondence between information structure diagrams produced in this phase and the associated commands and outputs to be produced in Phase 4 is not implied. More specifically, a single information structure diagram could lead to a multiplicity of inputs or outputs. Also, several information structure diagrams could lead to a single input or output. Additionally, information structure diagrams should not be interpreted as a specification of any software process required to implement the related inputs and outputs.

Input

List of MML functions.

Output

- Information Structure Diagrams of each MML function.
- Additional information (a list of possible parameter values associated with Information Structure Diagrams).

} Document D

Tools and methods

Each MML function derived in phase 2 is in essence an action upon an object (or set of objects). An Information Structure meta-language is used to produce the Information Structure Diagrams associated to each MML function, as described in Recommendation Z.333.

Terminology harmonization guidelines, as described in Recommendation Z.333.

Responsibility

Function experts of relevant Study Groups supported by MML experts.

3.4 *Phase 4: specification of the actual man-machine interface*

Purpose

To present each input and output as it might appear on a man-machine communication terminal in terms of the related syntactic structure and to identify any related special actions. Also to select the appropriate dialogue procedures related to the MML functions.

Input

- The information structure representation of each MML function.
- Additional information.

Output

- Specification of the man-machine interface:
  a) inputs
  b) outputs
  c) special actions
  d) dialogue procedures
  e) interrelationships among a) to d).

Tools and methods

The structure of inputs, outputs or special actions can be identified using guidelines as described in Recommendation Z.333.

A formal method to describe the syntactic structure of each MML input and output is given in Recommendation Z.333.

Recommendations Z.302, Z.314-Z.317, Z.323.

The use of SDL to describe the interactive operating sequences is recommended.

Responsibility

The execution of this phase is left to Administrations and suppliers. Z.300 series Recommendations do not deal with phase 4.

3.5 *Phase 5: verification and validation*

Purpose

To verify whether the MML functions identified previously together with their associated information structure lead to suitable procedures by which the users' needs can be satisfied.

To verify whether the man-machine interface identified in phase 4 leads to suitable procedures.

Input

- Information structure representations of each MML function.
- Preliminary man-machine interface.

Output

- An evaluation of the MML functions and their associated information structure.
- An evaluation of the preliminary man-machine interface.

} Document F

Tools and methods

- Procedure description method.
- Guidelines as described in Recommendation Z.333.

Responsibility

Function Experts of relevant Study Groups supported by MML experts for phases 1, 2 and 3. Administrations and suppliers for phase 4.

3.6 *Tools and methods*

Many tools and methods are available to provide assistance in reaching the goal of each phase described above. The applicability of each tool and method to a particular phase is dependent on the function being analysed. These tools and methods are described in Recommendation Z.333.

Examples of the use and application of these tools and methods for specifying functions are also included in Recommendation Z.333 and the Annexes to these Recommendations.

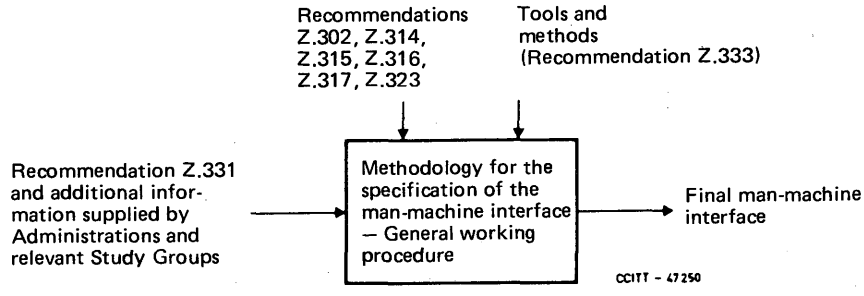


FIGURE 2/Z.332

High level view of the general working procedure of the methodology for the specification of the man-machine interface

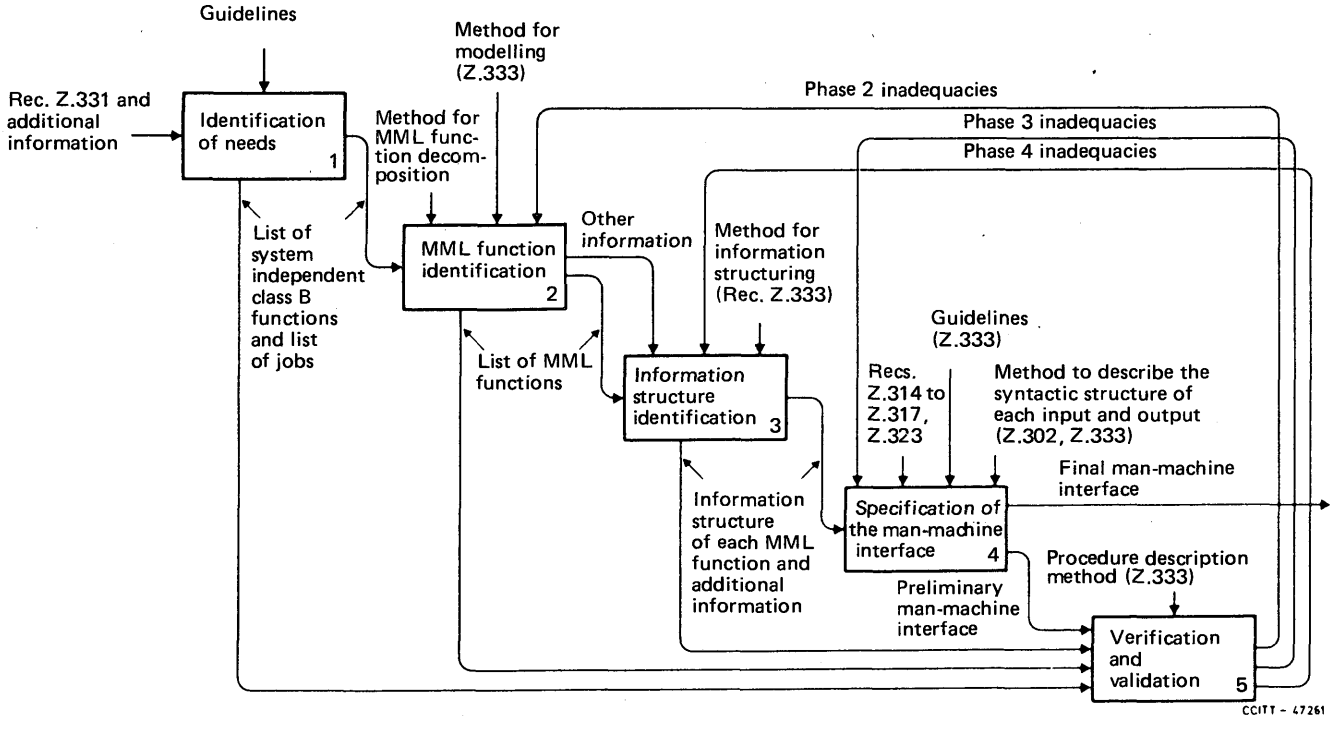


FIGURE 3/Z.332

General working procedure of the methodology for the specification of the man-machine interface

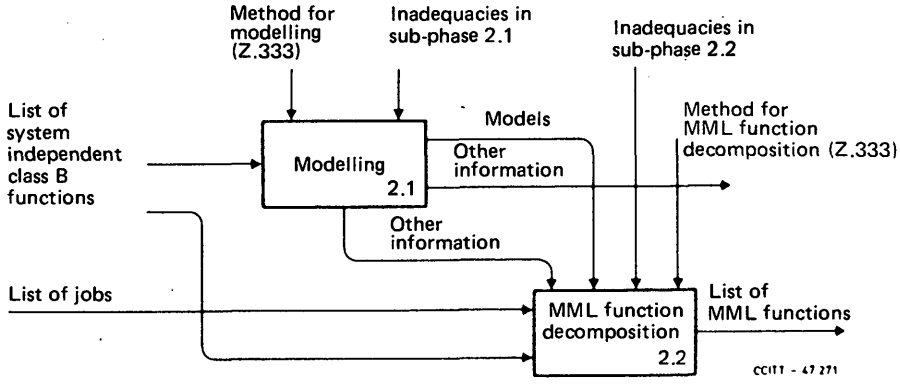


FIGURE 4/Z.332

Phase 2 of the general working procedure of the methodology for the specification of the man-machine interface

Recommendation Z.333

METHODOLOGY FOR THE SPECIFICATION OF THE MAN-MACHINE INTERFACE

TOOLS AND METHODS

1 Introduction

This Recommendation presents the tools and methods that support the general working procedure described in Recommendation Z.332. Taken together, Recommendations Z.332 and Z.333 constitute the methodology for the specification of the man-machine interface.

2 List of tools and methods [1)]

The following tools and methods are necessary to support the methodology for the specification of MML functions:

- guidelines,
- modelling,
- MML function decomposition method,
- information structure metalanguage,
- procedure description method,
- formal representation of the syntactic structure of each input and output.

[1)] The tools and methods may be improved on the basis of user experience leading to additions or revisions.

3 Description of available tools

3.1 *Guidelines*

3.1.1 *For phase 1*

Determine for each job:

- the purpose of the job;
- what the system is supposed to do;
- what the user is supposed to do;
- the complexity of the job from the users' perspective (see Note);
- the frequency of the job (see Note);
- at which level in the network hierarchy the job is supposed to be performed (exchange, OMC);
- safety aspects.

Note – The following assumptions have been taken to better identify what is meant for "frequency" and "complexity" of a job.

3.1.1.1 *Frequency*

Low:

- if the job is supposed to be performed weekly or at longer intervals.

Medium:

- if the job is supposed to be performed daily.

High:

- if the job is supposed to be performed several times in a day.

3.1.1.2 *Complexity*

Low:

- low number of parameters (in general sense) – max 0:3;
- most information associated with these parameters are not compound;
- there is no semantic relationship among different parameters and parameter values.

Medium:

- the number of parameters is greater than 4 but less than 6-8;
- much information associated to these parameters is compound;
- there is no semantic relationship among parameters and/or parameter values.

High:

- there are many parameters;
- most information associated to these parameters is compound;
- there are semantic relationships among parameters and/or parameter values.

3.1.2 *For Phase 4*

To define the individual inputs and outputs:

1) Consider what the system is supposed to do.
2) Select options in the function information structure.
3) Define the information to be represented by the command code or equivalent.
4) Define the information to be represented by the parameters and, if appropriate, their order.

5) For each parameter, when appropriate, identify the:
   - range of values,
   - default values,
   - information to be automatically supplied by the system.

6) Define the response outputs within dialogue, the interaction request outputs and outputs outside dialogue when applicable after considering the various mode operating sequences and the users' reactions to the outputs.

7) Define the associated syntactic structure.

8) Select terms and abbreviations for inputs and outputs.

3.1.3 *For phase 5*

1) Define a preliminary operational procedure in functional terms.

2) Finalize operational procedures.

3.1.4 *General guidelines*

1) Determine that the MML functions support the jobs to be performed.

2) It will be necessary to consider:
   - human factor aspects;
   - adequate allocation of authorities;
   - adequate definition of responsibility;
   - training of the user.

3.1.5 *Terminology harmonization guidelines for Phases 1-3*

To harmonize terminology:

1) Utilize existing CCITT vocabulary.

2) Select appropriate terms included in the general functional terminology (Appendix I).

3) Derive specific terms and their definitions pertinent to the functional area involved based on the following considerations:
   - common usage;
   - specificity;
   - translatability.

3.2 *Modelling*

Modelling involves the use of description text and/or figures drawn either with the support of formal symbology and rules (formal modelling) or without such rules (informal modelling).

3.2.1 *The need for models*

A tool available to Study Groups is the construction of informal models of those parts of telecommunications systems which have been selected for MML control. Also the organization of the Administration could be subject to modelling. Several models could apply when defining a job or an MML function. The use of models has the following advantages.

1) Models provide a means for the exchange of functional descriptions between Study Groups.

2) The validity of the derived man-machine interface can be consistently demonstrated by reference to the relevant models.

3.2.2 *Sources of information for models*

There are many sources of information on which models can be based but the approach by Study Group XI has so far been to draw on the experience and contacts of its own members. In the future, it is expected that models will be formed by cooperative effort between Study Group XI and other Study Groups. The models which are produced might have multiple applications and it is important that before embarking on extensive modelling effort, the Study Group should check whether or not suitable models exist for the applications in mind.

In order to ensure the authenticity of models, the Study Group must take all necessary steps to seek out and develop sources of information for models.

The support of other Study Groups is necessary in the light of their:

1) need for models in their own work;
2) ability to produce and/or provide information for models, and
3) interest in a separate capability within CCITT for modelling telecommunications systems for use in both CCITT and member organizations.

3.2.3 *Interpretation of models*

Models produced expressly to determine the MML control structure are interpreted purely with that use in mind. Other models must lend themselves to the generation of MML control message sequences. CCITT feels bound to produce models which can be linked with the methods for determining information structure of MML functions.

3.2.4 *Informal model examples*

Informal models covering traffic measurements administration and maintenance of circuits between exchanges are contained in the Annexes to this Recommendation.

3.3 *MML function decomposition*

The general MML functions are structured into component MML functions. Multiple levels of decomposition are allowed. For examples see the Annexes to this Recommendation.

3.4 *Information structure meta-language*

Each MML function identified at the lowest level of the MML function decomposition is structured into the information components needed to perform it. A top-down structuring is performed and multiple levels of information decomposition are allowed. The supporting tool is the meta-language presented below. For examples see the Annexes to this Recommendation.

An aid in understanding of information structuring is to view a MML function as an action on an object(s). Information composed may therefore relate either to objects or to actions.

A general action associated with a MML function can be decomposed into subsidiary actions and modifiers to those actions. It is possible that no decomposition will take place. However, if decomposition is necessary, it should be noted that "decomposition" with respect to actions means both determining subsidiary actions *and* determining any qualifiers (modifiers, options, etc.) associated with the action. The latter is not a true decomposition.

3.4.1 *Decomposition meta-language*

3.4.1.1 *General*

The representation of the information structure associated with a MML function involves the specification of all needed information entities and their inter-relationships.

This representation can be achieved in a consistent manner by means of Information Structure Diagrams, drawn using the meta-language described below, such meta-language consists of a set of symbols and drawing conventions.

A diagram represents the information structure in a top-down approach, starting from the identification of the MML function to be structured and ending with all the information components felt necessary in the man-machine interworking for that function.

The decomposition process is performed by the use of *sequences*, *selections* and *iterations*, by means of which any type of structure can be obtained.

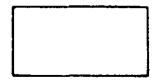
Unless otherwise stated, the sequence of information is not implied by the order in which different elements are presented in the diagram.

3.4.1.2 *Information entities*

3.4.1.2.1 *Composite parts*

A composite part is an information entity that can be further structured into smaller parts.

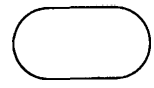
The following symbol is used:



3.4.1.2.2 *Component*

A component is an information entity that is not structured further.

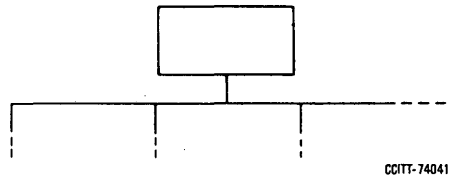
The following symbol is used:



3.4.1.3 *Structuring*

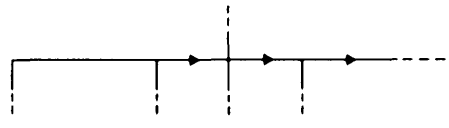
3.4.1.3.1 *Subdivision*

Subdivision in Information Structure Diagrams is shown in the following way:



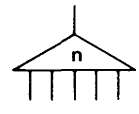
3.4.1.3.2 *Sequence*

When the order between information entities is relevant, these are specified in sequence. A left-to-right sequence is indicated by the use of arrowheads as follows:

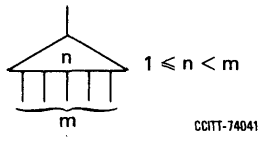


3.4.1.3.3 *Selection*

When a composite part is structured into a number of information entities, of which one or only some are relevant in any one case, a selection mechanism is used, represented by the following symbol:



In the general selection case, m possibilities exist from which selection must be made. Of these m possibilities a *specified* number, n, is to be selected, which implies $n < m$.



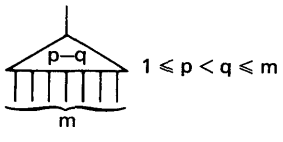
The number of possibilities to be selected, n, is given explicitly within the selection symbol, while the total number of possibilities, m, is given implicitly by the number of outlets from the selection symbol.

The following cases are allowed:

$n = 1$, $m > 1$ This is the most common selection case implying that one and only one of the possibilities is to be selected.

$n > 1$, $m > n$ Multiple selection of n of m possibilities.

If the number of choices to be made are variable between specified lower and upper limits, a number of possibilities are implied. In this case, both limits are given in the selection symbol:

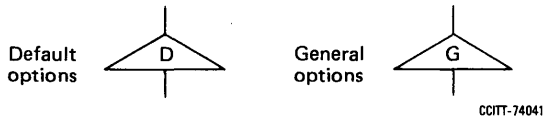


The lower limit p indicates the smallest number and q the largest number of *different* choices to be made out of the m possibilities. It should be noted that each choice may be selected only once.

3.4.1.3.4 *Options*

In some cases, options may be required such as default options or general options.

In these cases, the type of option is indicated by the appropriate capital letter only within the selection symbol, i.e. D for default options, G for general options. Only one outlet from the symbol is allowed:

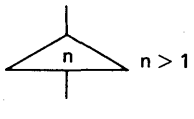


The use of a default option implies that the value taken by an information entity will be provided automatically if the user does not supply a value in the input.

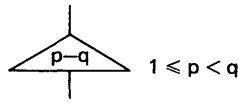
A general option is to be used for various reasons reflecting the needs of manufacturers and the needs of Administrations. The information entities that can be deduced from the outlet of this box can optionally be part of the man-machine interworking. This means either that the information exists in the system in a predetermined manner or that it is not needed. If this distinction must be made an annotation to the information structure diagrams should be made.

3.4.1.3.5 *Iteration*

When a composite part is structured into a number of information entities that can be repeated an arbitrary number of times, an iteration mechanism is used, represented by the following symbol, which has only one outlet:



If a number of interactions can vary within a range, the number of times a part is to be repeated is given as the lower limit p and the upper limit q.



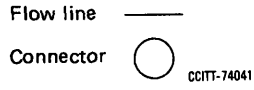
3.4.1.4 *Drawing conventions*

3.4.1.4.1 *Flow lines and connectors*

Every symbol is connected to the symbol it follows by a solid flow line.

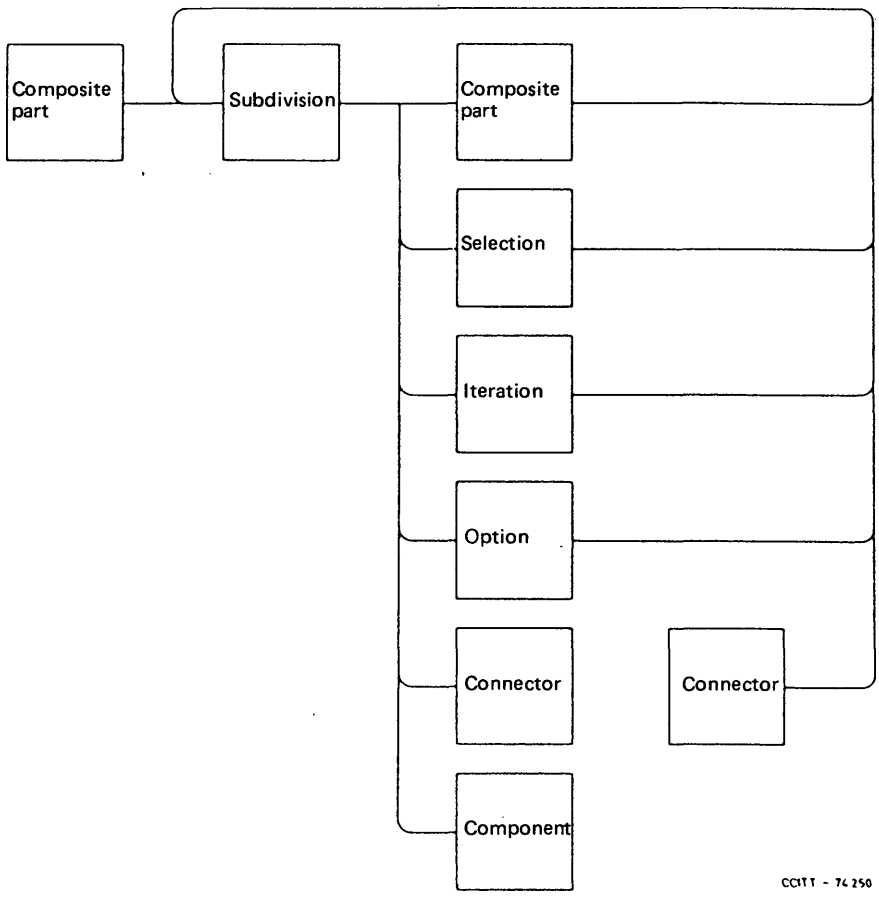
A solid flow line may be broken by a pair of associated connectors, with the flow assumed to be from the out-connector to its associated in-connector. Several out-connectors can be associated with the same in-connector.

Crossed flow lines should be avoided wherever possible.



3.4.1.4.2 *Connectivity rules*

Each information structure diagram begins with a composite part symbol and each path of a diagram ends with a component symbol. The drawing of diagrams must follow the connectivity rules represented below.



Note 1 – The symbol types and possible subdivision of flow lines are shown in square boxes.

Note 2 – Subdivision includes the trivial case of single continuous flow line.

3.4.1.4.3 *Annotations*

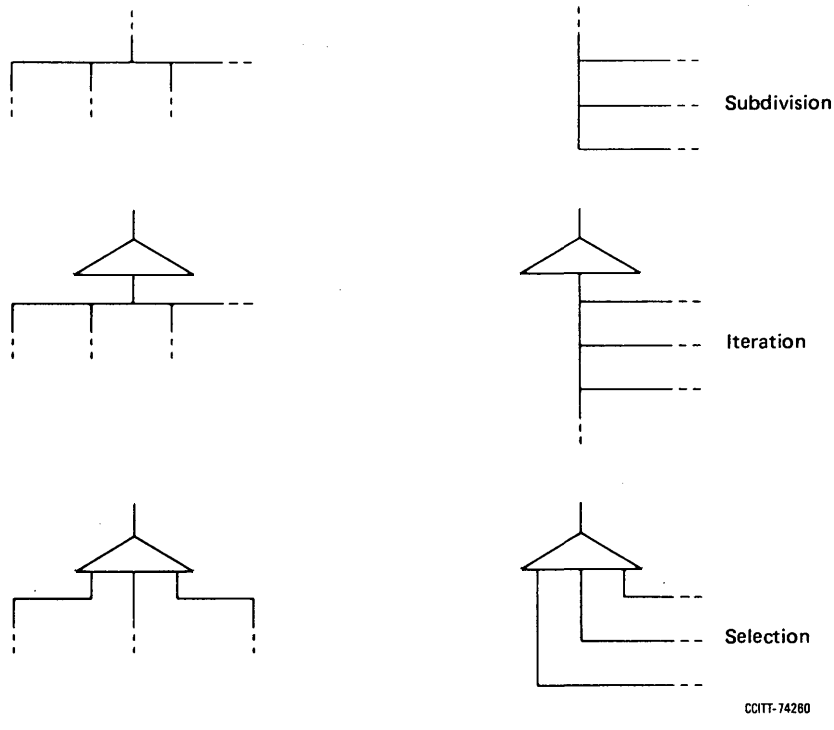
Annotations are denoted by the following symbol, where *n* is a number referencing a note providing descriptive and/or explanatory information.

Annotation ----[n .

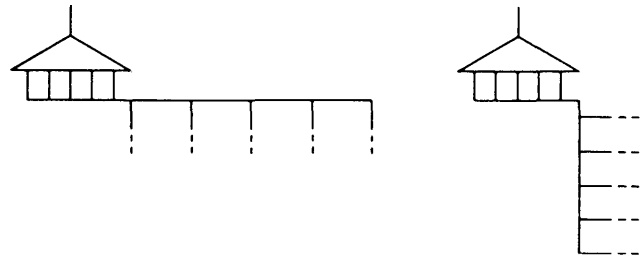
Annotations may be connected by a dashed line to any symbol or flow line.

3.4.1.4.4 *Special Notations*

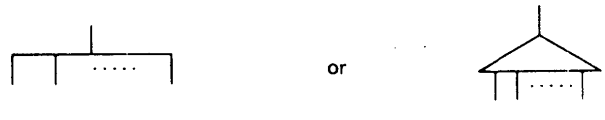
Instead of the normal structuring symbology where the structuring is shown horizontally, a vertical symbology may be used where this is advantageous, i.e. for saving of space. This vertical symbology may be used with all types of structuring.



For the selection symbol, in case of a high number of possibilities, the following drawing conventions are also allowed:



Where the number of information entities in a structure is undetermined, this could be shown as



depending on the type of structuring used.

CCITT - 74260

3.5 *Procedure description method*

Man-machine dialogue may be considered a feature of an SPC system and may be represented by means of two processes: one related to the user, the other related to the system. These two processes exchange information by means of signals that for MML purposes are intended to be mostly inputs and outputs.

In particular, the description of MML operational procedures may be made by focusing the attention on one of the machine logic functions, the associated MML function, and describing the process that performs this function.

To reduce the complexity of drawings, it seems useful to limit the description to the main signals between user and system, i.e. inputs and outputs, and to omit showing features such as timing, error reporting, editing procedures, etc., that may be described elsewhere by means of SDL if needed. For an example see Appendix II.

3.5.1 *Features to be used in the description*

A MML operational procedure can be considered as a process whose behaviour may be specified in terms of inputs, states, transitions, decisions, outputs and tasks.

In the following paragraphs, basic concepts of SDL are interpreted in the context of MML applications.

3.5.1.1 *Input*

An input is a set of data which is input by the user and which is recognized by the MML operational procedure. Input may be, e.g commands in direct information entry, or other types of data.

3.5.1.2 *State*

A state is a condition in which the action of the MML procedure is suspended awaiting an input.

3.5.1.3 *Transition*

A transition is a sequence of actions which occurs when a MML operational procedure changes from one state to another as a reaction to an input.

3.5.1.4 *Decision*

A decision is an action within a transition which asks a question to which the answer can be obtained at that instant and chooses one of several possible paths to continue the transitions.

3.5.1.5 *Output*

Output is a set of data which is output by the MML operational procedure and which in turn is used as an input to the operational process.

3.5.1.6 *Task*

A task is any action within a transition that is neither a decision nor an output.

3.5.1.7 *Symbols and rules*

Symbols and rules are those defined in the SDL Z.100 series Recommendations.

3.6 *Formal representation of the syntactic structure of specific inputs and outputs*

The formal representation of the syntactic structure of specific inputs and outputs may be provided by the use of the existing syntax metalanguage in Recommendation Z.302. The use of the Backus Naur Form (BNF) has also been suggested as possibly being more effective. As advanced terminal capabilities are being considered by the MML Sub-Working Party, additional methods may be needed. The suitability of these methods must be studied further, and if possible, a single method recommended.

3.6.1 *Backus Naur Form (BNF)*

Inputs and outputs are defined as sequences of terminal elements and/or non-terminal elements.

Terminal elements are characters belonging to the MML character set as defined in Recommendation Z.314 and the syntactic elements as defined in Recommendations Z.314, Z.315 and Z.316. Syntactic elements are indicated by means of their name written with small letters between angular brackets (< and >).

Non-terminal elements are elements that must be further defined again as sequences of terminal and/or non-terminal elements. They are indicated by one or more words written with small letters between angular brackets (< and >).

3.6.1.1 *Notation*

Definitions are inciated by writing commands or non-terminal elements on the left hand of the symbol ::= (double colon, equal sign) and, on the right hand side, one or more sequences of terminal and/or non-terminal elements.

Alternative choices are indicated separated by | (vertical bar).

Terminal and non-terminal elements may be grouped together by using braces ({ and }). Repetition of these groups is indicated by means of two subscripts after the braces, one for the minimum, one for the maximum number of times the group may be repeated.

If a group of terminal and non-terminal elements is written between brackets ([and]) the group is optional.

For an example see Appendix III.

APPENDIX I

(to Recommendation Z.333)

Glossary of common terms used in the specification of the man-machine interface

This glossary of common terms is to be utilized where applicable by CCITT bodies when applying phases 1-3 of the methodology. It is expected to evolve as the methodology is applied to a wider range of areas. This document is not intended to constrain manufacturers' and administrations' choice of terms to represent these concepts at the actual man-machine interface.

It has been noted in Recommendation Z.332 that it is useful to view MML functions as *actions* upon *objects*. The concepts represented by the terms in the present collection are limited to action concepts. It is expected that as this glossary evolves, most action concepts will be defined here since they generally have use in other functional areas. Conversely, object concepts will generally be specific to a functional area and thus defined in the glossary associated with a functional area.

Among the concepts for actions that may be performed at the man-machine interface are concepts for which the proper object of the action is:

- data only
- equipment only
- either data or equipment.

These three categories of actions correspond to the three major divisions of this glossary.

A number of the concepts below are best understood and normally utilized in complementary pairs; these cases will be indicated by notation such as CREATE/DELETE.

The examples given below of uses for these terms that are marked with a single asterisk (*) come from the traffic measurements application in Annex A and with a double asterisk (**), from the maintenance of the circuits between exchanges application in Annex B to this Recommendation.

I.1 *Data management actions*

The term **data set** is defined to be a user-accessible set of one or more data items characterized by a particular use, and also by the constraints on data format and/or values that make it suitable for this use.

I.1.1 *CREATE/DELETE*

The following concepts concern user control of the existence of data sets within the system.

CREATE: Establish in the system a new data set.

Examples: CREATE A MEASUREMENT SET *, CREATE AN OBJECT LIST *.

DELETE: Eliminate a data set from the system.

Examples: DELETE A MEASUREMENT SET *, DELETE AN OBJECT LIST *.

I.1.2 *CHANGE and EDIT*

The modification of data is normally accomplished by one of two basic methods. The first method (CHANGE) is through the use of functionally specific inputs and outputs intended to be used to modify particular data set types or even particular data items within those data sets. The second method of data modification (EDIT) allows the user to perform changes directly to a display of the data that is to be modified.

Taking this into account, CCITT organizations applying the methodology described in this recommendation should employ the term CHANGE for any data modification requirements, except in cases where the capability to EDIT would have clear advantages, such as in the example given below.

CHANGE: Modify specified data items in a data set via an input or inputs intended for that purpose.

Example: CHANGE ANALYSIS THRESHOLDS **.

EDIT: Display a specified data set and subsequently modify the data set. A common system capability, e.g., editor, is normally used to support such an action.

Example: EDIT TRAFFIC DATA RECORDS.

I.1.3 *ACTIVATE/DEACTIVATE*

The creation of a data set does not necessarily imply that that data is immediately available for use by the system for its intended purpose. The following concepts make a previously created data set available or unavailable to the system.

ACTIVATE: Initiate a system process that requires preliminary data entry, or make a previously entered data set available to the system for its intended use.

Examples: ACTIVATE A MEASUREMENT *, ACTIVATE A ROUTINE TEST **.

DEACTIVATE: Terminate a system process initiated by an ACTIVATE action, or make a data set unavailable for use by the system.

Examples: DEACTIVATE A MEASUREMENT *, DEACTIVATE A ROUTINE TEST **.

I.1.4 *FILTER and SORT*

These concepts allow the user to manipulate data to be subsequently accessed or stored.

FILTER: Form a subset of a data set consisting of all data items in the set meeting specified criteria. The original data set is unaffected by this action.

Example: FILTER TROUBLE OR RESTORAL REPORTS **.

SORT: Rearrange the order of a data set according to specified (or default) criteria. The contents of the original set is not affected by this action, only its order.

Example: SORT A FILE OF NAMES (e.g. in alphabetical order).

I.1.5 *INTERROGATE and BROWSE*

The concepts below describe system actions that allow user access to specified portions of the data that has been created by the user or by the system.

INTERROGATE: Provide a display of the current values of the items in one or more data sets.

Examples: INTERROGATE A MEASUREMENT *, INTERROGATE A MEASUREMENT TYPE *.

BROWSE: Display sequentially the current values of items in a data set. The user may examine the data items in either the forward or backward direction.

Example: BROWSE REPORT FILES **

I.1.6 *INPUT/OUTPUT and ROUTE*

The concepts in this section concern the transfer of data from one location to another.

INPUT: Enter data by means of a user terminal into the system.

Example: INPUT TROUBLE OR RESTORAL REPORT **.

OUTPUT: Transfer specified data from the system to the user terminal (e.g. VDT, printer).

Example: OUTPUT SUMMARY REPORT **.

The distinction between OUTPUT and INTERROGATE (1.5) is that INTERROGATE simply gives a read-back of user-created data, whereas OUTPUT refers to data upon which the system itself has acted in some way, e.g. reports.

ROUTE: Instruct the system that any subsequent messages, classes of data, or message types indicated should be output to specified media.

Example: ROUTE OUTPUT OF REPORTS **.

I.2 *Equipment Management Actions*

I.2.1 *REMOVE/RESTORE and SET*

Equipment units can often simply be placed either out of service or in service under software control. The pair REMOVE/RESTORE represents this pair of actions. Manipulation of the status of objects with a more complicated set of maintenance states is expressed by the system action SET, which normally also covers the out of service and in service states. The REMOVE/RESTORE pair is used frequently and is sufficient for a large range of equipment, hence is singled out here as an important special case of the SET action.

REMOVE: Take specified equipment units out of service. The system still retains knowledge of the units so that they may be returned to service by the RESTORE action defined below, automatic recovery, or manual override.

Example: REMOVE CIRCUIT **.

RESTORE: Return specified units to service.

Example: RESTORE CIRCUIT.**

SET: Place equipment in a specified state (number of states > 2). Possible states include in service and out of service.

Example: SET EQUIPMENT UNIT.

I.2.2 *ALLOW/INHIBIT*

Modern systems (e.g. for maintenance or control) utilize many system functions which occur automatically or dependent only upon the detection of certain conditions. Often it is essential to be able to instruct the system *not* to perform these functions, even should the appropriate set of conditions arise. The complementary capability to return the automatically controlled function to its normal state must then also be provided.

ALLOW: Permit specified system actions, system responses or functions to occur. These functions may be inhibited by system design or by the INHIBIT system action defined below.

Example: ALLOW THRESHOLD **.

INHIBIT: Prevent the specified system actions, system responses or functions from occurring. These functions may normally be allowed by the system design, or by the ALLOW action defined above.

Example: INHIBIT THRESHOLD **.

I.3 *Management actions that may apply to data or equipment*

INITIALIZE: Put specified data or equipment into a predefined initial (normal) condition or value.

Examples: INITIALIZE THRESHOLD COUNTER **, INITIALIZE OUTPUT DEVICE.

APPENDIX II

(to Recommendation Z.333)

Procedure description example

The job of "create a new traffic measurement" is described as a procedure in which two different SDL processes, the user process and the system process, are shown.

Only the relevant aspects of the procedure are represented in the diagrams; some features are omitted such as a rejection output due to syntactic errors and related correction procedures etc., which are common to the other procedures.

For further understanding of this example, see Annex A to this Recommendation.

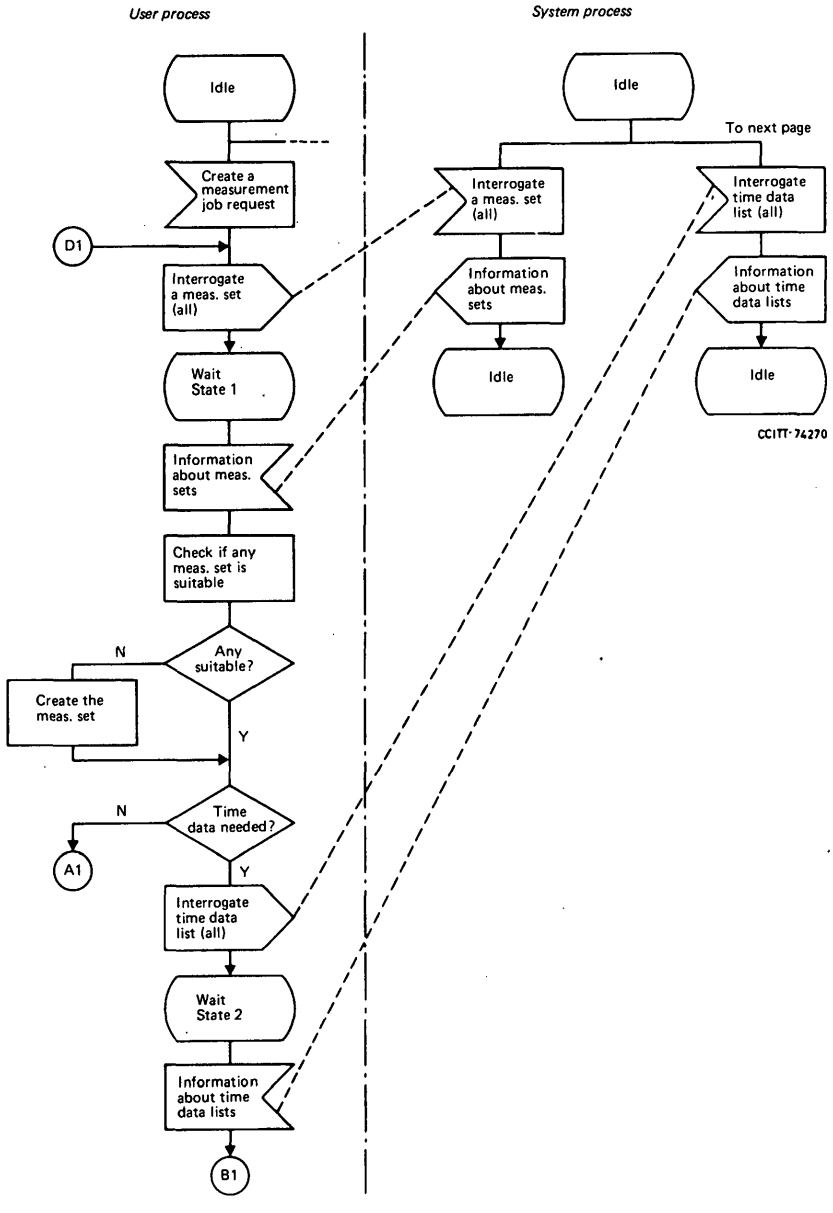


FIGURE II-1a/Z.333

Procedure description example

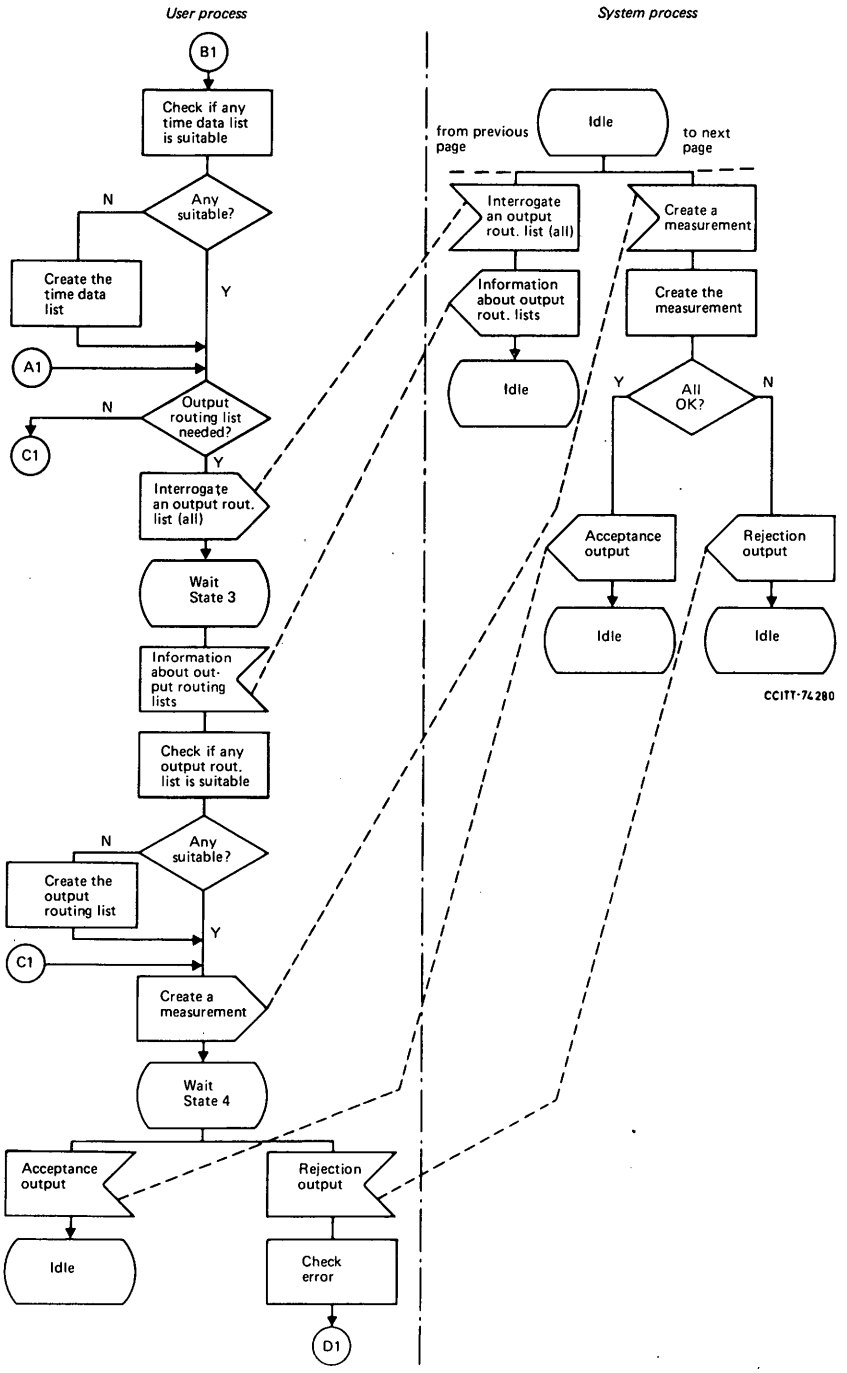


FIGURE II-1b/Z.333

Procedure description example (Continued)

APPENDIX III

(to Recommendation Z.333)

Examples of the use of the Backus Naur Form (BNF)

Applying the BNF metalanguage described in § 2.6.1 to the traffic measurement function example in Annex A of this Recommendation (Figures A-15 and A-20), the following BNF examples are derived with the assumption of a one to one relationship between the MML function and associated command:

a) *Function "create an object list":*

<create an object list> ::= <command code>:
<object list identity>
{,<list of objects of one type>}$^{1-N}$;

<object list identity> ::= <parameter name> = <symbolic name>

<list of object of one type> ::= <type of objects> = <objects identity>

<type of objects> ::= <parameter name>

<object identity> ::= <decimal numeral> {{&<decimal numeral>}|{&&<decimal numeral>}}$^{O-N}$
<symbolic name> {&<symbolic name>}$^{O-N}$

b) *Function "delete an object list"*

<delete an object list> ::= <command code>:
<list of identities of object list>;

<list of identities of object list> ::= <parameter name> =
<symbolic name> {&<symbolic name>}

ANNEX A

(to Recommendations Z.332 and Z.333)

Example of documents A, B, C, D and G for traffic measurements administration functions

A.1 *Document A*

A.1.1 *Introduction*

Traffic measurements administration functions are related to data production, collection and output.

These data are achieved by means of periodic and non-periodic traffic measurements carried out in the telecommunications system(s) and are output by the system(s) in a suitable form.

The traffic measurement result outputs should contain the measurement results and general information about the measurement itself and about the system which performed the measurement in order to ease the results analysis. Moreover, they should contain information summarizing the production of output blocks for checking purposes.

A.1.2 *List of Class B Functions*

1) Performing measurements of traffic parameters.
2) Scheduling traffic measurements execution and results output.
3) Managing measurements' data.
4) Retrieving measurements' data.

A.1.3 *List of jobs*

1) To create new measurements or measurements components and to modify old ones, by defining the entities to be measured and the objects and parameters of the measurements themselves (what and how to measure):
   - purpose of this job is to create and/or modify a set of data which is used by the system to perform a measurement in a given way;
   - the system is supposed to record the set of data of the measurement, and to check their static correctness;
   - the user is supposed to input/change all relevant data. The modification of data may be performed by means of different procedures, depending whether or not those data are related to activated measurements;
   - the complexity of the job could be high depending on the amount of data to be input;
   - the frequency of the job is low;
   - the job is supposed to be performed at exchange and/or OMC level.

2) To delete obsolete measurements or measurements components:
   - purpose of the job is to delete measurements of no further use or measurements components to release the employed resources;
   - the system is supposed to delete the data related to a specified measurement if the measurement is not active. The system is supposed to delete a measurement component only if it is not an active measurement component;
   - the user is supposed to input the identities of measurements or measurement components to be deleted;
   - the complexity of the job is low;
   - the frequency of the job is low;
   - the job is supposed to be performed at exchange and/or OMC level;

3) To define the measurement results output routing and scheduling (where and when the results will be output):
   - purpose of the job is to define where the measurement outputs have to be routed to and when they should be output;
   - the system has to route the measurement outputs toward the recording media or toward other systems specified, according to the results output schedule;
   - the user has to input the identity of the destination of the output and the results output schedule to be followed by the system;
   - the complexity of the job is low;
   - the frequency of the job is medium;
   - the job may be performed at exchange and/or OMC level.

4) To activate and to deactivate measurements (when to measure):
   - purpose of the job is to activate and/or to deactivate the performance of the measurements that have been previously defined;
   - the system is supposed to activate and/or deactivate a measurement and to start the production of the results;
   - the user is supposed to input the date and time of activation and/or deactivation;
   - the complexity of the job is low;
   - the frequency of the job is medium;
   - the job may be performed at exchange and/or OMC level.

5) To retrieve different kinds of information related to traffic measurements:
   - purpose of the job is to get information on measurements previously input in the system(s) in order to be aware of the current situation;
   - the system is supposed to output in suitable formats and on the selected device(s) the information requested;
   - the user is supposed to input the identity of the items to be interrogated and to select retrieving criteria;
   - the complexity of the job is low;
   - the frequency of the job is medium;
   - the job may be performed at exchange and/or OMC level.

A.2 *Document B*

A.2.1 *Introduction*

The traffic measurement model given in § A.2.3 is based upon a more general measurement model as given in § A.2.2.

A.2.2 *Measurement model*

A *measurement* is identified by three basic elements: *time, entities, object.*

Time includes all the necessary information to define the start, the duration and periodicity of a certain measurement.

Entities describe the quantities for which data collection must be performed with a certain measurement, e.g. traffic flow, number of call attempts, congestion time.

Objects are intended as individual items within each object type on which the measurements are performed. Examples of object types are subscriber lines, circuits, circuit groups, elements of switching networks, geographical areas with their corresponding dialled code. The definition of measurements is based on an abstract model which contains the definition of a *measurement matrix*, see Figure A-1, in which each row represents one uniquely definable entity, e.g. number of call attempts and each column represents a uniquely definable object type, e.g. incoming junction group, see Figure A-2.

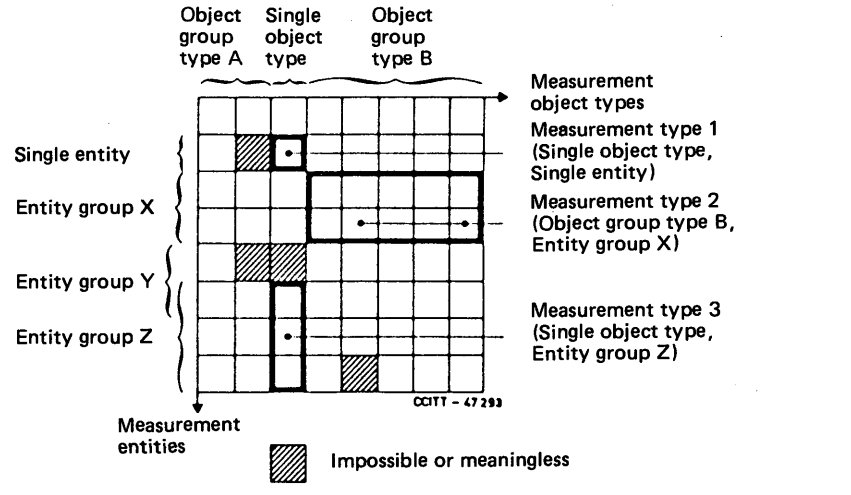


FIGURE A-1

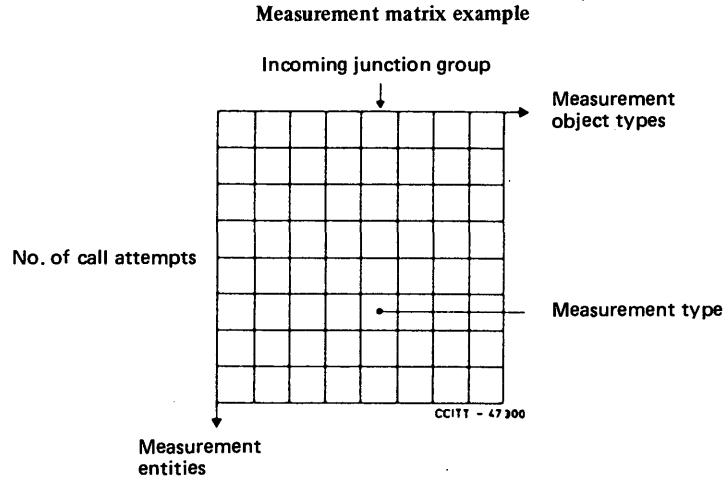


FIGURE A-2

Application of measurement matrix to a Traffic Measurement

A certain combination of entities and object types corresponds to certain entries in the measurement matrix and forms a *measurement type.* It is recognized that part of these measurement types may be standardized while the rest of them seem to be system and/or administration dependent. It should be noted that all the entries in the measurement matrix cannot be used because some of them will be impossible (e.g. call congestion on an incoming trunk) and some others may be more or less meaningless. A single object is defined by its type and/or its individual object identity. In some measurement types, the number of objects is fixed. In other types one can choose for the actual measurement some or all of the allowed objects by administration MML commands. Chosen (selected) objects form an object list.

The structure of division of object types and entities is open ended in such a way that any new object type or entity may be added.

A.2.3 *Traffic measurement model*

A.2.3.1 *Basic types of measurements*

Two basic types of measurements are envisaged (see Figure A-3). The first type (A) is measurement of undetermined duration while the second one (B),is intended to be performed only for a predetermined duration. The start of a measurement may be intended as instantaneous or delayed for a defined time duration t_1 from the activation of the measurement. Since the stop time of a measurement of type A is not given when the measurement is activated or created, it has to be given during the measurement unless the measurement is intended to go on forever.

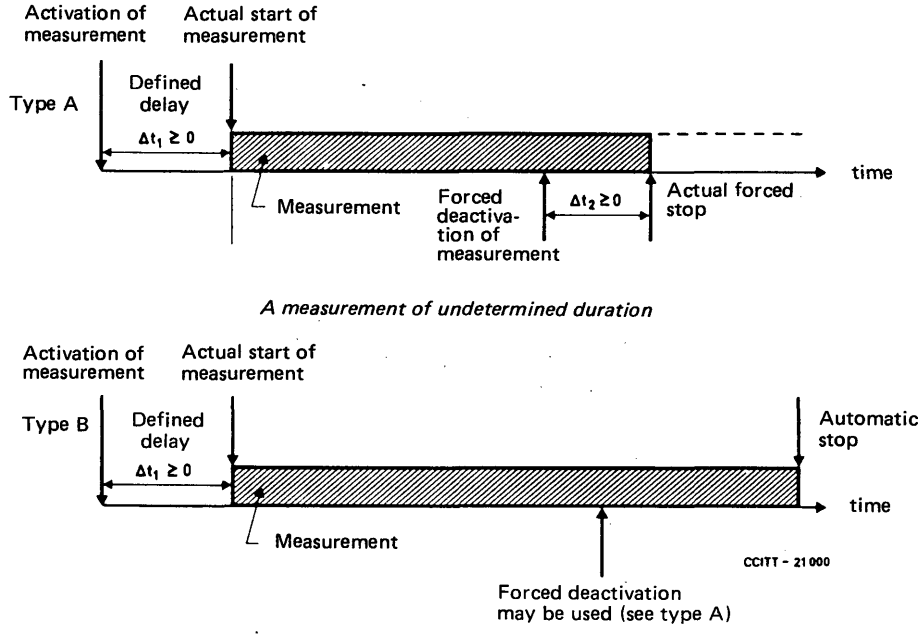


FIGURE A-3

Measurement duration types

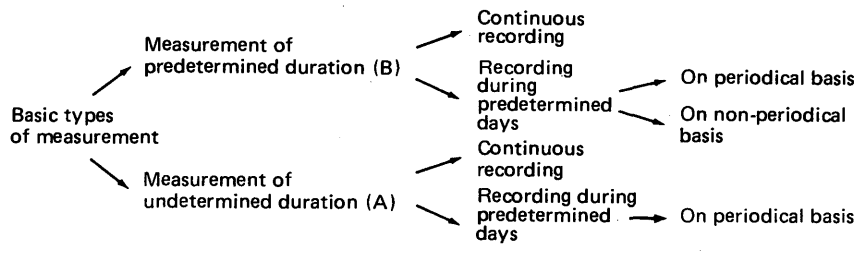


FIGURE A-4

Basic types of measurement

From the deactivation point of time there may be a defined delay of t_2 before the measurement is stopped. In the creation of a measurement a start time may optionally be provided, in which case for that particular measurement, the activation function is not necessary.

Time parameters needed to control a measurement can be divided into three groups:

1) measurement type dependent time parameters [interval parameters of a measurement type, e.g. sampling interval[2)]];
2) measurement dependent time parameters (e.g. time parameters which define the periodicity of measurement);
3) measurement independent time parameters (e.g. time parameters which are related to the actual start or stop of a certain measurement in activation and deactivation functions).

A.2.3.2 *Traffic measurement structure*

A traffic measurement (in the following called measurement) consists of:

- measurement set information,
- time information,
- output routing and scheduling information (output parameters).

Measurement set information, time information, output routing and scheduling information may be completely or partially pre-defined (initially provided by the supplier but changeable via MML inputs) or fixed (not changeable via MML inputs). The MML functions described for traffic measurements administration are intended to be supported to the extent that there is a need for user manipulation of the identified information items.

A.2.3.2.1 *Measurement set information*

Measurement set information consists of one or several selected measurement types with defined objects (object lists) and measurement type dependent parameters (e.g. sampling interval, number of events of a certain category, destination codes, etc.).

Note that for traffic administration purposes measurement types are fixed in the system by the design and implementation at a given moment in time and cannot be created, removed or changed by MML commands; only later supplier releases may change these types according to new requirements or they may be created, changed or deleted by MML commands as part of a system extension or upgrade operation. Therefore measurement types are not further defined in this example.

A.2.3.2.2 *Time information*

Measurements of types A and B may involve continuous recording or recording on predetermined days (recording days).

For measurements performing continuous recording only the start data is needed.

For recording on predetermined days, these days are determined on a periodical basis (periodicity pattern) in case of measurements of undetermined duration. For measurements of predetermined duration, the recording days are determined on a periodical basis or on a non-periodical basis (dates of recording days). These possibilities are summarized in Figure A-4.

Time data are defined at three main levels, as shown in Figure A-5.

Measurement level contains information about either:

- dates of recording days (in case of a non-periodical measurement). The start and stop date of the measurement are implicitly defined by the dates of the first and the last recording day. No activation function may be needed in this case;
- periodicity pattern (in case of periodical measurement) of recording and non-recording days.

Recording day level contains information about the start time and stop time for recording periods within a recording day. No overlap of recording periods is allowed for the same measurement.

Recording period level contains information about the periodicity of the data collection controlled by the results accumulation period. The results accumulation period can be shorter than the recording period; in that case more than one set of data is collected for each of the recording periods to be routed toward the output media according to the results output schedule.

---

2) Sampling interval, the time interval between two consecutive samplings.

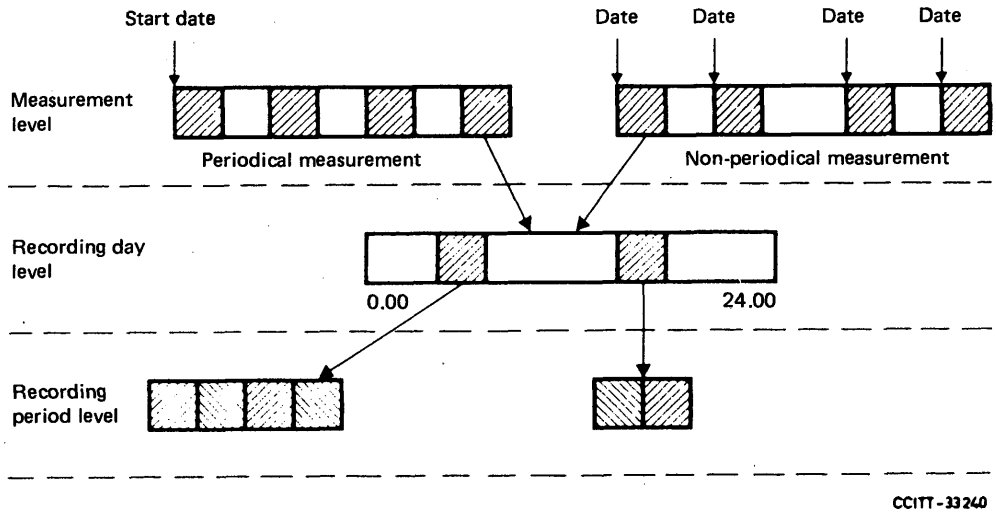


FIGURE A-5

Time information

A.2.4 *Other information*

A.2.4.1 *Measurement outputs' contents and procedures*

Activation of a traffic measurement causes the output of measurement results with the following procedures.

The produced output is routed toward the media specified in the output routing list associated to the measurement, e.g. printers, magnetic tapes, data links, system output files, etc. The output is made according to the output schedule.

The measurement results output is done according to time data related to the measurement. A measurement results output is made with the following logical blocks:

a) a "begin block" which contains measurement data, parameters i.e. measurement types data, time data, output data and data of interest related to the exchange configuration;

b) one or more "result blocks", one for each result output period, which contain the measurement results;

c) an "end block" which contains a general summary about the performance of the measurement, i.e. number of result blocks, number of interruptions of the measurement and the causes of the deactivation of the measurement (scheduled or forced).

If during the performance of the measurement, the measurement is suspended (e.g. due to a system crash) the measurement results output must be continued after the system restart with a new output of the begin block. This continuation may be accomplished automatically by the system or by user action. The system should notify the user via an output if the latter case applies.

The relationship between time data for the result accumulation period and time data defining the results output schedule is system or even measurement dependent and it is not considered herein.

A.3 *Document C*

A.3.1 *Introduction*

The list of the MML functions identified to support the traffic measurements administration is presented in the two forms below: a list and Figure A-6.

A.3.2 *List of MML functions*

1) *Creation*
   - create a measurement;
   - create a measurement set;
   - create an object list;
   - create a time data list;
   - create an output routing list;
   - create a results output schedule.

2) *Deletion*
   - delete a measurement;
   - delete a measurement set;
   - delete an object list;
   - delete a time data list;
   - delete an output routing list;
   - delete a results output schedule.

3) *Activation*
   - activate a measurement.

4) *Deactivation*
   - deactivate a measurement.

5) *Interrogation*
   - interrogate a measurement;
   - interrogate a measurement set;
   - interrogate a mesurement type;
   - interrogate an object list;
   - interrogate time data list;
   - interrogate an outputs routing list;
   - interrogate a results outputs schedule

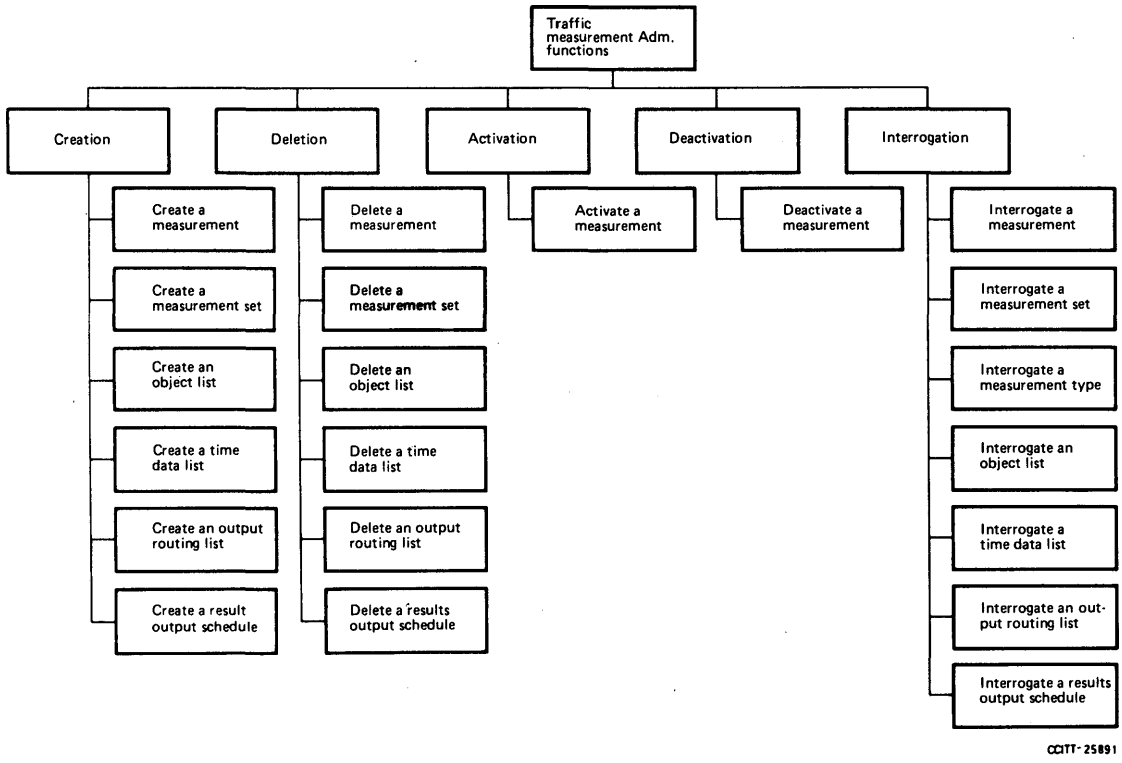


FIGURE A-6

MML functions for traffic measurement

A.3.3 *Modification functions*

The modification of measurements and measurements components data should be allowed, but no specialized modification functions are defined, provided that a general facility for editing data is contained among System Control Functions, which remain to be developed.

A.4 *Documents D*

A.4.1 *Introduction*

All the information entities needed for the MML functions previously derived have been identified and are reported in Document D by means of diagrams representing each MML function information structure (Figures from A-8 to A-30). In particular, the information structure diagrams of the measurement outputs are given in Figures from A-31 to A-36.

An overview of measurement data structure is also given in Figure A-7.

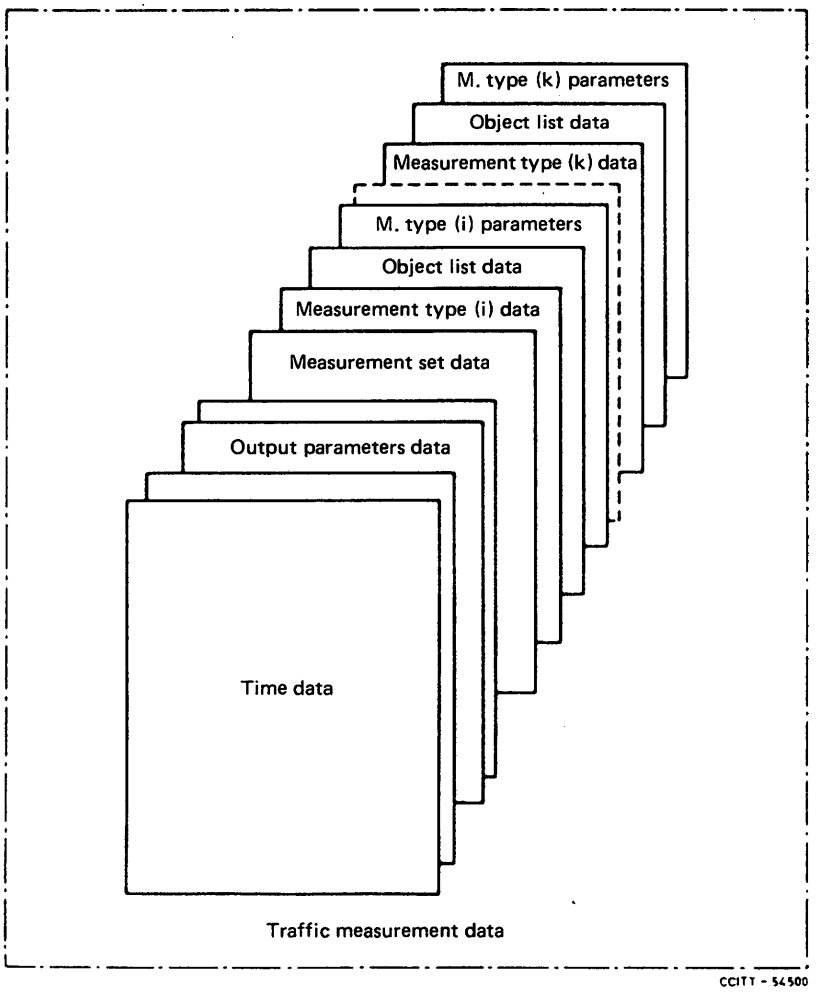


FIGURE A-7

Traffic measurement data overview

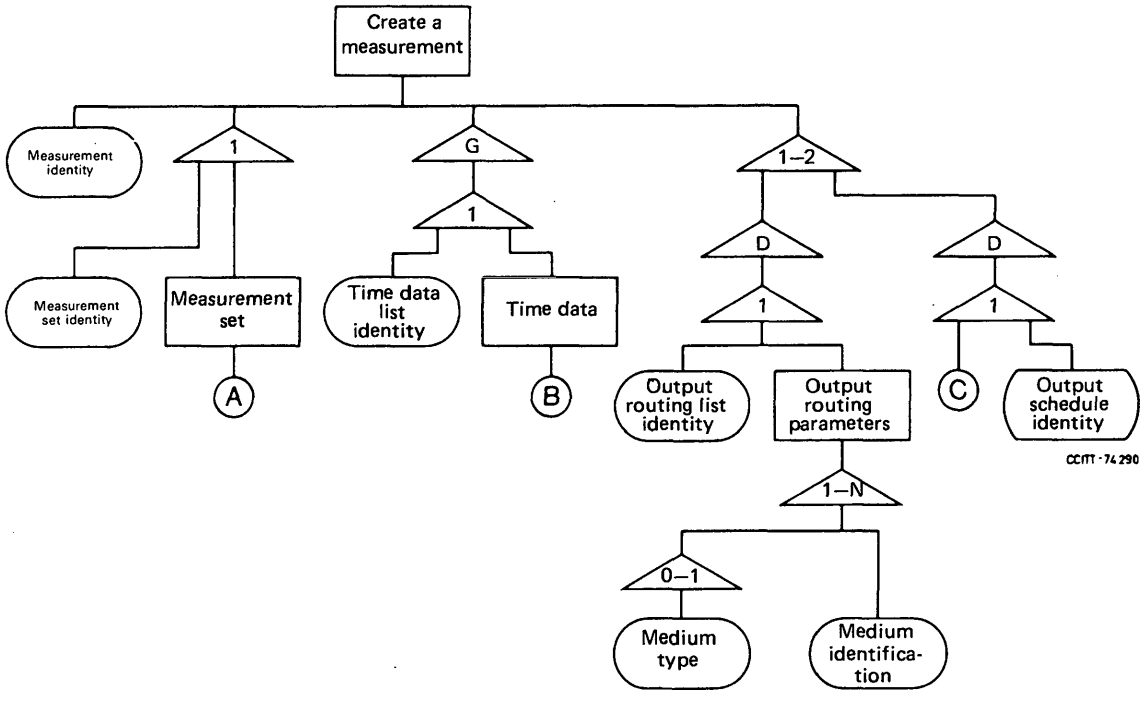
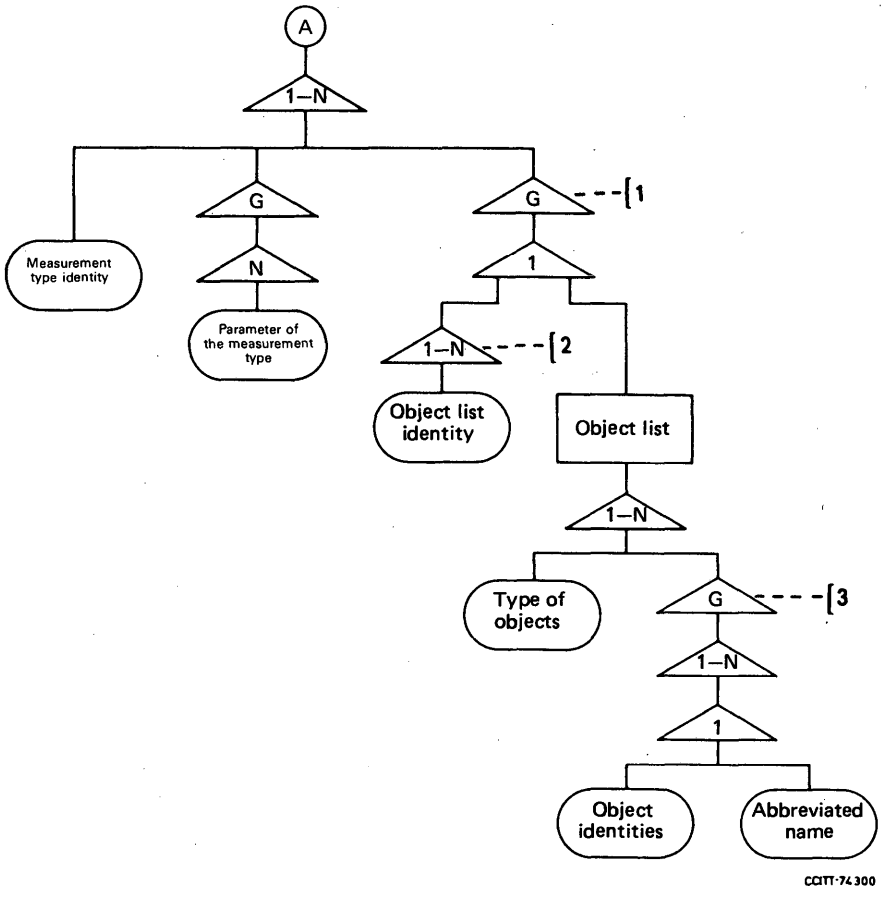


FIGURE A-8

Create a measurement



Note 1 – No object list needed if the measurement type implies global measurements on a certain object type.

Note 2 – Multiple object lists imply a resulting merged list.

Note 3 – Zero is meaningful only for the measurement types implying global measurements on selectable object types.

FIGURE A-9

Create a measurement (Continued)

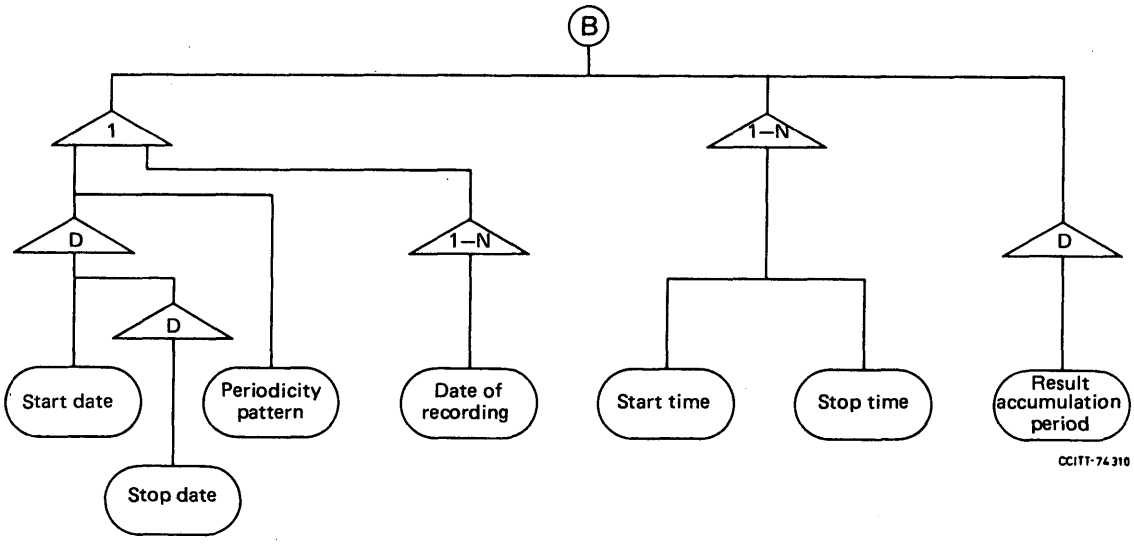


FIGURE A-10

Create a measurement (Continued)

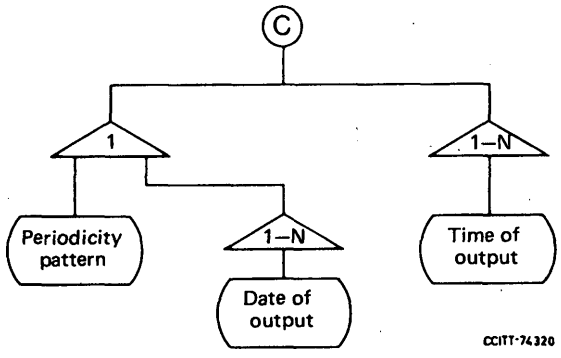


FIGURE A-11

Create a measurement (Continued)

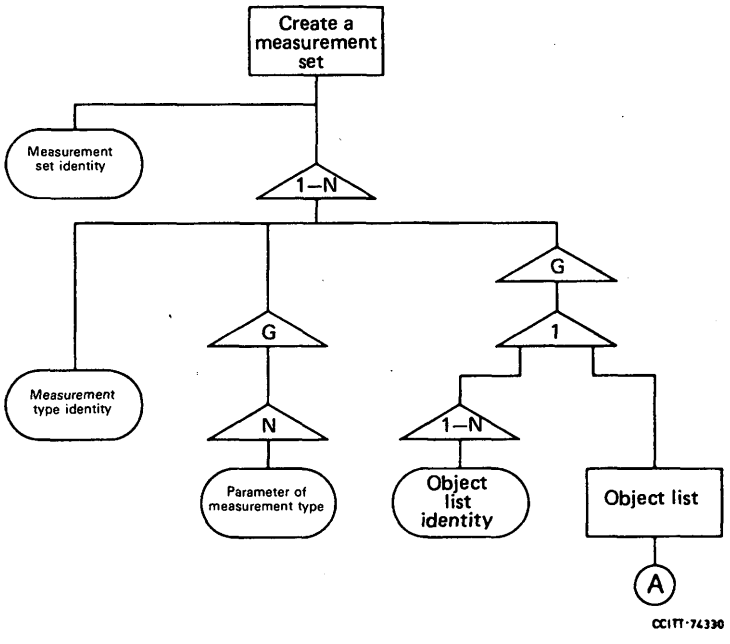


FIGURE A-12

Create a measurement set

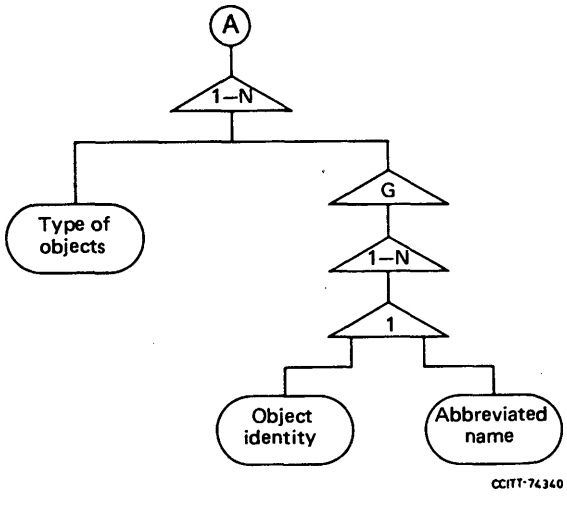
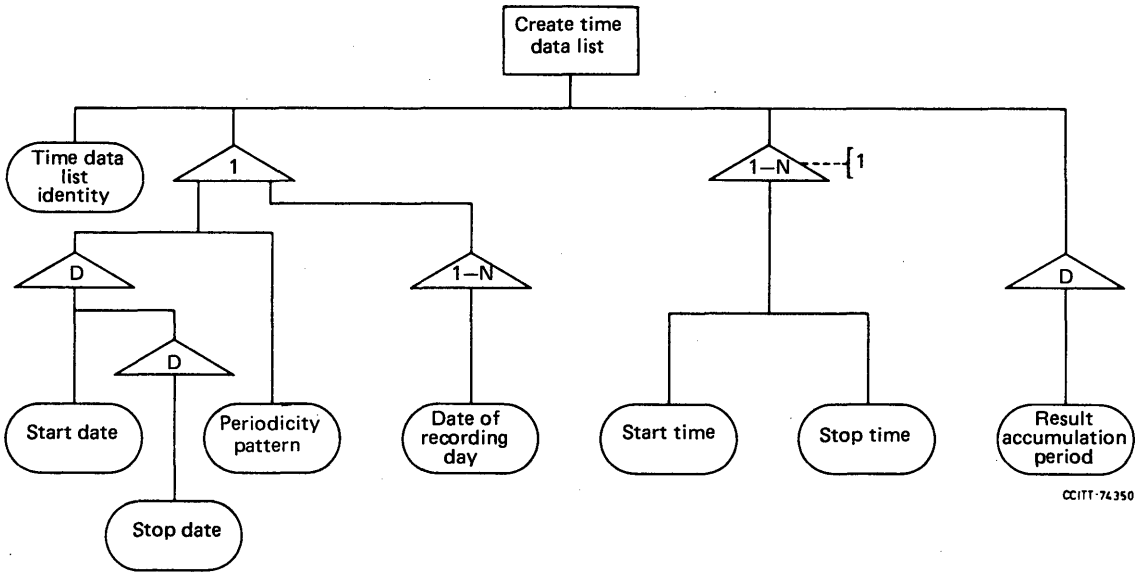


FIGURE A-13

Create a measurement set (Continued)



Note 1 – Each recording period is defined by giving its start and stop times.

FIGURE A-14

Create a time data list

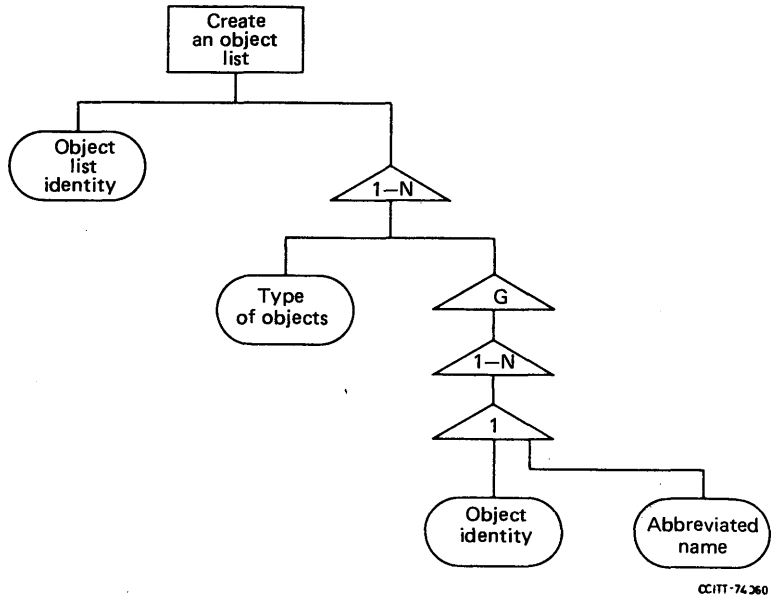


FIGURE A-15

Create an object list

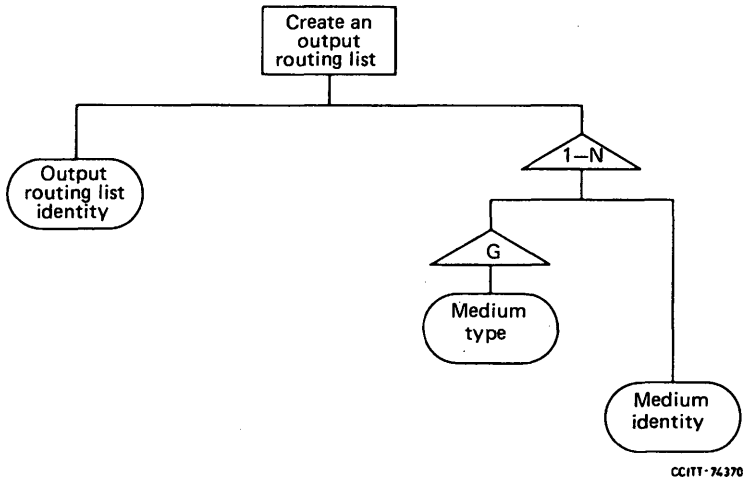
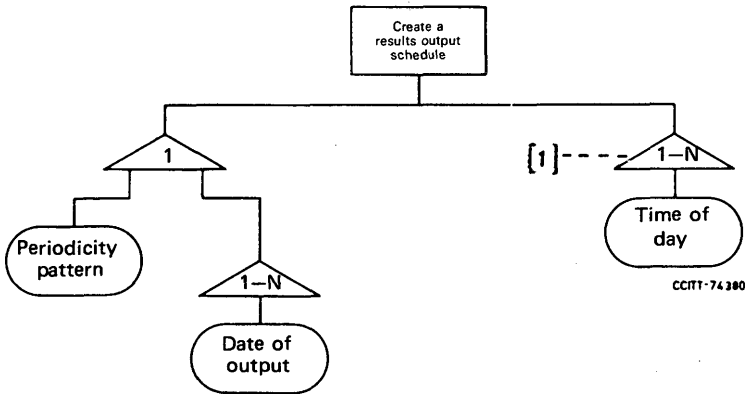


FIGURE A-16

Create an output routing list



Note 1 – Set of times may depend on output day.

FIGURE A-17

Create a results output schedule

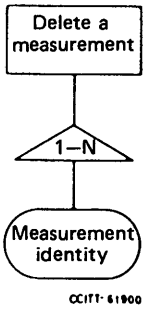


FIGURE A-18

Delete a measurement

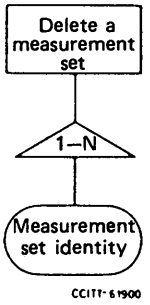


FIGURE A-19

Delete a measurement set

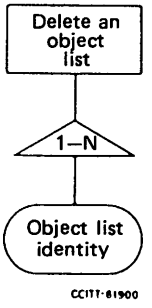


FIGURE A-20

Delete an object list

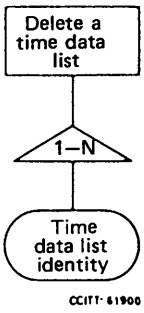


FIGURE A-21

Delete a time data list

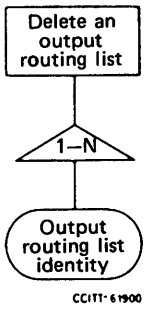


FIGURE A-22

Delete an output routing list

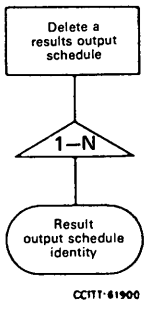


FIGURE A-23

Delete a results output schedule

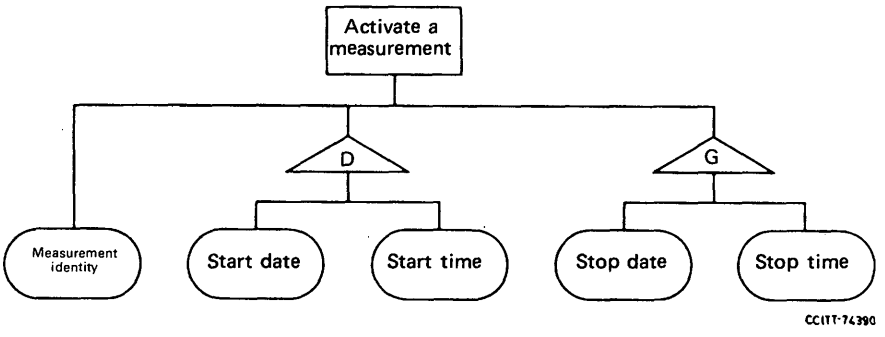


FIGURE A-24

Activate a measurement

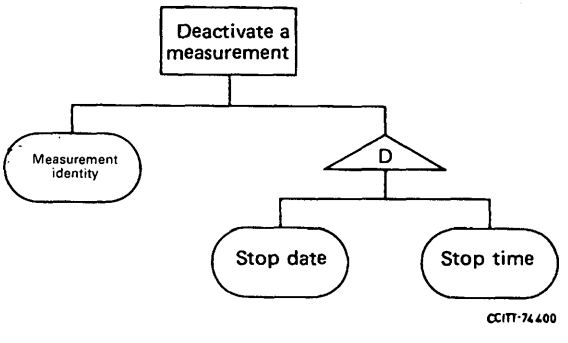
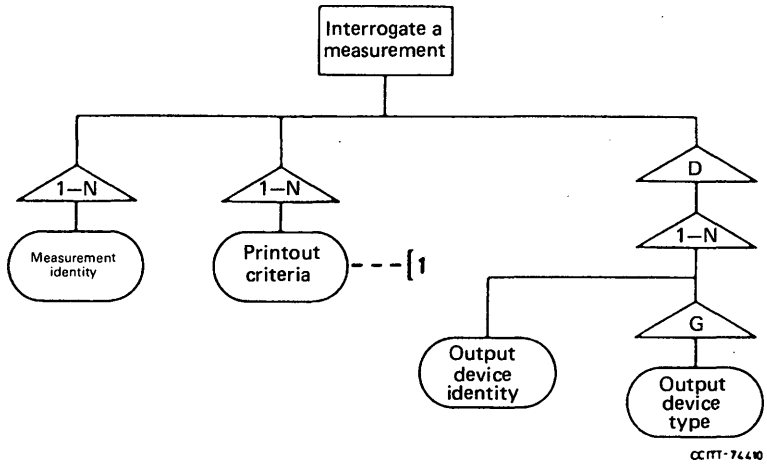


FIGURE A-25

Deactivate a measurement

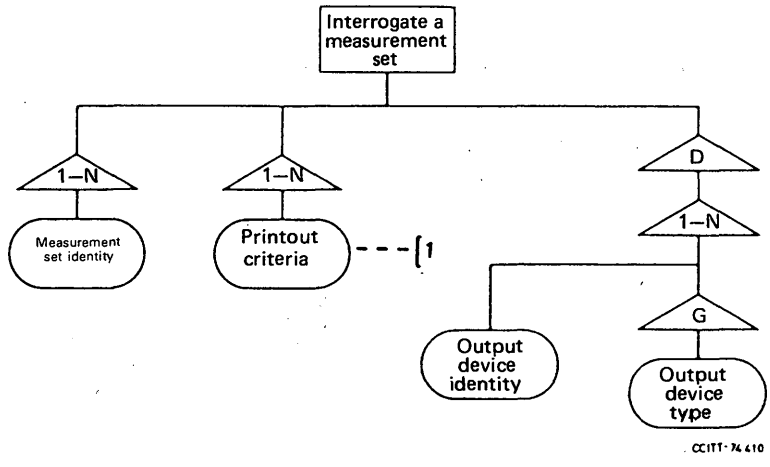


Note 1 – Possible parameter values:

- object list,
- object list identity,
- measurement types,
- parameters of measurement types,
- measurement set,
- measurement set identity,
- time data,
- time data list identity,
- output routing list,
- output routing list identity,
- output schedule,
- output schedule identity,
- status (activated or not activated).

FIGURE A-26

Interrogate a measurement

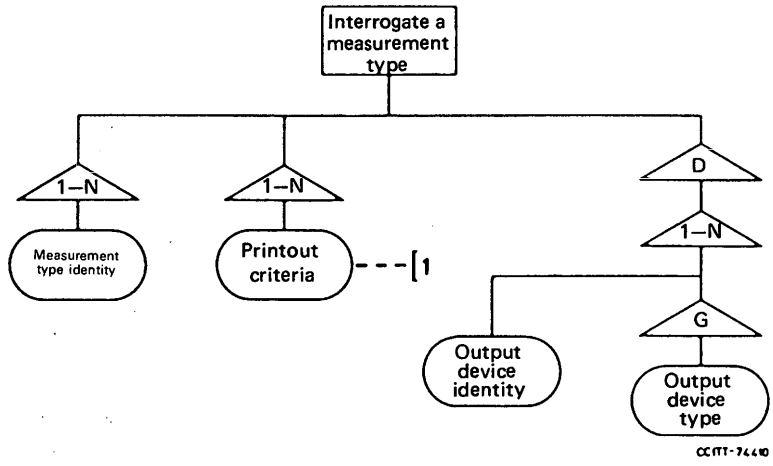


Note 1 – Possible parameter values:

- measurement type identities,
- parameters and associated values,
- object list,
- measurements utilizing the identified set.

FIGURE A-27

Interrogate a measurement set

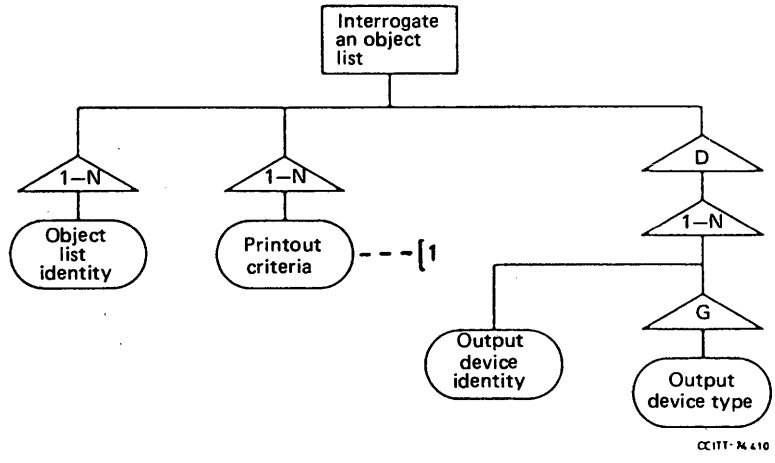


Note 1 – Possible parameter values:

- list of parameters of the measurement type,
- object lists associated with the measurement type,
- sets utilizing the measurement type,
- measurements utilizing the measurement type.

FIGURE A-28

Interrogate a measurement type

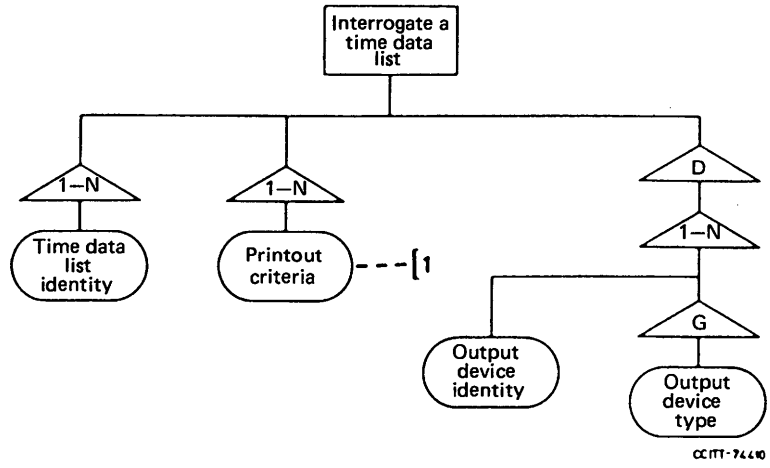


Note 1 – Possible parameter values:

- object type,
- object type and individual object identities,
- measurement utilizing the object list.

FIGURE A-29

Interrogate an object list

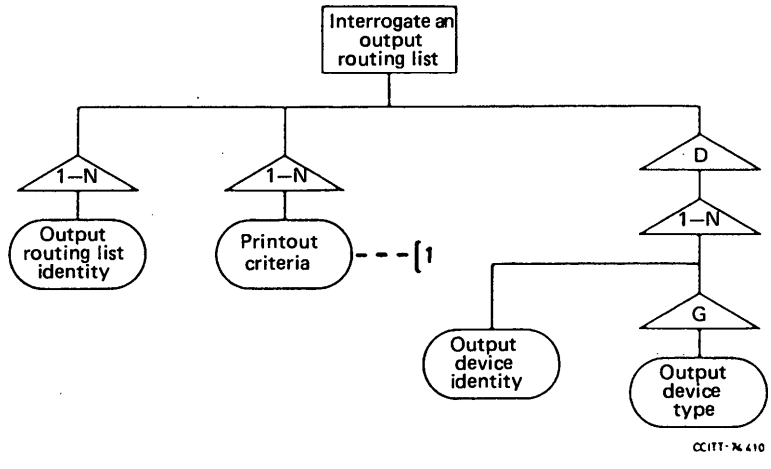


Note 1 – Possible parameter values:

– time data,
– measurement utilizing the time data list.

FIGURE A-30

Interrogate a time data list

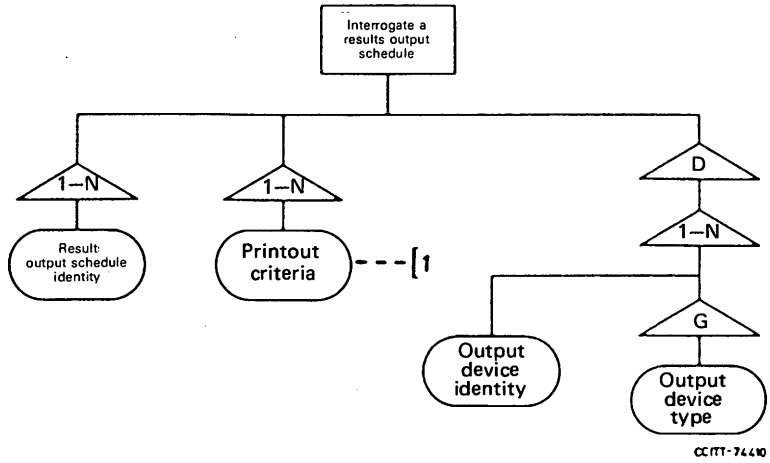


Note 1 – Possible parameter values:

– output routing data,
– measurement utilizing the output routing data list.

FIGURE A-31

Interrogate an output routing list



Note 1 – Possible parameter values:

– results output schedule data,
– measurement utilizing the results output schedule.

FIGURE A-32

Interrogate a results output schedule

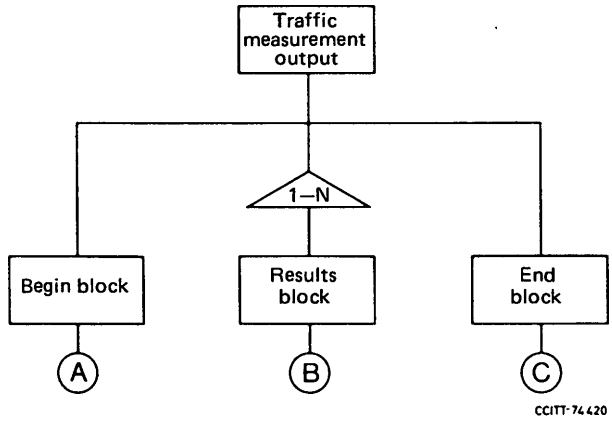
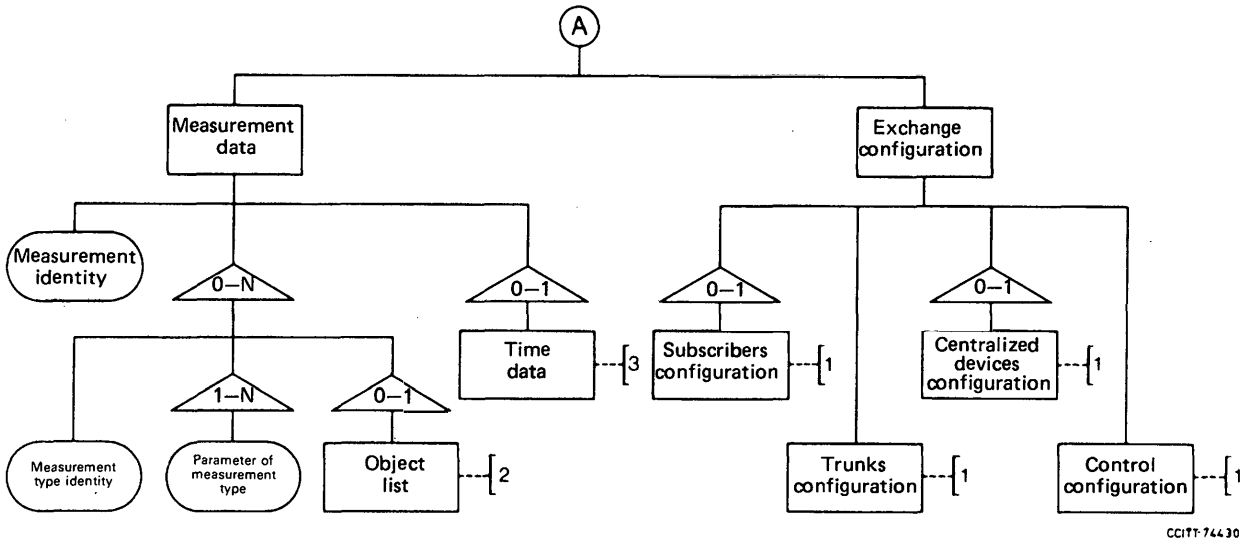


FIGURE A-33

Traffic measurement output



Note 1 – Not expanded further.

Note 2 – See Figure A-15.

Note 3 – See Figure A-14.

FIGURE A-34

Traffic measurement output (Continued)

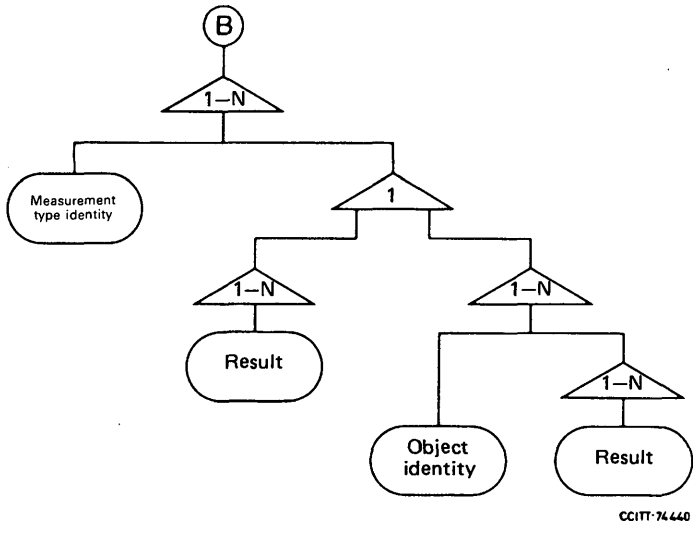


FIGURE A-35

Traffic measurement output (Continued)

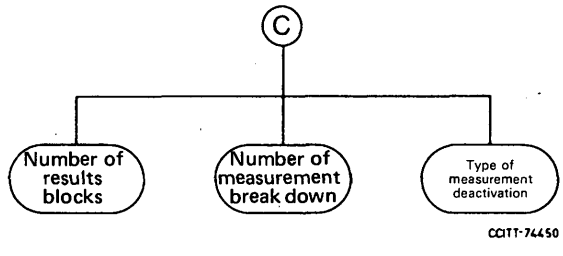


FIGURE A-36

Traffic measurement output (Continued)

A.5 *Document G*

A.5.1 *Introduction*

Terms related to documents from A to D are presented below. The production of other documents may require additional terms.

A.5.2 *Glossary of used terms*

Recording	Performance of the operations implied by the measurement entities in order to collect the required data.
recording day	Day when a recording is performed. Several recording periods are allowed within a recording day. No overlap of recording periods is allowed for the same measurement. Each recording period can have a different length.
start date	Start day for the measurement execution.
stop date	Stop day for the measurement execution.
periodicity pattern	A pattern which indicates which days are recording (or results output) days and which are not. The start day positions this time span. Once activated, the execution of the measurements (or of the results output) is performed according to this pattern, until disabled by a deactivation command.
start time	Time for beginning the recording period in a recording day.
stop time	Time for terminating a recording period in a recording day.
recording period	A period of recording during a recording day.
results accumulation period	Time interval within a recording period during which the required measurement entities are processed and at the end of which results are stored for immediate or later output.
output parameters	Data determining output routing and scheduling.
results output routing	Data defining the media to which results output is to be directed.
results output schedule	Data specifying a set of days (or a periodicity pattern) and of times during these days when the output of the results is to be made.

ANNEX B

(to Recommendations Z.332 and Z.333)

Example of documents A, B, C, D and G for Maintenance of circuits between exchanges and associated equipment

B.1 *Introduction*

The purpose of the maintenance is to detect, localize and repair failures. Failures can be detected by means of different methods, namely:

- test and measurements;
- observation and supervision;
- analysis.

All of these methods are useful in covering the variety of maintenance situations encountered. In order to aid in localizing the detected troubles to the failed equipment, so that repair may be instituted, maintenance actions such as on demand tests and measurements are required. Also in order to administer and control maintenance actions, certain supervision, observation, analysis and test/measurements functions along with their associated information and/or data are required.

The maintenance functions are based on the general network model reported in Figure B-1.

The functions for maintenance of circuits and associated equipments between exchanges may be divided into the following five functional sub-areas:

i) test and measurement;

ii) observation and Supervision;

iii) status of circuits and associated equipment;

iv) analysis of maintenance data;

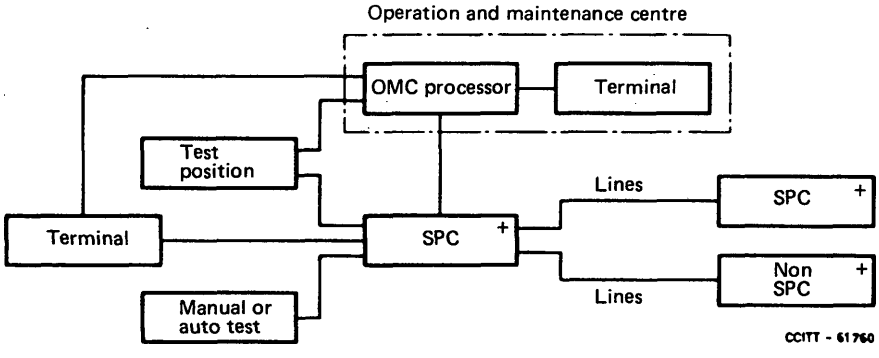
v) maintenance reports.

The remainder of this Annex is organized into five divisions corresponding to these sub-areas.

The lists of jobs included in the document A of these divisions are quite arbitrary. Certain jobs could be combined with other jobs to make a larger single job, while others could be broken down into smaller independent jobs. The key is not how many jobs are listed or what they are named, but after the jobs are listed, that they:

a) cover all the "maintenance jobs" that are needed for the maintenance of circuits between exchanges, and

b) allow all the required MML functions to be derived.



Test of single line and associated equipment

OMC Operation and maintenance centre.
SPC Stored programme control.
\+ International, toll or local exchange.

FIGURE B-1

Network model

B.2 *Tests and measurements*

B.2.1 *Document A*

B.2.1.1 *Introduction*

Maintenance tests and/or measurements may be performed on demand and routine basis, according to the maintenance strategies.

B.2.1.2 *List of class B functions*

B.2.1.2.1 Test/measurements of one circuit or a group of circuits and associated equipments.

B.2.1.3 *List of jobs*

B.2.1.3.1 *To plan a routine test/measurement*

- The purpose of the job is to create (or change) a list of tests and test programmes containing all of the data necessary for the tests/measurements to be run successfully and to identify the objects on which the tests/measurements are to be run.
- The system is supposed to record all of the necessary data and create (or change) required test/measurement sets.
- The user is supposed to introduce all the needed data.
- The complexity of the job may be high depending on the amount of data to be introduced.
- The frequency of the job is very low.
- The job is supposed to be performed at exchange and/or OMC level.

B.2.1.3.2 *To define/change/delete the schedule of routine tests/measurements*

- The purpose of the job is to schedule new (or change/delete existing) routine tests/measurements depending on the number of circuits to be tested/measured and the availability of test equipments and test circuits.
- The system is supposed to schedule (or change/delete) the requested tests/measurements according to the schedule input by the user.
- The user is supposed to input the test/measurement types and related data (test/measurement sets information) and time parameters, such as start time, stop time, etc., in order to obtain the required schedule.
- The complexity of the job is medium.
- The frequency of the job is low.
- The job is supposed to be performed at exchange and/or OMC level.

B.2.1.3.3 *To activate the execution of routine tests/measurements*

- The purpose of the job is to perform a routine test/measurement on one or more circuits and/or associated equipments according to a specified schedule. This allows the verification on a routine basis of the correct functioning of the circuits and/or associated equipments.
- The system is supposed to perform the tests/measurements according to the specified schedule. The results may be stored within the system for later analysis and/or output or routed to a designated hardcopy device. The system may also be required to provide an output error message if it is unable to perform some of the requested tests/measurements.
- The user may be required to input variables such as schedule identity, start time, stop time and a start point for a series of tests.
- The complexity of the job is low.
- The frequency of the job is medium.
- The job is supposed to be performed at exchange and/or OMC level.

B.2.1.3.4 *To stop/suspend the execution of a certain routine test/measurement*

- The purpose of the job is to stop/suspend the execution of the test/measurement before the scheduled stop time.
- The system is supposed to stop/suspend the execution of the test/measurement according to the time data introduced by the user.
- The user is supposed to introduce the identity of the test/measurement to be stopped/suspended and the time data of the actual stop/suspension.
- The complexity of the job is low.
- The frequency of the job is low.
- The job is supposed to be performed at exchange and/or OMC level.

B.2.1.3.5 *To activate the execution of on demand tests/measurements;*

- The purpose of the job is to perform on demand tests/measurements on one or more circuits and/or their associated equipments in order to verify the correct functioning of the circuits and/or associated equipments.
- The system is supposed to perform the actions requested on the specified objects as soon as possible. As many of the system parameters as possible should be system resident. The results may be displayed to the user, stored within the system and/or routed to hardcopy devices depending on the routing control information. The system should output an error message (or code) if it is unable to perform the requested test/measurement.
- The user is supposed to input the type of the test/measurement and the identities of the objects to be tested/measured. The user may also have to input relevant parameters. These would normally be modifications of the system resident default values for a particular test/measurement execution (e.g. the number of times the test is retried).
- The complexity of the job is low, unless the user has to input a large number of parameters values.
- The frequency of the job is high.
- The job is supposed to be performed at exchange and/or OMC level.

B.2.1.3.6 *To delete one or more obsolete test/measurement data*

- The purpose of the job is to delete the data related to a certain test/measurement component which are no more of interest.
- The system is supposed to delete the specified data, providing the necessary safety strategies.
- The user is supposed to specify the identity of the data to be deleted.
- The complexity of the job is low.
- The frequency of the job is low.
- The job is supposed to be performed at exchange and/or OMC level.

B.2.1.3.7 *To retrieve relevant data of tests/measurements*

- The purpose of the job is to retrieve information on the tests and/or measurements that are currently defined in the system.
- The system is supposed to provide to the user the requested information.
- The user is supposed to identify the requested information.
- The complexity of the job low.
- The frequency of the job is high.
- The job is supposed to be performed at exchange and/or OMC level.

B.2.1.3.8 *To retrieve the results of tests and/or of measurements already performed*

- The purpose of the job is to retrieve the recorded results in order to examine them.
- The system is supposed to provide to the user the requested information.
- The user is supposed to introduce the identity of the items to be displayed.
- The complexity of the job is low.
- The frequency of the job is high.
- The job is supposed to be performed at exchange and/or OMC level.

B.2.2 *Document B*

B.2.2.1 *Introduction*

The test/measurement model for maintenance of circuits between exchanges is based on the general model given in Annex A, § A.2.

This model is describing in a general (function independent) way those system functions called tests/measurements which can be controlled by the user by means of MML functions.

This model can be applied on measurements as well as on tests for maintenance purposes.

B.2.2.2 *Maintenance test/measurement model*

B.2.2.2.1 *Test/measurement elements*

A test/measurement is identified by three basic elements: *time, entities, objects.*

Time includes all necessary information to define the start, the duration and periodicity of a certain measurement.

Entities describe the quantities for which data collection must be performed with a certain measurement, e.g. loss, noise, gain/slope, signalling performance, etc.

Objects are intended as individual items within each object type on which the measurements are performed. Examples of objects types are circuits, group of circuits, transmission equipments, facilities, etc.

B.2.2.2.2 *Test/measurement matrix*

The definition of tests/measurements is based on an abstract model which contains the definition of a *measurement matrix* (see Figure B-2), in which each row represents one uniquely definable entity, e.g. transmission loss/noise test, and each column represents a uniquely definable object type, e.g. a group of circuits, a destination.

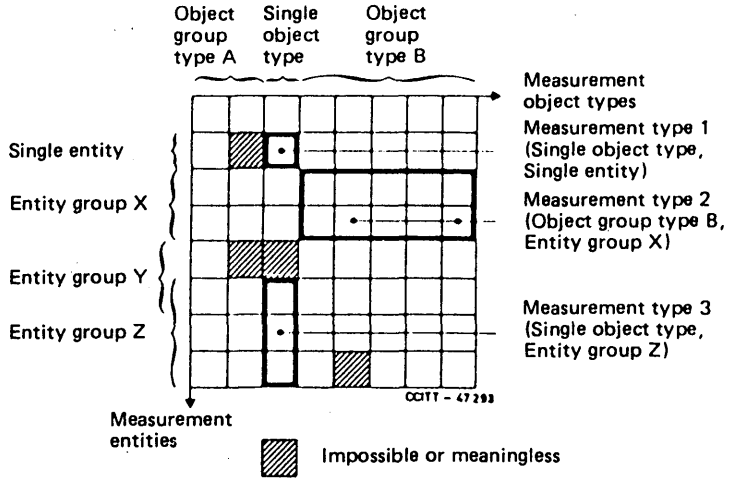


FIGURE B-2

Test/measurement matrix

A certain combination of entities and object types corresponds to certain entries in the test/measurement matrix and form a *test/measurement type.*

It is recognized that part of these test/measurement types may be standardized while the rest of them could be system and/or administration dependent. It should be noted that not all the entries in the test/measurement matrix can be used because some of them will be meaningless (e.g. signalling tests on incoming circuits).

A single object is defined by its type and/or its individual object identity. In some test/measurement types, the number of the objects is fixed, in other types one can choose for the actual test/measurement some or all the allowed objects by means of administration MML commands. Chosen (selected) objects form an *object list.*

The structure of division of object types and entities is open-ended in such a way that any new object type or entity may be added.

If the start of a measurement is instantaneous, it can also be called "on-demand or one-shot test/measurement".

B.2.2.2.3 *Basic types of measurements*

Two basic types of measurements are envisaged (see Figure B-3). The first type (A) is measurement of undetermined duration while the second one (B) is intended to be performed only for a predetermined duration. Examples in the maintenance of circuits area for a Type A measurement would be a maintenance alarm on an associated facility or equipment terminal. Examples of a Type B measurement would be a routine or on-demand test/measurement on a circuit or group of circuits.

The start of a measurement may be intended as instantaneous or delayed for a defined time duration Δt_1 from the activation of the measurement. Since the stop time of a measurement of type A is not given when the measurement is activated or created, it has to be given during the measurement unless the measurement is intended to go on forever.

Whether deactivation is requested, there may be a defined delay of Δt_2 before the measurement is stopped. In the creation of a measurement (or test), a start time may optionally be provided, in which case for that particular measurement, the activation function is not necessary.

Time parameters needed to control a measurement can be divided into three groups:

1) measurement type dependent time parameters (interval parameters of a measurement type, e.g. repeat test interval[1]);
2) measurement dependent time parameters (e.g. time parameters which define the periodicity of measurement). These parameters refer always to relative time or specific dates;
3) measurement independent time parameters (e.g. time parameters which are related to the actual start or stop of a certain measurement in activation and deactivation functions).

B.2.2.2.4 *Structure of a measurement*

A measurement consists of:

- measurement set information;
- time information;
- output routing information.

Figure B-4 shows a model relating these parameters to maintenance test/measurements. This model is useful in illustrating the relationships between test/measurement sequences (measurement sets), time parameters, some of which relate to routine tests only (i.e., they have no relevance to on-demand test/measurements) and the specification of output media (which can be assumed to be by the specification of output destination(s)).

Test/measurement set information, time information and output media information may be predefined as well as circuits lists. It should be noted that predefinition characteristics are system dependent.

B.2.2.2.4.1 *Measurement set information*

Measurement set information consist of one or more selected measurement types with defined objects (objects lists) and measurement type dependent parameters.

Note that measurement types are fixed in the system by the design and implementation at a given moment in time and cannot be created, removed or changed by MML commands; only later supplier release may change these types, according to new requirements or they may be created, changed or deleted by MML commands as a part of a system extension or upgrade operation. Therefore measurement types are no further defined in MML specification of functions.

B.2.2.2.4.2 *Time information*

Measurements of types A and B (see Figure B-3) may perform continuous recording or recording on predetermined days (recording days).

For measurements performing continuous recording only the start date is needed.

B.2.2.2.4.3 *Output routing information*

Output routing information defines the output destinations (there may be more than one), the output formats and the number of copies required. An output destination may be an internal (system resident) log or file. This file may be analyzed at a later time and its data used to provide reports both to the users and for administrative purposes.

---

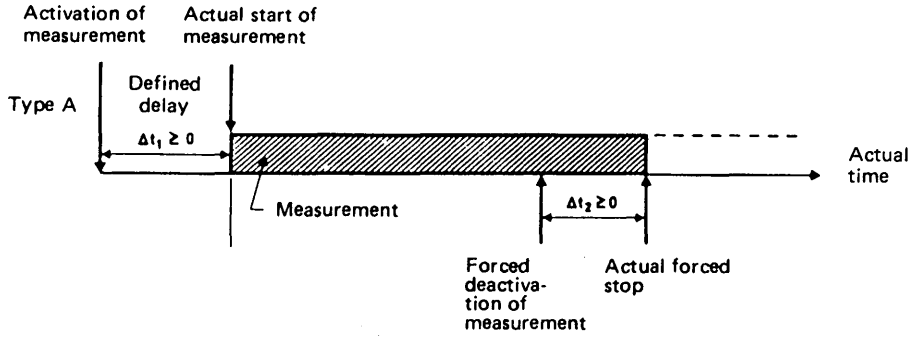
[1] A repeat test interval is the minimum time interval before the repetition of a test can be attempted)

B.2.2.2.4.4 *Summary*

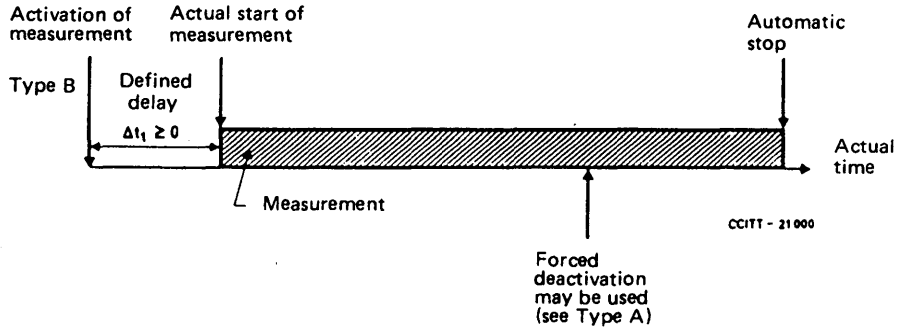
Measurement set information, time information and output routing information may be *predefined* as well as object lists. It should be noted that predefinition characteristics are normally system dependent.

B.2.2.3 *Modification functions*

The modification of measurements and measurements components data should be allowed, but no specialized modification function is defined, provided that a general facility for editing data will be contained among System Control Functions, which remain to be developed.



A measurement of undetermined duration



A measurement of predetermined duration

FIGURE B-3

Basic types of measurements

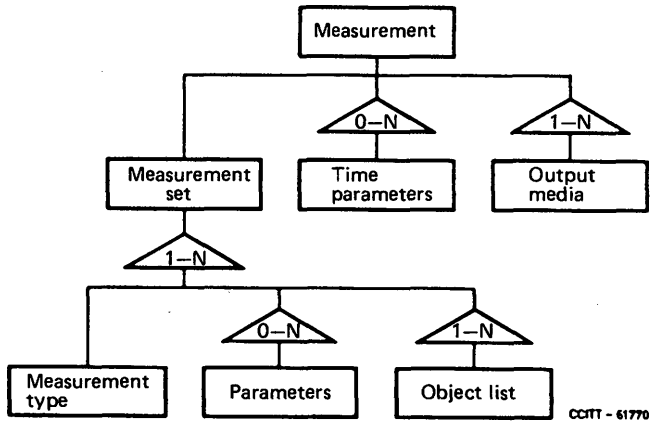


FIGURE B-4

Maintenance test/measurement model

B.2.3 *Document C*

B.2.3.1 *List of MML functions*

1) Creation
   - create a test set;
   - create a measurement set;
   - create a list of circuits;
   - create a time data list;
   - create an output media list;
   - create a routine test.

2) *Delete*
   - delete a test set;
   - delete a measurement set;
   - delete a list of circuits;
   - delete a time data list;
   - delete an output media list;
   - delete a routine test.

3) *Interrogation*
   - interrogate a routine test;
   - interrogate a test set;
   - interrogate a measurement;
   - interrogate a measurement set;
   - interrogate a list of circuits;
   - interrogate a time data list;
   - interrogate an output media list.

4) *Activation*
   - activate a routine test;
   - activate a routine measurement;
   - activate an on-demand test;
   - activate an on-demand measurement.

5) Deactivation
   - deactivate a routine test;
   - deactivate a routine measurement.

6) Output
   - output the results of a routine test;
   - output the results of a routine measurement.

B.2.4 *Document D*

B.2.4.1 *Introduction*

All the information entities needed for the MML functions related to the maintenance tests administration have been identified and are reported in Document D by means of diagrams representing each MML function information structure.

The same information structure diagrams apply to maintenance measurements administration functions.

B.2.4.2 *Information structure diagrams for each MML function*

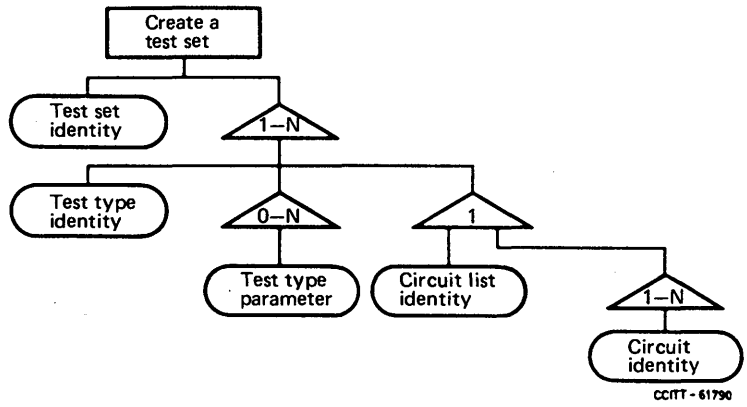


FIGURE B-5

Create a test set

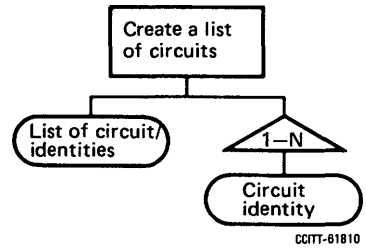


FIGURE B-6

Create a list of circuits

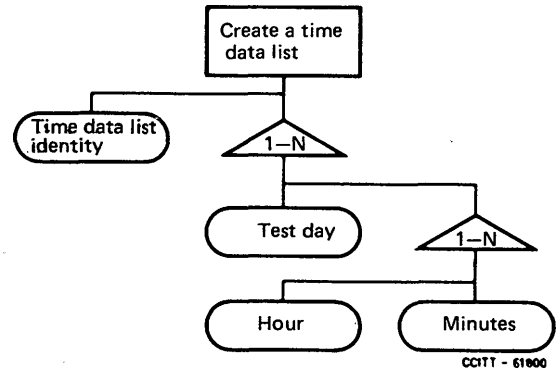


FIGURE B-7

Create a time data list

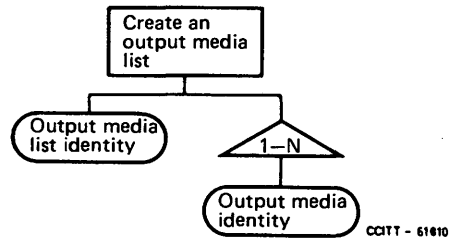


FIGURE B-8

Create an output media list

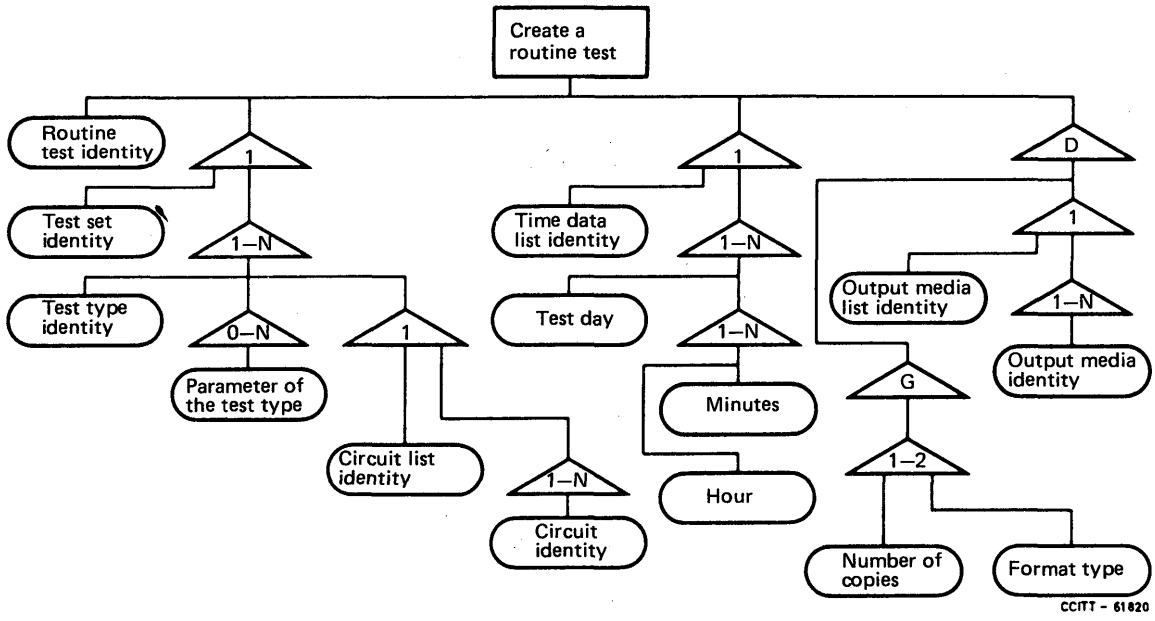


FIGURE B-9

Create a routine test

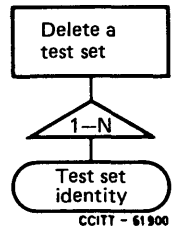


FIGURE B-10

Delete a test set

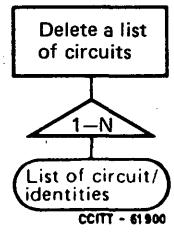


FIGURE B-11

Delete a list of circuits

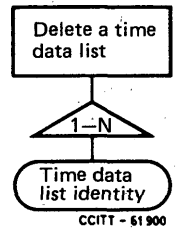


FIGURE B-12

Delete a time data list

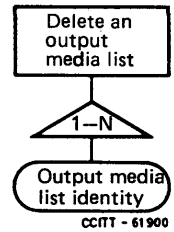


FIGURE B-13

Delete an output media list

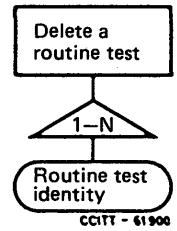
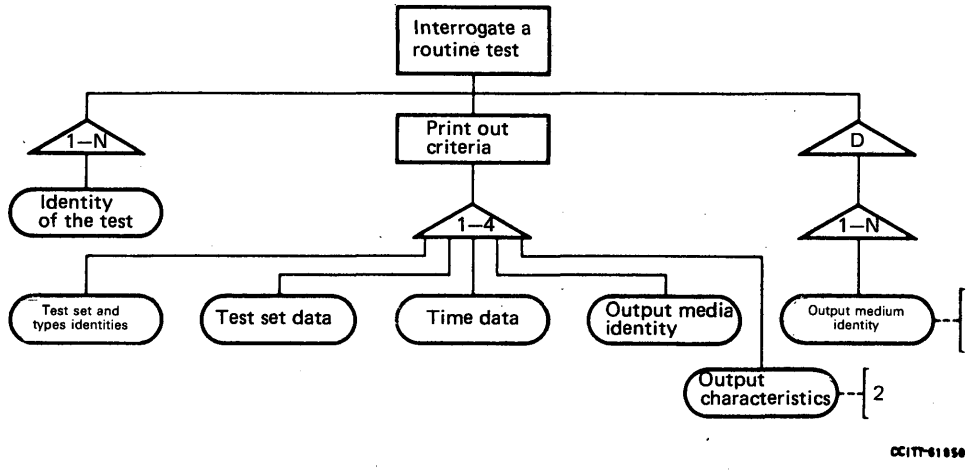


FIGURE B-14

Delete a routine test



Note 1 – For response to the interrogation.

Note 2 – Format types. No. of pages.

FIGURE B-15

Interrogate a routine test

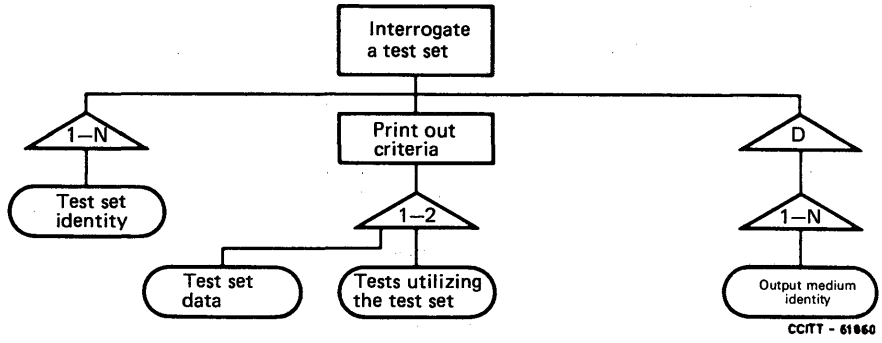


FIGURE B-16

Interrogate a test set

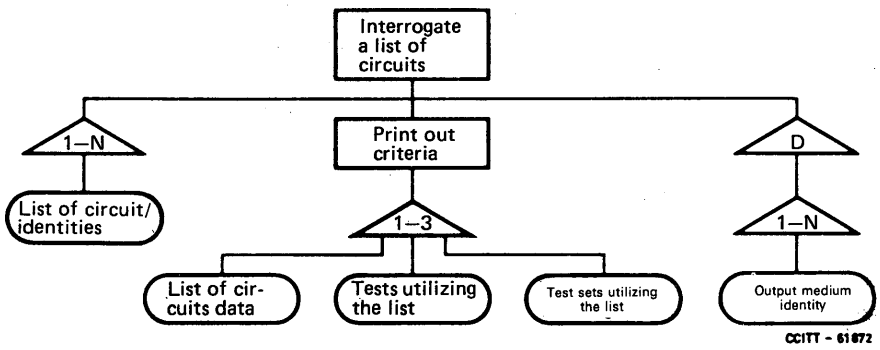


FIGURE B-17

Interrogate a list of circuits

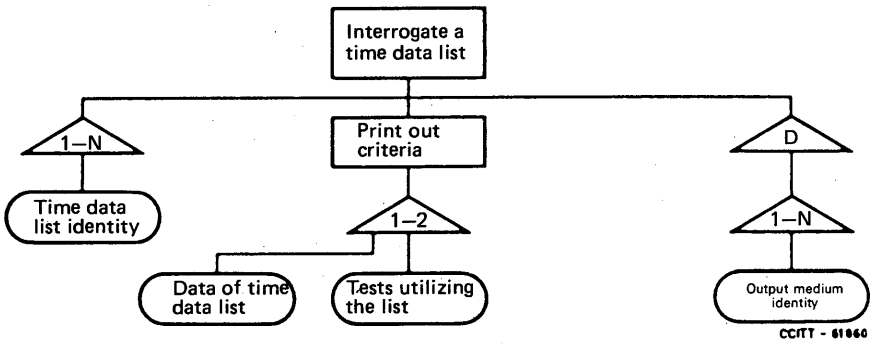


FIGURE B-18

Interrogate a time data list

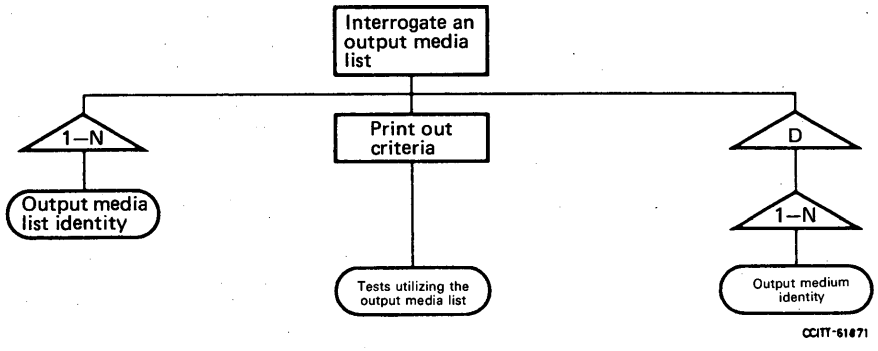
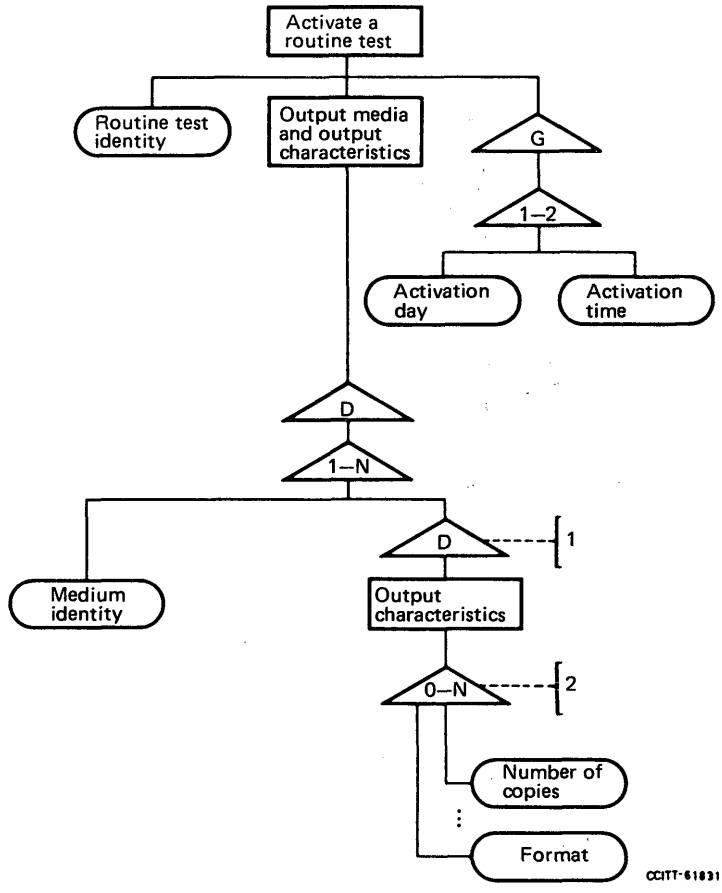


FIGURE B-19

Interrogate an output media list



Note 1 – Further default with respect to particular output medium.

Note 2 – Zero applies when there is no choice for a particular medium; this branch is most likely to apply to hard copy output.

FIGURE B-20

Activate a routine test

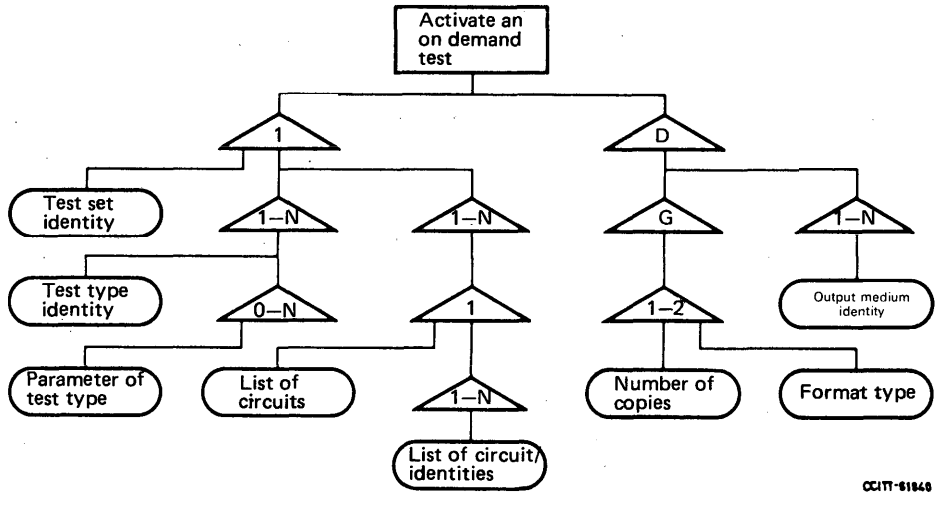
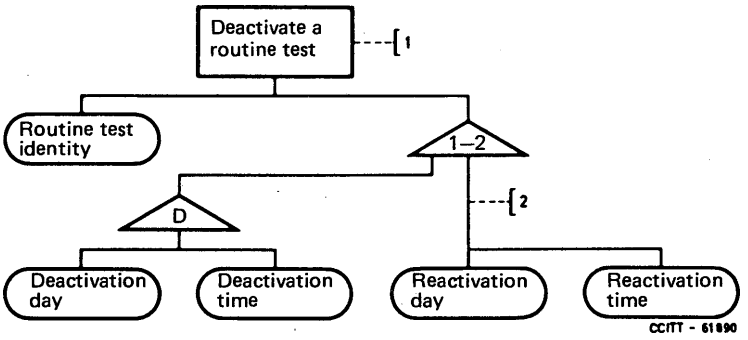


FIGURE B-21

Activate an on demand test



Note 1 – This function is intended to support also the stopping/suspending of a test execution.

Note 2 – This branch allows the possibility of stopping/suspending the test execution.

FIGURE B-22

Deactivate a routine test

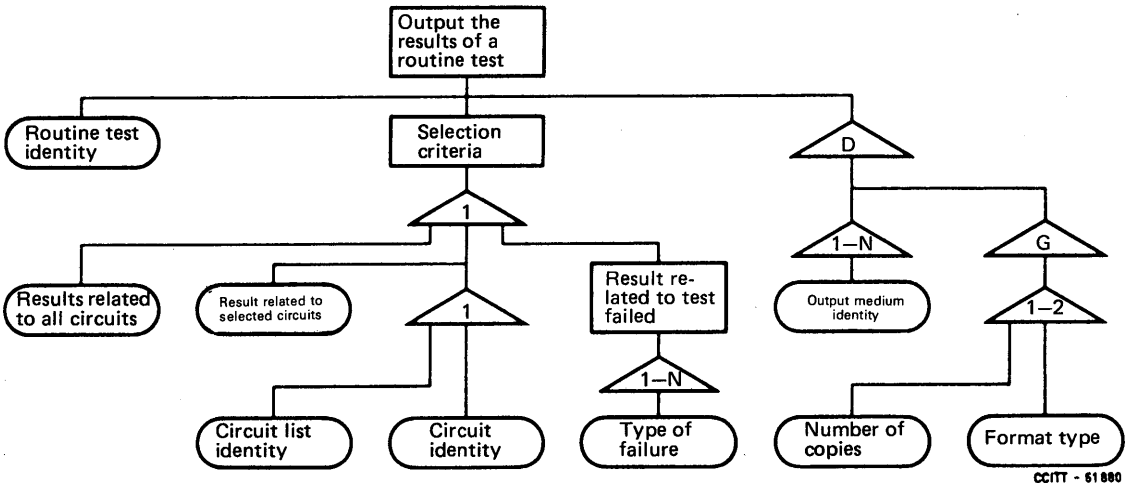


FIGURE B-23

Output results of a routine test

B.2.5 *Document G*

B.2.5.1 *Introduction*

Terms related to documents from A to D are presented below. The production of the other documents in § B.1 may require additional terms.

B.2.5.2 *Glossary of used terms*

start/stop date	The start/stop day of a test/measurement on a routine basis.
start/stop time	The start/stop time of a test/measurement on a routine basis.
test/measurement day	Day in which the test/measurement is performed according to the associated schedule.

B.3 *Observation and supervision*

B.3.1 *Document A*

B.3.1.1 *Introduction*

The observation and supervision of circuits and associated equipments between exchanges mainly consists of reporting the occurrence and/or restoral of failures.

B.3.1.2 *List of class B functions*

B.3.1.2.1 *Observation and supervision of circuits and associated equipments between exchanges*

B.3.1.3 *List of jobs*

B.3.1.3.1 *To interrogate the status of a circuit or a group of circuits and/or associated equipment(s)*

- The purpose of the job is to determine the status of a circuit or a group of circuits and/or their associated equipments. This includes the ability to retrieve the status of all relevant alarms and other performance indications.
- The system is supposed to provide the user with the requested status information. It should also update the status of the circuit within the system on any internally generated lists (work lists, trouble logs, etc.), on which the object circuit may be listed.
- The user is supposed to request the system to provide the requested status information. The user may have to designate where the information is to be displayed.
- The complexity of the job is low.
- The frequency of the job is high.
- The job is supposed to be performed at exchange and/or OMC level.

B.3.1.3.2 *To input the occurrence of failures, circuit irregularities and degraded performance messages*

- The purpose of the job is to input failures, irregularities and performance degradations. In most cases, failures, irregularities and performance degradations are detected by the systems involved. However, there is a need for entering manually into the system to report failures from other sources or to append some additional information to reports already stored in the system.
- The system is supposed to register the information entered by the user and/or the reporting system or exchange. Add the required system resident information, required format and complete the report. Output and/or store the report within the system as required. Alert the user according to the alarm level associated with the report if required.
- The user is supposed to input the proper information on a failure report record.
- The complexity of the job is variable.
- The frequency of the job is low (for manual entries).
- The job is supposed to be performed at exchange and/or OMC level.

B.3.1.3.3 *To input the restoral of failures*

- The purpose of the job is to input the restoral of failures. Normally the restoral of failures will be automatically reported by the system. Manual effort may be needed to remove the restored entity from the fault register (file or log), work lists, trouble tickets, etc., to close out the initial report.
- The system is supposed to register the information entered and take what automatic action is required to close out any system resident work list, trouble ticket, etc. Provide any required output in response to the reported restoral. Restore any system alarms associated with the restored circuit(s).
- The user is supposed to input the proper information.
- The complexity of the job is variable.
- The frequency of the job is low (for manual entries).
- The job is supposed to be performed at exchange and/or OMC level.

B.3.1.3.4 *Access static and dynamic data of the system*

Not available.

B.3.2 *Document B*

B.3.2.1 *Introduction*

No specific model has been developed so far for the "observation and supervision administration" sub-area. Only some supplementary information has been collected and it is reported below.

B.3.2.2 *Supplementary information*

The jobs associated with this Class B function include reporting the occurrence and/or restoral of single or multiple circuit and equipment failures. The reporting mechanism may take the form of a specific message generated by a switching system (or support system) or an alarm report from either a piece of equipment or a transmission facility. The reports may indicate either degraded performance (such as slips on a digital facility) or a total circuit or equipment failure.

B.3.2.2.1 *Report the occurrence of failures, circuit irregularities and degraded performance messages*

In most cases the failures, irregularity and performance degradations are detected by the switching system or an associated support system. The output report may be:

a) filed in a log until a specified threshold is exceeded;

b) filed in a log and only accessed on a demand or browse basis. Both this log and the log in item a) would be periodically purged on a first in first out basis. If desired, a hard copy output of any purged files could be automatically provided;

c) immediately put on a work list for a designated user;

d) directed to a hard copy device.

Failure reports from other sources, such as user reports, may be entered manually into a support system by a technician using an input command.

A failure report may involve a single circuit, a group of circuits (all or part of a traffic group), a facility or a piece of transmission or signalling equipment. Circuit irregularity reports may take many different forms depending on the switching and signalling systems involved. They mainly involve different types of intermittent call processing failures associated with a circuit or group of circuits. Facility or equipment failure reports will normally also involve an alarm indication. The type of alarm indication (e.g. major, minor or critical) will vary with the severity of the failure. Degraded performance is normally detected and reported by facility terminals. Depending on the facility type involved there may be several types of degraded performance output reports (bit error rate, slips, out-or-frames, etc.) at one or more threshold levels.

Associated with all the above reports should be the ability for the user to add by means of MML additional information to the report such as the reason for the failure, tests that have been performed, to whom, and when the failure has been referred for repair, etc.

B.3.2.2.2 *Report the restoral of failures*

The restoral output report should be directed to the proper log file and/or terminal depending on how the failure was originally reported. This restoral function may be provided automatically by some systems for some classes of reports.

B.3.2.3 *Modification functions*

The modification of observation and supervision and observation and supervision components data should be allowed, but no specific modification function is defined, provided that a general facility for editing data will be contained among system control functions, which remain to be developed.

B.3.3 *Document C*

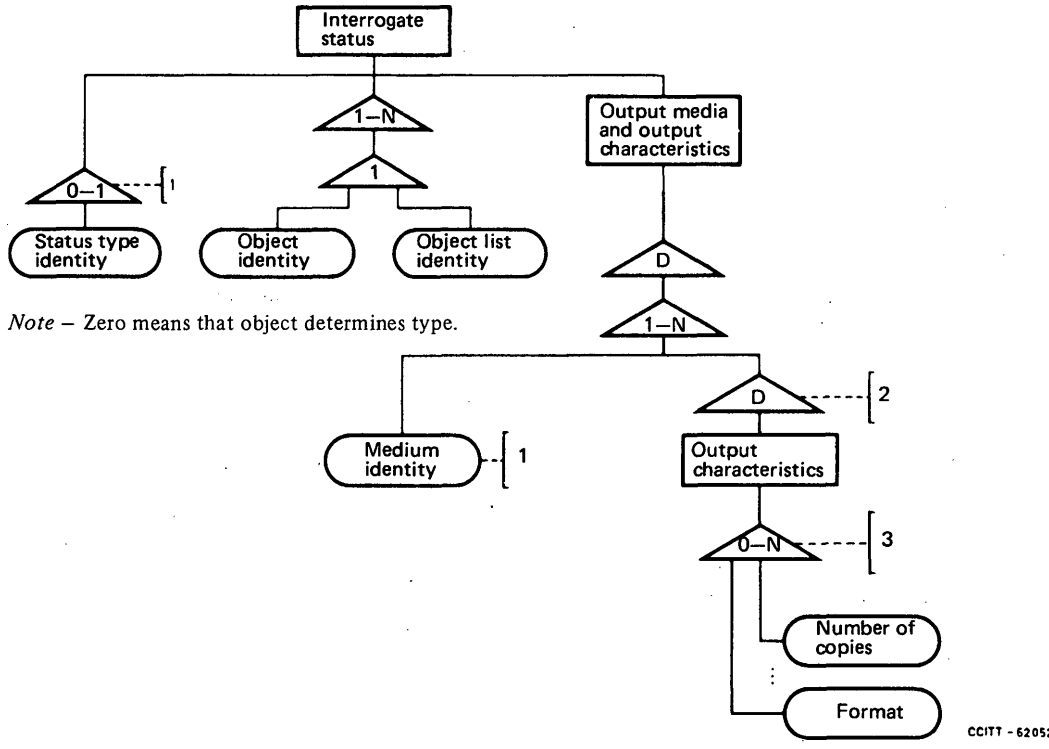
B.3.3.1 *List of MML functions*

1) Interrogate the status of a circuit(s) and/or associated equipment(s).

2) Input trouble or restoral report.

B.3.4 *Document D*

B.3.4.1 *Information structure diagrams for each MML function*



Note – Zero means that object determines type.

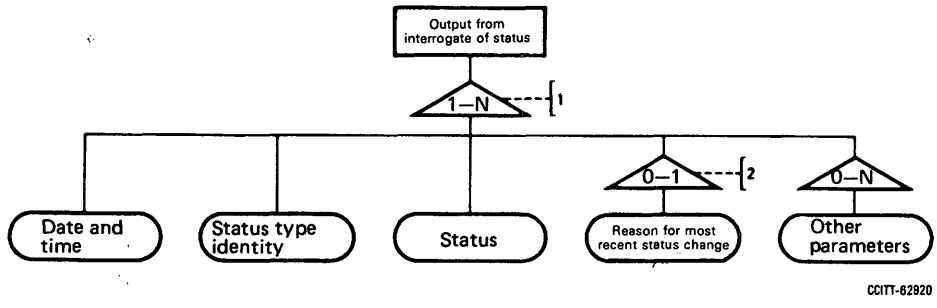
Note 1 – e.g. software file, work list, printer, terminal.

Note 2 – Further default with respect to particular output medium.

Note 3 – Zero applies when there is no choice for a particular medium; this branch is most likely to apply to hard copy output.

FIGURE B-24

Interrogate the status of a circuit(s) and/or associated equipment(s)

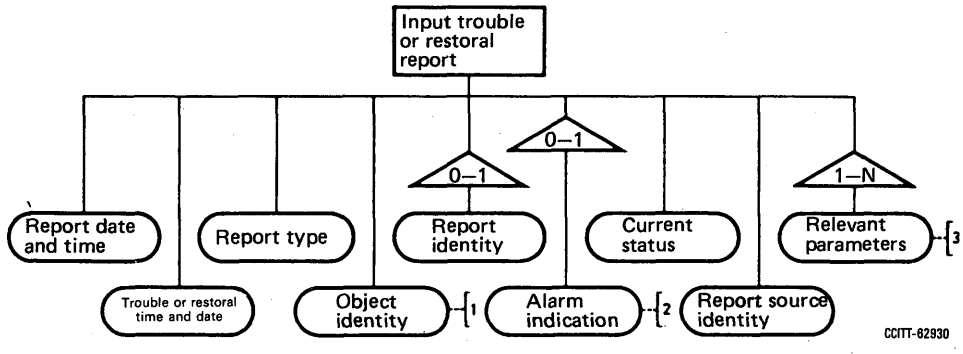


Note 1 – 1–N required for group status check.

Note 2 – Required if out of service.

FIGURE B-25

Interrogate status of a circuit(s) and/or associated equipment(s) (Output information structure)



Note 1 – e.g. restoral, failure, irregularity, degraded performance.

Note 2 – e.g. major, minor, critical, etc.

Note 3 – May be any data to be stored with the report (e.g. test results).

FIGURE B-26

Input trouble or restoral report

B.4 *Status of circuits and associated equipment*

B.4.1 *Document A*

B.4.1.1 *Introduction*

In order to operate and maintain the circuits and associated equipment of an exchange, facilities are required to alter the status of such items.

B.4.1.2 *List of Class B functions*

B.4.1.2.1 Control the status of a circuit or a group of circuits and associated equipments.

B.4.1.3 *List of jobs*[2)]

B.3.1.3.1 *To change the status of a circuit or a group of circuits and associated equipments.*

- The purpose of the job is to change the status of one or more circuits and/or its associated equipments for maintenance purposes (i.e., remove or restore to service, etc.).
- The system is supposed to modify the status of the circuit(s) and associated equipment(s) according to the user input.
- The user is supposed to introduce the identity of the circuit(s) or equipment for which the status *modification* is desired.
- The complexity of the job is low.
- The frequency of the job is high.
- The job is supposed to be performed at exchange and/or OMC level.

---

2) The interrogation of the status of a circuit or a group of circuits and associated equipments is also a job relevant to this sub-area. For the description of such a job and the relevant MML function and associated diagram see §§ B.3.1.3.1, B.3.3.1.1, B.3.4.1, Figure B-24.

B.4.2 *Document B*

B.4.2.1 *Model for the control of the status of circuits or groups of circuits and associated equipment*

B.4.2.1.1 *Introduction*

The state-transition diagram contained in Figure B-27 provides a model for control of the status of a circuit or a group of circuits and associated equipments.

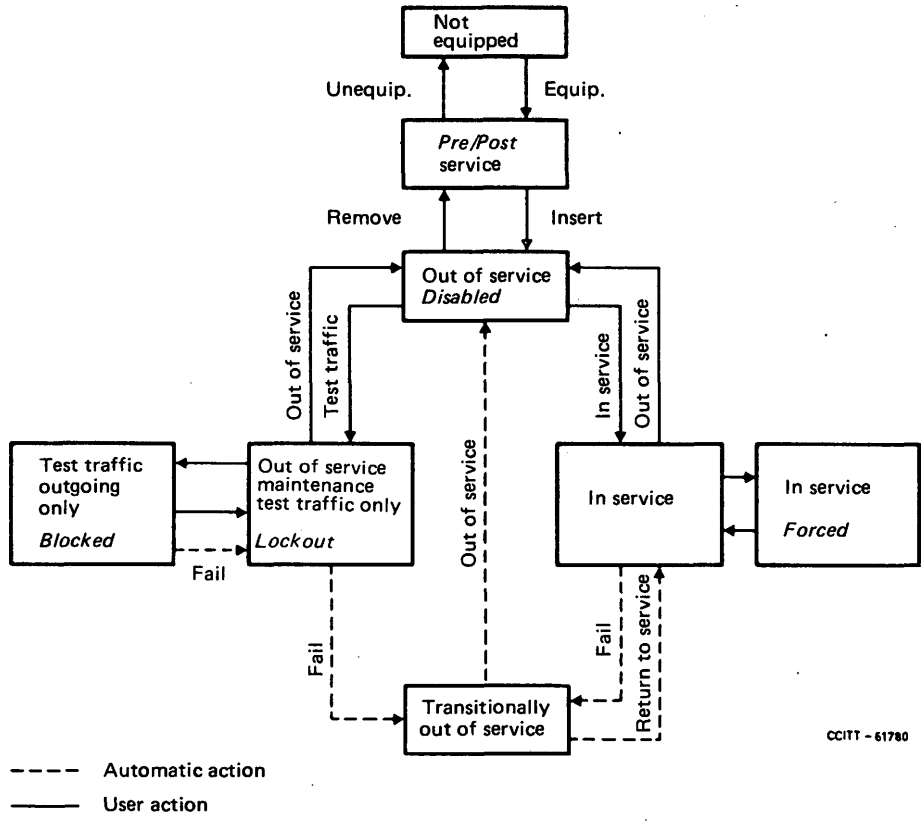


FIGURE B-27

System state diagram for the class B function: Control the status of a circuit or group of circuits

According to this model, a circuit can be in one of only two basic states:

i) in service;

ii) out of service.

Within these basic states a number of sub-states exist as shown in the Table B-1.

TABLE B-1

Basic states and sub states

State		Explanatory information	
Basic state	Sub state	Reference paragraph	Comment
In service	–	B.4.2.1.2.1	Fully in service
	Forced	B.4.2.1.2.1(a)	In service restricted
Out of service	–	B.4.2.1.2.2	–
	Locked out	B.4.2.1.2.2(a)	Maintenance condition, test traffic only
	Blocked	B.4.2.1.2.2(b)	Test traffic in outgoing only incoming disabled
	Pre/Post service	B.4.2.1.2.2(c)	Circuits do not form part of operational network
	Disabled	B.4.2.1.2.2(d)	No traffic

B.4.2.1.2 *In-service circuit states and explanatory information*

B.4.2.1.2.1 *Basic state: In service*

Describes the basic state of a circuit that is fully available to perform the call-handling functions for which it was designed.

Sub states:

a) *Forced:*

Qualifies an in-service circuit functioning with an impairment which normally would automatically be removed from service. However, the automatic removal mechanism has been manually overriden and the circuit caused to remain in service. In this state, the circuit is generally not subject to further automatic removal from service. (Common channel signalling may be an exception in that a failure of the common signalling channel will cause circuits to be unusable.)

B.4.2.1.2.2 *Basic state: out of service*

Depicts the basic state of a circuit that is restricted to some degree from performing its call-handling functions due to trouble or for administrative reasons.

Sub states:

a) *Locked out*

Indicates that an outgoing circuit cannot be selected for traffic calls but can be selected for outgoing test calls. Non-common channel signalling circuits with incoming capabilities will accept all incoming calls. For common channel signalling circuits with incoming capabilities, blocking will have been sent to the far-end and, therefore, except for test calls, no incoming call will be expected or accepted. Locked Out is always associated with maintenance, pre/post service, or circuit administration. The out-of-service maintenance, locked out state is the usual state of a circuit detected to be faulty.

b) *Blocked*

Qualifies the out-of-service condition for systems having the capability to effect near-end control over the maintenance state of a circuit by action at far-end. It is primarily intended for use in common channel signalling but may cover other situations as well. A circuit that has been blocked cannot be selected for outgoing traffic calls but can be selected for outgoing test calls. Receipt of blocking will not in itself prohibit incoming calls and, therefore, all circuits having incoming capabilities will accept incoming calls if presented. The method of accomplishing blocking is generally through the transmission of a blocking signal from the near-end switching exchange to the far end. This signal or message could be on on individual circuit or on a predetermined group of circuits basis. Unblocking is similarly accomplished on an individual or group basis. Functionally, whether entered into on an individual group basis or on a group basis, the Blocked state of a circuit is the same and, either an individual signal or, group signal can unblock the circuit thereafter.

This term does not require an associated operational restriction, as this is already an inherent part of its definition.

Note – An essential feature of this restriction is that since it is imposed by the far-end it can normally only be removed by the far-end.

c) *Pre/post service*

Qualifies the basic out-of-service state of new circuits that exist in machine translation memory but are awaiting initial turnup for service and, recently discontinued circuits awaiting removal from machine translation memory. Depending upon the status of hardware and software and, the suitability and need for testing, these circuits would be further categorized as being locked-out or disabled as required.

It is expected that large blocks of circuits will reside in the Pre/Post state for extended periods of time. The intent is that these circuits be in a unique state that precludes them from being considered as part of an available circuit group and therefore would not be included in any type of service or maintenance measurements or any index plan. In addition, circuit maintenance people should be able to easily distinguish these circuits from already existing circuits that have been temporarily removed from service for short periods for rearrangements or other administrative reasons. It is expected that the record of circuits in the pre/post-service state would be in protected memory so that they would not be inadvertently placed In-Service as a result of a machine failure.

d) *Disabled*

Indicates that no calls can be placed on the circuit and all supervision will be ignored. However, existing systems may choose to retain the capability to attempt outgoing test calls on disabled circuits. Currently, only a near-end originated state is implied. A similar restriction originated at the far-end could be added in the future if the need arises. On common channel signalling circuits with incoming capabilities, blocking will have been sent to the far-end. Disabled is always associated with maintenance, pre/post-service, or circuit administration.

B.4.2.2 *Supplementary information*

B.4.2.2.1 *Maintenance actions*

The following paragraphs further detail the maintenance actions related to the introduced model.

B.4.2.2.1.1 *Set a circuit out of service*

This action has the purpose to condition a circuit or a group of circuits so that is:

a) closed to all types of traffic (this action includes the pre-service or post-service conditioning);

b) open to maintenance tests from an identified maintenance source but closed to normal (customer) traffic.

The system should allow each circuit to be conditioned to a "Forced" in-service condition. Such circuits should be prevented from having their state changed by automatic actions and would need to be redesignated from the "Forced" condition before they become responsive to system failures procedures.

All considered traffic-carrying circuits are bothway, incoming only or outgoing only.

B.4.2.2.1.2 *Set a circuit in service*

The purpose of this action is restricted to condition the circuit to be fully available for all traffic states. The transition from this state to all other circuit conditions is accomplished by the "set out of service" action.

B.4.2.2.2 *Reaction of the system*

The system should execute the required function conditioning the circuit(s) as requested by the user and, in addition, it should make some checks and action as in the following described.

For the "set out of service" function, the system should:

a) check that the circuit is not already in the required state. If it is, then report the fact to the initiator without attempting to execute;

b) print out with the condition report, relevant circuit details and characteristics as defined in the system tables.

If the function has been required for more than one circuit and some of them are already in the required state, the system should:

a) move to the required state those circuits for which this is possible;

b) make no attempt to initiate a change on those circuits already in the required state;

c) respond to the initiator or to a designated report printer with a message containing, for each of the list requested, the proper information (e.g. "State changed", "Already in maintenance", etc.).

B.4.2.2.3 *Users aspects*

The user may be handed daily a list of circuit identities (faulty or otherwise) which require conditioning to the "Out of service" condition.

The command layout should allow state changes for one or more circuits via a single command by entering the relevant number of circuit identities.

For those circuits not moved to another state but reported to the user (or other report position) as being already in the required circuit condition, an investigation should be made as to why the circuit condition records for these are wrong.

B.4.2.2.4 *Characteristics of the jobs*

The jobs associated with these actions appear relatively simple but only practical in an SPC-controlled environment. They may also be performed from an auxiliary system in a maintenance centre. The system should allow for the alternatives "Camp on busy" and "Forced release" (as methods of changing the state of a busy circuit) to be specific by command parameter (optional) or to be predefined system resident default values.

The actions may be performed probably daily and at any network level.

B.4.2.2.5 *Safety aspects*

To ensure some protection to the grade of service offered by the switching systems, the system should check that the number of circuits in a Fault State or conditioned for "out of service" does not exceed a fixed proportion of the total circuits available. The precise proportions permitted may be variable over a 24-hour period in two or more time slots defined by system-resident parameters. Separate values could be provided for special days (weekends, holy days, etc.).

For example, a minimum proportion could be allowed during the system busy hour, while a maximum proportion could be allowed during the system quiet period.

B.4.2.3 *Modification functions*

The modification of status of circuits and associated equipment and status of circuits and associated equipment components data should be allowed, but no specific modification is defined, provided that a general facility for editing data will be contained among System Control Functions, which remain to be developed.

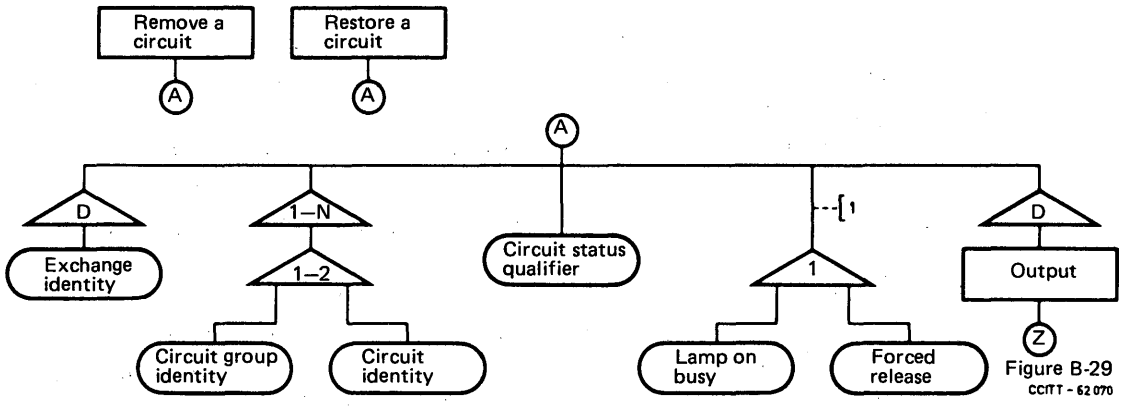
B.4.3 *Document C*

B.4.3.1 *List of MML functions*

1) Remove a circuit (or group of circuits).
2) Restore a circuit (or group of circuits).

B.4.4 *Document D*

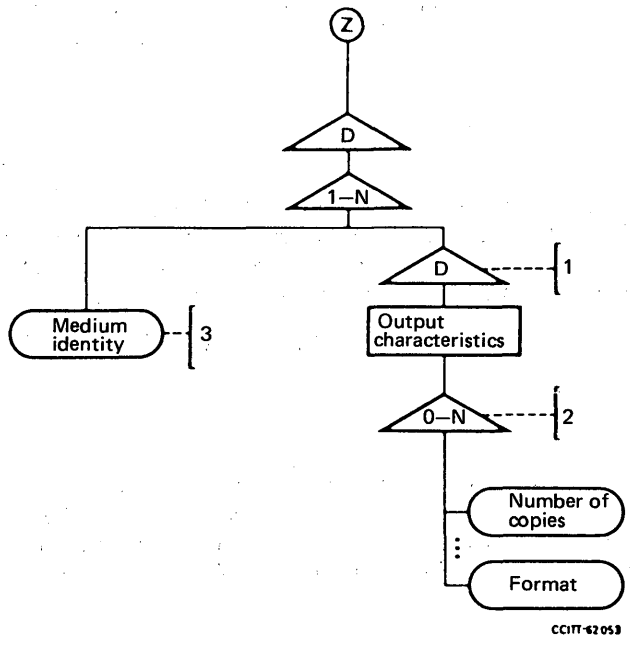
B.4.4.1 *Information structure diagrams for each MML function*



Note 1 – Only for busy condition.

FIGURE B-28

Remove a circuit (or group of circuits)
Restore a circuit (or group of circuits)



Note 1 – Further default with respect to particular output medium.

Note 2 – Zero applies when there is no choice for a particular medium; this branch is most likely to apply to hard copy output.

Note 3 – e.g. software file, work list, printer, terminal.

FIGURE B-29

Continuation of Figure B-28

B.5 *Analysis of maintenance data*

B.5.1 *Document A*

B.5.1.1 *Introduction*

The results of maintenance tests and/or measurements and of the observation and supervision may be analyzed by means of suitable analysis functions.

B.5.1.2 *List of Class B functions*

B.5.1.2.1 *Analysis of maintenance data*

B.5.1.3 *List of jobs*

B.5.1.3.1 *To administer maintenance analysis groups*

- Purpose of the job is to administer (create, change, delete) the maintenance analysis groups.
- The system is supposed to update the maintenance analysis groups as required by the user.
- The user is supposed to input the proper information to update the groups to be modified and to input the desired new values.
- The complexity of the job is low.
- The frequency of the job is medium.
- The job is supposed to be performed at exchange and/or OMC level.

B.5.1.3.2 *To administer user selectable maintenance analysis thresholds*

- Purpose of the job is to specify and/or change thresholds for the proper execution of the maintenance analysis.
- The system is supposed to update the specified threshold values that are used by the maintenance analysis functions.
- The user is supposed to specify the desired parameters and key in new values.
- The complexity of the job is low.
- The frequency of the job is medium.
- The job is supposed to be performed at exchange and/or OMC level.

B.5.1.3.3 *To interrogate different kinds of information related to maintenance analysis functions*

- The purpose of the job is to retrieve information on the currently active maintenance analysis functions.
- The system is supposed to output in suitable format and on the selected devices the information requested.
- The user is supposed to select the information to be retrieved.
- The complexity of the job is low.
- The frequency of the job is medium.
- The job is supposed to be performed at exchange and/or OMC level.

B.5.1.3.4 *To activate and deactivate maintenance analysis functions*

- The purpose of the job is to activate and/or deactivate maintenance analysis functions on specified report files.
- The system is supposed to make the report files available to the maintenance analysis functions.
- The user is supposed to input the data needed for the activation or the deactivation request along with maintenance analysis functions identities.
- The complexity of the job may be high depending on the amount of data.
- The frequency of the job is low.
- The job is supposed to be performed at exchange and/or OMC level.

B.5.1.3.5 *To allow, inhibit or initialize a threshold*

- The purpose of the job is to allow the users and/or system manager to control the thresholds that are used by analysis functions.
- The system is supposed to allow, inhibit, initialize the selected threshold as instructed by the user.
- The user is supposed to input the required commands and parameters.
- The complexity of the job is low.
- The frequency of the job is low.
- The job is supposed to be performed at exchange and/or OMC level.

B.5.2 *Document B*

B.5.2.1 *Introduction*

No specific model has been developed up to now for the "Maintenance analysis functions administration". Only some supplementary information have been collected and they are reported.

B.5.2.2 *Supplementary information*

B.5.2.2.1 *Modification of analysis thresholds*

One of the tools of the statistical analysis is the use of thresholds to help reduce the amount of data to be displayed to the users. In software systems, some of these thresholds may be made variable under control of the user. MML functions are required to allow the user to control these thresholds. The MML function should specify the type of the threshold to be modified, the new threshold value(s), the object to which the threshold is to be applied. The threshold modification should be verified by an output verification message.

B.5.2.2.2 *Modification of analysis groups*

There is the need to create or to modify analysis groups also if it is assumed that in general the ability to create new analysis groups is a system dependent function and it is not under control of all the users.

The user should be able to specify:

a) the type or identity of the analysis group to be modified;

b) the type of modification to be made (e.g., a change in parameters, thresholds, group size, or to delete certain members from the analysis groups that may be disrupting the analysis).

All modifications to an analysis group should be verified by an output verification message.

B.5.2.2.3 *Allow, inhibit or initialize a threshold*

Systems should permit the users to allow, inhibit or initialize maintenance thresholds. Depending on the system or exchange type involved, thresholds may be provided for various performance degradations, circuit irregularity counts and failure rates. Tripping of a threshold may require an alarm indication along with an output report or logged message.

B.5.3 *Document C*

B.5.3.1 *List of MML functions*

1) Activate maintenance analysis functions.
2) Deactivate maintenance analysis functions.
3) Change analysis thresholds.
4) Change analysis groups.
5) Interrogate analysis thresholds.
6) Interrogate analysis groups.
7) Allow, inhibit, initialize a threshold.

B.5.4 *Document D*

B.5.4.1 *Information structure diagrams for each MML function*

Not available

FIGURE B-30

Activate maintenance analysis functions

Not available

FIGURE B-31

Deactivate maintenance analysis functions

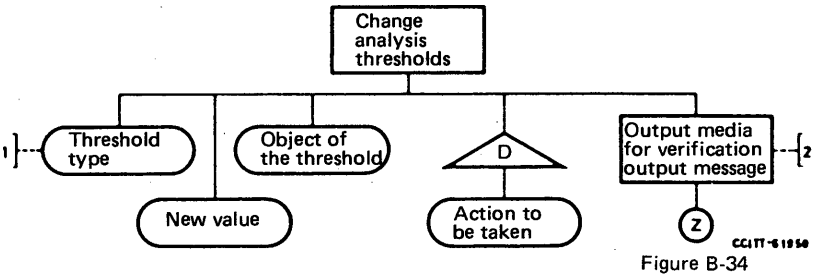


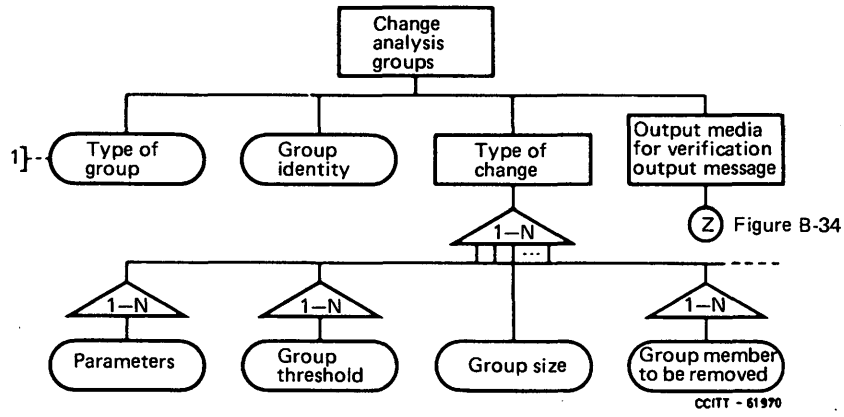
Figure B-34

Note 1 – There may be thresholds for general or specific trouble type possibly with respect to one particular equipment, facility, line group or signalling type.

Note 2 – Message automatically generated to inform the user that change has been accomplished, may be the same as output from INTERROGATE ANALYSIS THRESHOLD (Figure B-35).

FIGURE B-32

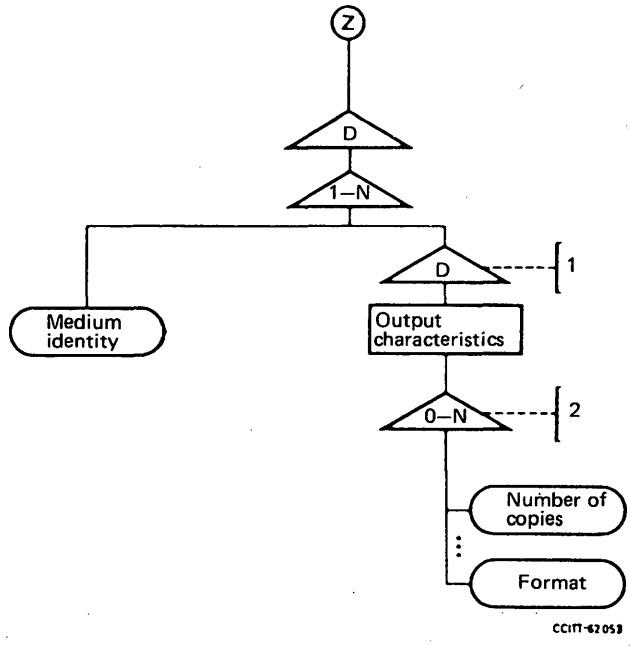
Change analysis thresholds



Note 1 – e.g. equipment, line, facility, or signalling type, or exchange.

FIGURE B-33

Change analysis groups



Note 1 – Further default with respect to particular output medium.

Note 2 – Zero applies when there is no choice for a particular medium; this branch is most likely to apply to hard copy output.

FIGURE B-34

Continuation of Figures B-32, B-33 and B-37

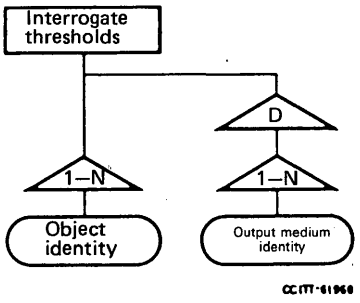


FIGURE B-35

Interrogate analysis thresholds

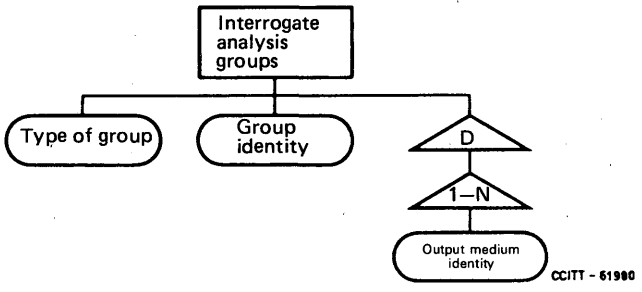
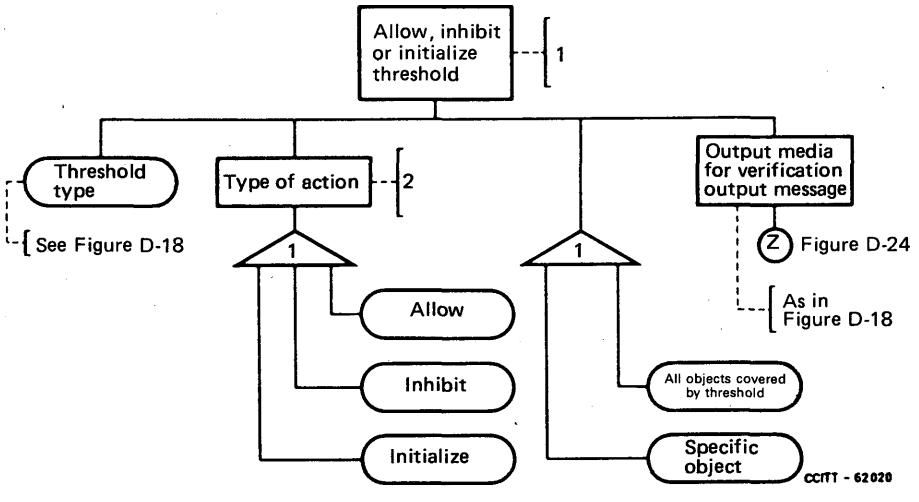


FIGURE B-36

Interrogate analysis groups



Note 1 – For temporary suspension of thresholds set as in Figure D-18.

Note 2 – This will normally be three separate commands.

FIGURE B-37

Allow, inhibit, initialize a threshold

B.6 *Maintenance report*

B.6.1 *Document A*

B.6.1.1 *Introduction*

Maintenance activities are carried out on the basis of the various report messages received from the switching exchange and/or associated processor, including all the report messages associated with "Observation, supervision and analysis of circuits and equipments".

Systems are required to provide the users the ability, using MML functions, to administer such reports, e.g., to create summary reports and work lists such as a list of the "N" worst circuit groups in an exchange.

B.6.1.2 *List of Class B functions*

- Administration and control of maintenance reports.

B.6.1.3 *List of jobs*

B.6.1.3.1 *Plan and/or interrogate reports*

- Purpose of the job is to plan and/or interrogate various types of maintenance-related reports. These include scheduled reports (hourly, daily, etc.) and on-demand reports using system processes such as sort techniques.
- The system is supposed to format and output the interrogated reports at the indicated time(s) and output device(s).
- The user is supposed to provide the system with the needed data and parameters when planning to input the necessary commands when interrogating a report. This may include the use of a system resident generalized report generating capability to provide special reports.
- The complexity of the job is medium for planning, and low for interrogating.
- The frequency of the job is very low.
- The job may be performed at exchange and/or OMC level.

B.6.1.3.2 *Plan and prioritize maintenance activities*

- Purpose of the job is to allow the users to plan and prioritize daily maintenance activities. This includes reviewing available failure, status, supervision, and observation information in order to efficiently direct and prioritize the ongoing maintenance activities.
- The system is supposed to supply the required browse, sort and file movement control commands to support this job.
- The user is supposed to use the system capabilities as required depending on the failure and other data available.
- The complexity of the job is medium.
- The frequency of the job is medium to high.
- The job may be performed at exchange and/or OMC level.

B.6.1.3.3 *Route of reports and outputs*

- Purpose of the job is to allow the users to route maintenance reports (failures, irregularities, performance, etc.) and outputs. Rerouting may be required as result of work loads, equipment failures and user availability.
- The system is supposed to reroute the maintenance reports and outputs as instructed by the user.
- The user is supposed to input the required commands when needed.
- The complexity of the job is low.
- The frequency of the job is low.
- The job may be performed at exchange and/or OMC level.

B.6.2 *Document B*

B.6.2.1 *Introduction*

No specific model has been developed.

B.6.2.2 *Supplementary information*

B.6.2.2.1 *Sorting functions*

Sorting functions are needed in those cases where various report type messages are logged in the same file. This file might reside either in the exchange or in its associated auxiliary systems. It is assumed that MML sorting function should provide the ability to sort these files by one or more categories. Some of the possible categories may be:

a) circuit group (traffic group)
   - traffic number;

b) exchange;

c) facility identity;

d) equipment identity;

e) message or device type:
   - failure,
   - circuit irregularity,
   - performance degradation,
   - signalling,
   - facility type,
   - equipment type.

The ability to control the time period over which the sorting is done should also be provided (e.g., a list of all circuit failures that occurred on a certain traffic group within specified hours).

B.6.2.2.2 *Create report summaries*

In order to properly administer the maintenance of circuits between exchanges, the technicians, using MML input commands, must have the ability to create and/or modify the report summaries provided by the exchange and/or its associated auxiliary systems.

This job function assumes that these processors have generalized reports generating software under the control of users. The MML commands supporting this job should be able to specify:

a) the type of report to be created or modified;

b) the period of the report;

c) the report format, if more than one is available;

d) if it is a routine or a demand report;

e) if it is a routine report, the automatic start and stop times for the report;

f) the output destination for the report.

The new or modified report should be verified by an output verification message.

B.6.2.2.3 *Move report to other files*

This is a part of the maintenance administrative job function that allows the user, using MML input commands, to mark and/or refile designated report messages. An example of this function would be to move or delete reported troubles that had been cleared (restored) from an active trouble log (or file) to a cleared trouble log referred out for repair either to the far-end or to the local repair forces into separate logs or files. This feature would also be used to reassign trouble reports from one user to another.

B.6.2.2.4 *Browse report files*

This job function allows the users, using MML input commands, the ability to access a maintenance file (or log) and browse through the file. The users should have the ability to start their browse at either the beginning or the end of the log and work toward the other end. They should also have the ability to specify a specific start time and date along with the browse direction.

B.6.2.2.5 *Interrogate summary reports*

This job function allows the technicians, using the MML input commands, the ability to interrogate the various report types described previously in this section. The report intervals may include a portion of an hour, hourly, daily, weekly, etc., reports. The ability to demand current summaries since the last report interval should also be provided. Various summary status reports may also be interrogated depending on the switching exchange and/or its associated auxiliary systems.

B.6.3 *Document C*

B.6.3.1 *List of MML functions*

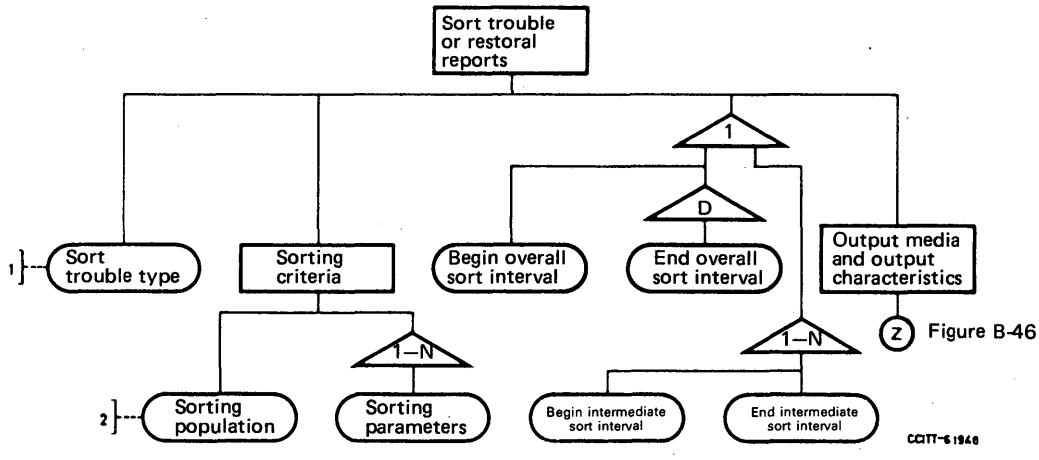
1) sort trouble or restoral reports;
2) move reports to other files;
3) browse report files;
4) create summary reports;
5) activate a report on demand;
6) activate a report on routine;
7) deactivate a report on routine;
8) change report classification;
9) output summary reports;
10) route output of reports.

B.6.3.2 *Modification function*

The modification of maintenance report and maintenance report components data should be allowed, but no specific modification function is defined, provided that a general facility for editing data will be contained among System Control Functions, which remain to be developed.

B.6.4 *Document D*

B.6.4.1 *Information structure diagrams for each MML function*



Note 1 – e.g. failure, irregularity, degraded performance, restoral.

Note 2 – This may include such criteria as by line group, exchange, facility or equipment ID, and/or by more specific trouble type, signalling type, equipment or facility type.

FIGURE B-38

Sort trouble or restoral reports

Not available

FIGURE B-39

Move reports to other files

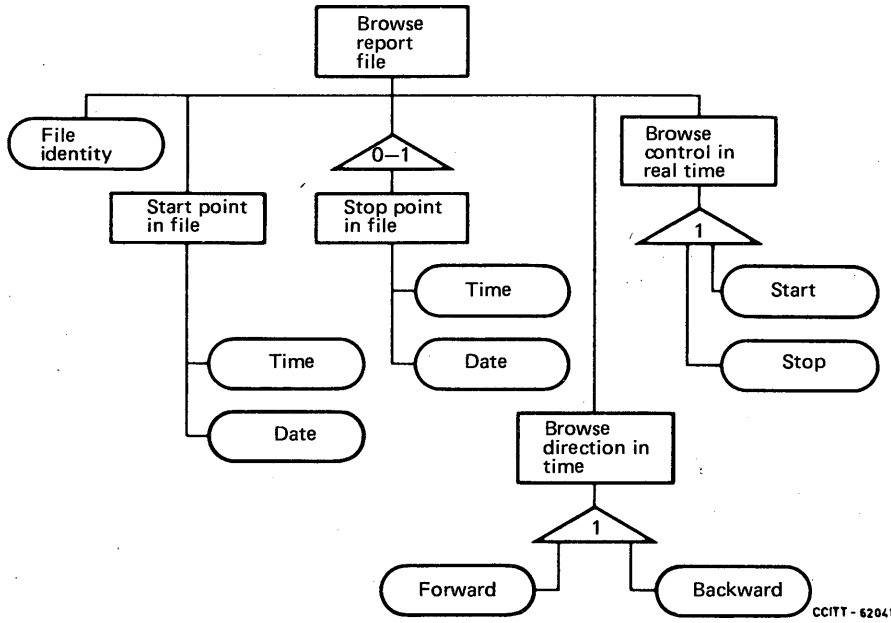
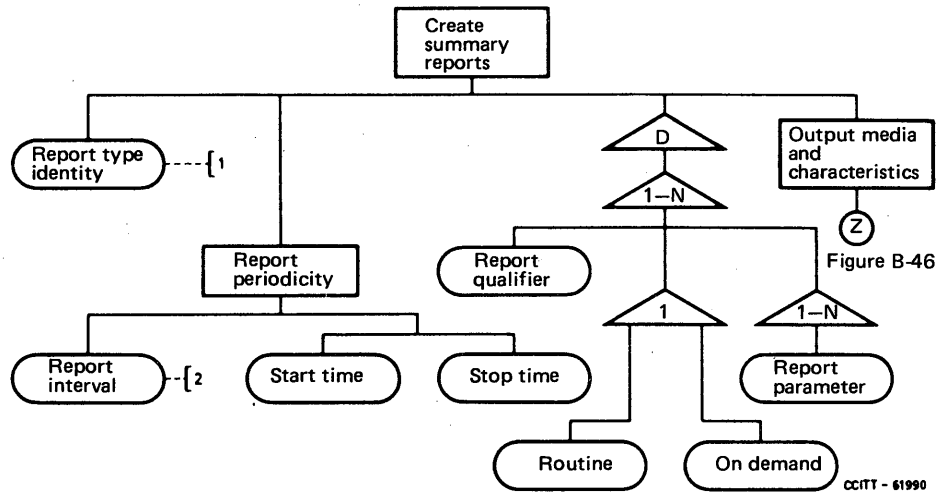


FIGURE B-40

Browse report files



Note 1 – Equipment alarms, transmission, outages, etc.

Note 2 – e.g. N minutes, hourly, weekly, monthly, etc.

FIGURE B-41

Create summary reports

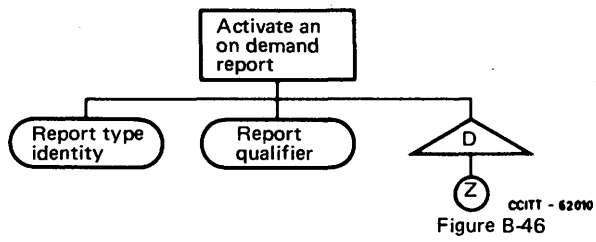
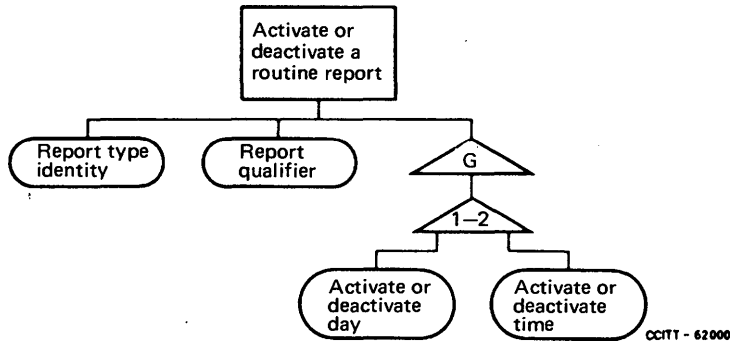


Figure B-46

FIGURE B-42

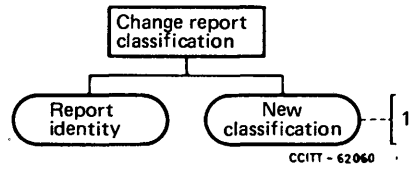
Activate a report on demand



Note 1 – Suspend equals deactivate for a limited time.

FIGURE B-43

Activate/deactivate or suspend a report on routine



Note 1 – e.g. active, referred locally or to far end, cleared.

FIGURE B-44

Change report classification

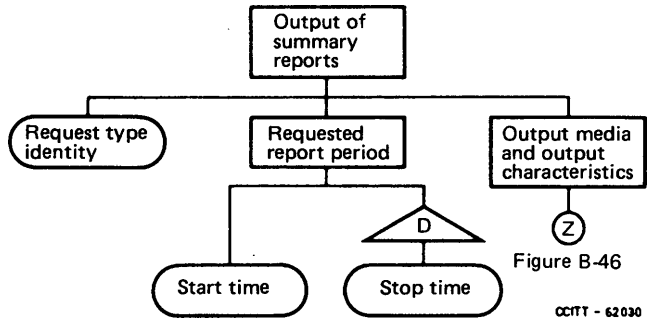
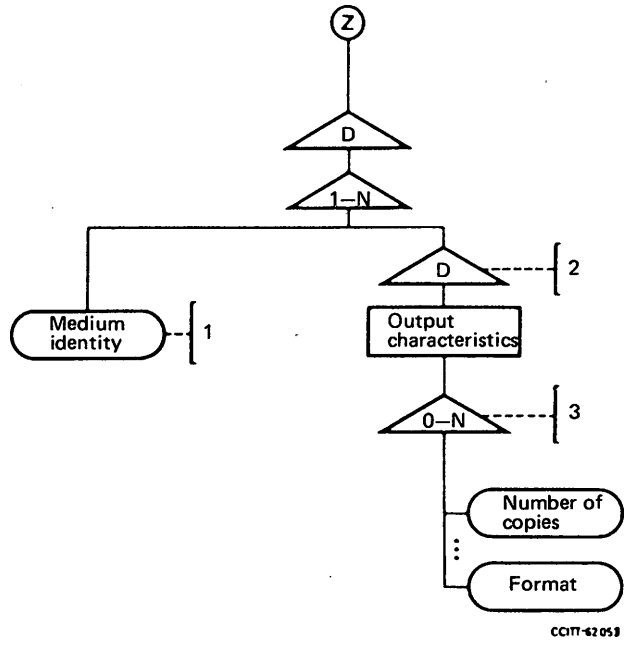


FIGURE B-45

Output summary reports



Note 1 – e.g. software file, work list, printer, terminal.

Note 2 – Further default with respect to particular output medium.

Note 3 – Zero applies when there is no choice for a particular medium; this branch is most likely to apply to hard copy output.

FIGURE B-46

Continuation of Figures B-38, B-41, B-42 and B-45

Not available

FIGURE B-47

Route output of reports

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

SECTION 5

GLOSSARY OF TERMS

Recommendation Z.341

GLOSSARY OF TERMS

1 General

The aim of the glossary for the man-machine language is to include terms used in describing the man-machine language. It comprises, in alphabetical order, terms used in the Z.300-series Recommendations that have a special significance in the MML context and hence require definition. Terms comprising words used in their ordinary, every day sense, i.e. unambiguous and self-explanatory words, are not included.

The terms in italics in the text of the definitions are defined elsewhere in this glossary. If a term has one meaning within the context of Recommendations Z.321-Z.323, and another within the context of Recommendations Z.331-Z.333, then the meaning in the former context follows i), and the meaning in the latter follows ii).

2 List of terms

acceptance input

An *input* used to allow the *system* to *output* a high priority message, announced by a *message waiting indication.*

acceptance output

An *output* message indicating 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 actual result.

accessible field

A *field* for writing by the *user* and the *system.*

action

The process of performing an *MML function*; usually represented by a verb.

action modifier

A qualification of an *action.*

activate

An *action* to initiate a *system* process that requires preliminary data entry, or an *action* to make previously entered *data set* available to the *system* for its intended use; opposite of *deactivate.*

additional header information

Provides information supplementary to the actual *output header,* such as sequence number, processor number, *output* device, or day of the week.

additional information

i) General information on how to proceed, e.g. how to select an item, a *form,* a *menu* or how to submit a *form* to the *system.*

ii) List of possible values to be associated with one or more *information entities* in *information structure diagrams.*

administrative system

A *system* which supports administration personnel in performing administrative *jobs,* e.g. billing, related to *SPC systems.*

alarm statement

A statement providing information concerning an alarm condition, such as the degree (level) of alarm or the source of the alarm.

allow

An *action* to permit specified *system actions,* responses, or *functions* to occur; these *functions* may be inhibited by *system* design or by use of the *inhibit action.*

annotation

An aspect of the *drawing convention* of the *syntax and decomposition meta-language* indicating how descriptive or explanatory notes may be presented for clarification purposes.

annotation symbol

A *symbol* (– – – – –[n where *n* is a number referencing a note) used in the syntax *meta-language* for *annotation* purposes.

arithmetic delimiter

A *symbol* used to delimit an *arithmetical expression*: ((left parenthesis) for the opening *delimiter* and) (right parenthesis) for the closing *delimiter.*

arithmetic operator

A *symbol* used to denote the arithmetic operation(s) to be performed in an *arithmetical expression.* Allowed operators are: + (plus sign), – (hyphen), / (solidus), * (asterisk).

arithmetical expression

A combination of *arithmetic operators, numerals* (*decimal, hexadecimal, octal* or *binary*) and *identifiers* enclosed by *arithmetic delimiters.*

auxiliary system

A *system* that supports *SPC systems* in performing their tasks. It may be either an *operation and maintenance system* or an *administrative system.*

Backus Naur Form (BNF)

A syntactic *meta-language* for use in specifying the syntax structure of inputs and outputs of an actual *man-machine interface.*

binary numeral

A *numeral* in the binary (base 2) *numbering system*, represented by the *characters* 0 (zero), 1 (one) and optionally preceded by B' (B apostrophe).

block mode transmission

A transmission characteristic in which all of the regular typewriter keys and some of the special purpose keys are only transmitted to the controlling processor, in a block, when a "send" key is activated.

block of parameters

A set of *parameters* containing information necessary for the *system* to perform the *function* specified in the *command.*

border area

That part of a *visible display* which is physically unavailable for displaying or entering data.

browse

An *action* to display sequentially the current values of items in a *data set*; the *user* may examine the data items in either the forward or backward direction.

CCITT MML

The *man-machine language* (MML) developed by the International Telegraph and Telephone Consultative Committee (CCITT) for *stored program-controlled systems* and *operation and maintenance systems.*

change

An *action* to modify specified data items in a *data set.*

character mode transmission

A transmission characteristic in which each and every character *input* at the keyboard is sent to the controlling processor one at a time.

character set

The finite set of different characters used in *CCITT MML.*

clarifying text

A set of *information units* used to make the purpose and content of the *output* clearer.

class A function

A *function* which provides the user with the means to control *system functions* via MML *inputs* and *outputs*; also known as an *MML function.* It can be viewed as an *action* upon an *object.*

class B function

A *function* which can be controlled at least partially by the *user* by means of *class A* (or *MML*) *functions.*

class C function

A *function* which is not controllable by the *user* in a given *system.*

command

The complete specification of a *function* that the *system* is required to perform. It comprises a *command code* followed generally (but not necessarily) by one or more *blocks of parameters.*

command code

A set of up to 3 *identifiers*, each separated by a – (hyphen), used to define the nature of the *command*.

command entry sequence

The sequence of operations required to input a *command* or a series of *commands*.

command reference

A reference to a previously given *command*, appearing in *output outside dialogue* and *dialogue procedures*, in the form of a *command sequence number* and, possibly, *clarifying text*.

command sequence number

A reference number uniquely identifying a *command* recognized by the *system*.

comment

A character string enclosed between the *separators* /* (solidus asterisk) and */ (asterisk solidus). It has no MML syntactical or semantical meaning.

component

A *decomposition meta-language* symbol for an *information entity* that cannot be divided further.

composite part

A *decomposition meta-language* symbol for an *information entity* that can be divided into smaller parts.

compound parameter argument

A *parameter argument* made up of more than one *information unit*. It is used to specify a multidimensional *object* or value, e.g. a date can be expressed as 1979-12-31.

concealment

A *video attribute* by which information is hidden, e.g. secret parts of a password.

connectivity rules

An aspect of the *drawing convention* of the *decomposition meta-language* indicating *symbol* interrelationship.

connector

An aspect of the *drawing convention* of the *decomposition meta-language* indicating how *flowlines* may be broken.

continuation character

A special *execution character* implying a similar *command code* for the next *command* and hence allowing the *system* to *prompt* directly for the next *block of parameters*.

control character

A character whose occurrence in a particular context initiates, modifies, or stops an *action* that affects the recording, processing or interpretation of data.

control functions

Functions related to the *man-machine interface* that are applied by the *user* independently while in a *dialogue* with the *system* application *functions*. *Control functions* have no direct impact on the *system functions*.

control key

A key which when pressed performs a *control function.*

correction character

A character used to invoke correction facilities prior to analysis of *input* by the *system.*

cursor control functions

Functions influencing the position or movement of the *cursor.*

create

An *action* to establish in the *system* a new *data set*; opposite of *delete.*

cursor

The item in the *display area* which identifies the position appropriate to the task at hand, e.g. where the next character will appear.

data set

A user-accessible set of one or more data items characterized by a particular use and also by the constraints on data format and/or values that make it suitable for this use.

deactivate

An *action* to terminate a *system* process initiated by an *activate action,* or an *action* to make a *data set* unavailable for use by the *system*; opposite of *activate.*

decimal numeral

A *numeral* in the decimal (base 10) *numbering system,* represented by the characters 0 (zero), 1, 2, 3, 4, 5, 6, 7, 8, 9 optionally preceded by D' (D apostrophe).

decomposition meta-language

A graphical *meta-language* to describe the structure of the *information entities* associated with an *MML function.*

default option

A *symbol* of the *decomposition meta-language* which indicates that the value taken by an *information entity* will be provided automatically if the user does not supply a value in the *input* for such an *information entity.*

default value

The value given to any *parameter* by the *system* in the absence of a specific value in the *user's input.*

delete

An *action* to eliminate a *data set* from the *system*; opposite of *create.*

delimiter

A character that organizes and separates items of data.

destination identifier

Identifies, after *input,* the system (destination) that, from the *user*'s perspective, becomes the new partner in a *dialogue.*

destination prologue

An operating sequence causing subsequent *inputs* to be processed in the *system* defined by the *destination identifier.*

dialogue

See *dialogue procedure.*

dialogue element

Element of a set of three types of *information entry* in a *man-machine communication*: viz. *direct information entry, information entry* through *menu-item selection* or through *form filling.*

dialogue procedure

The complete interactive procedure for interchanging data between *user* and *system* comprising *procedure prologue, procedure body* and *procedure epilogue.* In the Z.300-series Recommendations, the terms *dialogue* and *dialogue procedure* are interchangeable.

dialogue window

That part of a VDT *display area* where related *input* and *output* is displayed during the *dialogue.*

digit

A character of the *character set* representing an integer listed in Table 1/Z.314, column 3, positions 0 (zero) to 9.

direct information entry

A *dialogue element* whereby the *input* of a *command* or *destination identifier* is done without the aid of *menus* and/or *forms.*

directive

User input to direct the *system* to present information rather than to execute a *command*; can also be used in the interaction between the *user* and *system* prior to *command* execution. Directives can never cause any change in the state of the *system.*

display area

That part of a *visible display* which is available for displaying or entering data.

display area format

Expression specifying the *display area* in terms of the number of lines and columns available.

display area size

Expression specifying the height and width dimensions of the *display area.*

display memory

A memory containing characters, of which a certain number is displayed somewhere in the *display area.*

display memory recall functions

Control functions used for recalling information from the *display memory* not currently displayed in the *display area.*

documents A through G

Specially formatted information generated during various *phases* of the *methodology* for the specification of the *man-machine interface.*

drawing convention

A set of rules provided by the *decomposition meta-language* to indicate the allowed use of the *symbols* and their interconnection.

edit

An *action* to display a specified *data set* and subsequently to modify the *data set.*

end of dialogue

The indication that *dialogue* has finished.

end of output

The indication that *output outside dialogue* has finished.

end statement

Terminates *output* information from the *system* in an operating sequence where termination is not obvious.

error correction

The activity of correcting *input* which has been offered to but not accepted by the *system.*

escape indication

A mechanism to indicate that following character(s) are not to be interpreted according to the normal *syntax* rules.

exchange

SPC switching *system.*

execution character

A character which requests that the *command* be executed.

field

A part of a *window* (sometimes the entire *window*), which is used either for entering or displaying information.

filter

An *action* to form a subset of a *data set* consisting of all data items in the *data set* meeting specified criteria; the original *data set* is unaffected by this *action.*

flowline

A line representing a connection between *symbols* in:

i) a *syntax diagram*;

ii) an *information structure diagram.*

form

A list of *parameters,* including empty positions for insertion of *parameter values* by the *user.*

form filling

The activity of inserting *parameter values* into a *form,* that lists all the *parameters* of a requested *command,* and submitting the *block of parameters* contained in the completed *form* to the *system* under *user* control.

form identity

An identity unique to a *form* so that it can be distinguished from other *forms.*

form output

An *output* of a *form* belonging to a *command,* used in certain *information entry* procedures.

format effector

Any character(s) used to control the position of printed, displayed or recorded data.

function

A *system* activity necessary to the performance of a duty for which the *system* was designed (see also *class A, B, and C functions*).

functional area (or sub-area)

A set of related operation, maintenance, installation or acceptance testing *functions* to be controlled by means of *MML* (*class B functions*).

function key

A key which when pressed causes the *man-machine terminal* or the *system* to perform a specific *function.*

function model

A formal or informal representation of one or more aspects of those parts of telecommunication systems which should be controlled by means of MML.

general option

A *symbol* of the *decomposition meta-language* which indicates either that an *information entity* exists in the *system* in a predetermined manner or that it is not needed.

graphic characters

A collection of characters within the *character set* used to improve readability of *output.*

graphic terminals

Terminals which provide graphic capability (line drawing, circles, etc.) using other than alphanumeric means.

guidance output

Output which appears as a *comment* in *output* providing assistance to the *user* in a *man-machine* communication.

guidelines

General directions by which the purpose of one or more phases of the *methodology* may be accomplished.

header

General information which could comprise identification information, date and time, etc.

help output

The *output* resulting from a request for *user assistance.*

hexadecimal numeral

A *numeral* in the hexadecimal (base 16) *numbering system*, represented by the characters 0 (zero), 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, optionally preceded by H' (H apostrophe).

highlighting

Techniques used to emphasize visually a portion of the *display area* to make it stand out from adjacent portions, i.e. to call the viewer's attention to it.

identification invitation

A prompt to request the *user* to identify himself by means of a *password* and/or an identity card.

identifier

A representation of an entity, typically consisting of one or more *characters*. It is used to identify or name a unique item of data. In the *man-machine language*, the first character is a letter.

inaccessible field

A field for writing only by the *system*.

indicator

A *character input* by a *user* or *output* by a *system* to indicate a state or to request *user* or *system* action.

information entity

An information element associated with an *MML function* and usually represented in an *information structure diagram*.

information entry

General term for each of the three *dialogue elements*.

information entry through form filling

A *dialogue element* whereby the *input* of *parameter values* is done by means of *form filling*.

information entry through menu-item selection

A *dialogue element* whereby the *input* of a *command* or *destination identifier* is done by means of *menu-item selection*.

information structure (diagram)

A representation of the *information entities* associated with an *MML function* and their interrelationships.

information structure meta-language

See *decomposition meta-language*.

information unit

The smallest part of data in the *input* or *output.*

inhibit

An *action* to prevent the specified *system actions, system* responses or *functions* from occurring; these *functions* may normally be allowed by the *system* design or by the *allow action.*

initialize

An *action* to put specified data or equipment into a predefined initial (normal) condition or value.

input

i) Information that is transferred to the *system* by the *user,* e.g. *commands, directives, menu-item selections, form identities,* etc.

ii) An *action* to enter data by means of a *man-machine terminal* into the *system.*

input acknowledgement

Termination of *information entry* through *menu-item selection* or *form filling.*

input error

A *system*-detected error in *input* information.

input error information

Information describing the location and nature of an *input error.*

input field

See *accessible field.*

interaction request output

System output inviting further *user actions.*

interactive operating sequence

A sequence which may consist of a single *command entry sequence* terminated by an optional *end statement* or of a series of *command entry sequences* and/or *manual responses.* The latter occurs when, as a result of partial execution of a *function,* the *system* requests the *user* to supply it with further information in the *form* of *manual responses* or further *commands* for which *user* judgement and/or decision is required.

interface control functions

Functions used to force specific *actions* relating to the interface.

interrogate

An *action* to provide a display of the current value of the items of one or more *data sets.*

inverse video

A *video attribute* by which information can be displayed by inverting the image of the characters, such as going from light characters on a dark background to dark characters on a light background.

item description

A brief description of the nature of the item in a *menu.*

item selection procedure

A procedure to select an item out of a list of items on a *menu output.*

iteration

A *symbol* of the *decomposition meta-language* which indicates that a repetitive use of one or more *information entities* is possible.

I/O device

Device for entering or receiving data to or from a *system.* Can be controlled manually for entering or receiving data.

job

A discrete administrative activity within a telecommunications business which is designated as a part of the overall plan for running the business and characterized by *man-machine communication.*

job area

A collection of jobs particular to a given *functional area,* e.g. subscriber line maintenance, trunk line maintenance, call routing administration, etc.

keyed numeral

A *numeral* in a *numbering system* based on keypad *input,* represented by the characters 0 (zero), 1, 2, 3, 4, 5, 6, 7, 8, 9, *, #, A, B, C, D, optionally preceded by K' (K apostrophe).

layout option

A combination of *format effectors* and/or *graphic characters* used to bound elements of the *output* in a clear and readable form.

letter

A character of the *character set* representing the alphabet, listed in Table 1/Z.314, columns 4, 5, 6 and 7 excluding table positions 5/15 and 7/15.

machine

See *system.*

man

See *user.*

man-machine communication

The interchange of data between *user* and *system.*

man-machine interface

The set of *inputs, outputs,* and special *actions* as well as the man-machine interaction mechanism, including *dialogue procedures* and the interrelationships identified for these entities in the various *functional areas.*

man-machine language (MML)

The means of expression used in communication between the *user* and the *system.*

man-machine terminal

An *input/output device,* that enables the *user* and the *system* to communicate with each other, e.g. visual display terminal, printer.

manual response

A *user* response to a *system* invitation that may comprise the actuation of keys on *terminals* or switch frames, replacement of equipment, etc.

menu

A list of items, from which a selection can be made by the *user.*

menu identity

An identity unique to a *menu* so that it can be distinguished from other *menus.*

menu item

A brief description of an item in a *menu,* optionally accompanied by a *selection identity,* in order to allow a choice to be made by inputting such an identity.

menu-item selection

The activity of selecting an item using the item selection procedure and the repetition of this activity for subsequent *menus* until ultimately the procedure results in something other than further *menu output.*

menu output

An *output* of a *menu,* used in *information entry* procedures.

message waiting indication

A means of announcing, within a *dialogue procedure,* the presence of a high priority *output* addressed to this *man-machine terminal.*

meta-language

Formal means of representation using defined *symbols* according to specific rules.

methodology (for the specification of the man-machine interface)

A five-phase general working procedure that (1) provides for the generation of *MML function semantics* and (2) provides for the creation of an actual *man-machine interface* using *syntax, dialogue procedures,* and *MML function semantics.*

MML function

See *class A function.*

MML function decomposition

The division of a *function* into its constituent parts.

MML function semantics

Semantics peculiar to one or more *MML functions* within the *functional areas* (or sub-areas) that were generated by the application of the *methodology* for the specification of the *man-machine interface.* It is based upon *actions, objects, information entities* and their interrelationships.

MML syntax and dialogue procedures meta-language

A graphical *meta-language* for representing MML *input* and *output syntax* as well as *dialogue procedures.*

monologue output

Output from the *system* which occurs outside a *dialogue.*

name-defined parameter

A *parameter* which is identified by its *parameter name.*

non-decimal numeral

A *numeral* in a *numbering system* other than *decimal.*

non-terminal symbol

Representation, within a *syntax diagram,* of another *syntax diagram* by name. It is an abbreviated *symbol* for a more complex construct.

numbering system

Any notation for the representation of numbers.

numeral

A discrete representation of a number within a *numbering system.*

object

An *information entity,* usually the *system* part towards which the *action* of a *function* is directed.

octal numeral

A *numeral* in the octal (base 8) *numbering system,* represented by the characters 0 (zero), 1, 2, 3, 4, 5, 6, 7, optionally preceded by O' (letter O apostrophe).

operation and maintenance system

A *system* which supports administration personnel in performing operation and maintenance *jobs* related to *SPC systems.*

operational procedure

A process illustrating the interrelationship of *user* and *system* in performing an operation, maintenance, installation or acceptance testing *job.*

Operation and Maintenance Centre (OMC)

A physical location staffed by administration personnel responsible for operation and maintenance (O&M) of *SPC systems.*

other information

General information which may accompany the *function models* and the lists of *MML functions* in the documents B and C.

output

i) Information that is transferred from the *system* to the *user*, e.g. *help output*, etc.

ii) An *action* to transfer specified data from the *system* to a *man-machine terminal.*

output field

See *inaccessible field.*

output outside dialogue

A spontaneous *output* indicating a certain event, e.g. an alarm situation, or an output in response to a *command* previously entered in an *interactive operating sequence*, e.g. a traffic measurement result.

parameter

Data which identifies and contains pieces of information necessary to execute a *command.*

parameter argument

The smallest portion of a *parameter value* which specifies an appropriate object or value. It can be a *simple* or a *compound* structure and may be used singularly or as part of a group.

parameter block

See *block of parameters.*

parameter block entry sequence

A procedure used to input a *block of parameters.*

parameter block request indication

An indication from the *system* to the *user* to proceed with *input* of *parameters.*

parameter identity

The *parameter name* or the *parameter position* identifying a *parameter* in a *block of parameters.*

parameter name

An *identifier* which indicates unambiguously the meaning and structure of the subsequent *parameter value.*

parameter position

The sequence number of a *parameter* in a *block of parameters.*

parameter value

The part of a *parameter* that contains the information required to specify any appropriate object(s) or value(s). It consists of one or a group of *parameter arguments.*

parameter value input field

An *accessible field* that is normally empty or filled in by the *system* and should be filled in or overwritten by the *user.*

password

A character string used for identification and authorization of a *user.*

phase

One of five steps of the general working procedure that forms the *methodology* for the specification of a *man-machine interface.*

position-defined parameter

A *parameter* whose nature is identified by its position in the *parameter block* of a *command.*

procedure body

That part of a *dialogue procedure* where *commands* can be entered and new physical areas can be addressed, dependent on the authority of the *user.*

procedure description

A method of representing an *operational procedure.*

procedure epilogue

The procedure used to terminate the *dialogue procedure.* It consists of an *action* by the *user* to deactivate the *dialogue* and/or an *output* from the *system* to indicate the *end of dialogue.*

procedure prologue

A set of *actions* to activate the *man-machine terminal,* to call the *system* and to identify the *user.*

prompting

A method used by the *system* to request *input* from the *user* in a *dialogue procedure.*

prompting output

An *output* from the *system* providing guidance on the next *input* requirement.

ready indication

An *output* element used in a *dialogue procedure* to indicate that the direction of the dialogue has changed and that the *system* is ready to receive a *command* or a *destination identifier.* It is also used as the *identification invitation.*

ready indicator

An *indicator* used in the *ready indication* to indicate that the *system* is ready to receive information.

rejection output

An *output* message indicating that an *input* to the *system* is invalid and will not be acted upon, and corrections cannot be applied.

remove

An *action* to request the *system* to take specified equipment units out of service; the system still retains knowledge of the units so that they may be returned to service by the *restore action.*

request

A manual *action* used to activate a *man-machine terminal* and the *system.*

request output

A type of *response output* requesting further *input action* from the *user,* e.g. correction of an erroneous *parameter,* or supplying further information.

response output

An *output* message in the *dialogue procedure* which gives information about the state of an *input.* The output can be any one of the following types: *acceptance output, rejection output* and *request output.*

restore

An *action* to return specified equipment units to service; opposite of *remove.*

route

An *action* to instruct the *system* that any subsequent *output* of a certain type should be routed to specified media.

scrolling (windowing)

A mechanism for retrieving data not currently displayed from the *display memory* into the *display area.*

selection

A *symbol* of the *decomposition meta-language* which indicates that the choice among several *information entities* is possible.

selection identity

An identity unique to a *menu item* so that it can be distinguished from other *menu items* within the same *menu.*

semantics

The rules and conventions governing the interpretation and assignment of meaning to constructions in a language.

separator

A character used to delimit *syntax* elements.

sequence

A *symbol* of the *decomposition meta-language* which indicates a left-to-right ordering of *information entities.*

session

See *dialogue procedure.*

session status

Information reflecting the current status of the *session* in terms of *user* identity, destination identity, etc.

set

An *action* to place equipment units in a specified state (number of possible states greater than 2); possible states include in service and out of service.

simple parameter argument

A *parameter argument* made up of only one *information unit.*

sort

An *action* to rearrange the order of a *data set* according to specified (or default) criteria; the contents of the original set are not affected by this *action*, only its order.

source identifier

One or more *information units* indicating the physical area where an *output* was generated.

Specification and Description Language (SDL)

The specification and description language specified in the Z.100-series Recommendations.

spontaneous menu

A *menu* that is automatically given at the start of an *information entry.*

spontaneous output

An *output* generated by internal events of the *system*, e.g. an alarm.

status window

See *system status window.*

Stored Program Control (SPC) system

A *system* (this includes switching *systems*) that provides telecommunication services to the subscriber.

subdivision

A symbolic means in the *decomposition meta-language* of indicating the division of an entity into its constituent parts.

supplementary information

Information presenting an explanation to the *user* if required so as to ease the *input* of the *parameter value.*

symbol

A conventional representation of a concept or a representation of a concept upon which agreement has been reached.

symbolic name

A character string used for the representation of an entity.

syntax

The rules for the formation of permissible constructions (e.g. character strings) in a language, without regard to meaning.

syntax diagram

A representation either of the syntactic structure of the construct or of a portion of the *dialogue procedure.*

system

Computer-based equipment used in telecommunications to provide service to the subscriber or to support administration personnel in their *jobs.*

system information

Information related to the status of the *system.* It may contain items such as: system status *indicators,* alarm *indicators,* and a *message waiting indicator.*

system status window

A *window* indicating whether the sub-*systems* are working normally or are affected by some fault.

table

An ordered presentation of interrelated information.

terminal

Abbreviation for *man-machine terminal.*

terminal symbol

A *symbol* containing a character or string of characters which actually appear in the *input* or *output.*

terminology harmonization

Standardization of the terminology to be used in the generation of *MML function semantics.*

text block

Any combination of *clarifying texts, name-defined parameters* and/or *tables* which gives *output* information wherever it is needed or requested.

text string

A character string (excluding " (quotation mark) and *correction characters*) not interpreted within the *man-machine language* but stored in the *system* for later *output* in its original form.

tool

A means by which the task of one or more *phases* of the *methodology* for the specification of the *man-machine interface* may be accomplished.

user

The human being in *man-machine communication.*

user assistance

Information displayed by the *system* to help the *user* to perform the task.

variable text

A string of *information units* which contains information unique to the event which caused the *output.*

video attributes

Attributes to distinguish certain important information (e.g. a title, a message, a chosen item) in order to attract the attention of the *user.* They work on the characters of the information shown within an entire *window,* a part of a *window,* an entire *field* or within a part of a *field.*

visible display

The entire physical screen of a visual display terminal.

window

A part of the *display area* (sometimes the entire *display area*) which is used for entering and/or displaying functionally related data.

windowing (scrolling)

A mechanism for retrieving data not currently displayed from the *display memory* into the *display area.*

PART II

Supplements to Recommendations Z.301 to Z.341

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

MML PUBLICITY FILE

PUBLICATIONS

1.	HORNBACH, B. H. (AT&T)	MML, IEEE transactions on communications, special issue on communications software. June 1982.
2.	STEENHUISEN, A. C. (PTI)	Man-machine language for digital PRX telephone systems, Philips Telecommunications Review, Vol. 38, No. 4, November 1980.
3.	MORO, A. M. and CREMIEUX, J. P.	Observation de trafic dans le système de commutation E12, commutation et transmission No. 2, 1981 (abstract in English).
4.	BENEDETTI, M., ed. (SIP)	MML Newsletter, Volume 1, Number 1, December 1982 – Swedish Telecommunications Administration, publisher.

CONFERENCES

ISS 79 – Paris

1.	BAGNOLI, P. (SIP), GORLA, R. (ITALTEL) and SARACCO, R. (CSELT)	Proteo system – studies and experiences for man-machine dialogue implementation.
2.	BORSOTTI, A. (TELETTRA) and FRASSANI, L. (TELETTRA)	The MML for operational control, administration, and maintenance of the AFDTI switching system.
3.	BIRCHALL, S. A.	Man-machine language interface in SPC telecommunications.

Fourth International Conference on Software Engineering for Telecommunications Switching Systems, University of Warwick, United Kingdom, 20-24 July 1981.

1.	BREUNING, T. K. (SIEMENS)	Implementation of CCITT man-machine language in EWSD.
2.	GADEFAIT, J. P., KONRAT, J. L. and SURLEAU, P.	Execution mechanisms for administration programs in the EIO system.

National Telecommunications Conference, New Orleans, United States, December 1981.

1.	HORNBACH, B. H. (AT&T)	CCITT man-machine language (MML).

1982 International Zurich Seminar on Digital Communications, Swiss Federal Institute of Technology, 9-11 March 1982.

1.	SIDOR, D. J. and MICHELSEN, R. W. (AT&T)	Man-machine interaction in Bell system SPC exchanges – No. 5 ESS, the latest view.
2.	BREUNING, T. (SIEMENS)	Architecture and features of the man-machine interface in EWSD.
3.	BENEDETTI, M., (SIP) and LANZINI, G. (CSELT)	Toward a new man-machine interaction in SPC switching systems.
4.	BORSOTTI, A. (TELETTRA)	Man-machine interaction aspects and implementation concepts.

MEDARABTEL – Seminar on Maintenance, Khartoum, 13-24 March 1982.

1.	SIDOR, D. J. (AT&T)	Computer aided maintenance.

International Conference on Man-Machine Systems, UMIST, 6-9 July 1982.

1.	HILL, A. J. (Shape Technical Centre, The Hague, Netherlands)	A man-machine language for control of a telecommunications network.

5th International Conference on Software Engineering for Telecommunications Switching Systems, Lund, Sweden, 4-8 July 1983.

1.	Discussion Session Speaker: N. R. Brown (ITT)	Aspects of man-machine interaction.

National Electronics Conference, Chicago, Illinois, 24-26 October 1983.

1.	HORNBACH, B. H. (AT&T)	Status report: the CCITT languages SDL, CHILL, and MML.

NT-P Symposium (Languages and Methods for Telecommunications Applications) Turkv, Finland, 6-8 March 1984.

1.	ERIKSSON, E. (Swedish Admin.)	Presentation of CCITT Man-Machine Language (MML).

SEMINARS

1.	HORNBACH, B. H. (AT&T)	*CCITT MAN-MACHINE LANGUAGE (MML)* Delivered to Swedish Telecommunications Administration, 18 March 1983, 50 attendees.

Supplement No. 2

MML IMPLEMENTATION RECORD

The MML implementation record is designed to present in an organized and concise manner the available information relating to the applications of the CCITT MML in switching systems, systems to support operations and maintenance and other systems. The information is compiled by country and by Administrations and scientific/industrial organizations within the country. For each exchange, OMC, or other systems, current status is identified as required and/or implemented, in service, planned or under development.

The definitions for planned, under development, implemented and in service are non-overlapping, i.e. CCITT MML cannot be both planned and under development on a system. However, CCITT MML can be both required for a system and either planned, under development, implemented or in service.

Required: The CCITT MML is required for a system by an Administration. The incorporation of the CCITT MML on the system can be in the planning, under development, implemented or in service stage. This state is only to be reported by Administrations.

Planned: The CCITT MML is planned to be implemented on a system but the implementation of the CCITT MML on the system is not yet under development. A scientific or industrial organization can plan to incorporate the CCITT MML in a system.

Under development: The CCITT MML has been planned for a system but is not yet available for installation.

Implemented: The CCITT MML is implemented on a system which is either installed and operational or available for installation.

In service: The CCITT MML has been implemented on a system which is in service operationally with an Administration.

MML implementation record

Country	Organization	Conforming to MML Recommendations	Applications and status			Remarks	Application of extended MML
			Exchange	OMC	Other		
Germany (F.R.)	Deutsche Bundespost	Yes	S(EWSO) [see remark 1)]			1) SPC exchange local.	P(Digital SPC systems)
	Standard Elektrik Lorenz AG	Yes	S(System 12)				P(System 12)
	TEKADE-FGF	Yes	S(PRX-D)		U(Mobile automatic telephone)		
	Siemens AG	Yes	S(EWSD)	S(OMDS)	U(Mobile telephone exchange)		U(Exchanges and OMC)
Belgium	BTM	Yes	S(System 12) [see remark 1)] U(System 12) [see remark 2)]			1) For early 1240 systems this implementation covers the interactive direct input mode. 2) Fully interactive menu mode is under development.	
Brazil	CPqD-Telebrás	Yes	U(Trópico system)	P			
Canada	Northern Telecom	Yes	S(DMS 100)	–	S(SL 100)		
Republic of Korea	Ministry of Communications	Yes					
Egypt		Yes					
Spain	CTNE	Yes	R(AXE, 1240)	P			
United States	GTE		U(OTD 5 EAX)				U(Experimental system) 1240
	ITT	Yes	U(Experimental system)				
	AT&T	Yes	R-S(No. 5 ESS)	R-S(RMAS)	R(AMA-teleprocessing [see remark 1)]	1) The AMA teleprocessing system is a centralized billing data collection system.	U(5 ESS)
Finland	TELENOKIA	Yes	S(DX200)	S(DX200/MSW) [see remark 1)]		1) OMC is implemented by message switch (MSW) by which it is possible to centralize the use of the peripheral units of the exchanges.	
France	PTT	Yes	R [see remark 1)]	R [see remark 1)]	U [see [remark 2)]	1) These systems are in accordance with the MML defined in the Operational Standards (MEF) which date from 1976. The language is based on a sub-set of the MML and obviously goes much further than it. While essentially it	U experimental intelligent terminal for exchanges and OMC

Key: I - Implemented; P - Planned; R - Required; U - Under development; S - In service.

MML implementation record *(cont.)*

Country	Organization	Conforming to MML Recommendations	Applications and status			Remarks	Application of extended MML
			Exchange	OMC	Other		
France *(cont.)*	PTT	Yes	R [see remark 1)]	R [see remark 1)]	U [see remark 2)]	is in accordance with the MML, there are a number of divergences due to the date at which the language was constructed. 2) An experimental system.	U experimental intelligent terminal for exchanges and OMC
	CIT-ALCATEL	Yes [see remark 1)]	S [see remark 2]	I [see remark 2)]		1) Compliant with MML philosophy (Z.311-Z.318). Often strictly in conformity with CCITT MML, but some differences (generally minor ones) because we must obey the France PTT detailed specifications (issued in 1976) which, among other topics, define the syntax, procedures, commands, inputs and outputs required for operation and maintenance. 2) Three possibilities (depending on the system and the network): – only exchange, – only OMC, – option to be chosen by the customer.	
	Thomson CSF	Yes	S(MT-20/25)				
Italy	SIP	Yes [see remark 1)]	R	R		1) Other suppliers operating in Italy but related to organizations (GTE, FACE and FATME) are going to implement or have planned to implement CCITT MML conforming to SIP specifications.	R, P, U (Exchanges and OMC)
	ASST	Yes	R	R			
	ITALCABLE	Yes	R,U(System 12)	U			R,U(Exchange, System 12, and OMC)
	ITALTEL	Yes	S(CT2)S(TW16) I(UT 10/3)				

Key: I - Implemented; P - Planned; R - Required; U - Under development; S - In service.

MML implementation record *(cont.)*

Country	Organization	Conforming to MML Recommendations	Applications and status			Remarks	Application of extended MML
			Exchange	OMC	Other		
Italy *(cont.)*	TELETTRA	Yes	S(AFDT1)S(DTN) I(SPMT)I(ITZ)				R,U(PROTEO, 2 system)
	CSELT	Yes			I [see remark 1)]	1) Research applications.	
Japan	NTT	Yes	R,S(D10, D20, D30, D60, D70)				
	KDD	Yes	R,U(XE-10, XE-20) [see remark 1)]			1) Digital SPC Exchange for Telephone INTS.	
	CIAJ	Yes	S(D10)			CIAJ is composed of NEC, HITACHI, OKI and FUJITSU.	
	NEC	Yes	S(NEAX 61)	S(NCOM)			
	HITACHI	Yes	S(NDX-10)	S(NDX10-CMOC)			
	OKI	Yes	I(KB 70)	U(KL 70)			
	FUJITSU	Yes	S(FETEX-150) [see remark 1)]	U(CMOC) [see remark 2)]		1) SPC exchange. 2) Centralized maintenance and operation centre.	
Norway	Norwegian Telecommunications Administration	Yes	U(System 1240)	U			
Netherlands	PTT	Yes		I(PMT)	U(TMAS) I(SDL-System)	PMT – Program Controlled Management System for Telecommunication Systems. TMAS – Traffic Measurement and Analysis Systems.	
	Philips Telecommunicatie Industrie	Yes	S(PRX/D) [see remark 1)]			1) See Philips Telecommunicatie Review Volume 38, No. 4, November 1980.	
	AT&T and Philips Telecommunications	Yes	U(5 ESS/PRX)				U(5 ESS/PRX)
Poland	Ministry of Post and Telecommunication				I [see remark 1)]	1) Telegraph and data exchange.	

Key: I - Implemented; P - Planned; R - Required; U - Under development; S - In service.

MML implementation record *(end)*

Country	Organization	Conforming to MML Recommendations	Applications and status			Remarks	Application of extended MML
			Exchange	OMC	Other		
United Kingdom	British Telecom	Yes	R,S(System X)	R,S [see remark 1)]		1) Implemented for exchange resource management (OMC acts as MML message switch). Other OMC administrative functions under development.	
Singapore	Telecoms	Yes	S(D10) S(FETEX 150)	S(D10-CMOC) U(FETEX 150-CMOC) [see remark 1)]		1) CMOC-Centralized Maintenance and Operation Centre.	
Sweden	Swedish Telecommunications Administration	Yes	S(AXE, AXB)	S(AOM)	S(MD,TIG) [see remark 1)]	1) TIG is Time Internal Generator system MD is a PABX system.	
	LM Ericsson	Yes	S(AXE, AXB)	S(AOM)	S(MD) [see remark 1)]	1) MD is a PABX system.	

Key: I - Implemented; P - Planned; R - Required; U - Under development; S - In service.

Printed in Switzerland — ISBN 92-61-02261-8