This electronic version (PDF) was scanned by the International Telecommunication Union (ITU) Library & Archives Service from an original paper document in the ITU Library & Archives collections.

La présente version électronique (PDF) a été numérisée par le Service de la bibliothèque et des archives de l'Union internationale des télécommunications (UIT) à partir d'un document papier original des collections de ce service.

Esta versión electrónica (PDF) ha sido escaneada por el Servicio de Biblioteca y Archivos de la Unión Internacional de Telecomunicaciones (UIT) a partir de un documento impreso original de las colecciones del Servicio de Biblioteca y Archivos de la UIT.

هذه النسخة الإلكترونية (PDF) للاتصالات الدولي الاتحاد في والمحفوظات المكتبة قسم أجراه الضوئي بالمسح تصوير نتاج (ITU) والمحفوظات المكتبة قسم في المتوفرة الوثائق ضمن أصلية ورقية وثيقة من نقلاً.

此电子版（PDF版本）由国际电信联盟（ITU）图书馆和档案室利用存于该处的纸质文件扫描提供。

Настоящий электронный вариант (PDF) был подготовлен в библиотечно-архивной службе Международного союза электросвязи путем сканирования исходного документа в бумажной форме из библиотечно-архивной службы МСЭ.

© International Telecommunication Union

THE INTERNATIONAL TELEGRAPH AND TELEPHONE CONSULTATIVE COMMITTEE

# CCITT

# SIXTH PLENARY ASSEMBLY

GENEVA, 27 SEPTEMBER - 8 OCTOBER 1976

## ORANGE BOOK

## VOLUME VI.4

# PROGRAMMING LANGUAGES FOR STORED-PROGRAMME CONTROL EXCHANGES

Published by the
INTERNATIONAL TELECOMMUNICATION UNION
GENEVA, 1977

THE INTERNATIONAL TELEGRAPH AND TELEPHONE CONSULTATIVE COMMITTEE

# CCITT

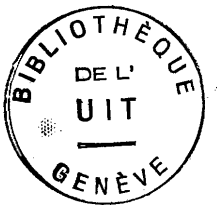
# SIXTH PLENARY ASSEMBLY

GENEVA, 27 SEPTEMBER - 8 OCTOBER 1976

## ORANGE BOOK

## VOLUME VI.4

# PROGRAMMING LANGUAGES FOR STORED-PROGRAMME CONTROL EXCHANGES




Published by the
INTERNATIONAL TELECOMMUNICATION UNION
GENEVA, 1977

ISBN 92-61-00421-0

## CONTENTS OF THE CCITT BOOK
## APPLICABLE AFTER THE SIXTH PLENARY ASSEMBLY (1976)

### ORANGE BOOK

**Volume I** — Minutes and reports of the VIth Plenary Assembly of the CCITT.
— Resolutions and Opinions issued by the CCITT.
— General table of Study Groups and Working Parties for the period 1977-1980.
— Summary table of abridged titles of Questions under study in the period 1977-1980.
— Recommendations (Series A) on the organization of the work of the CCITT.
— Recommendations (Series B) relating to means of expression.
— Recommendations (Series C) relating to general telecommunication statistics.

**Volume II.1** — General tariff principles — Lease of circuits for private service: Series D Recommendations and Questions (Study Group III).

**Volume II.2** — Telephone operation, quality of service and tariffs: Series E Recommendations and Questions (Study Group II).

**Volume II.3** — Telegraph operations and tariffs: Series F Recommendations and Questions (Study Group I).

**Volume III** — Line transmission: Series G, H and J Recommendations and Questions (Study Groups XV, XVI, XVIII, CMBD).

**Volume IV.1** — Line maintenance and measurement: Series M and N Recommendations and Questions (Study Group IV).

**Volume IV.2** — Specifications of measuring equipment: Series O Recommendations and Questions (Study Group IV).

**Volume V** — Telephone transmission quality and telephone sets: Series P Recommendations and Questions (Study Group XII).

**Volume VI.1** — General Recommendations relating to telephone switching and signalling: Series Q Recommendations and Questions (Study Group XI).

**Volume VI.2** — Signalling System No. 6: Recommendations.

**Volume VI.3** — Signalling Systems R1 and R2: Recommendations.

**Volume VI.4** — Programming languages for stored-programme control exchanges: Series Z Recommendations.

**Volume VII** — Telegraph technique: Series R, S, T and U Recommendations and Questions (Study Groups VIII, IX, X, XIV).

**Volume VIII.1** — Data transmission over the telephone network: Series V Recommendations and Questions (Study Group XVII).

**Volume VIII.2** — Public data networks: Series X Recommendations and Questions (Study Group VII).

**Volume IX** — Protection: Series K and L Recommendations and Questions (Study Groups V, VI).

Each volume also contains, for its field and where appropriate:
— definitions of specific terms used;
— supplements for information and documentary purposes.

© I.T.U.

## CONTENTS OF VOLUME VI.4 OF THE ORANGE BOOK


Page

Part I – *Series Z Recommendations (Z.101 to Z.104)*

Functional specification and description language (SDL) . . . . . 1

Part II – *Series Z Recommendations (Z.311 to Z.359)*

Man-machine language (MML) . . . . . 25


---

## PRELIMINARY NOTE

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

# PART I

## Series Z Recommendations (Z.101 to Z.104)

## FUNCTIONAL SPECIFICATION AND DESCRIPTION LANGUAGE (SDL)

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

# FUNCTIONAL SPECIFICATION AND DESCRIPTION LANGUAGE (SDL)

## TABLE OF CONTENTS


| | | | Page |
|---|---|---|---|
| Rec. Z.101 | 1. | *General explanation of the SDL* | |
| | 1.1 | Introduction | 5 |
| | 1.1.1 | Methods of presentation | 5 |
| | 1.1.2 | General objective | 5 |
| | 1.1.3 | Area of application | 5 |
| | 1.2 | Framework | 5 |
| | 1.2.1 | Basic definitions | 5 |
| | 1.2.2 | Specifications and descriptions | 6 |
| | 1.2.3 | Functional specifications and functional descriptions | 6 |
| | 1.2.4 | Functional specification blocks and functional description blocks | 6 |
| | 1.3 | Basic concepts for the SDL | 7 |
| | 1.3.1 | Signals | 7 |
| | 1.3.2 | Inputs | 7 |
| | 1.3.3 | States | 7 |
| | 1.3.4 | Transitions | 7 |
| | 1.3.5 | Outputs | 7 |
| | 1.3.6 | Decisions | 7 |
| | 1.3.7 | Tasks | 7 |
| Rec. Z.102 | 2. | *Symbols and rules* | |
| | 2.1 | General | 8 |
| | 2.2 | Symbols | 8 |
| | 2.3 | Sequence rules | 8 |
| | 2.4 | Flow lines and connectors | 9 |
| | 2.5 | Flow line rules | 9 |
| | 2.6 | Annotations | 9 |
| | 2.7 | Annotation rules | 9 |
| | 2.8 | Connectivity diagram | 10 |
| | 2.9 | Draughting conventions | 10 |
| Rec. Z.103 | 3. | *Optional use of pictorial elements within state symbols* | 11 |

Page

Rec. Z.104 4. (Has not yet been defined)

Annex to Recommendations Z.101 to Z.103 – *Examples of the use of SDL*

A. General introduction . . . . . 11
A.1 General . . . . . 11
A.2 Functional block interaction . . . . . 12
B. Figures and examples . . . . . 12

*Figure 1* – Example 1 of a local call handling process . . . . . 13
*Figure 2* – Example 2 of a local call handling process using state pictures . . . . . 16
*Figure 3* – Functional block interaction for Examples 1 and 2 . . . . . 17
*Figure 4* – Example 3 – a system configuration process during fault conditions, using state pictures . . . . . 18
*Figure 5* – Functional block interaction for Example 3 . . . . . 19
*Figure 6* – Example 4, an R2 outgoing line-signalling process . . . . . 20
*Figure 7* – Example 5, an R2 outgoing line-signalling process using state pictures . 23
*Figure 8* – Functional block interaction for Examples 4 and 5 . . . . . 24

**Recommendation Z.101**

## 1. GENERAL EXPLANATION OF THE SPECIFICATION AND DESCRIPTION LANGUAGE (SDL)

This Recommendation deals with the presentation of the functional specification and of the description of the internal logic processes in stored programmed control (SPD) telephone exchanges.

### 1.1 *Introduction*

#### 1.1.1 *Methods of presentation*

The methods of presentation of functional specifications and of descriptions of internal logic processes in SPC telephone exchanges can be subdivided into the following categories:

- narrative methods (natural language and numerical information supported by drawings and lists etc.);
- formalized presentation methods.

The narrative methods, which can be used to a large extent for both specifications and descriptions of SPC telephone switching systems, need no standardization by the CCITT.

Considering the formalized methods of presentation, the subject of this Recommendation is a graphical method, based on state transition diagrams, using the symbols and rules of the Functional Specification and Description Language (SDL) described in the following sections. (It may be noted that some processes of an SPC switching system may require specifications and/or descriptions by methods other than in this Recommendation.) Wherever appropriate the symbols of the SDL have been taken from the ISO standard for flow charts (ISO/R 1028-1969).

#### 1.1.2 *General objective*

The objective of the SDL is to provide a standardized method of presentation:

- that is easy to learn, to use and to interpret in relation to the needs of operating organizations;
- that provides unambiguous specifications and/or descriptions for tendering and ordering;
- that provides the capability of meaningful comparisons between competitive types of SPC telephone exchanges;
- that is open-ended to be extended to cover new developments.

#### 1.1.3 *Area of application*

The main area of application covers all types of SPC telephone switching systems. Within these systems the following functions are included amongst others:

- call processing (e.g. call handling, routing, signalling, metering, etc.);
- maintenance and fault treatment (e.g. alarms, automatic fault clearing, configuration control, routine tests, etc.);
- system control (e.g. overload control, modification and extension procedures, etc.).

### 1.2 *Framework*

#### 1.2.1 *Basic definitions*

a) To clarify the meaning of terms used in the SDL, a number of definitions are given below.

b) Some of the terms defined below have been in use in other fields and will have connotations related to those fields. Care should therefore be exercised by those concerned with the SDL that their use and understanding of such terms is in accord with the definitions contained in this section.

### 1.2.2 *Specifications and descriptions*

a) The requirements of a system are defined in a *specification* of that system and the implementation of those requirements in a *description* of the system.

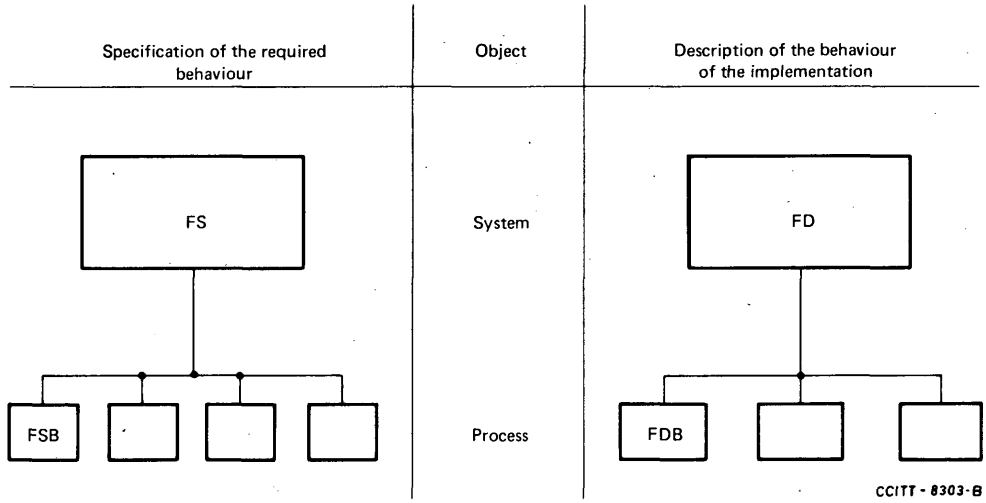
b) Both *specifications* and *descriptions* consist of two parts: the former comprises *general parameters* required of the system and the *functional specifications* (FS) of its required behaviour; the latter comprises *general parameters* of the system as implemented and the *functional description* (FD) of its actual behaviour.

c) The *general parameters* in both cases relate to such matters as temperature limits, transmission limits, construction, exchange capacity, grade of service, etc.

### 1.2.3 *Functional specifications and functional descriptions* (See Figure 1/Z.101)

The *functional specification* (FS) of a system is a specification of the total functional requirements of that system from all significant points of view.

The *functional description* (FD) describes the actual behaviour of the implementation of those functional requirements in terms of the internal structure and logic processes within the system.




FS = functional specification
FD = functional description
FSB = functional specification block
FDB = functional description block

*Note.* – The partitioning of an FS into FSBs for a particular system does not necessarily correspond to the partitioning of the FD into FDBs for the same system.

FIGURE 1/Z.101 – **Partitioning**

### 1.2.4 *Functional specification blocks and functional description blocks*

a) *functional specification blocks* (FSB) and *functional description blocks* (FDB), as tools for specification or description, are both entities of manageable size and relevant internal relationship. (See Figure 1/Z.101.)

A functional specification block specifies the desired behaviour of one or more processes.

A functional description block describes the means by which the required behaviour of processes is achieved.

b) The behaviour of a process is described in terms of *inputs*, *states*, *transitions*, *decisions*, *tasks* and *outputs*.

c) FSBs and/or FDBs might relate to such processes as call handling, traffic recording, signalling, switchboard operations, man-machine procedures, queueing, etc.

d) FSBs and FDBs are formed by partitioning their parent FSs and FDs. There may be more than one way of partitioning any FS or FD.

e) Both FSBs and FDBs may be partitioned further to form subsidiary FSBs and FDBs so that the hierarchical structure is extended downwards.

f) The partitioning of an FS into FSBs for a particular system does not necessarily correspond to the partitioning of the FD into FDBs for the same system. The boundaries of FSBs and FDBs do not necessarily correspond at any hierarchical level.

g) At any given hierarchical level a process appears only in one block.

h) Every process specified or described by a *block* should have a well-defined boundary across which *signals* pass as explicitly indicated. Interfaces between FSBs, and between FDBs, should similarly be well defined.

## 1.3 *Basic concepts for the SDL*

The SDL is based on the following definitions:

### 1.3.1 *Signals*

a) A *signal* is a flow of data conveying information to a process.

b) A *signal* may be either in hardware or in software form.

c) If the information flow is from a process described by a *block* to a process described by another *block* it is an *external signal.* If the flow is between processes described by the same *block* it is an *internal signal.*

### 1.3.2 *Inputs*

An *input* is an incoming *signal* which is recognized by a process. (It is not to be confused with input as applied to normal data processing.)

In accordance with the definition of *signals*, an input can be internal or external.

### 1.3.3 *States*

A *state* is a condition in which the action of a process is suspended awaiting an *input.*

### 1.3.4 *Transitions*

A *transition* is a sequence of actions which occurs when a process changes from one *state* to another in response to an *input.*

A process can be either in one of its states or in a transition at any one instant.

### 1.3.5 *Outputs*

An *output* is an action within a *transition* which generates a *signal* which in turn acts as an *input* elsewhere. (It is not to be confused with output as applied to normal data processing.)

In accordance with the definition of signals an output can be either internal or external.

### 1.3.6 *Decisions*

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 paths to continue the *transition.*

### 1.3.7 *Tasks*

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

**Recomendation Z.102**

## 2. SYMBOLS AND RULES

### 2.1 *General*

Each process represented consists of several states and the various transitions between them. An input will trigger the leaving of a state to travel along a transition, executing tasks, generating outputs and branching on decisions until another state is reached. The representations may be linear, with multiple appearances of a single state if convenient, or may be of mesh form or any combination of the two.

The concepts of state, input, task, output and decision are represented by their respective symbols. The appropriate interconnection of such symbols by flow lines represents the logical flow of a process.

### 2.2 *Symbols*

The recommended symbols appear in Figure 2/Z.102.

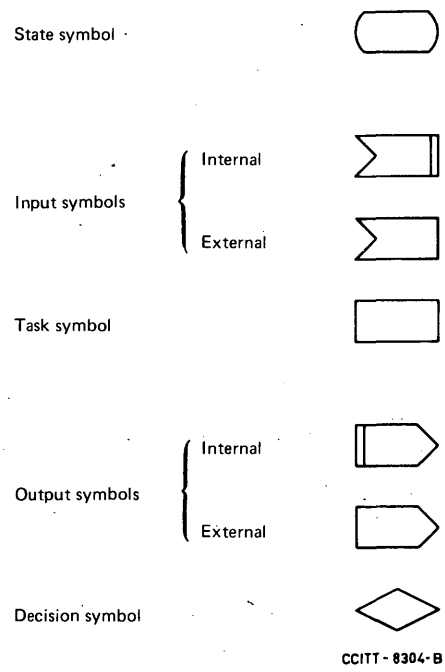



FIGURE 2/Z.102 – Recommended symbols

### 2.3 *Sequence rules*

Certain rules for the use of the symbols and their interconnections are required to ensure a valid representation of a process. For the purpose of these rules, *follow* means *follow immediately.*

2.3.1 A state symbol may only be followed by one or more input symbols.

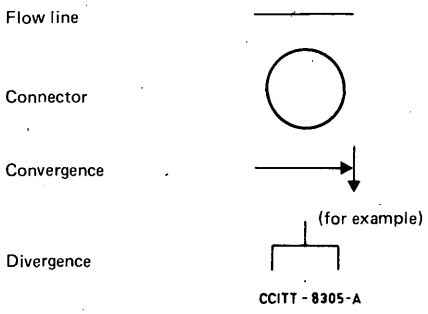
2.3.2 Each input symbol follows one and only one symbol which must be a state symbol.

2.3.3 Each input symbol is followed by one and only one symbol, which may be any symbol except another input symbol.

2.3.4 Each task or output symbol is followed by one and only one symbol, which may be any symbol except an input symbol.

2.3.5 A decision symbol must be followed by two or more symbols, which may not be input symbols.

## 2.4 *Flow lines and connectors*




## 2.5 *Flow line rules*

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

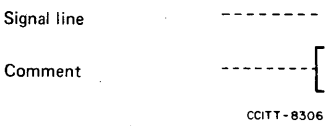
2.5.2 A solid flow line may be broken by a pair of associated connectors, with the flow assumed to be from the out-connector to its associated in-connector.

2.5.3 Where two or more symbols are followed by a single symbol the flow lines leading to that symbol converge. This convergence may appear as one flow line flowing into another or as more than one out-connector associated with a single in-connector.

2.5.4 Where a symbol is followed by two or more other symbols a flow line leading from that symbol may diverge into two or more flow lines.

2.5.5 Arrow heads are required whenever two flow lines converge and whenever a flow line enters an out-connector or a state symbol. Arrow heads are prohibited on flow lines entering input symbols.

## 2.6 *Annotations*




## 2.7 *Annotation rules*

2.7.1 Where an output and an associated input symbol represent a signal from one process to another a dashed line from one symbol to the other may be included to indicate the association.

These dashed lines representing signals may diverge, converge or be broken by connectors.

2.7.2 Comments may be enclosed by a single square bracket connected by a dashed line to any symbol or flow line.

## 2.8 *Connectivity diagram*

Figure 3/Z.102 summarizes the sequence rules and flow line rules stated in the text.

*Note.* – Divergence and convergence both include the trivial case of a single continuity flow line.

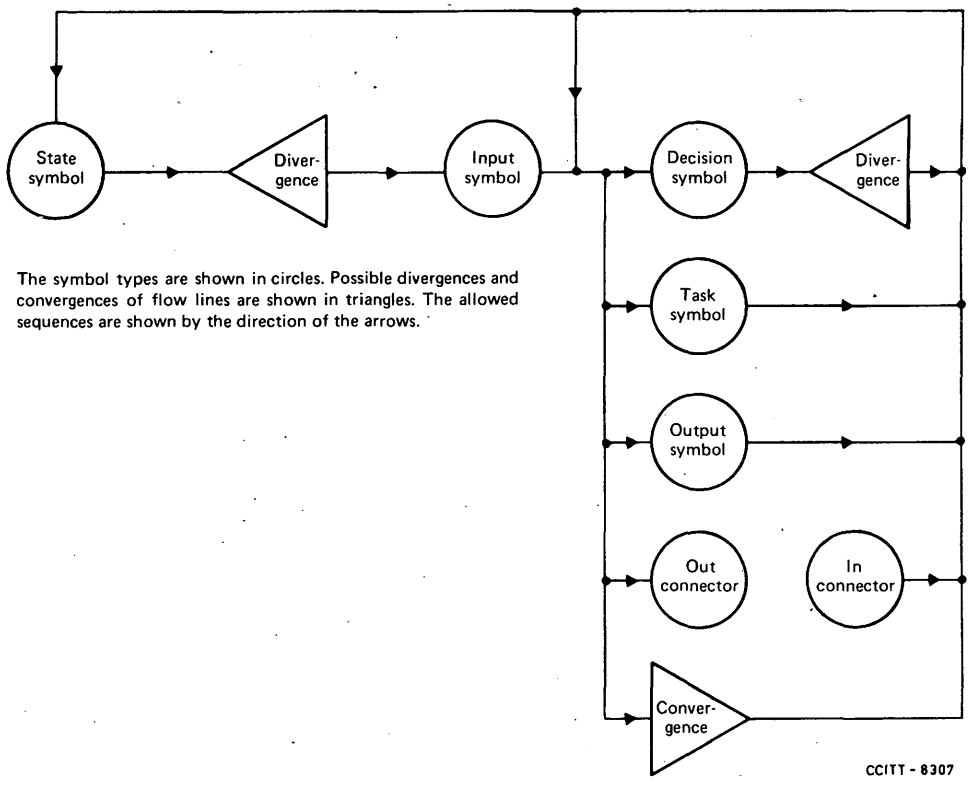



FIGURE 3/Z.102 – **Connectivity diagram**

## 2.9 *Draughting conventions*

2.9.1 All symbol boxes of the same type shall preferably be of the same size within any one diagram.

2.9.2 Mirror images of input and output symbols are allowed.

2.9.3 Flow lines are horizontal or vertical and have sharp corners.

2.9.4 Flow lines that cross have no logical relationship.

2.9.5 The preferred aspect ratio of symbols is 2:1.

2.9.6 The text associated with the symbols should be placed within those symbols where practicable.

**Recommendation Z.103**

## 3. OPTIONAL USE OF PICTORIAL ELEMENTS WITHIN STATE SYMBOLS [1)]

### 3.1 *General*

3.1.1 The use of pictorial elements within a state symbol forms an optional part of the SDL.

Such pictorial elements can provide advantages when applied to certain functional specifications and functional descriptions, resulting in a more compact and less verbal diagram.

3.1.2 With pictorial elements each state is represented by a state symbol containing a state picture and a state identifier (normally consisting of a state number and a state title) with the format shown in Figure 4/Z.103.

3.1.3 The total processing involved when going from one state to the following state is that required to effect the changes in the state pictures, together with the processing indicated in any decisions, outputs or tasks appearing in the transition between the states.

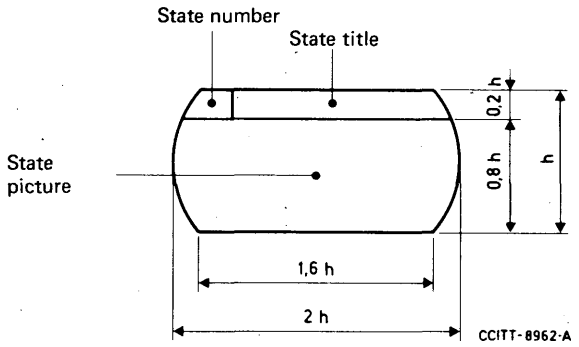



FIGURE 4/Z.103 – **Recommended format for a state symbol with pictorial elements**

**Recommendation Z.104** (Has not yet been defined)

[1)] The standardization of symbols for pictorial elements is the subject of further study in new Question 7/XI. The preliminary results from the Study Period 1973-1976 are presented in Annex 2 to the new Question 7/XI, Contribution COM XI-No. 1.

ANNEX TO RECOMMENDATIONS Z.101-Z.103

## Examples of the use of SDL

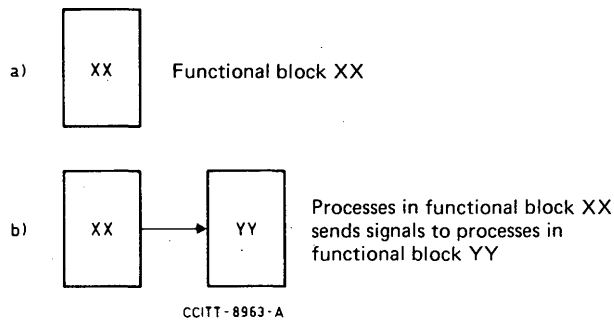
A. *General introduction*

A.1 *General*

The examples in this section are intended to illustrate the versatility of the SDL in application to the specification (before design) and the description (after design) of several processes typical of SPC switching systems.

A.2 *Functional block interactions*

To provide a framework in which each example can be understood, the concepts of Recommendation Z.101 have been applied in the form of a simple diagram of the interactions between functional blocks for each example. The interpretation of the diagrams is as follows:




A.3 These examples have been designed to show the use of the SDL, and are not international specifications.

B. *Figures and examples*:

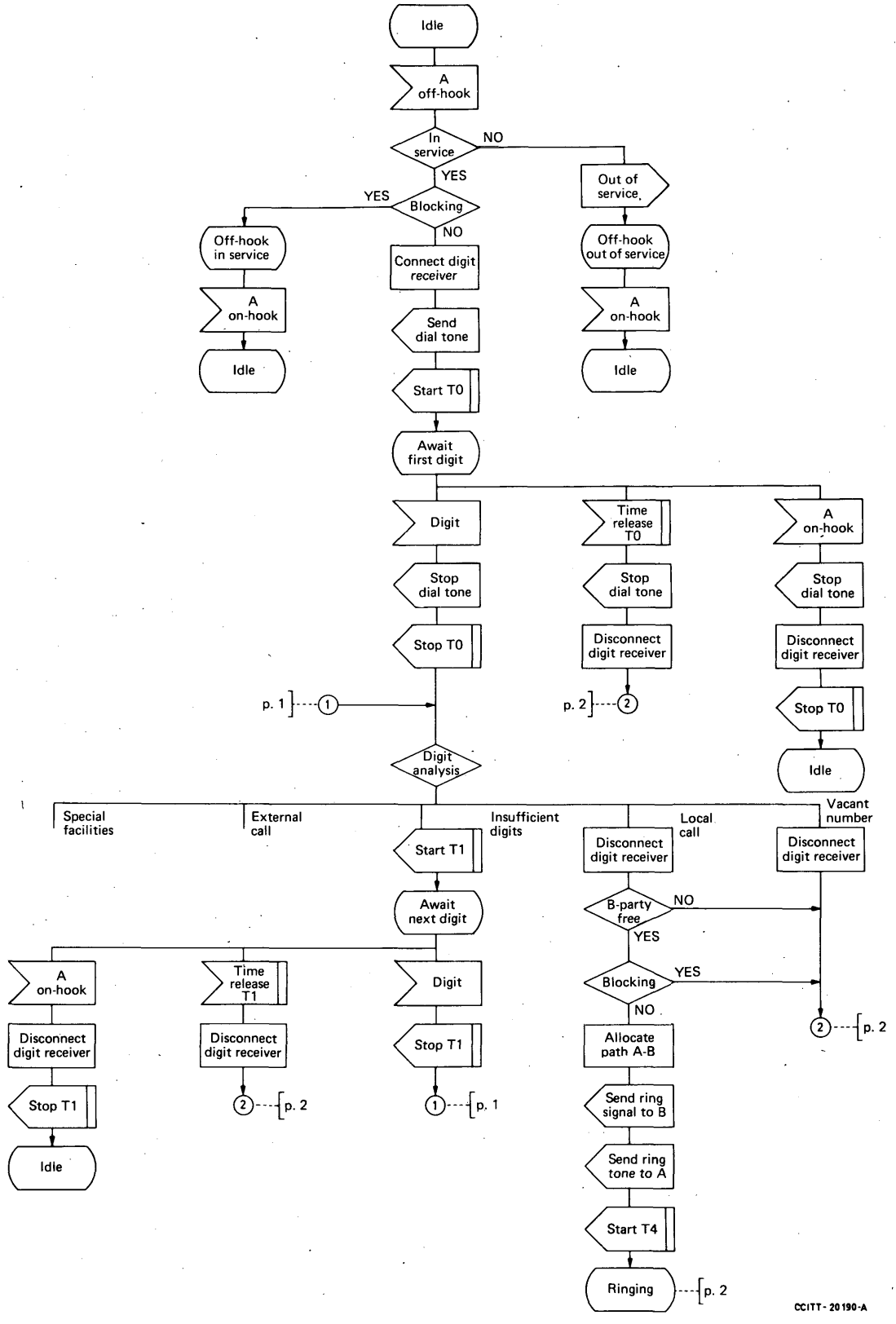



FIGURE 1 – **Example 1 of a local call handling process, page 1**

For notes to this Figure, see end of this Annex.

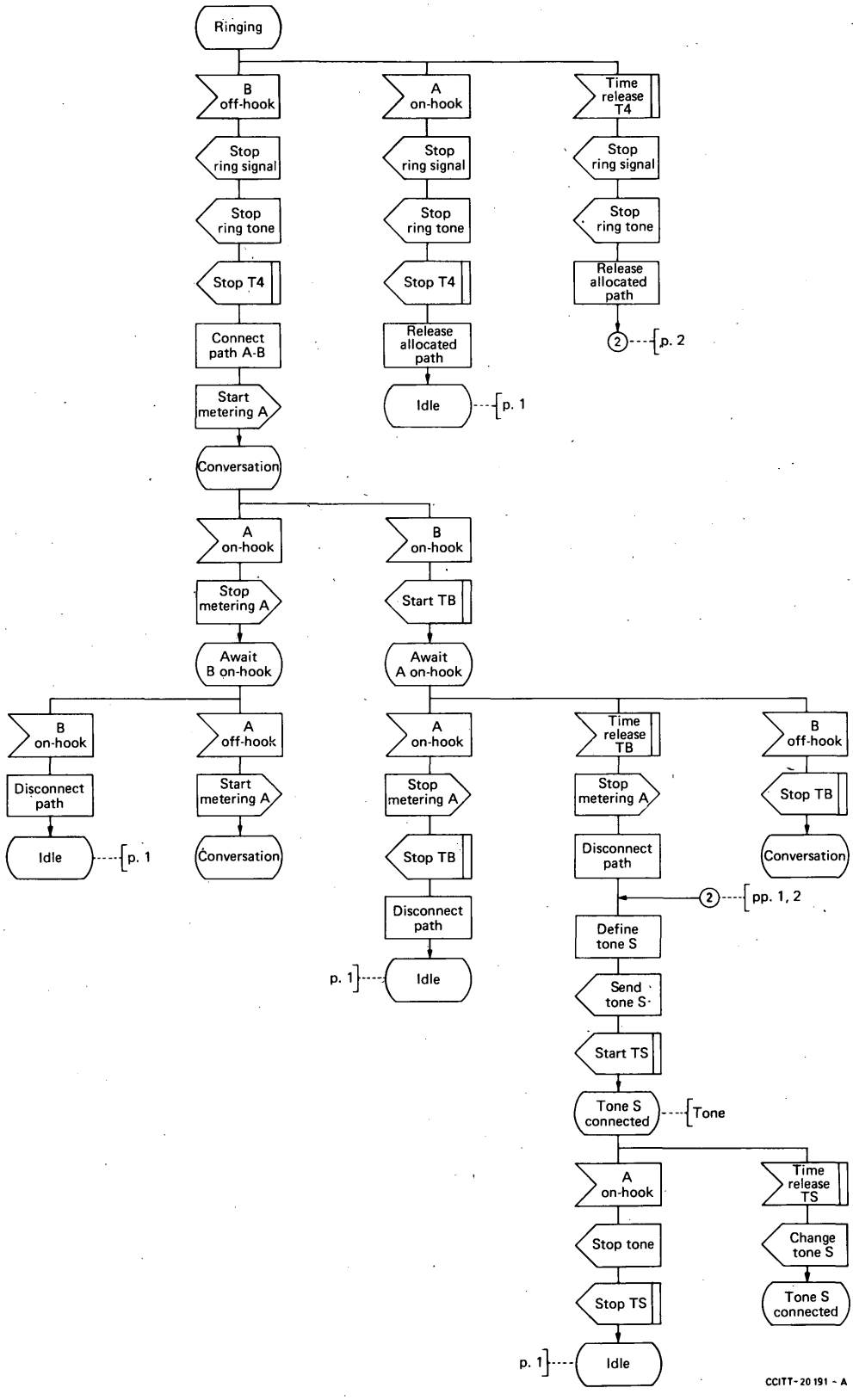



FIGURE 1 – Example 1 of a local handling process, page 2

For notes to this Figure, see end of this Annex.

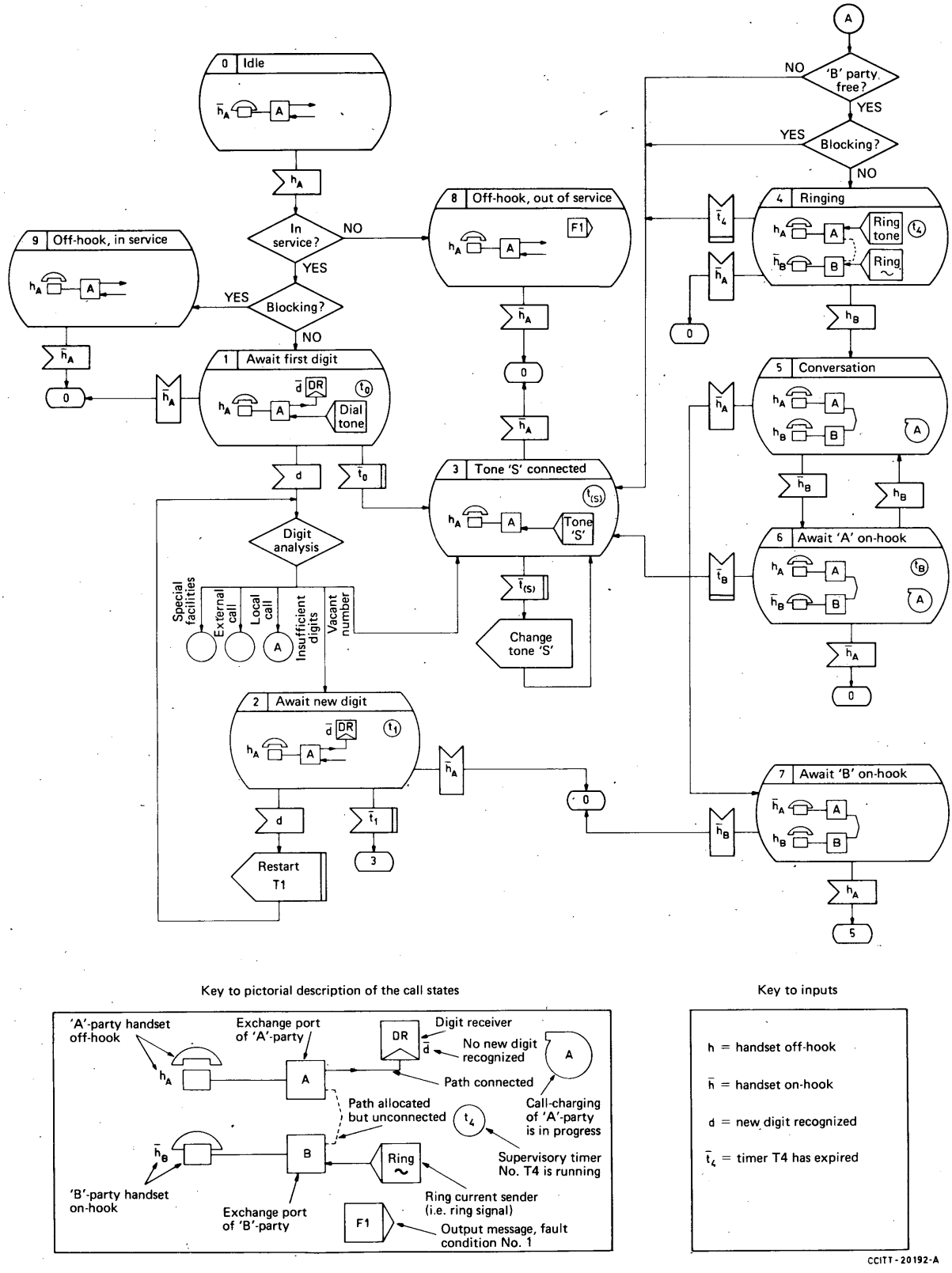



FIGURE 2 – **Example 2 of a local call handling process, using state pictures**

For notes to this Figure, see end of this Annex.

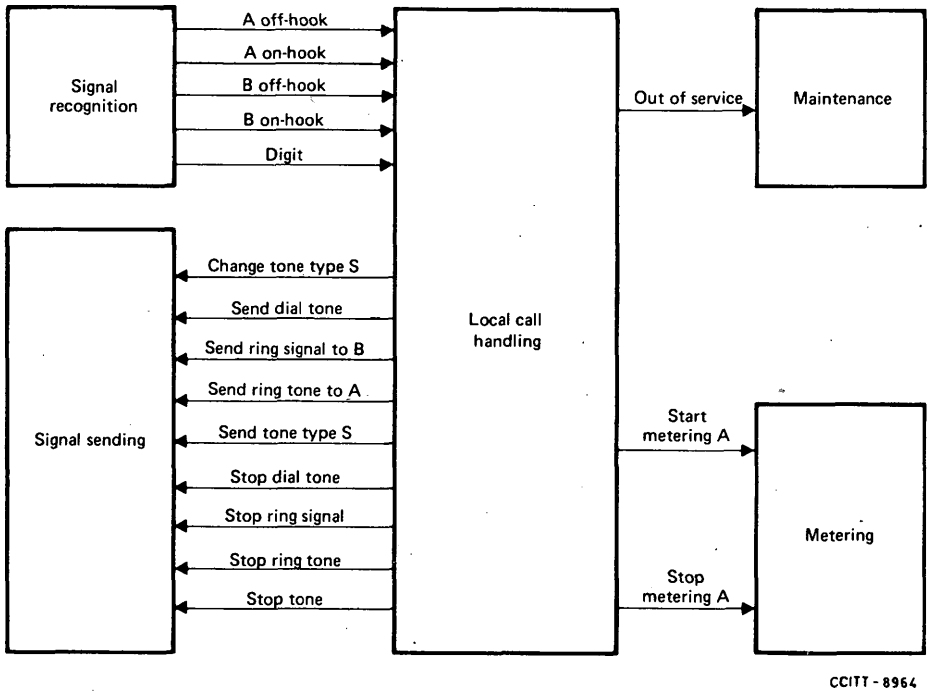



FIGURE 3 – Functional block interaction, for Examples 1 and 2

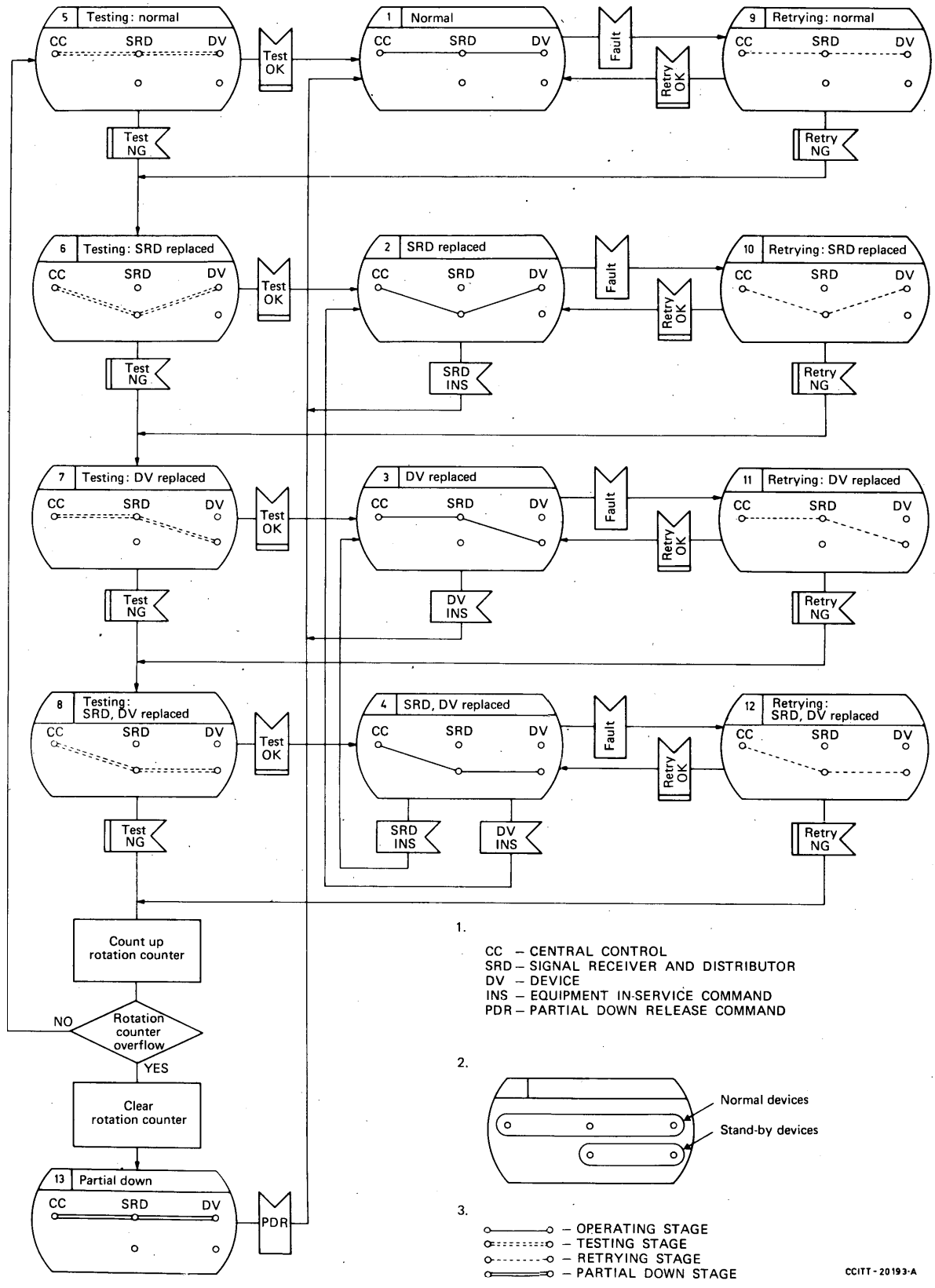



FIGURE 4 – Example 3, a system configuration process during fault conditions using state pictures

For notes to this Figure, see end of this Annex.

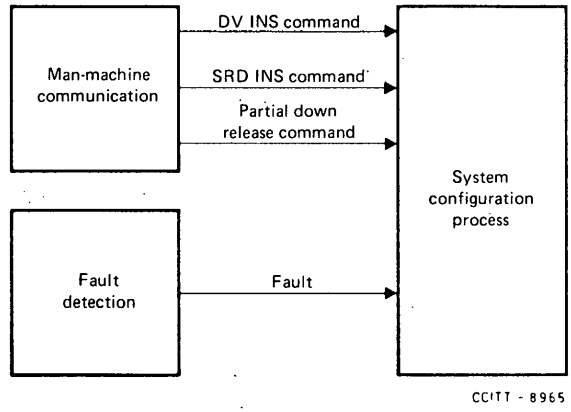



FIGURE 5 – **Functional block interaction for Example 3**

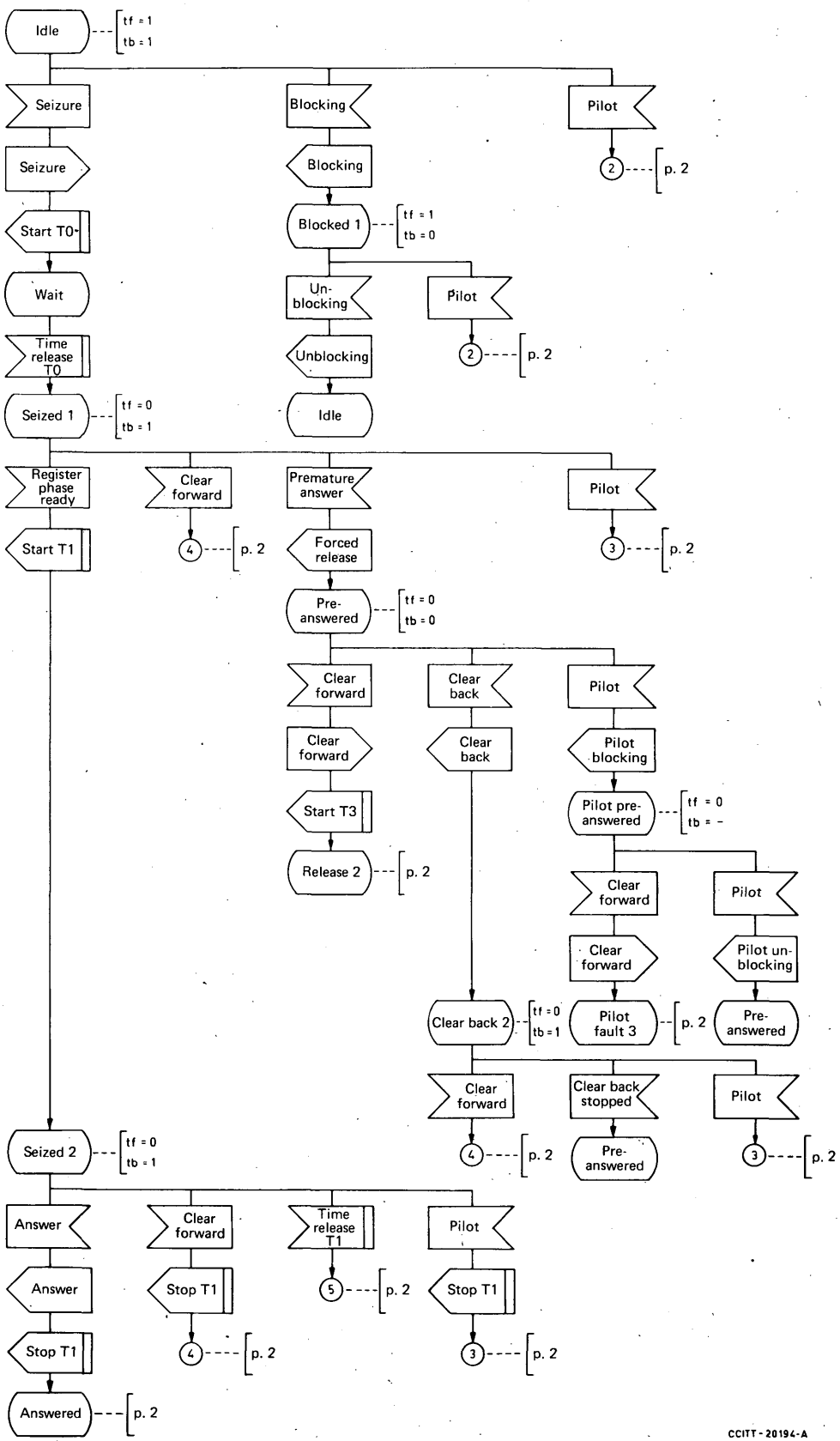



FIGURE 6 – **Example 4, an R2 outgoing line-signalling process, page 1**

For notes to this Figure, see end of this Annex.

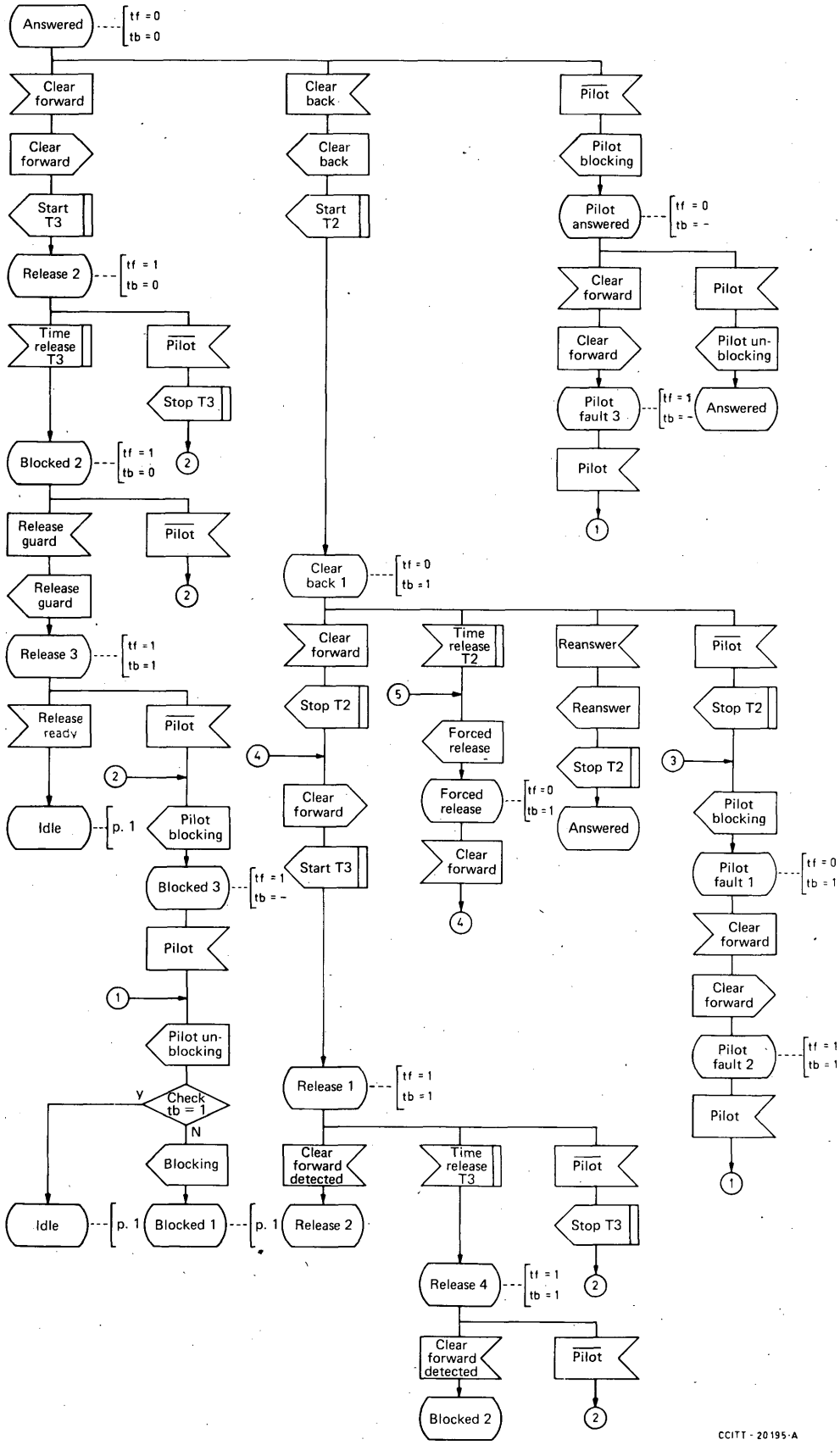



FIGURE 6 – **Example 4, an R2 outgoing line-signalling process, page 2**

For notes to this Figure, see end of this Annex.

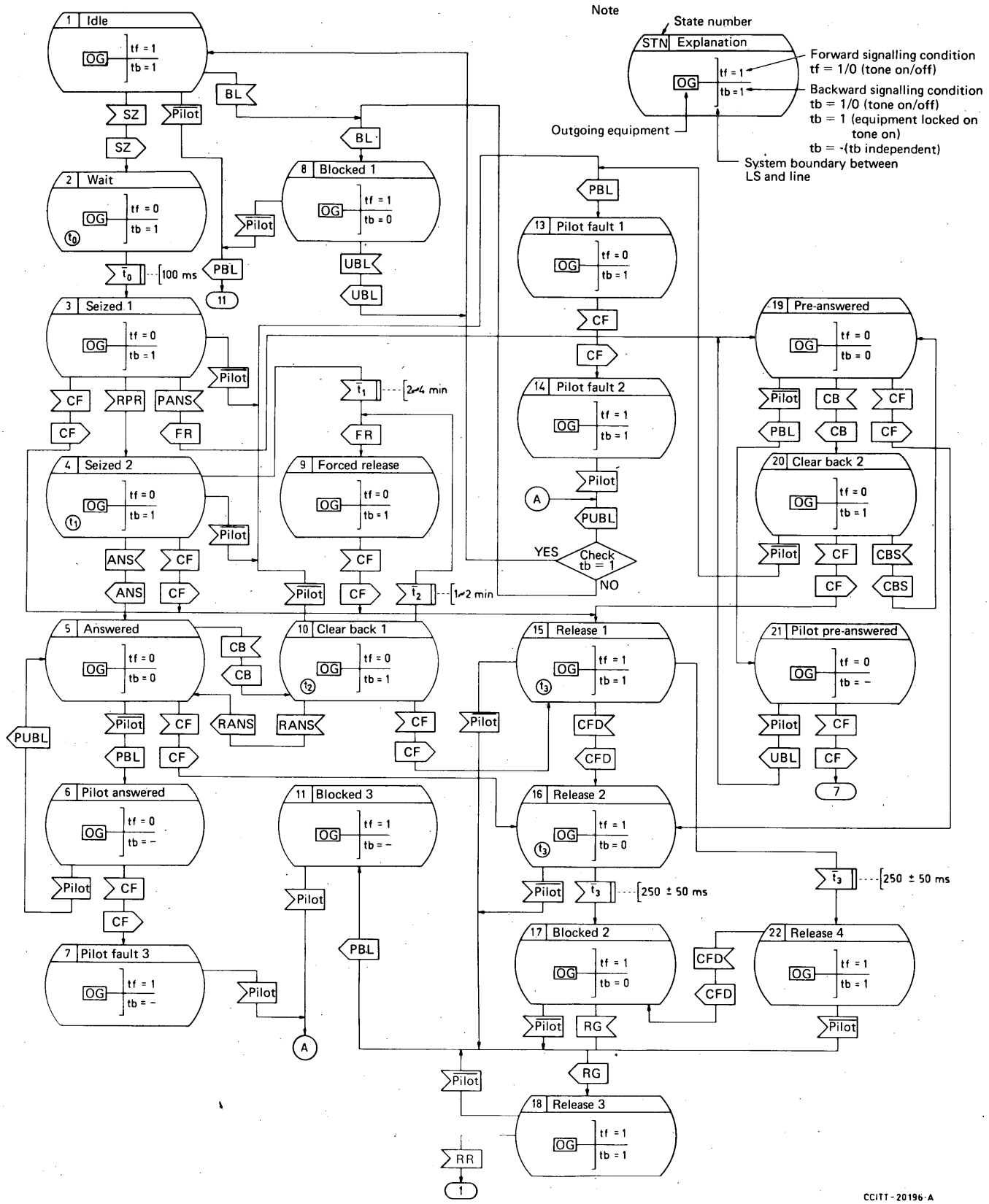



FIGURE 7 – **Example 5, an R2 outgoing line-signalling process using state pictures**

For notes to this Figure, see end of this Annex.

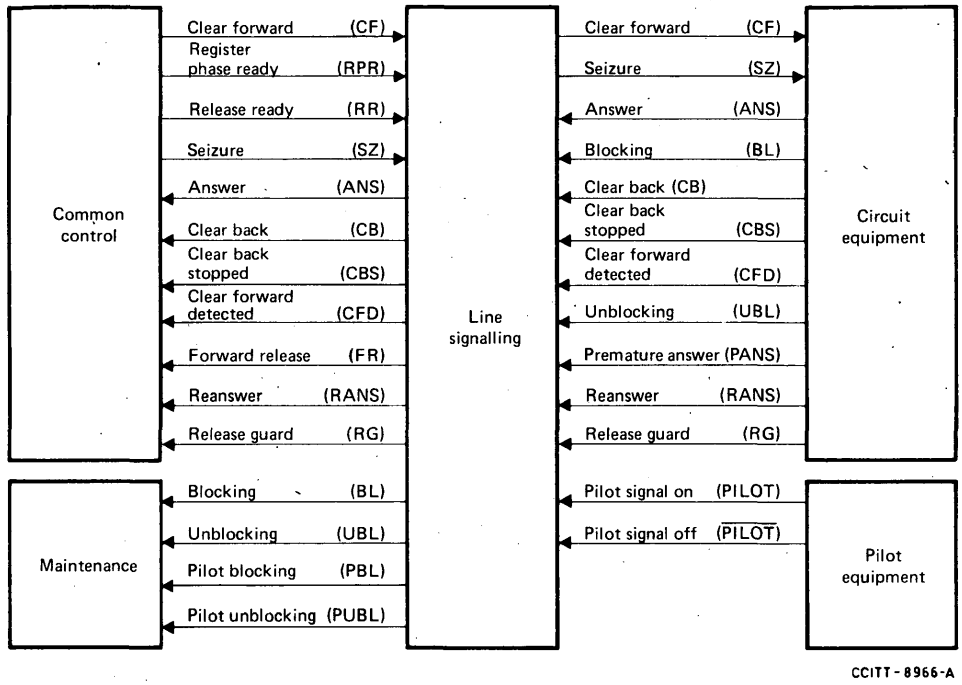



FIGURE 8 – **Functional block interaction for Examples 4 and 5**

*Notes to Figure 1* (Example 1)

*Note 1.* – This example has been designed to show the use of the SDL, and is not an international specification for call handling.

*Note 2.* – The use of the *blocking* decisions in the SDL diagram accounts for the possibility of there being no free device or free path before device connection or path connection.

*Note 3.* – The service tones are considered to be of various types S = 1, 2, 3, ... that are not identified separately in Figure 1.

*Notes to Figure 2* (Example 2)

*Note 1.* – This example has been designed to show the use of the SDL, and is not an international specification for local call handling.

*Note 2.* – The use of the *blocking* decisions in the SDL diagram accounts for the possibility of there being no free device or free path before device connection or path connection.

*Note 3.* – The service tones are considered to be of various types S = 1, 2, 3, ... that are not identified separately in Figure 2.

*Note 4.* – Each state is shown only once in full, i.e. with its state number, state title and state picture. All other appearances of the state are shown by a diminished size state symbol containing the state number alone as sufficient identification.

*Note 5.* – The functional block interactions of this Example 2 are shown in Figure 3.

*Note to Figure 4* (Example 3) – This Example has been designed to show the use of the SDL, and is not an international specification.

*Notes to Figure 6* (Example 4)

*Note 1.* – This example has been designed to show the use of the SDL, and is not an international specification.

*Note 2.* – The input and output symbols have been drawn in such a way that their orientation indicates the direction of the signals between the appropriate FSBs. Common control and maintenance is depicted to the left; circuit equipment and pilot equipment is depicted on the right.

*Note 3.* – The abbreviation tf = 1 (tf = 0) indicates that the forward tone is on (off). tb refers to the backward tone.

*Note 4.* – The functional block interactions of this Example 4 are shown in Figure 8.

*Notes to Figure 7* (Example 5)

*Note 1.* – This example has been designed to show the use of the SDL and is not an international specification.

*Note 2.* – The functional block interactions of this Example 5 are shown in Figure 8.

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

# PART II

## Series Z Recommendations (Z.311 to Z.359)

## MAN-MACHINE LANGUAGE (MML)

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

# MAN-MACHINE LANGUAGE (MML)

## TABLE OF CONTENTS


Page

SECTION 1 – *General principles*

Rec. Z.311 1. *Introduction*

1.1 Field of application . . . . . 31
1.2 Basis of MML . . . . . 31
1.3 Input/output . . . . . 32
1.4 Extensibility and sub-setting . . . . . 33

Rec. Z.312 2. *Basic form lay-out*

2.1 General . . . . . 33
2.2 Recommended formats for presenting informations in MML . . . . . 33
2.2.1 Format F1 . . . . . 33
2.2.2 Format F2 . . . . . 33

Rec. Z.313 3. *The character set*

3.1 General . . . . . 34
3.2 The sub-set selected for the MML . . . . . 34
3.3 Classification of characters . . . . . 35

Rec. Z.314 4. *The meta-language for describing the syntax and procedures*

4.1 Introduction . . . . . 36
4.2 Basic elements . . . . . 37
4.3 Rules . . . . . 37

Rec. Z.315 5. *Input (command) language syntax specification*

5.1 General . . . . . 38
5.2 Command . . . . . 38
5.3 Command code . . . . . 38
5.4 Block of parameters . . . . . 38
5.5 Parameters . . . . . 38
5.5.1 Parameter name defined parameters . . . . . 38
5.5.2 Position defined parameters . . . . . 38
5.6 Parameter name . . . . . 38
5.7 Parameter value . . . . . 39
5.8 Parameter argument . . . . . 39
5.9 Information unit . . . . . 39

Page

5.10 Basic elements . . . 39
5.10.1 Identifier . . . 39
5.10.2 Symbolic-name . . . 39
5.10.3 Decimal numeral . . . 39
5.10.4 Non-decimal numeral . . . 39
5.10.5 Separator and separator string . . . 39
5.10.6 Indicator . . . 40
5.10.7 Control character . . . 40
5.10.7.1 Format effector . . . 40
5.10.8 Undefined characters . . . 40
5.11 Information grouping . . . 40
5.11.1 General . . . 40
5.11.2 Blocks of parameters . . . 40
5.11.3 Parameter arguments . . . 40
5.11.4 Information units . . . 40
5.12 Summary of use of characters . . . 41
5.13 Formal specification of the input (command) language syntax in diagrams . 42
5.13.1 Command . . . 42
5.13.1.1 Command format . . . 42
5.13.1.2 Command code . . . 42
5.13.1.3 Block of parameters . . . 42
5.13.1.4 Position defined parameter . . . 42
5.13.1.5 Parameter name defined parameter . . . 42
5.13.1.6 Parameter name . . . 43
5.13.1.7 Parameter value . . . 43
5.13.1.8 Parameter argument . . . 43
5.13.1.9 Information unit . . . 43
5.13.2 Syntactic basic elements . . . 43
5.13.2.1 Identifier . . . 43
5.13.2.2 Symbolic name . . . 44
5.13.2.3 Decimal numeral . . . 44
5.13.2.4 Hexa-decimal numeral . . . 44
5.13.2.5 Octal numeral . . . 45
5.13.2.6 Binary numeral . . . 45
5.13.3 Alphabet . . . 45
5.13.3.1 Letter . . . 45
5.13.3.2 Digit . . . 45

Rec. Z.316 6. *Output language, syntax and general semantics* [1] (provisional contents list only) . . . 45

6.1 Definitions
6.2 Output format
6.3 Header
6.4 Prefix statement
6.5 Information statement
6.6 End statement
6.7 Types of output
6.7.1 Response outputs
6.7.2 Acceptance output
6.7.3 Solicited outputs

[1] There are no Recommendations yet for 6. The matter is still under study.

Page

6.7.4 Automatic outputs
6.7.5 Alarm outputs
6.7.6 Ready indicator
6.8 Summary of formal syntax

Rec. Z.317 7. *Man-machine dialogue*

7.1 Description of dialogue procedures . . . . . 46
7.1.1 General . . . . . 46
7.1.2 Procedure prologue . . . . . 46
7.1.2.1 Request . . . . . 46
7.1.2.2 Ready indicator . . . . . 47
7.1.2.3 Password . . . . . 47
7.1.2.4 Header . . . . . 47
7.1.2.5 Destination identifier . . . . . 47
7.1.3 Direct mode . . . . . 47
7.1.3.1 Response output . . . . . 47
7.1.3.2 End statement . . . . . 47
7.1.4 Continuation mode . . . . . 47
7.1.5 Parameter introduction sequence . . . . . 48
7.1.6 End code . . . . . 48
7.2 Formal specification of the dialogue procedure syntax in diagrams . . . . . 48
7.2.1 Sentence and dialogue procedure . . . . . 48
7.2.1.1 Sentence . . . . . 48
7.2.1.2 Dialogue procedure . . . . . 48
7.2.2 Procedure prologue . . . . . 49
7.2.2.1 Request . . . . . 49
7.2.2.2 Ready indicator . . . . . 49
7.2.2.3 Password . . . . . 49
7.2.2.4 Header . . . . . 50
7.2.2.5 Destination identifier . . . . . 50
7.2.3 Direct mode operating sequence . . . . . 50
7.2.3.1 Response output . . . . . 50
7.2.3.2 End statement . . . . . 50
7.2.4 Continuation mode operating sequence . . . . . 50
7.2.5 Parameter introduction sequence . . . . . 51
7.2.6 End code . . . . . 51

Rec. Z.318 8. *List of functions*

8.1 Introduction . . . . . 51
8.2 Functions . . . . . 51
8.2.1 General functions . . . . . 51
8.2.2 Operational functions . . . . . 51
8.2.2.1 Subscriber administration . . . . . 51
8.2.2.2 Routing administration . . . . . 52
8.2.2.2.1 Changing data relating to a group of circuits . . . . . 52
8.2.2.2.2 Changing routing and analysis data . . . . . 52
8.2.2.3 Traffic administration . . . . . 52
8.2.2.4 Tariff and charging administration . . . . . 52
8.2.2.5 System control operation . . . . . 52
8.2.3 Maintenance functions . . . . . 53
8.2.3.1 Maintenance of subscriber's lines . . . . . 53
8.2.3.2 Maintenance of lines between exchanges . . . . . 53

Page

8.2.3.3 Switching network maintenance . . . . . 53
8.2.3.4 Control system maintenance . . . . . 53
8.2.4 Plant installation functions . . . . . 54
8.2.4.1 Switching system installation . . . . . 54
8.2.4.2 System control installation . . . . . 54
8.2.5 Testing functions . . . . . 55

SECTION 2 – *Specification of functions* (not yet defined; it will contain Recommendations Z.321 to Z.329)

SECTION 3 – *Users manual* (not yet defined; it will contain Recommendations Z.331 to Z.339)

SECTION 4 – *Glossary of terms* [2] (not yet defined; it will contain Recommendations Z.341 to Z.349

SECTION 5 – *Implementors guide* (not yet defined; it will contain Recommendations Z.351 to Z.359)

[2] See Annex 6 to the new Question 9/XI, Contribution COM XI-No. 1, 1977-1980.

SECTION 1

# GENERAL PRINCIPLES

**Recommendation Z.311**

## 1. INTRODUCTION

### 1.1 *Field of application*

The man-machine language (CCITT MML) can be used to facilitate operation and maintenance functions of SPC switching systems of different types. According to different national requirements CCITT MML can also be used to facilitate installation and testing of such systems. The contribution of the man-machine language toward testing will also include diagnosis of hardware faults and hardware/software design deficiencies.

A list of such functions is given in Recommendation Z.318.

In many cases, SPC systems will be supported by auxiliary processors (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 network configuration is shown in Figure 1/Z.311 which shows the case of three separate processors. Local and remote man-machine terminals may be used (remote including the international case). The configuration of processors in a system may vary but this has no effect on 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.

### 1.2 *Basis of MML*

The MML contains inputs (commands), outputs, control actions and procedures sufficient to ensure that all relevant functions for the operation, maintenance, installation, testing of SPC systems can be performed.

It is understood that some of these matters will arise by being initiated by man and some by being initiated by the machine.

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

*a)* The MML is easy to learn and to use, it is easy to input commands and to interpret outputs.

*b)* Both direct and continuation modes of operation are available as well as compact and expanded form of input/output.

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

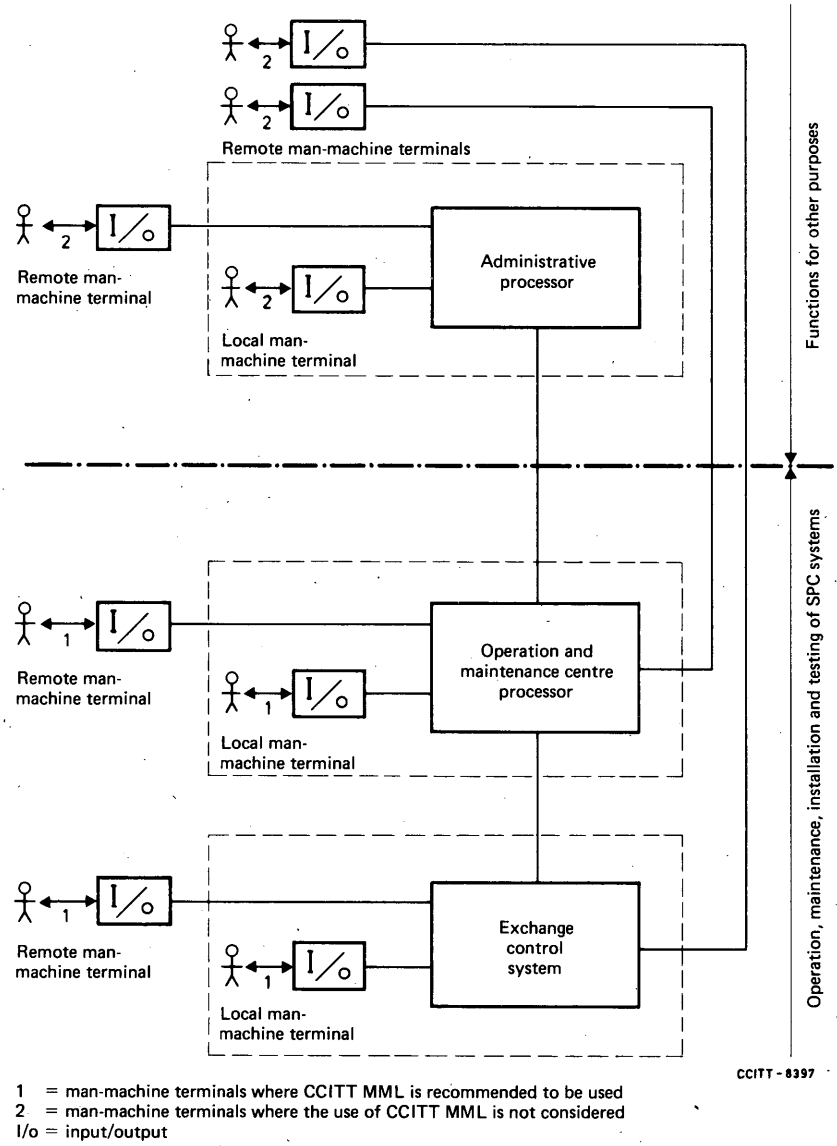



FIGURE 1/Z.311

According to different Administration requirements, the use of certain commands or procedures may be restricted by the implementation to certain staff, terminals, etc.

The MML should be implemented in such a way that errors in commands or control actions must not cause the system to stop, unduly alter the system configuration or take up undue resources.

## 1.3 *Input/output*

Input can be from any device that produces the code of the characters of the CCITT No. 5 alphabet.

A keyboard input device will normally be used; however, recorded input may be used (e.g., paper tape, magnetic tape, cassette, etc.).

Output can be to any device that accepts the code of the characters of the CCITT No. 5 alphabet (paper tape punchers, teletypewriters, line printers, visual display units, etc.).

Two basic formats F1 and F2 for printing are recommended (see Recommendation Z.312).

### 1.4 *Extensibility and sub-setting*

The MML has an open ended structure in such a way that any new function or requirement added 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 the selection is done for needs of certain sections of staff; application sub-sets, etc., in which selection is made for convenience of application, etc.

**Recommendation Z.312**

## 2. BASIC FORM LAYOUT

### 2.1 *General*

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

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 form, as this format can be accommodated on both the A4 and the 11-inch paper standard size and can be printed by standard teletypewriters.

Where a number of characters/line in excess of 72 is required, a second format is recommended. This accommodates 120 characters/line and would be used, for example, on typewriters and 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 has been taken into account.

### 2.2 *Recommended formats for presenting information in MML*

#### 2.2.1 *Format F1*

According to this format which is based on the A4 and 11-inch paper standard sizes, the maximum number of characters per line is 72. The number of lines per form 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 be displayed on most of the visual alpha-numerical displays 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.

#### 2.2.2 *Format F2*

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

**Recommendation Z.313**

## 3. THE CHARACTER SET

### 3.1 *General*

The character set to be used for the CCITT MML is a sub-set of the International Alphabet No. 5 which has been established jointly by the CCITT and the International Organization for Standardization (CCITT *Orange Book*, Volume VIII, Recommendation V.3).

To allow for possible implementations for the CCITT MML using national languages, the sub-set is taken from the basic code table given in Recommendation V.3. 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 transmission control characters 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.

### 3.2 *The sub-set selected for the MML*

The characters selected for use in the CCITT MML are given in Table 1/Z.313 below.

TABLE 1/Z.313 – 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 | | | SP | 0 | [a] | P | | |
| 0 | 0 | 0 | 1 | 1 | | | ! | 1 | A | Q | | |
| 0 | 0 | 1 | 0 | 2 | | | " | 2 | B | R | | |
| 0 | 0 | 1 | 1 | 3 | | | # | 3 | C | S | | |
| 0 | 1 | 0 | 0 | 4 | | | $ | 4 | D | T | | |
| 0 | 1 | 0 | 1 | 5 | | | % | 5 | E | U | | |
| 0 | 1 | 1 | 0 | 6 | | | & | 6 | F | V | | |
| 0 | 1 | 1 | 1 | 7 | | | ' | 7 | G | W | | |
| 1 | 0 | 0 | 0 | 8 | | CAN | ( | 8 | H | X | | |
| 1 | 0 | 0 | 1 | 9 | | | ) | 9 | I | Y | | |
| 1 | 0 | 1 | 0 | 10 | LF | | * | : | J | Z | | |
| 1 | 0 | 1 | 1 | 11 | | | + | ; | K | [a] | | |
| 1 | 1 | 0 | 0 | 12 | | | , | < | L | [a] | | |
| 1 | 1 | 0 | 1 | 13 | CR | | – | = | M | [a] | | |
| 1 | 1 | 1 | 0 | 14 | | | . | > | N | [a] | | |
| 1 | 1 | 1 | 1 | 15 | | | / | ? | O | _ | | |

CCITT - 7729

[a] These positions are reserved for national use

*General remarks.* – The characters on the open positions are considered as outside the MML. They are implementation dependent and may be used in accordance with the rules given in Recommendation V.3. 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. 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.

## 3.3 *Classification of characters*

The characters can be classified according to their use in the MML as follows:

### 3.3.1 *letters*

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

### 3.3.2 *digits*

0 1 2 3 4 5 6 7 8 9

### 3.3.3 *separators*

| | |
|---|---|
| , | (comma) |
| . | (full stop) |
| : | (colon) |
| = | (equal sign) |
| - | (hyphen) (see also 3.3.6) |
| ' | (apostrophe) |
| & | (ampersand) |
| / | (solidus) (see also 3.3.6) |
| > | (greater than) |
| ( | (left parenthesis) |
| ) | (right parenthesis) |

### 3.3.4 *indicators*

| | |
|---|---|
| < | (less than) |
| * | (asterisk) (see also 3.3.6) |
| ? | (question mark) |
| ; | (semi-colon) |

3.3.5 *control characters*

3.3.5.1 *format effectors*

LF (line feed)
CR (carriage return)
SP (space)

3.3.5.2 *other control characters*

Not yet defined.

3.3.6 *operators in arithmetical expressions*

\+ (plus sign) (see also 3.3.7)
\- (hyphen, minus sign) (see also 3.3.3)
* (asterisk) (see also 3.3.4)
/ (solidus) (see also 3.3.3)

3.3.7 *characters which may be used in symbolic names*

# (number sign)
\+ (plus sign) (see also 3.3.6)
% (percent sign)

3.3.8 *undefined characters*

For the present time the use of the following characters is not defined and therefore not classified.

! (exclamation mark)
_ (underline)
$ (dollar sign)
CAN (cancel)
" (quotation mark)

**Recommendation Z.314**

## 4. THE META-LANGUAGE FOR DESCRIBING THE SYNTAX AND PROCEDURES

### 4.1 *Introduction*

Syntax diagrams are a method of defining language syntax. [1)]

A syntax diagram consists of terminal and non-terminal symbol boxes connected by directed 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 defines a particular non-terminal symbol. In the MML Recommendations syntax diagrams are used for specifying the syntax of the MML input and MML output. For the purpose of describing the man-machine dialogue procedures the method has been extended. A path through a syntax diagram defines an MML input, MML output or a man-machine dialogue structure.

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

---

[1)] *PASCAL, User manual and report* ; K. Jensen, N. Wirth, Springer Verlag.

## 4.2 *Basic elements*

4.2.1 Terminal symbols are those characters or strings of characters which actually appear in the input or output.

4.2.2 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. So it is an abbreviated symbol for a more complex construct (consisting of a set of terminal or non-terminal symbols) used in several places.

4.2.3 Annotation symbols (see 4.3.7 below) are used to insert comments. For example, they may be used to put a limit on the number of times a symbol is repeated.

## 4.3 *Rules*

4.3.1 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.

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

4.3.3 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.

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

4.3.5 Terminal symbols are surrounded by boxes with rounded corners [see Figure 1(a)/Z.314]. The width of the box is proportional to the number of characters contained. For short terminal symbols, the box may become a circle [see Figure 1(b)/Z.314].

When describing procedures in the MML, terminal symbols generated by machine are surrounded by double-framed boxes with rounded corners [see Figure 1(c)/Z.314] or circles [see Figure 1(d)/Z.314].

4.3.6 Non-terminal symbols are surrounded by rectangular boxes, [see Figure 1(e)/Z.314]. The name of the non-terminal symbol must be written in lower case characters. The non-terminal symbol used to name a particular syntax diagram must appear underlined at the upper left corner of the diagram.

When describing procedures in the MML, non-terminal symbols generated by the machine are surrounded by double-framed rectangular boxes [see Figure 1(f)/Z.314].

4.3.7 An annotation is denoted by the following symbol:

– – – – [n

where n is a number referring to a comment. The text of the comment must be located at the foot of the diagram (this splitting of reference and text should help to reduce translation problems).

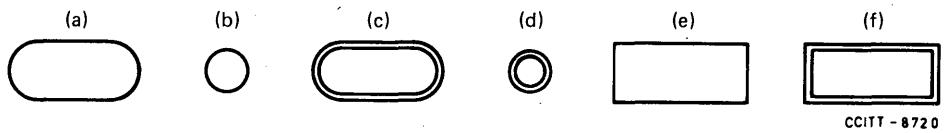



FIGURE 1/Z.314 – Terminal and non-terminal symbols to be used for the CCITT man-machine language

**Recommendation Z.315**

## 5. INPUT (COMMAND) LANGUAGE SYNTAX SPECIFICATION

### 5.1 *General*

The following describes the elements of the input language only. Procedure aspects are not taken into account.

### 5.2 *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. The parameter part consists of one or more *blocks of parameters* and is separated from the command code by : (colon). The command is always terminated by the indicator ; (semi-colon).

### 5.3 *Command code*

It is recommended that all the command codes in a certain application consist of the same number of characters. The command code should be an *identifier* in the form of a mnemonic abbreviation (preferably of letters) of the appropriate function. [2)]

### 5.4 *Block of parameters*

A block of parameters shall contain the 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 the separator , (comma). All parameters in any one block shall be of the same kind i.e. either *parameter name defined parameters* or *position defined parameters.*

### 5.5 *Parameters*

A parameter identifies and contains a piece of information and may be either parameter name defined or position defined. Non-relevant parameters may be omitted.

5.5.1 *Parameter name defined parameters* consist of a *parameter name* usually followed by a *parameter value* from which it is separated by = (equal sign). They may be given in any arbitary order within a parameter block.

5.5.2 *Position defined parameters* consist of a parameter value which may be preceded by a parameter name from which it is separated by = (equal sign). They must be given in a predetermined order. Where non-relevant parameters are omitted, their position in the parameter block must be indicated.

### 5.6 *Parameter name*

A parameter name shall unambiguously indicate the kind and structure of the subsequent *parameter value* and thereby define the parameter value and how it shall be interpreted. It is an identifier.

---

[2)] The use of a string of identifiers is under study, see Annex 4 to new Question 9/XI, Contribution COM XI-No. 1, 1977-1980.

### 5.7 *Parameter value*

A parameter value consists of one or more *parameter arguments* each of which shall contain the information required to specify the appropriate object or value.

### 5.8 *Parameter argument*

A parameter argument consists of one or more *information units* separated by - (hyphen) or &- (ampersand hyphen) or &&- (ampersand ampersand hyphen). See also 5.11 below, Information grouping.

### 5.9 *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 expressed numerically in the form of a *decimal numeral* or *non-decimal numeral* or non-numerically in the form of an *identifier* or a *symbolic name.*

### 5.10 *Basic elements*

#### 5.10.1 *Identifier*

An identifier is defined as a string of one or more characters which begins with a letter and otherwise only contains digits and/or letters. For example U, UPDATE, UPD8.

#### 5.10.2 *Symbolic name*

A symbolic name is defined as a string of one or more characters containing letters and/or digits and/or the graphic characters +, #, %. For example 24H, #6, +4687191818.

#### 5.10.3 *Decimal numeral*

A decimal numeral is defined as a character combination which only contains digits, e.g. 123456. An optional D' may precede a decimal numeral in line with 5.10.4 below, e.g. D'123456.

#### 5.10.4 *Non-decimal numeral*

A non-decimal numeral is defined as a character combination preceded by a special character combination indicating the type of numeral.

H' is used to indicate hexa-decimal numerals, the following characters thereby being digits or any of the letters A to F.

O' is used to indicate octal numerals, the following characters thereby being any of the digits 0-7.

B' is used to indicate binary numerals, the following characters thereby being either of the digits 0 or 1.

The special character combination can be omitted if confusion does not arise.

#### 5.10.5 *Separator and separator string*

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

### 5.10.6 *Indicator*

An indicator is a character given by man or machine used to indicate a state or a request, e.g. ; indicates the end of the command.

### 5.10.7 *Control character*

This class of characters comprises format effectors but is not yet fully defined.

#### 5.10.7.1 *Format effector*

A format effector is a character used to format an input or output in a suitable manner. In input it has no significance for the command but terminates a syntactic basic element in outputs. The characters space (SP), line feed (LF) and carriage return (CR) are the format effectors.

### 5.10.8 *Undefined characters*

Those characters not used as letters, digits, separators, indicators or control characters or used in symbolic names are reserved for later use. They may be used in outputs to improve the legibility.

## 5.11 *Information grouping*

### 5.11.1 *General*

Information grouping is used to enter information for the execution of combined functions or to improve the speed and ease of the operator's activities. It is performed by giving sets of information of the same type within the same command. When combined functions are used there is a semantic relationship between the different sets of information of the same type.

### 5.11.2 *Blocks of parameters*

If several groups of parameters should be entered in one command, it is possible to form blocks of parameters separated by : (colon). Each block represents a group of parameters belonging together.

### 5.11.3 *Parameter arguments*

If several parameter arguments should be entered within one parameter in one command it is possible to indicate the parameter arguments within the same parameter value, separated by & (ampersand). If the parameter arguments have consecutive values (incrementing by 1) and all consist of only one information unit it is possible to indicate a suite of parameter arguments by writing the first and last parameter arguments separated by && (ampersand ampersand).

### 5.11.4 *Information units*

If several parameter arguments, each consisting of more than one information unit, should be entered within one parameter in one command it is possible to indicate them in the same parameter argument if the arguments only differ in the last information unit. All the parameter arguments except the first are represented only by the last information unit and they are separated by &- (ampersand hyphen). If the information units have consecutive values (incrementing by 1) it is possible to indicate a suite of information units by writing the first and the last information units, separated by &&- (ampersand ampersand hyphen).

## 5.12 *Summary of use of characters*

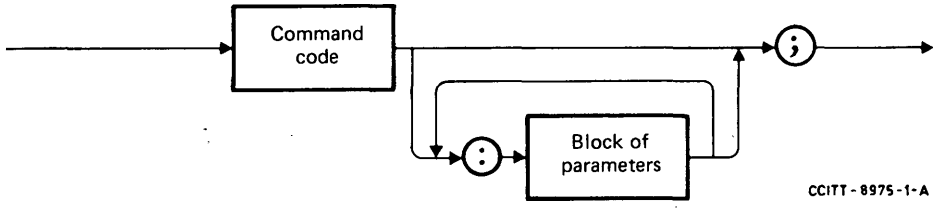
The use of characters in the character set, except letters and digits, is described below. The CCITT International Alphabet No. 5 code is indicated in parentheses. For the sake of completeness, the characters to be used outside the input language are also included.

| | | |
|---|---|---|
| LF | (0/10) | A format effector without syntactical meaning in input. |
| CR | (0/13) | A format effector without syntactical meaning in input. |
| CAN | (1/8) | Use not yet defined. |
| SP | (2/0) | A format effector without syntactical meaning in input. |
| ! | (2/1) | Use not yet defined. |
| " | (2/2) | Use not yet defined. |
| # | (2/3) | A character which may be used in symbolic names. |
| $ | (2/4) | Use not yet defined. |
| % | (2/5) | A character which may be used in symbolic names. |
| & | (2/6) | A separator for information grouping. |
| ' | (2/7) | 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. |
| ( | (2/8) | Reserved for delimiting arithmetical expressions. |
| ) | (2/9) | Reserved for delimiting arithmetical expressions. |
| * | (2/10) | An indicator used in procedures. (Continuation character in input language). Also reserved for use in arithmetical expressions. |
| + | (2/11) | A character which may be used in symbolic names, also reserved for use in arithmetical expressions. |
| , | (2/12) | A separator used to separate parameters (if more than one) within a block of parameters. |
| - | (2/13) | A separator used to separate information units (if more than one) within a parameter argument. Also reserved for use in arithmetical expressions. |
| . | (2/14) | A separator reserved for sub-dividing a number into an integer part and a fraction part. |
| / | (2/15) | Reserved for use as a separator. Also reserved for use in arithmetical expressions. |
| : | (3/10) | A separator used to separate blocks of parameters from each other and from the command code. |
| ; | (3/11) | An indicator used to terminate a command (execution character). |
| < | (3/12) | An indicator used as a ready indicator for the machine to tell the man that it is ready to receive information. |
| = | (3/13) | A separator used to separate the parameter name and the parameter value of a parameter. |
| > | (3/14) | A separator to separate the destination field and the command code. |
| ? | (3/15) | An indicator reserved for procedures. |
| _ | (5/15) | Use not yet defined. |
| && | (2/6, (2/6) | Separator string used for information grouping. |
| &- | (2/6, 2/13) | Separator string used for information grouping. |
| &&- | (2/6, 2/6, 2/13) | Separator string used for information grouping. |

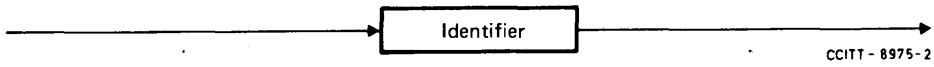
## 5.13 *Formal definition of the input (command) language syntax in diagrams*

### 5.13.1 *Command*

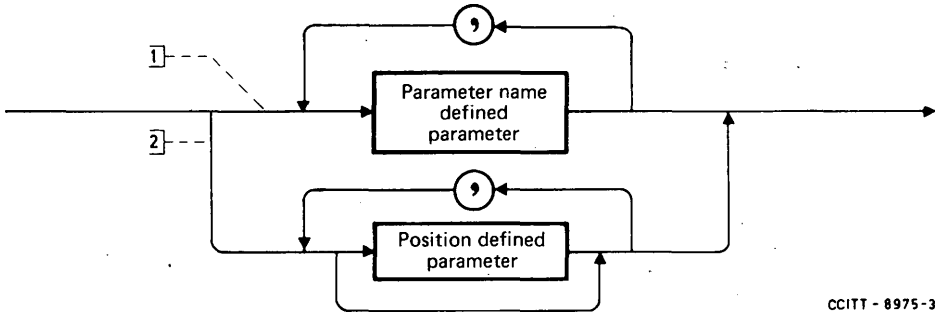
5.13.1.1 *command format* (for the procedural aspect see Recommendation Z.317).




5.13.1.2 *command code*

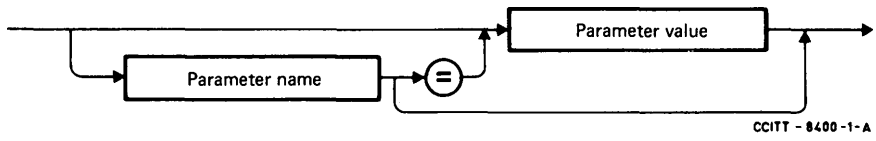



5.13.1.3 *block of parameters*

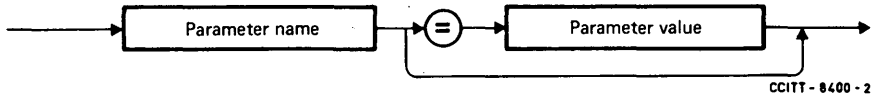



1. Upper main branch valid only for block of parameter name defined parameters.
2. Lower main branch valid only for block of position defined parameters.

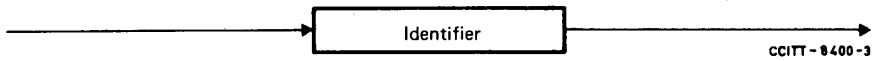
5.13.1.4 *position defined parameter*



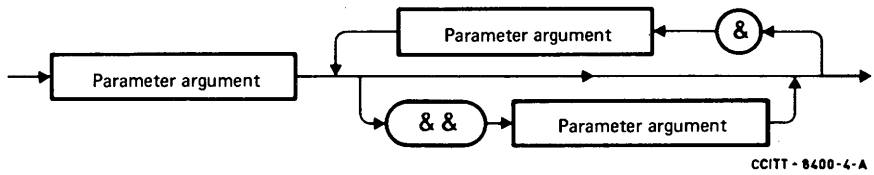

5.13.1.5 *parameter name defined parameter*



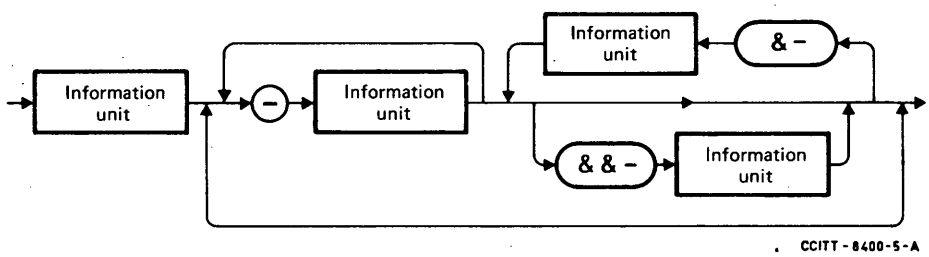
5.13.1.6 *parameter name*



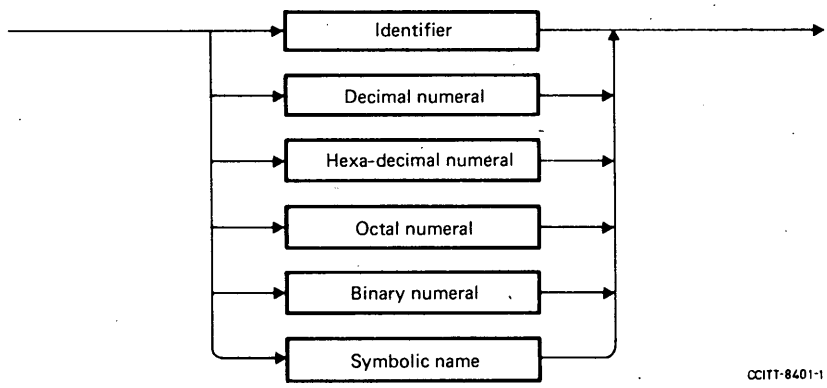

5.13.1.7 *parameter value*




5.13.1.8 *parameter argument*

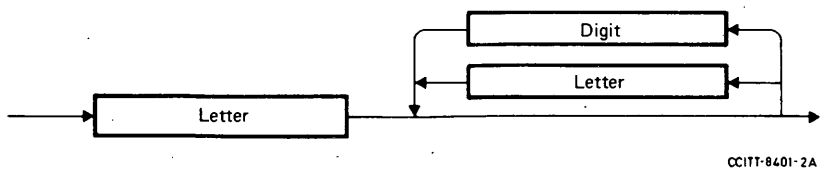



5.13.1.9 *information unit*

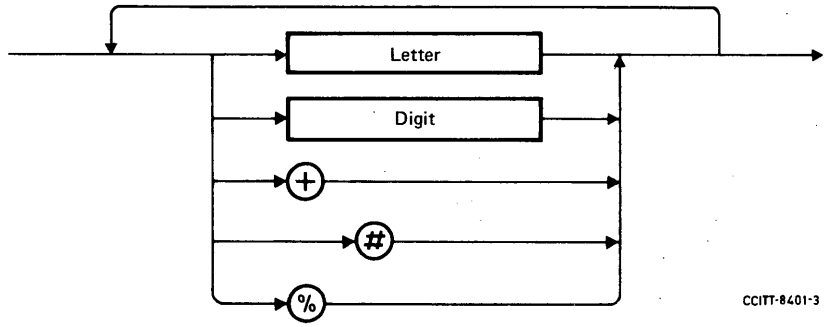



5.13.2 *Syntactic basic elements*

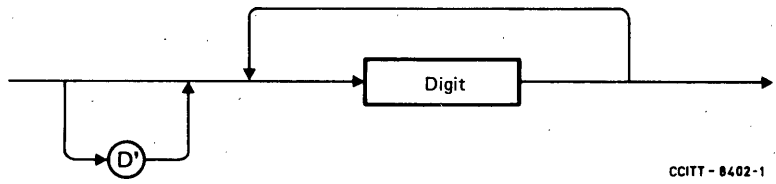
5.13.2.1 *identifier*



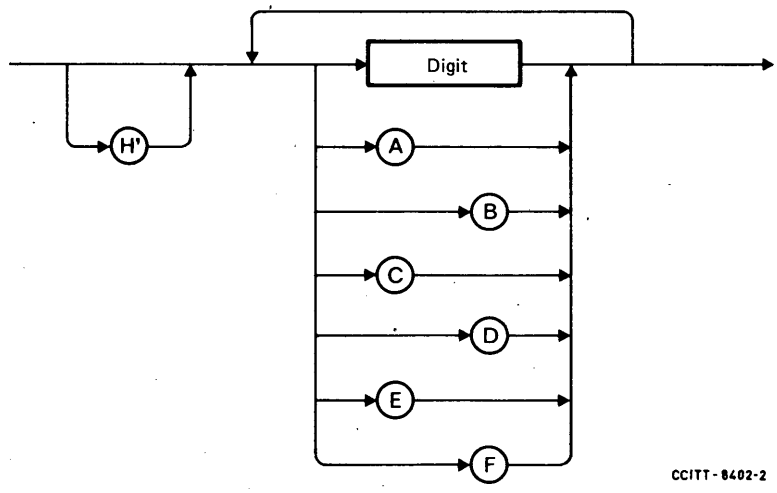
5.13.2.2 *symbolic name*



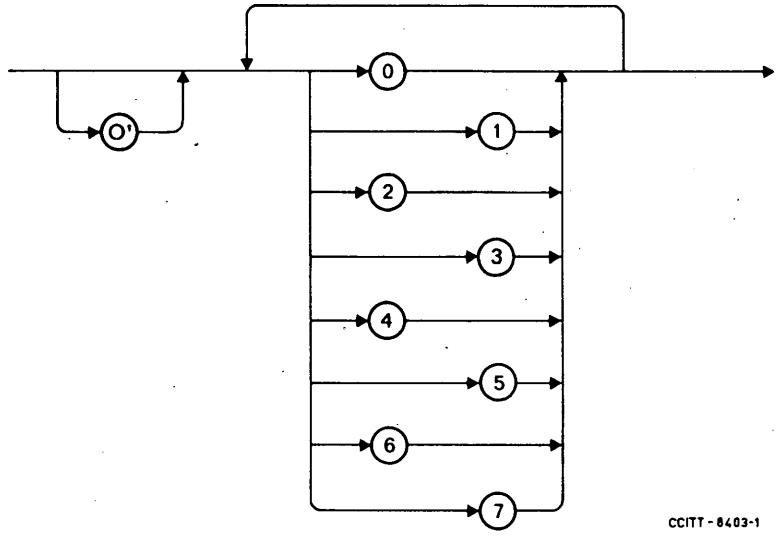

5.13.2.3 *decimal numeral*



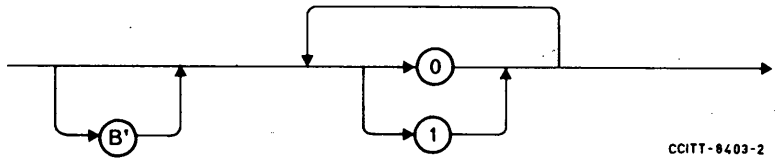

5.13.2.4 *hexa-decimal numeral*



5.13.2.5 *octal numeral*

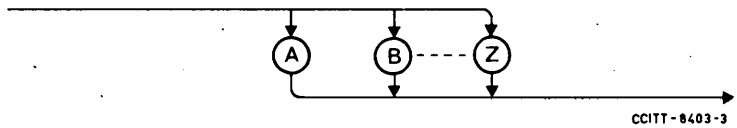



5.13.2.6 *binary numeral*

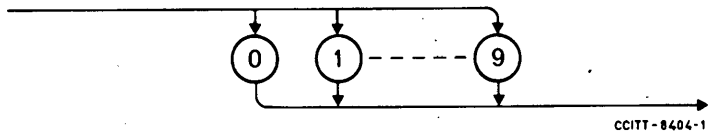



5.13.3 *Alphabet*

5.13.3.1 *letter*




5.13.3.2 *digit*




**Recommendation Z.316**

## 6. OUTPUT LANGUAGE, SYNTAX AND GENERAL SEMANTICS

(This Recommendation is still under study. See the table of contents for a provisional summary of this Recommendation.)

**Recommendation Z.317**

# 7. MAN-MACHINE DIALOGUE

## 7.1 *Description of dialogue procedures*

### 7.1.1 *General*

In the man-machine information interchange one can define two types of sentences, i.e. dialogue procedure and automatic output.

Within the dialogue procedure only two mode operating sequences have yet been defined, namely direct mode operating sequence and continuation mode operating sequence. The direct mode is the basic mode and is a sub-set of the continuation mode. Additional modes can be added. Some error and editing sequences are also to be defined later.

The direct mode operating sequence and the continuation mode operating sequence are defined in 7.1.3 and 7.1.4 respectively, below.

The operating sequences can be preceded by the procedure prologue. The prologue contains the various preparations an operator must perform before a command can be written. It may include also a written header from the machine and an indication from the machine to the operator that a command can be written. The procedure prologue is defined in 7.1.2 below.

### 7.1.2 *Procedure prologue*

The procedure prologue may consist of six parts:

- the request, which is an action from the operator to activate the man-machine terminal and the command system;
- the password given from the operator for identification and authorization. The password can be optional for a system or within a system for a certain operator or a certain terminal;
- the header, which is given from the machine and contains the exchange identification, information relating to time and date, etc. The header can constitute the top line on a display or on a paper form. The header can be optional for a system or within a system for a certain terminal;
- the ready indicator (<) which is always given from the machine as an indication to the operator to give the command;
- the destination identifier, which indicates the physical area where the command is to be processed, may then be given by the operator to the machine, followed by the separator (>) to terminate the destination identifier and to separate it from the command code to follow. The destination can also be defined by a parameter in the command;
- a header and ready indicator which may be given by the machine to indicate that a selected destination is available and ready.

The prologue is intended to be passed only once for a set of commands.

The request, ready indicator, password, header, and destination identifier are defined in 7.1.2.1, 7.1.2.2, 7.1.2.3, 7.1.2.4 and 7.1.2.5 respectively, below.

#### 7.1.2.1 *request*

The request is a manual action from the operator to activate the man-machine terminal and the system. The composition of the request is highly dependent on the type of man-machine terminal and system used for a certain SPC-system.

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

7.1.2.2 *ready indicator*

The ready indicator tells the operator that the direction of the dialogue has changed so that the machine is waiting for a character to be typed at the man-machine terminal. The ready indicator is defined as the character < (less than).

7.1.2.3 *password*

The password is used for identification and authorization of an operator. The password can give an operator access to different groups of commands. The groups can have different security classifications or can have different functional tasks (e.g. traffic-measuring functions).

The password consists of a symbolic name.

7.1.2.4 *header*

The header is given from the machine to the man in the procedure prologue.

The main purpose of the header is to mark the command-form for later identification and information. The header can also be used for special purposes for an operation and maintenance centre. Recommended contents are information related to exchange-name, date and time. More information can be added to the header.

As the header is an output, no syntax definition is given here but in 6. (Recommendation Z.316). [3)]

7.1.2.5 *destination identifier*

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

7.1.3 *Direct mode*

In the direct mode, the complete command is given without questions from the machine. Machine response, including header and ready indicator, can be given in the procedure prologue (see 7.1.2 above).

The command can be given with none, one or several blocks of parameters. When the operator has finished the command by giving the execution character (;) the command is interpreted by the machine.

If the command is built up by several blocks of parameters the interpretation can be done in parts as if the command consisted of several commands (with a common command code and different parameter blocks). The interpretation of a multi-block command can also be non-divisable, which permits semantic relationships between the blocks.

When the command is interpreted and executed a response output is given to the operator, followed by an optional end statement.

7.1.3.1 *response output*

See 6. (Recommendation Z.316) [3)].

7.1.3.2 *end statement*

See 6. (Recommendation Z.316) [3)].

7.1.4 *Continuation mode*

In the continuation mode the operator may give several blocks of parameters to the system as in the direct mode, but in an alternating sequence of one or more parameter blocks and a correspondig number of executions. Execution of a new sequence of one or more blocks of parameters for the same command is initiated by giving the continuation character (*) instead of the execution character (;). The continuation character functions as a signal to the machine to return a ready indicator (<) after the response outputs associated with

[3)] Under study, see new Question 9/XI, Volume VI.1, *Orange Book*.

present blocks of parameters have been generated. The ready indicator in turn functions as a signal to the operator to proceed with the next block or blocks. If the operator gives the execution character (;) instead of the continuation character (*) a ready indicator (<) will not be given following the response outputs and an end statement may then be given by the machine.

After the machine gives the ready indicator (<), the operator may leave the continuation mode by giving the end code followed by the execution character (;).

As the interpretation and execution of input information is done after each of the given continuation indicators, semantic relationships between sets of information, separated by the continuation indicators, are not allowed.

7.1.5 *Parameter introduction sequence*

The parameter introduction sequence consists of a block of parameters. Block of parameters is defined in 5.13.1.3 above.

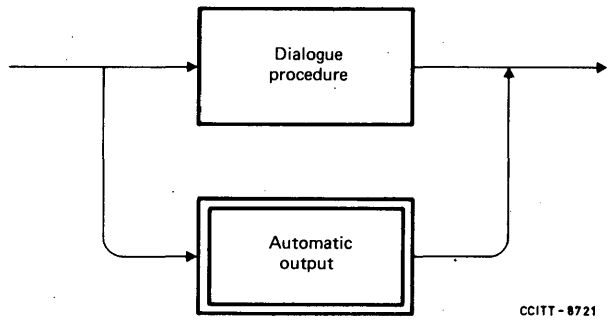
7.1.6 *End code*

The end code consists of a symbolic name. Symbolic name is defined in 5.13.2.2 above.

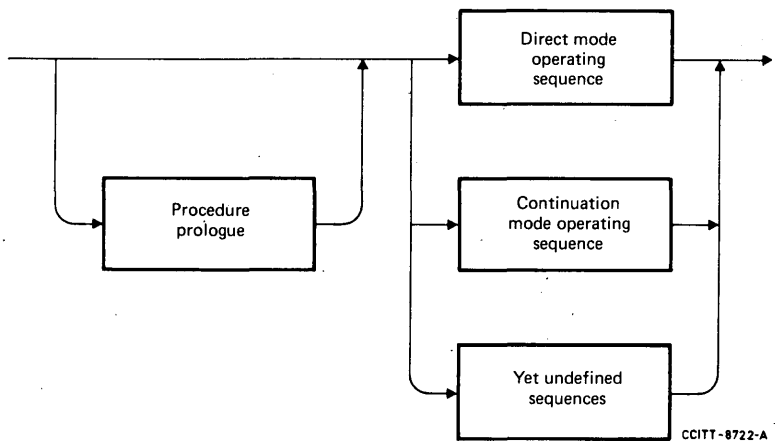
7.2 *Formal specification of the dialogue procedure syntax in diagrams*

7.2.1 *Sentence and dialogue procedure*

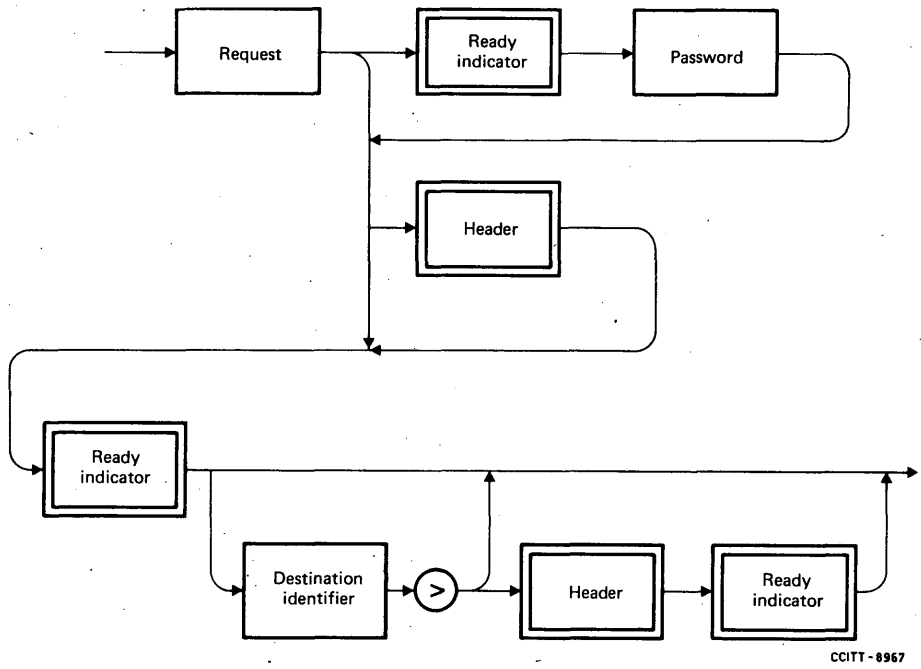
7.2.1.1 *sentence*



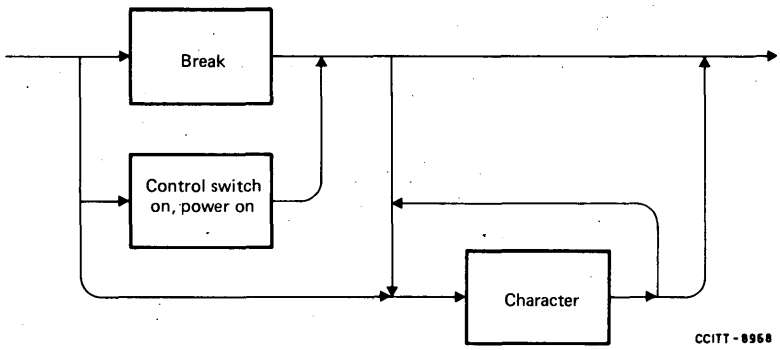

7.2.1.2 *dialogue procedure*



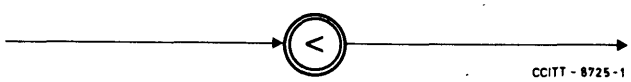
7.2.2 *Procedure prologue*



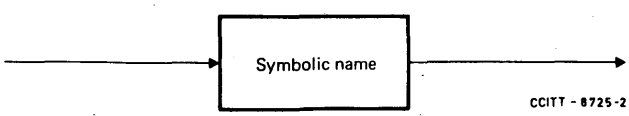

7.2.2.1 *request*




7.2.2.2 *ready indicator*



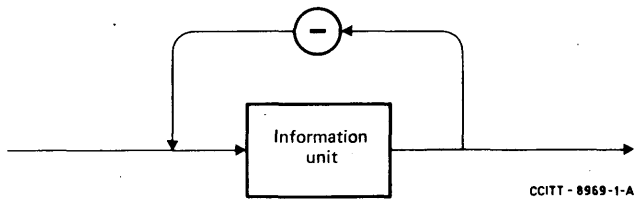

7.2.2.3 *password*



7.2.2.4 *header*

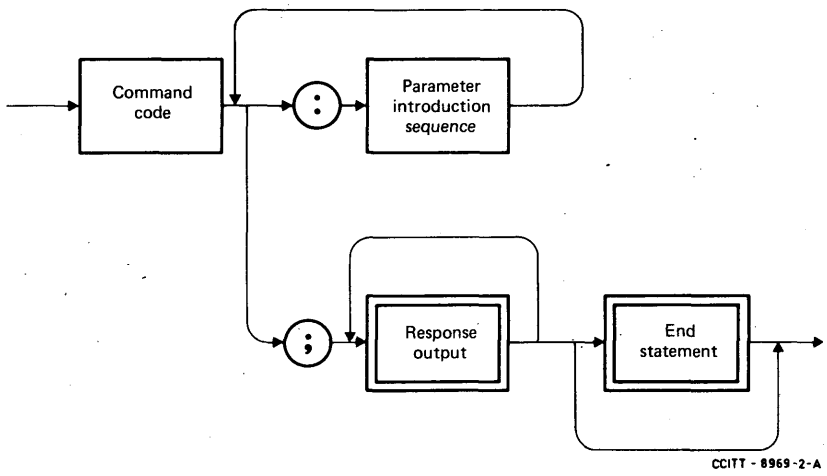
(See 6., Recommendation Z.316.)

7.2.2.5 *destination identifier*




7.2.3 *Direct mode operating sequence*

Although the direct mode operating sequence is the basic operating sequence of the MML its syntax diagram forms part of the syntax diagram of the continuation mode operating sequence shown in 7.2.4 and is shown here separately only for clarity.



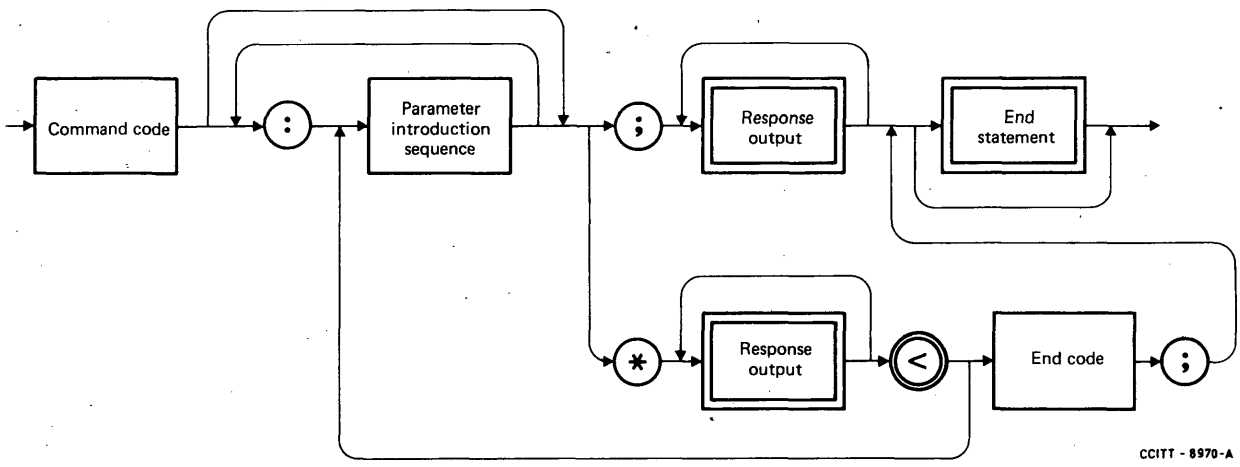

7.2.3.1 *response output*

(See 6. of Recommendation Z.316.)

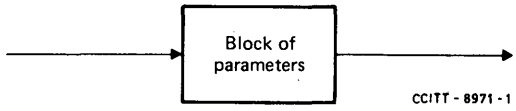
7.2.3.2 *end statement*

(See 6. of Recommendation Z.316.)

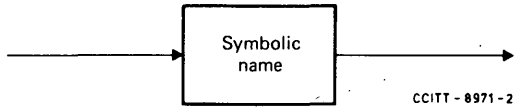
7.2.4 *Continuation mode operating sequence*



7.2.5 *Parameter introduction sequence*




7.2.6 *End code*




**Recommendation Z.318**

## 8. LIST OF FUNCTIONS

8.1 *Introduction*

A preliminary list of functions which are expected to be controlled by means of the MML is given in this Recommendation [4].

The functions are listed under four main function areas:

operation, maintenance, installation and testing.

Because of potentially different national needs some functions appear under more than one heading.

8.2 *Functions*

8.2.1 *General functions*

In addition to those functions indicated below which imply changing or setting data there are also the corresponding interrogation functions.

8.2.2 *Operational functions*

8.2.2.1 *Subscriber administration*

- taking subscribers' lines into/out of service [5];
- allocate change and remove classes of subscriber service;
- changing of a subscriber's number;
- blocking and unblocking a subscriber's line;
- interrogation of a subscriber's classes of service;
- interrogation of blocked subscriber lines;
- reading of subscriber's charging information;

---

[4] This list was agreed jointly in studies of Study Groups XI and XIII during the 1973-1976 study period. The list is not to be regarded as complete. The purpose of preparing a list of functions was to ensure that the MML recommended by the CCITT would be adequate to cover all the necessary functions.

[5] Subscribers' lines includes lines on PABX.

- retrieval of charging information;
- malicious call tracing;
- putting a subscriber on to *subscriber charging observation*, etc.

8.2.2.2 *Routing administration*

8.2.2.2.1 *Changing data relating to a group of circuits*

- changing of signalling system;
- inserting a new group of circuits;
- changing 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.

8.2.2.2.2 *Changing routing and analysis data*

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

8.2.2.3 *Traffic administration*

- traffic recording according to CCITT method No. 1 (Recommendation E.261);
- traffic recording according to CCITT method No. 2 (Recommendation E.261);
- changing of traffic load control criteria;
- analysis of destinations of incoming traffic;
- measurements of subscribers' behaviour;
- means for interrogating the measurement parameters of different measurement groups;
- means for inserting a given route, circuit, etc., into, or removing it from, a series of measurements;
- outputting of data relating to grade of service.

See network management actions in 5.2 of Recommendation E.410 (former Recommendation Q.55).

See measuring and recording of traffic, Recommendations E.500 and E.501.

See Recommendations E.420, E.421 and E.422 to E.425 (formerly Recommendations Q.60, Q.60 *bis*, and Q.61 to Q.63).

8.2.2.4 *Tariff and charging administration*

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

8.2.2.5 *System control operation*

- setting and reading of calendar;
- loading of overlay programmes;
- changing authority for an input device;
- changing output device for a certain output;
- changing the code to be transmitted by a tape reader unit or to a tape punch as, for example, in ATME;
- defining a file being set up on a magnetic tape equipment;

- changing working mode for a certain programme;
- changing of system configuration;
- starting tape reader or magnetic tape equipment;
- loading a new programme package.

8.2.3 *Maintenance functions*

8.2.3.1 *Maintenance of subscribers' lines*

- test of one subscriber's line and associated equipment;
- test of a group of subscribers' lines and associated equipment;
- measurement of one subscriber's line and associated equipement;
- measurement of a group of subscribers' lines and associated equipment;
- blocking or unblocking a subscriber's line for maintenance purposes;
- observation or supervision of subscriber's lines and equipment;

8.2.3.2 *Maintenance of lines between exchanges*

- as in 8.2.3.1 above (Deleting subscriber's or subscribers').

8.2.3.3 *Switching network maintenance*

- making test calls;
- call trace;
- holding faulty connections;
- testing and measuring peripheral equipment (relay sets, signalling receivers and senders, etc.);
- testing and measuring switch units;
- reducing service for low priority subscribers;
- setting up a connection via a specific path through the network;
- supervision and measurement of quality of service of switching network;
- fault localisation in the speech path network;
- access for traffic observation for maintenance purposes;
- alarm reporting;
- switch unit status recording.

8.2.3.4 *Control system maintenance*

Hardware:

- system status reporting;
- alarm reporting and fault localisation;
- functional testing after repair;
- initiating periodic testing operations;
- changing system configuration for maintenance purposes.

Software:

- checking consistency of data (e.g. checksumming);
- initiating restart;
- setting traps for programme fault tracing;
- changing memory contents;
- memory dumping for maintenance purposes.

Load control:

- omitting non-essential functions in overload situations;
- changing the criteria for the recognition of degradation of service;
- reducing service for low-priority subscribers.

Network management:

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

8.2.4 *Plant installation functions*

The configuration of an exchange is described by several parameters of limits such as:

- maximum number of subscribers;
- maximum number of trunks;
- wiring of equipments;
- list of existing services or facilities;
- list of existing signalling systems;
- maximum size of data tables, etc.

Any change in software or hardware under these limits is considered as an operational function, for example adding equipments or features.

To initiate or change some of these limits (first installation or extension) specific functions have to be executed (the same kind of functions in both cases are called plant installation functions).

8.2.4.1 *Switching system installation*

Testing and data generation functions for:

- new network-blocks installation;
- new trunks installation;
- new signalling-equipment installation;
- new test-equipment installation;
- new blocks of subscriber-circuits installation;
- new interface-equipment installation.

8.2.4.2 *System control installation*

Installation of new software:

- new operational packages;
- new test programmes;
- new statistics programmes;
- new overlay programmes;
- new signalling systems;
- new services, facilities and tariff.

Extension of system control:

- control unit;
- memory;
- input/output devices.

Table generation:

- subscribers;
- routing and tariffs.

8.2.5 *Testing functions*

The memory dump expressing:

- the begin and end of the dump in absolute addressing;
- the begin and end of the dump in symbolic addressing;
- the wish to have the dump in a binary (hexa-decimal/decimal) form;
- changing programmes and data.

Adding patches:

- patches in a case of programme changing;
- reloading patches in a reload phase;
- starting and stopping a programme trace;
- starting and stopping a data trace.

Under simulated conditions:

test programmes;

- test hardware equipment;

start test calls;

- stop at a particular address;
- loading new programme packages;
- restarting the machine (from certain conditions).

ISBN 92-61-C0421-0

PRINTED IN SWITZERLAND