INTERNATIONAL TELECOMMUNICATION UNION

# ITU-T Z.315

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

## MAN-MACHINE LANGUAGE

---

# INPUT (COMMAND) LANGUAGE SYNTAX SPECIFICATION

## ITU-T Recommendation Z.315

(Extract from the *Blue Book*)

---

**NOTES**

1 ITU-T Recommendation Z.315 was published in Fascicle X.7 of the *Blue Book*. This file is an extract from the *Blue Book*. While the presentation and layout of the text might be slightly different from the *Blue Book* version, the contents of the file are identical to the *Blue Book* version and copyright conditions remain unchanged (see below).

2 In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

**Recommendation Z.315**

**INPUT (COMMAND) LANGUAGE SYNTAX SPECIFICATION**

## 1 General

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

## 2 Command structure

### 2.1 *Command*

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

### 2.2 *Command code*

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

### 2.3 *Block of parameters*

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

### 2.4 *Parameters*

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

#### 2.4.1 *Position defined parameter*

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

#### 2.4.2 *Parameter name defined parameter*

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

2.5 *Parameter name*

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

2.5.1 *Simple parameter name*

A simple parameter name consists of one identifier.

2.5.2 *Compound parameter name*

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

2.5.2.1 *Index number*

An index number is one or more digits.

2.6 *Parameter value*

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

2.7 *Parameter argument*

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

2.7.1 *Simple parameter argument*

A simple parameter argument consists of one information unit.

2.7.2 *Compound parameter argument*

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

2.8 *Information unit*

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

2.9 *Information grouping*

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

2.9.1 *Grouping of blocks of parameters*

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

2.9.2 *Grouping of parameter arguments*

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

2.9.2.1 *Grouping of simple parameter arguments*

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

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.

2.10 *Data base queries*

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

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

*Examples:*

query-dbx: projection = field a,

selection = (field c = 0);

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

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

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

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

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

---

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.

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

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

*Warning*

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

*Restriction*

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

## 3 Corrections and delete command

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

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

### 3.1 *Delete last character*

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

### 3.2 *Delete to last system output*

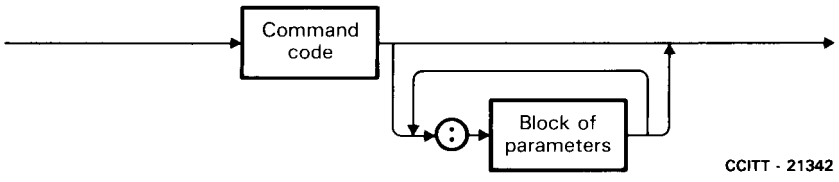
This facility deletes all input characters after the last system output, being either the ready indication or prompting output (see Recommendation Z.317).

### 3.3 *Delete command*

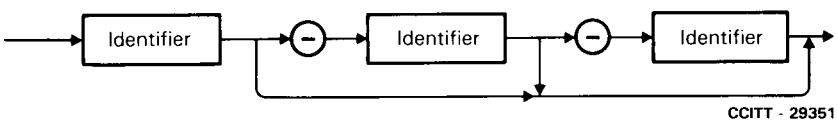
The delete command request is conveyed by the CAN character (cancel). The use of this character causes the system to respond with an acknowledgement that presents input after the last command executed is cancelled. The system should respond with a new ready indication to indicate that it is waiting for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

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

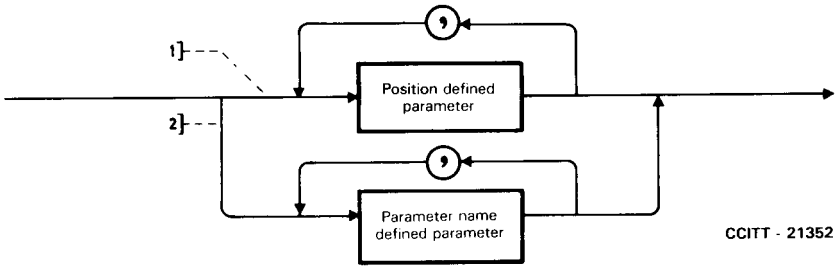
### 4.1 *Command*




### 4.2 *Command code*



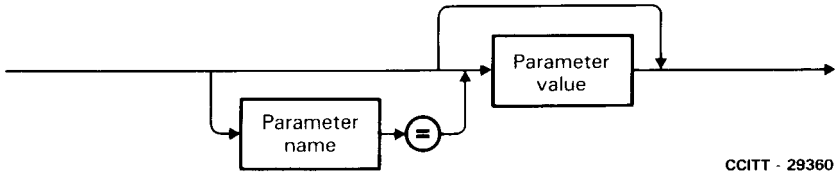
4.3 *Block of parameters*



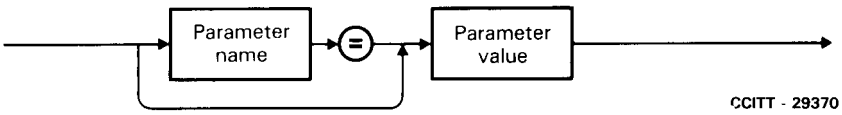

1) Upper main branch valid only for block of position defined parameters.
2) Lower main branch valid only for block of parameter name defined parameters.

4.4 *Parameters*

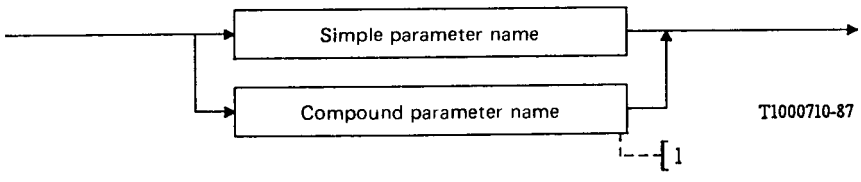
4.4.1 *Position defined parameter*




4.4.2 *Parameter name defined parameter*

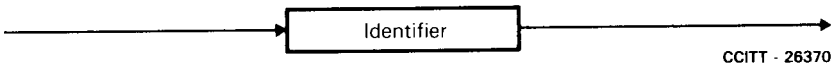



4.5 *Parameter name*

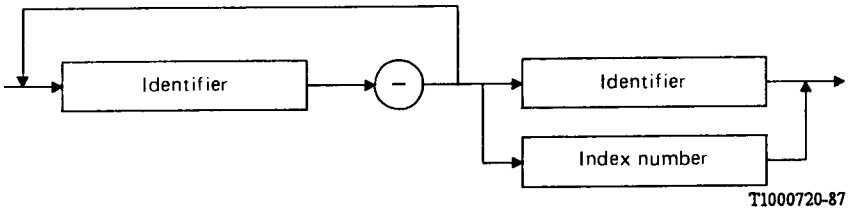



1) This is optional.

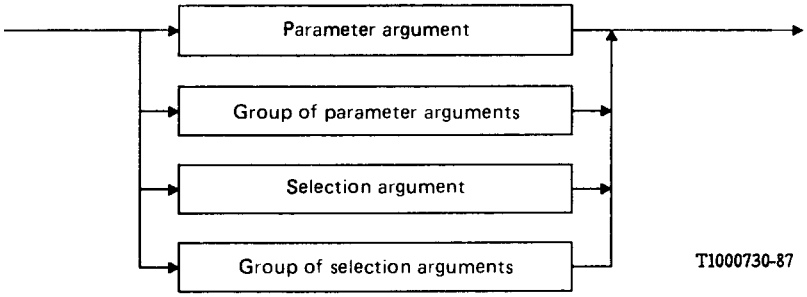
4.5.1 *Simple parameter name*



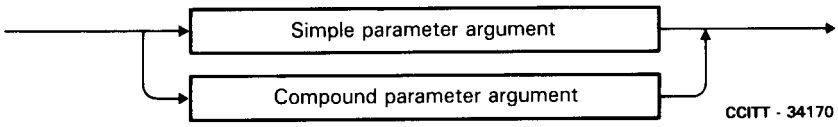
4.5.2 *Compound parameter name*



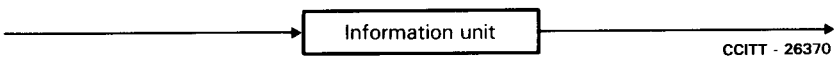

4.6 *Parameter value*



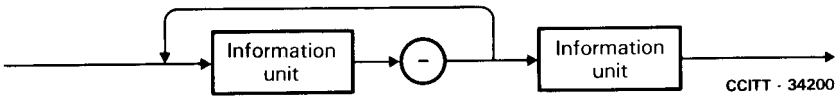

4.7 *Parameter argument*



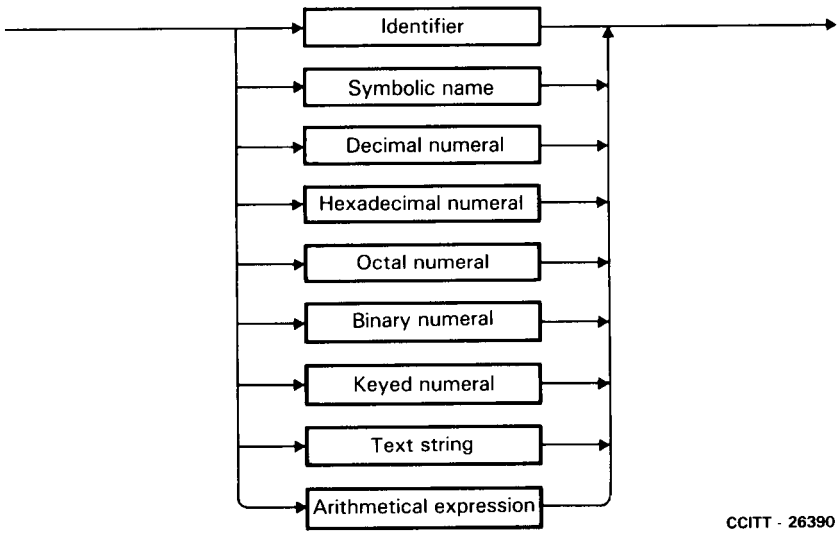

4.7.1 *Simple parameter argument*



4.7.2 *Compound parameter argument*




4.8 *Information unit*

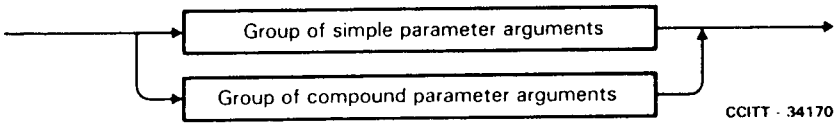



4.9 *Information grouping*

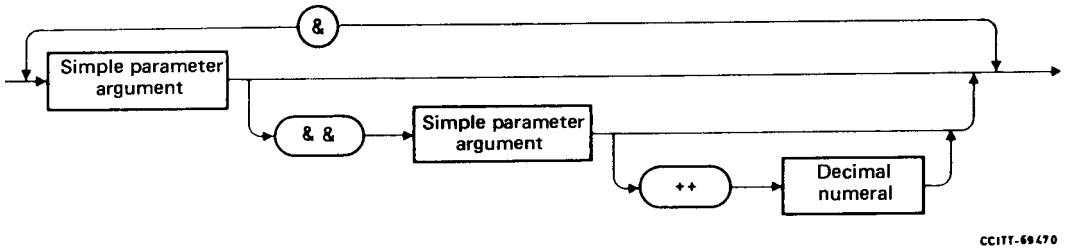
4.9.1 *Group of blocks of parameters*

See syntax diagram § 4.1.

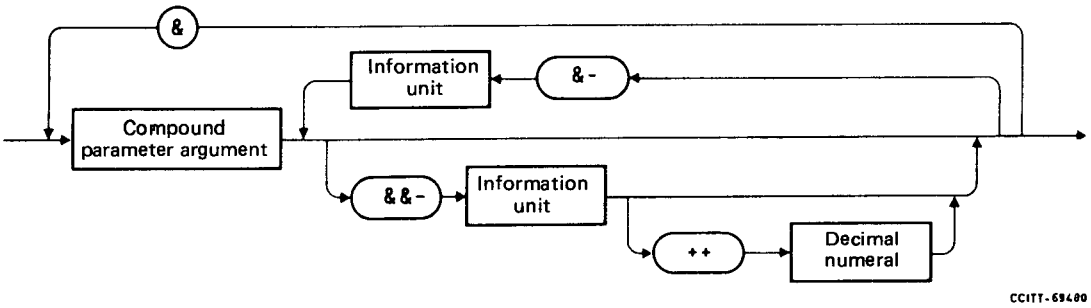
4.9.2 *Group of parameter arguments*



4.9.2.1 *Group of simple parameter arguments*

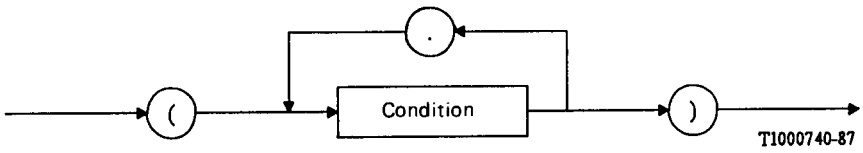



4.9.2.2 *Group of compound parameter arguments*

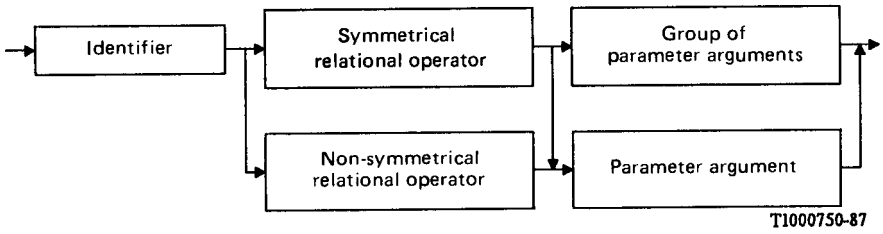



4.10 *Data base queries*

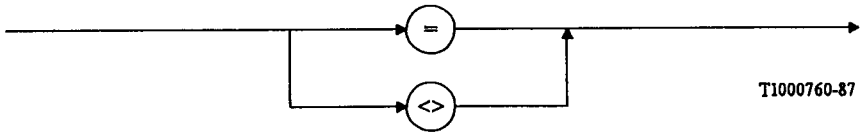
4.10.1 *Selection argument*



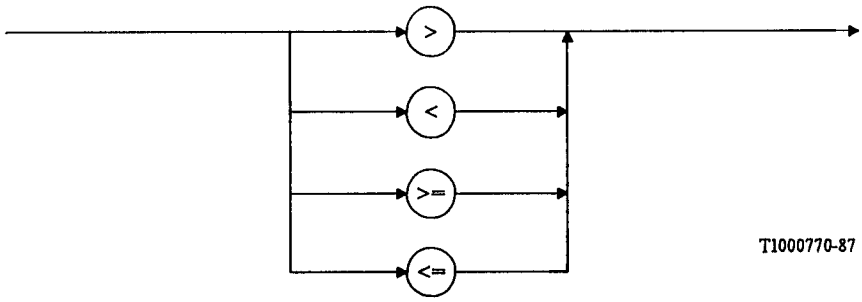
4.10.1.1 *Condition*




4.10.1.2 *Symmetrical relational operator*




4.10.1.3 *Non-symmetrical relational operator*







4.10.2 *Group of selection arguments*