JTC 1

INTERNATIONAL STANDARD

ISO 9040

First edition
1990-11-15

AMENDMENT 2
1992-10-15

Corrected and reprinted
1994-02-01

Information technology — Open Systems Interconnection — Virtual Terminal Basic Class Service

AMENDMENT 2: Additional functional units

Technologies de l'information — Interconnexion des systèmes ouverts — Service de classe de base de terminal virtuel

AMENDEMENT 2: Unités fonctionnelles supplémentaires

Reference number
ISO 9040 : 1990/Amd.2: 1992 (E)

CONTENTS

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) together form a system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

Amendment 2 to International Standard ISO 9040:1990 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

Annex A forms an integral part of this amendment.

Introduction

This amendment extends the service defined in ISO 9040 to provide ripple mode editing, exception reporting and retention of VT-context across negotiation.

The following clauses and annexes in ISO 9040:1990 apply unchanged:

Introduction
clause 1 Scope
clause 2 Normative references
clause 5 Conventions
clause 8 Modes of operation
clause 9 Access-rules
clause 11 VT Environment Profiles
clause 12 The VTE Model
clause 13 Display Objects
clause 15 Reference Information Objects
clause 16 Device Objects
clauses 21–25, 29, 31–33
Annexes A, B and D

Instructions for amending ISO 9040:1990 are given in italics; clause numbers and titles in this amendment (in bold type) are those of this amendment and do not correspond to those of ISO 9040:1990.

This amendment will be consolidated into a new edition of ISO 9040 at its next revision.

Information technology – Open Systems Interconnection – Virtual Terminal Basic Class Service –

Amendment 2: Additional functional units

1 Scope

ISO 9040, clause 1 applies.

2 Normative references

The references given in clause 2 of ISO 9040 apply to this amendment.

3 Definitions

ISO 9040, clause 3 applies with the following additions:

3.3.73 ripple: Ripple is a mechanism whereby the contents of array elements of the Display Object may be moved into adjacent array elements without the need of reconveying the contents from one VTE user to the other.

3.3.74 ripple mode control object (RMCO): The Ripple Mode Control Object is a control object associated with the ripple mechanism which controls the operation of the ripple, for example, the extent of the ripple.

3.3.75 ripple-extent: A ripple-extent is that part of a DO within which the ripple mechanism operates.

3.3.76 copy buffer: A copy buffer stores the content and structure of an extent of the Display Object to allow subsequent transfer of this structure and content to some other extent of the DO.

3.3.77 extended-y-array: The set of array elements which are within the currently defined blocks of a particular y-array.

3.3.78 extended-z-array: The set of array elements which are within the currently defined blocks of some y-array of the z-array.

3.3.79 filling: Is the operation which defines the values of the primary and secondary attributes of character box elements which are left undefined as the result of ripple operations.

3.3.80 ripple coordinate: Determines the coordinate direction of the ripple, i.e. the coordinate which is altered as the result of ripple operations, and hence determines the unit of ripple. It takes the values "x", "y", "z", "k".

3.3.81 ripple direction: Specifies whether units are to be moved in a forward (increasing coordinate) or a backward (decreasing coordinate) direction, in a ripple operation.

3.3.82 unit of ripple: Determines the units which the ripple operations work upon. It takes a values from the set "array element", "x-array" and "y-array".

4 Abbreviations

ISO 9040, clause 4 applies with the following additions:

RMCO Ripple Mode Control Object

5 General features

5.1 References

ISO 9040, subclause 6.2 applies with the extensions defined in 5.2 of this amendment.

5.2 Extension to features of the service

This amendment provides additional functional units which, when selected for a VT-association, add the following features to the list of features in ISO 9040, clause 6.2:

r) display object update with ripple;

s) the reporting of exception conditions by the VT-service-provider;

t) the selective retention of VT-context between negotiation of successive VTEs.

6 Communication facilities

ISO 9040, clause 7 applies with the addition of the facility defined below.

7.8 Exception reporting facility

The exception reporting facility provides services which allow the VT-service-provider to report abnormal conditions to the VT-users without causing termination of the VT-association.

7 VT functional units

The following list of functional units is added to the list in ISO 9040, clause 10.

k) Ripple

l) Exceptions

m) Context Retention

The following subclauses are added to ISO 9040, clause 10.

10.10 Ripple functional unit

The Ripple functional unit provides insertion, deletion and copy operations for a display object. These operations are available in two forms, either using basic addressing or using logical addressing. If this functional unit is selected, the structured control objects functional unit must also be selected.

NOTE – These functions allow simple text editing to be performed with much lower communications overhead than if the functional unit were not selected. Without this facility, editing of text with insertion and deletion would require the transmission of large parts of the Display Object between the peer VT-users. With this facility, only information relating to the actual insertion and deletion operations need be transmitted. Considerable communication savings are therefore possible.

10.11 Exceptions functional unit

The Exceptions functional unit provides mechanisms by which non-fatal exception conditions may be reported by the VT-service-provider to both VT-users. A facility is provided to ensure that the integrity of the VT-context is maintained.

The Exceptions functional unit may only be selected if the Break functional unit is selected.

NOTE – Without this facility, the VT-Service-provider normally aborts the VT-association if any exception condition is discovered. With this facility, exception conditions may be classified as either fatal or non-fatal. Fatal exception conditions will still cause the VT-service-provider to issue a VT-P-ABORT primitive.

10.12 Context Retention functional unit

The Context Retention functional unit allows the retention of the information stored in selected VT-objects (DOs and COs) to be retained between successive VTEs within the life time of a VT-association.

The Context Retention functional unit may be selected only if the Switch Profile Negotiation functional unit is selected.

NOTE – This functional unit will allow, for instance, the contents of a display object to be retained when a new control object is brought into existence through negotiation. Without this functional unit, such contents would be lost.

8 Control objects

The following subclause is added to ISO 9040, clause 14.

14.6 Standard CO for ripple mode editing

The Ripple Mode Control Object (RMCO) is used as part of the ripple mode editing facility made available with the Ripple functional unit. There is one such control object if the Ripple functional unit is selected. Use of this CO type requires the Structured Control Objects functional unit to be selected. Clause 20.3.9 defines the VTE-parameters for this CO and gives further details of its use.

Use of this CO is mandatory if the Ripple functional unit is used for the updating of a display object. The following information is held:

a) a boolean switch to allow a VT-user to alternate between insertion and overwrite operation of DO updates;

b) the primary and secondary attributes to be applied to empty space resulting from ripple operations;

c) a control which governs the action to be taken in overflow situations at the end of a line or a page;

d) a control which governs the action to be taken in overflow situations when using logical addressing;

e) a control to determine whether insert and delete array operations are to be before or after the display pointer.

9 Directed graph of VTE–parameters

Figure 7 of ISO 9040 applies with the extensions defined in Figure 1 of this amendment.

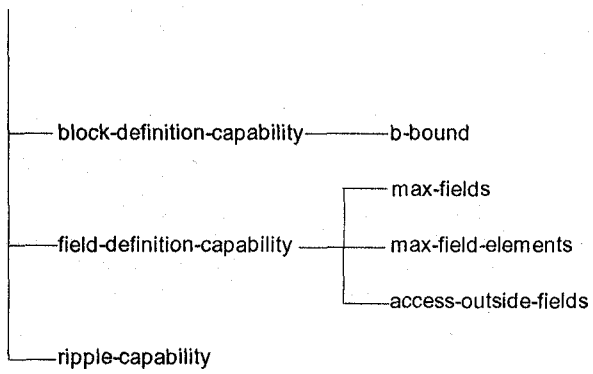

Figure 1 – Extension to directed graph

10 Display object VTE-parameters

10.1 References

Use of any of the additional functional units does not affect any of the provisions of ISO 9040, clause 18 except as defined in 10.2 of this amendment.

10.2 Ripple-capability primary VTE-parameter

The following line is added to ISO 9040, table 2.

ripple-capability "yes"/"no" (default = "no")

The following paragraph is added to ISO 9040, subclause 18.1.

The VTE-parameter ripple-capablility may only be defined if the Ripple functional unit is selected.

11 Operations on display objects

This clause defines changes to DO update operations when a DO has ripple-capability = "yes" and ripple-mode = "insert". It also defines additional DO update operations which are available when a DO has ripple-capability = "yes".

11.1 References

ISO 9040, subclause 19.1.1.1 (Implicit addressing with the primitive display pointer) is modified by 11.2 of this amendment.

ISO 9040, subclause 19.1.2.1 (Implicit addressing with extended display pointer) is modified by 11.3 of this amendment.

ISO 9040, subclause 19.1.3.1 (Implicit logical addressing) is modified by 11.4 of this amendment.

The text of 11.5 of this amendment is inserted immediately preceding ISO 9040, subclause 19.2.1.1 (TEXT operation).

ISO 9040, subclause 19.2.1.1 (TEXT operation) is qualified by 11.6 of this amendment.

ISO 9040, subclause 19.2.1.2 (REPEAT-TEXT operation) is qualified by 11.7 of this amendment.

ISO 9040, subclause 19.2.1.4 (ERASE operation) is qualified by 11.8 of this amendment.

The text of 11.9 to 11.13 of this amendment is inserted logically before ISO 9040, subclause 19.2.2 (Update operations using the logical pointer).

The text of 11.14 of this amendment is inserted before ISO 9040, subclause 19.2.2.1 (LOGICAL-TEXT operation).

ISO 9040, subclause 19.2.2.1 (LOGICAL-TEXT operation) is qualified by 11.15 of this amendment.

ISO 9040, subclause 19.2.2.2 (REPEAT-LOGICAL-TEXT operation) is qualified by 11.16 of this amendment.

ISO 9040, subclause 19.2.2.4 (LOGICAL-ERASE operation) is qualified by 11.17 of this amendment.

The text of 11.18 of this amendment is inserted before ISO 9040, subclause 19.3 (Access control over display object).

11.2 Implicit addressing with ripple

In ISO 9040, subclause 19.1.1.1, insert "and ripple is not enabled" *after* "array element".

Insert a new paragraph at the end of ISO 9040, subclause 19.1.1.1.

When a value is written to a primary attribute value and ripple is enabled, the display pointer is updated to point to the next element of the ripple-extent (or to point immediately after the ripple-extent, if there is no next element).

11.3 Implicit addressing with extended display pointer and ripple

In ISO 9040, subclause 19.1.2.1, insert "and ripple is not enabled" *after* "for a DO".

The following paragraph is added at the end of ISO 9040, subclause 19.1.2.1.

When a value is written to a primary attribute value and ripple is enabled, the display pointer is updated to point to the next element of the ripple-extent (or to point immediately after the ripple-extent, if there is no next element).

11.4 Implicit logical addressing with ripple

In ISO 9040, subclause 19.1.3.1, insert "and ripple is not enabled" *after* ", 19.2.3.1)".

The following paragraph is added at the end of ISO 9040, subclause 19.1.3.1.

When a value is written to a primary attribute value and ripple is enabled, the display pointer is updated to point to the next element of the ripple-extent (or to point immediately after the ripple-extent, if there is no next element).

11.5 Ripple operations

The following subclauses are inserted in front of ISO 9040, subclause 19.2 and subsequent text is renumbered.

19.2 Ripple operations

Ripple operations provide for the bulk movement of character box elements either to create space for an insertion or to close up a space after a deletion. Ripple operations are not provided as separate DO updates. Instead, they are used as part of the definition of DO updates such as TEXT, COPY-FROM-BUFFER and INSERT-X-ARRAY when ripple is enabled.

Ripple is enabled when the DO has *ripple-capability* "yes" and the ripple-mode element of the RMCO is "true".

A ripple operation is specified by defining:

- the elements affected by the ripple (the ripple-extent), (see 11.4.1 of this amendment);
- the ripple coordinate (x, y or z)
- *ripple direction* (forwards for insertion, backwards for deletion);
- the number of units of ripple, N.

The coordinate, direction and number of units depend on the DO update performing the ripple operation.

In a ripple operation, the ripple-extent is taken as a sequence of units, the unit depending on the coordinate direction of the ripple:

- x array element;
- y x-array;
- z y-array.

The content of a unit is the collection of primary and secondary attributes of all the array elements of the unit.

For a forward ripple, the content of each unit M after the Nth is set to the previous content of unit M – N. The new content of the first N units is specified by the DO update initiating the ripple. If the ripple extent is bounded, the old content of the last N units is lost.

For a backward ripple, the content of each unit M except the last N is set to the previous content of unit M + N. If the ripple extent is bounded, each array element of the last N units is set according to the fill procedure, (see 11.4.2 of this amendment).

19.2.1 Definition of ripple–extent

The ripple-extent for an operation consists of the extent for that operation, extended either to an end point determined by ripple-limit or to any array element which is below any update- window, whichever is smaller. The end point determined by the value of the ripple-limit element in the RMCO is the end of an array containing the end of the operation extent:

	ripple-limit	array
primitive operations		
	x	x-array
	y	y-array
	z	z-array
extended operations		
	p	p-array
	q	block
	b	extended-y-array
	z	extended-z-array

NOTE – The above definition ensures that the ripple-extent does not contain any array elements which are not updatable because of update-window constraints, unless they are already contained in the operation extent (in which case the operation is invalid).

19.2.2 Fill operations

Filling determines the primary and secondary attributes of array elements which are emptied by ripple operations.

If the fill-mode element of the RMCO has value "erase", the primary attribute is unset and all secondary attributes are set to the explicit modal default.

If the fill-mode element of the RMCO has value "fill", the primary attribute is set to the value of the fill-character element of the RMCO. The secondary attribute character-repertoire is set from the CO-repertoire VTE-parameter of the fill-character element of the RMCO. Each rendition secondary attribute is determined by the following conditions, in order:

a)
1) if an operation using the display pointer is being performed, the corresponding modal attribute, if not "null";
2) if an operation using the logical pointer is being performed, the corresponding FDR attribute if set, else the field modal attribute if not "null";

b) the corresponding global attribute value, if not "null";

c) the explicit modal default.

11.6 TEXT operation with ripple

The following paragraph is added at the end of ISO 9040, subclause 19.2.1.1 before renumbering.

If ripple is enabled, a forward ripple is performed before the primary and secondary attributes are updated, with ripple coordinate x and ripple of one unit. The ripple-extent is derived as specified in 11.4.1 of this amendment from an operation extent consisting of the single array element at the display pointer.

11.7 REPEAT–TEXT operation with ripple

The following paragraph is added at the end of ISO 9040, subclause 19.2.1.2 before renumbering.

If ripple is enabled, a forward ripple is performed before the REPEAT-TEXT operation, with ripple coordinate x and the number of units equal to the number of array elements in the repeat-extent. The ripple-extent is derived from the repeat-extent as specified in 11.4.1 of this amendment.

11.8 ERASE operation with ripple

The following paragraph is added at the end of ISO 9040, subclause 19.2.1.4 before renumbering.

If ripple is enabled, the ERASE operation is replaced by a backward ripple performed with ripple coordinate x, the number of units equal to the number of array elements in the erase-extent and ripple-extent derived from the erase-extent as specified in 11.4.1 of this amendment.

11.9 INSERT–X–ARRAY operation

The following subclause is inserted in ISO 9040, subclause 19.2 before renumbering.

19.2.1.6 INSERT–X–ARRAY operation

This operation is available when ripple is enabled, and has the form

INSERT–X–ARRAYno-of-arrays

A forward ripple operation is performed, with ripple coordinate y and the number of units equal to no-of-arrays. The ripple-extent is derived as defined in 11.4.1 of this amendment from an operation extent consisting of no-of-arrays x-arrays (or p-arrays).

If the backwards-forwards element of the RMCO has value "backwards", the x-array (or p-array) containing the display pointer is the first unit of the operation extent;

If backwards-forwards has value "forwards", the x-array (or p-array) following that containing the display pointer is the first unit of the operation extent.

NOTE – If blocks are not in use, the operation extent may cross more than one y-array.

Each array element of the first no-of-arrays x-arrays (or p-arrays) in the ripple-extent is filled as defined in 11.2.2 of this amendment.

If blocks are in use, INSERT–X–ARRAY is only permitted if the ripple-extent lies within one block. In particular, ripple-limit must be "p" or "q".

It is an error if any element of the ripple-extent is below any update-window.

If the backwards-forwards element of the RMCO has value "backwards", INSERT-X-ARRAY leaves the display pointer unchanged. If the backwards-forwards element of the RMCO has value "forwards", INSERT-X-ARRAY updates the y and z coordinates of the display pointer to point to the last x-array of the operation-extent, leaving the x coordinate unchanged.

11.10 DELETE–X–ARRAY operation

The following subclause is inserted in ISO 9040, subclause 19.2 before renumbering.

19.2.1.7 DELETE–X–ARRAY operation

This operation is available when ripple is enabled, and has the form

DELETE–X–ARRAYno-of-arrays

A backward ripple operation is performed, with ripple coordinate y and the number of units equal to no-of-arrays. The ripple-extent is derived as defined in 11.4.1 of this amendment from an operation extent consisting of no-of-arrays x-arrays (or p-arrays).

If the backwards-forwards element of the RMCO has value "backwards", the x-array (or p-array) containing the display pointer is the first unit of the operation extent.

If backwards-forwards has value "forwards", the x-array (or p-array) containing the display pointer is the last unit of the operation extent. The operation is only defined if there are at least (no-of-arrays – 1) x-arrays (or p-arrays) in the display object before that containing the display pointer.

NOTE – If blocks are not in use, the operation extent may cross more than one y-array.

If blocks are in use, DELETE–X–ARRAY is only permitted if the ripple-extent lies within one block. In particular, ripple-limit must be "p" or "q".

It is an error if any element of the ripple extent is below any update-window.

If the backwards-forwards element of the RMCO has value "backwards", DELETE–X–ARRAY leaves the display pointer unchanged. If the backwards-forwards element of the RMCO has value "forwards", DELETE–X–ARRAY updates the y and z coordinates of the display pointer to point to the x-array immediately before the operation-extent, leaving the x coordinate unchanged.

11.11 INSERT–Y–ARRAY operation

The following subclause is inserted in ISO 9040, subclause 19.2 before renumbering.

19.2.1.8 INSERT–Y–ARRAY operation

This operation is available when ripple is enabled and blocks are not in use, and has the form

INSERT–Y–ARRAYno-of-arrays

A forward ripple operation is performed with ripple coordinate z and the number of units equal to no-of-arrays. The ripple-extent is derived as defined in 11.4.1 of this amendment from an operation extent consisting of no-of-arrays y-arrays.

If the backwards-forwards element of the RMCO has value "backwards", the y-array containing the display pointer is the first unit of the operation extent;

If backwards-forwards has value "forwards", the y-array following that containing the display pointer is the first unit of the operation extent.

Each array element of the first no-of-arrays y-arrays in the ripple-extent is filled as defined in 11.4.2 of this amendment.

It is an error if any element of the ripple-extent is below any update-window.

If the backwards-forwards element of the RMCO has value "backwards", INSERT–Y–ARRAY leaves the display pointer unchanged. If the backwards-forwards element of the RMCO has value "forwards", INSERT–Y–ARRAY updates the z coordinate of the display pointer to point to the last y-array of the operation-extent, leaving the x and y coordinates unchanged.

11.12 DELETE–Y–ARRAY operation

The following subclause is inserted in ISO 9040, subclause 19.2 before renumbering.

19.2.1.9 DELETE–Y–ARRAY operation

This operation is available when ripple is enabled and blocks are not in use, and has the form

DELETE–Y–ARRAYno-of-arrays

A backward ripple operation is performed with ripple coordinate z and the number of units equal to no-of-arrays. The ripple-extent is derived as defined in 11.4.1 of this amendment from an operation extent consisting of no-of-arrays y-arrays.

If the backwards-forwards element of the RMCO has value "backwards", the y-array containing the display pointer is the first unit of the operation extent.

If backwards-forwards has value "forwards", the y-array containing the display pointer is the last unit of the operation extent. The operation is only defined if there are at least (no-of-arrays – 1) y-arrays (or q-arrays) in the display object before that containing the display pointer.

It is an error if any element of the ripple extent is below any update-window.

If the backwards-forwards element of the RMCO has value "backwards", DELETE-Y-ARRAY leaves the display pointer unchanged. If the backwards-forwards element of the RMCO has value "forwards", DELETE-Y-ARRAY updates the z coordinate of the display pointer to point to the y-array immediately before the operation-extent, leaving the x and y coordinates unchanged.

11.13 Copy operations

The following subclause is inserted in ISO 9040, subclause 19.2 before renumbering.

19.2.1.10 Copy operations

Operations are provided to copy the contents of a region of a display object to or from an external buffer. This buffer can be either a (named) RIO record or a temporary buffer. Copy operations are not available when blocks are in use.

19.2.1.10.1 Copy buffer

A copy buffer stores the content and structure of an extent of the display object to allow subsequent transfer of this structure and content to some other extent of the DO.

A copy buffer may be held either in a RIO record (the choice is available only if the RIO functional unit is selected and a RIO has been included in the VTE), or in a special temporary buffer. There is one temporary buffer associated with each DO. Each temporary buffer is empty when a new full-VTE is established. The content of temporary buffers is only changed by COPY-TO-BUFFER and COPY-LOGICAL-TO-BUFFER.

The content of the copy buffer is a sequence of DO updates taken from the set:

- TEXT
- ATTRIBUTE
- NEXT X–ARRAY
- NEXT Y–ARRAY
- POINTER–RELATIVE
- LOGICAL–TEXT
- LOGICAL–ATTRIBUTE
- NEXT FIELD
- LOGICAL–RELATIVE

NOTES

1 Each ATTRIBUTE or LOGICAL–ATTRIBUTE operation applies to exactly one array element.

2 The above encoding provides a formal definition of the copy buffer. Typical implementations will use some other more efficient encoding with the same semantics, for example by grouping TEXT and ATTRIBUTE operations.

19.2.1.10.2 COPY–TO–BUFFER operation

This operation takes the form

COPY-TO-BUFFER end-address buffer-name rendition structure

The arguments are

a) end-address is a display pointer value greater than or equal to "current". It is either an explicit value or one of the symbolic values defined in ISO 9040, subclause 19.1.1.4 as valid as an extent end;

b) buffer-name is one of

– a pair < RIO-name, record-id> , identifying a RIO record as in ISO 9040, subclause 22.2.2. RIO-name is optional if the VTE contains only one RIO;

– the symbolic value "temporary", identifying the temporary buffer associated with the DO;

c) rendition takes one of the values "copy attributes", "no attribute copy";

d) structure takes one of the values "none", "x", "x and y".

The operation initialises then fills in the copy buffer designated by buffer name.

A copy buffer is initialised by creating it (if it is a RIO record which does not exist) followed by setting its contents to the empty sequence.

The copy operation applies to the set of array elements (x; y; z) which satisfy

$x_c \leq x \leq x_f$; $y_c \leq y \leq y_f$; $z_c \leq z \leq z_f$
(if structure is "x and y")

$x_c \leq x \leq x_f$; $(y_c , z_c) \leq (y , z) \leq (y_f , z_f)$
(if structure is "x")

$(x_c , y_c , z_c) \leq (x , y , z) \leq (x_f , y_f , z_f)$
(if structure is "none")

where (x_c , y_c , z_c) is the current value of the display pointer and (x_f , y_f , z_f) is end-address.

The selected elements are processed in order of address, as follows. For each array element, DO updates are added in order to the end of the current contents of the buffer:

a) for each element after the first, if the element is in a different x-array or y-array, add DO updates as specified in table 8.

Table 8 – Copy insertion

structure	new Y-array?	new X-array?	add
x&y	Y	–	NEXT-Y-ARRAY
x&y	N	–	NEXT-X-ARRAY
x	Y	–	NEXT-X-ARRAY
x	N	Y	NEXT-X-ARRAY

b) if rendition = "copy attributes", for each secondary attribute which has a value assigned, add

ATTRIBUTE attribute-id attribute-value
< "current","current">

If both chracter-repertoire and font have assigned values, characeter-repertoire must be processed before font.

c) if the primary attribute has a value assigned, add

TEXT primary-attribute-value

else if the primary attribute has no value assigned, add

POINTER-RELATIVE x := x + 1

The COPY-TO-BUFFER operation does not affect any DO content or any display pointer.

19.2.1.10.3 COPY–FROM–BUFFER operation

This operation takes the form

COPY–FROM–BUFFER start-address buffer-name
rendition structure ripple

The arguments are

a) start-address is either an explicit value for the display pointer or is one of the symbolic values defined in ISO 9040, subclause 19.1.1.4 which is valid as an extent start.

b) The definition of buffer-name, rendition and structure are the same as the corresponding parameters in COPY–TO–BUFFER in 11.12.2 of this amendment.

c) ripple takes one of the values "on", "off".

The operation proceeds as follows:

– set the display pointer to start-address

– obey the sequence of DO updates in the copy buffer designated by buffer-name.

In obeying the DO updates, the following interpretations apply.

Let $(x_s; y_s; z_s)$ be the start address.

– if structure is "none", NEXT–X–ARRAY is ignored , otherwise, it causes y := y + 1 or, if y= y-bound, NEXT–Y–ARRAY, and x := x_s

– if structure is "none", NEXT–Y–ARRAY is ignored; if structure is "x", NEXT-Y-ARRAY causes y := y + 1 or, if y= y-bound, NEXT-Y-ARRAY, and x := x_s; if structure is "x and y", NEXT-Y-ARRAY causes z := z + 1; y := y_s and x := x_s.

– ATTRIBUTE is ignored if rendition = "no attribute copy" and otherwise processed as below.

If ripple is "off", ATTRIBUTE, TEXT and POINTER-RELATIVE are processed as if ripple-mode in the RMCO were set to "false".

If ripple is "on", ATTRIBUTE, TEXT and POINTER-RELATIVE updates are processed in groups. Each group consists of zero or more ATTRIBUTE updates, followed by either TEXT or POINTER-RELATIVE. For each group, a forward ripple occurs with ripple of one unit, then the ATTRIBUTE, TEXT and POINTER–RELATIVE updates are processed as if ripple-mode in the RMCO were set to "false". POINTER–RELATIVE is interpreted as meaning "move to the next element of the ripple-extent". The ripple-extent is based on an operation extent consisting of the single array element at the display pointer. The ripple coordinate is always x.

The COPY-FROM-BUFFER operation is invalid if any of the individual operations generated are invalid, for example because bounds are violated.

A COPY-FROM-BUFFER from a RIO record is only defined if the RIO record content was created by a COPY-TO-BUFFER operation.

The COPY-FROM-BUFFER operation is invalid if the copy buffer contains any logical pointer operations.

No change is made to the source copy buffer.

11.14 Logical ripple operations

The following subclauses are inserted in front of ISO 9040, subclause 19.2 and subsequent text is renumbered.

19.3 Logical ripple operations

Logical ripple operations provide for the bulk movement of character box elements either to create space for an insertion or to close up space after a deletion. Logical ripple operations are not provided as separate logical DO updates. Instead, they are used as part of the definition of logical DO updates such as LOGICAL–TEXT and COPY–LOGICAL–FROM–BUFFER when ripple is enabled.

Logical ripple is enabled when the DO has ripple-capability "yes" and the ripple-mode element of the RMCO is "true".

A logical ripple operation is specified by defining:

– the elements affected by the logical ripple (the ripple-extent) see 11.14.1 of this amendment;

– the ripple direction (forwards for insertion, backwards for deletion);

– the number of units of ripple, N.

The ripple direction and the number of units of ripple depend on the logical DO update performing the ripple operation.

In a logical ripple operation, the ripple-extent is taken as a series of units, the content of each being the collection of primary and secondary attributes of the single array element, i.e. the ripple coordinate takes the value "k".

For a forward ripple, the content of each unit M after the Nth is set to the previous content of M – N. The new content of the first N units is specified by the logical DO update initiating the ripple. If the ripple-extent is bounded, the old content of the last N units is lost.

For a backward ripple, the content of each unit M except the last N is set to the previous content of unit M+ N. If the ripple-extent is bounded, each array element of the last N units is set according to the fill procedure, see 11.4.2 of this amendment.

19.3.1 Ripple-extent for logical operations

The ripple-extent for a logical operation consists of the logical extent for that operation extended to an end point determined by logical-ripple-limit.

If logical-ripple-limit = "k", the end point is the end of the field containing the end of the operation extent.

If logical-ripple-limit = "f", the end point is the last active field on the f-dimension of the y-array containing the end of the operation extent.

If the logical-ripple-limit = "z", the end point is the last active field defined (ie the field with the highest (f,z) coordinate).

11.15 LOGICAL–TEXT operation with ripple

The following paragraph is added at the end of ISO 9040, subclause 19.2.2.1 before renumbering.

If ripple is enabled, a forward ripple with ripple of one unit is performed before the primary and secondary attributes are updated. The ripple-extent is defined as specified in 11.14.1 of this amendment from the extent consisting of the single array element at the logical pointer.

11.16 REPEAT–LOGICAL–TEXT operation with ripple

The following paragraph is added at the end of ISO 9040, subclause 19.2.2.2 before renumbering.

If ripple is enabled, a forward ripple is performed before the REPEAT-LOGICAL-TEXT operation, with the number of units equal to the number of array elements in the repeat-extent. The ripple-extent is derived from the repeat-extent as specified in 11.14.1 of this amendment.

11.17 LOGICAL–ERASE operation with ripple

The following paragraph is added at the end of ISO 9040, subclause 19.2.2.4 before renumbering.

If ripple is enabled, the LOGICAL-ERASE operation is replaced by a backward ripple performed with the number of units equal to the number of array elements in the logical-erase-extent and with ripple-extent derived from the logical-erase-extent as specified in 11.14.1 of this amendment.

11.18 Logical copy operations

The following subclauses are inserted before ISO 9040, subclause 19.3.

19.2.2.5 Logical copy operations

Operations are provided to copy the contents of a region of the display object to or from an external buffer. This buffer can either be in a (named) RIO (as an identified record) or can be a temporary buffer.

19.2.2.5.1 COPY–LOGICAL–TO–BUFFER operation

This operation takes the form

COPY–LOGICAL–TO–BUFFER end-address
buffer-name rendition structure

The arguments are

a) end-address is a logical pointer value greater than or equal to log-current. It is either an explicit value or one of the symbolic values defined in ISO 9040, subclause 19.1.3.5 as valid as an extent end;

b) buffer-name is one of

– a pair < RIO-name, record-id> , identifying a RIO record as in ISO 9040, subclause 22.2.2. RIO-name is optional if the VTE contains only one RIO;

– the symbolic value "temporary", identifying the temporary buffer associated with the DO;

c) rendition takes one of the values "copy attributes", "no attribute copy";

d) structure takes one of the values "none", "k".

The operation initialises and then fills in the copy buffer designated by buffer name.

A copy buffer is initialised by creating it (if it is a RIO record which does not exist) followed by setting its contents to the empty sequence.

The copy operation applies to the set of array elements (k, f, z) which lie within active fields and which satisfy

$k_c \leq k \leq kf, (f_c , z_c) \leq (f , z) \leq (f_f , z_f)$
(if structure value of "k")

$(k_c , f_c , z_c) \leq (k , f , z) \leq (k_f , f_f , z_f)$
(structure value of "none")

where (k_c , f_c , z_c) is the current value of the logical pointer and (k_f , f_f , z_f) is the end address.

The selected elements are processed in order of address, as follows. For each array elment, DO updates are added in order to the end of the current contents of the buffer:

a) if structure = "k" and this is not the first element to be processed and this element is in a different field from the previous element then add NEXT FIELD;

b) if rendition = "copy-attributes", for each secondary attribute which has a value assigned, add LOGICAL–
ATTRIBUTE attribute-id attribute-value < "log-current", "log-current"> . Attribute-value shall not take the value "field-explicit-value". If both character-repertoire and font have assigned values, repertoire must be processed before font.

c) if the primary attribute has a value assigned, add LOGICAL–TEXT primary-attribute-value FDR-attribute = "no" else add LOGICAL–RELATIVE k:= k+ 1

The COPY–LOGICAL–TO–BUFFER operation does not affect any DO content or the logical pointer.

19.2.2.5.2 COPY–LOGICAL–FROM–BUFFER operation

This operation takes the form

COPY–LOGICAL–FROM–BUFFER start-address buffer-name rendition structure ripple

The arguments are

a) end-address is an explicit value or one of the symbolic values defined in ISO 9040, subclause 19.1.3.5 as valid as an extent start;

b) buffer-name is one of

– a pair < RIO-name, record-id> , identifying a RIO record as in ISO 9040, subclause 22.2.2. RIO-name is optional if the VTE contains only one RIO;

– the symbolic value "temporary", identifying the temporary buffer associated with the DO;

c) rendition takes one of the values "copy attributes", "no attribute copy";

d) structure takes one of the values "none", "k";

e) ripple takes one of the values "on", "off".

The operation proceeds as follows:

– set the logical pointer to start-address

– obey the sequence of DO updates in the copy buffer designated by buffer-address

In obeying the DO updates, the following interpretations apply:

Let (k_S,f_S,z_S) be the start address.

– NEXT–FIELD is ignored if structure = "none". If structure = "k" and there is another active field in the forward f direction, f is incremented to a value where the next active field exists on the current y-array. If such a field does not exist on the current y-array, z:= z+ 1, f:= 1. This process is repeated until an active field is reached. Then k:= k_S. The operation terminates if no further active fields can be found;

– LOGICAL–ATTRIBUTE is ignored if rendition = "no attribute copy" and otherwise processed as below.

If ripple is "off", LOGICAL–ATTRIBUTE, LOGICAL–TEXT and LOGICAL–RELATIVE are processed as if ripple-mode in the RMCO were set to "false".

If ripple is "on", LOGICAL–ATTRIBUTE, LOGICAL–TEXT and LOGICAL–RELATIVE updates are processed in groups. Each group consists of zero or more LOGICAL–ATTRIBUTE updates, followed by either LOGICAL–TEXT or LOGICAL–RELATIVE. For each group, a forward ripple occurs with a ripple of one unit, then the LOGICAL–ATTRIBUTE, LOGICAL–TEXT and LOGICAL–RELATIVE updates are processed as if ripple-mode in the RMCO were set to "false". LOGICAL–RELATIVE is interpreted as "move to the next element in the ripple-extent.

The logical-ripple-extent is based upon an operation extent consisting of the single array element at the logical pointer.

The COPY–LOGICAL–FROM–BUFFER operation is invalid if any of the individual operations generated are invalid, for example because bounds are violated.

A COPY–LOGICAL–FROM–BUFFER from a RIO record is only defined if the RIO record content was created by a COPY–LOGICAL–TO–BUFFER operation.

The COPY–LOGICAL–FROM–BUFFER operation is invalid if the copy buffer contains any primitive pointer operations.

No change is made to the source copy buffer.

12 Control object parameters

A new standard control object type is defined in this amendment, see clause 8 of this amendment. This clause defines the values for the CO generic VTE-parameters which enable the new RMCO type to be included in a VTE. It also defines the nature of the information field for the RMCO.

The following subclause is added to the end of ISO 9040, subclause 20.3

20.3.9 Ripple mode control object RMCO

The Ripple Mode CO (RMCO) is used in conjunction with the ripple-capability available with the Ripple functional unit. Use of this CO requires the structured control objects functional unit to be selected.

20.3.9.1 VTE-parameters for RMCO

CO-name : any value unique within the VTE;
CO-type-identifier : object identifier value vt-b-sco-rmco, see Annex A of this amendment, identifying the CO as being of type RMCO;
CO-structure : 6
CO-access : as for the corresponding DO
CO-priority : "normal"
CO-trigger : "not selected"

a) ripple-mode
CO-element-id : 1
CO-category : "Boolean"
"true" - insert
"false" - overwrite (the default)
CO-size : 1

b) fill-mode
CO-element-id : 2
CO-category : "boolean"
"true" - fill
"false" - erase
CO-size : 1

c) fill-character
CO-element-id : 3
CO-category : "character"
CO-repertoire-assignment : (default - < space>
- IRV ISO 646)
CO-size : 1

d) ripple-limit
CO-element-id : 4
CO-category : "symbolic" from the set ("x", "y", "z", "p", "q", "b").
CO-size : 6
The initial value for ripple-limit is "x"

e) logical-ripple-limit
CO-element-id : 5
CO-category : "symbolic" from the set ("k", "f", "z").
CO-size : 2
The initial value for logical-ripple-limit is "k"

NOTE – The value of this element is only relevant if field-definition-capability is selected.

f) backwards-forwards
CO-element-id : 6
CO-category : "symbolic" from the set ("backwards", "forwards")
CO-size : 2
The initial value for backwards-forwards is "backwards"

13 VT services

13.1 References

ISO 9040 table 16 is extended by the entry in table 1 of this amendment.

Table 1 – VT services available with functional units

Functional Unit	Facility	Service	Structure	Clause
Exceptions	Exception Reporting	VT–EXCEPTION	Prl, Nsq, Dst	35.1

14 VT service requirements

14.1 References

ISO 9040, subclause 27.4 is extended by 14.2 of this amendment.

ISO 9040 subclause 27.5 is extended by 14.3 of this amendment.

14.2 Availability and usage conditions of VT services

The following item is added to the list in ISO 9040, subclause 27.4. The references relate to ISO 9040.

f) in the event of a collision of VT–SWITCH–PROFILE.request, VT–START–NEG.request or VT–RELEASE.request with VT–P–EXCEPTION, the VT–P–EXCEPTION wins and is performed, see 35.1. Unlike other collisions (see 27.5), a confirm primitive is not generated for the destroyed request.

14.3 Service collisions in A-mode

Note 3 of ISO 9040, subclause 27.5 is amended as follows.

3 A collision between VT–BREAK and any other service primitive, except VT–P–ABORT, VT–U–ABORT and VT–P–EXCEPTION is resolved in favour of the VT-user issuing the VT–BREAK.

The following note is added to ISO 9040, subclause 27.5.

5 A collision between VT–P–EXCEPTION and any other service primitive, except VT–P–ABORT, VT–U–ABORT and VT–BREAK is resolved in favour of the VT–P–EXCEPTION.

15 Establishment facility

15.1 References

ISO 9040, subclause 28.1.3.5 is extended by 15.2 of this amendment.

15.2 Service parameters

The following entries are added to the list in ISO 9040, subclause 28.1.3.5:

k) Ripple
l) Exceptions
m) Context Retention

16 Negotiation facilities

16.1 References

ISO 9040, subclause 30.1.1.3 is extended by 16.2 of this amendment.

ISO 9040, subclause 30.1.1.4 is extended by 16.3 of this amendment.

ISO 9040, subclause 30.2.2.3 is extended by 16.4 of this amendment.

ISO 9040, subclause 30.2.2.4 is extended by 16.5 of this amendment.

16.2 Service parameters for VT-SWITCH-PROFILE service

ISO 9040 table 22 is extended by the entry in table 2 of this amendment.

Table 2 – VT–SWITCH–PROFILE service parameters

Parameter name	Req.	Ind.	Rsp.	Cfm.
VT-object-retention-list	O	C	C	C=

A new subclause 30.1.1.3.5 is added at the end of ISO 9040, subclause 30.1.1.3 as follows:

30.1.1.3.5 VT-object-retention-list may only be present if the Context Retention functional unit is selected. This parameter is then optional and if present in a request contains a list of the names of those DOs and COs whose contents are requested to be retained. This parameter is present in the indication and response if and only if it is present in the request. It is present in the response or confirmation if and only if it is present in the request and VT-result= "success". The service provider may reduce this list as provided in the request depending on its capability and on the specific items being negotiated. The list may be reduced to the empty list. The peer VT-user may further reduce this list. The service provider must not change this parameter as provided in the response when delivering the confirmation.

16.3 Usage and effects

In ISO 9040, subclause 30.1.1.4, replace the second paragraph from "the stored VT-context value..." *to the end that paragraph by the following text.*

the reset-context is set from the new full-VTE. The new full-VTE is installed as current-VTE with the VT-context-value set to its initial value for this full-VTE, see 13.1.1 and 20.2.4, except that objects listed in VT-object-retention-list take their values from the saved VT-context-value.

The following text is added at the end of ISO 9040, subclause 30.1.1.4.

The VT-object-retention-list parameter is used when it is desired to retain the contents of certain VT-objects across an instance of the VT–SWITCH–PROFILE service. The service provider and/or the peer VT-user may decide that it is not possible to retain the contents of specific items listed in this parameter; these items shall be removed from the VT-object-retention-list parameter. For instance:

a) if the result of the use of this service is a change to the VTE-parameters associated with a particular object, then the contents of that object may not be retained;

b) some COs are part of a complex linked data structure. Where that structure is changed as a result of the use of this service, it may not be possible to retain the contents of certain COs which are part of that linked structure.

16.4 Service parameters for VT-END-NEG service

ISO 9040 table 24 is extended by the entry in table 3 of this amendment.

Table 3 – VT–END–NEG service parameters

Parameter name	Req.	Ind.	Rsp.	Cfm.
VT-object-retention-list	O	C	C	C=

A new subclause 30.2.2.3.6 is added at the end of ISO 9040, subclause 30.2.2.3 as follows:

30.2.2.3.6 The definition of VT-object-retention-list is the same as the corresponding parameter on VT–SWITCH–PROFILE, see 16.2 of this amendment.

16.5 Usage and effects

In ISO 9040, subclause 30.2.2.4; under (a), delete "the reset context is set from the chosen VTE."; *in the paragraph* "In case (a)...", *replace* ".. then the draft-VTE ... full-VTE" *by the following paragraph.*

then the reset-context is set from the draft-VTE. The draft-VTE is installed as current-VTE with the VT-context-value set to its initial value for this full-VTE, see 13.1.1 and 20.2.4, except that objects listed in VT-object-retention-list take their values from the saved VT-context-value.

The following text is added at the end of ISO 9040, subclause 30.2.2.4.

The VT-object-retention-list parameter is used when it is desired to retain the contents of certain VT-objects across an instance of multiple interaction negotiation when the result is the adoption of a new full-VTE. The service provider and/or the peer VT-user may decide that it is not possible to retain the contents of specific items listed in this parameter; these items shall be removed from the VT-object-retention-list parameter. For instance:

a) if the result of the use of this service is a change to the VTE-parameters associated with a particular object, then the contents of that object may not be retained;

b) some COs are part of a complex linked data structure. Where that structure is changed as a result of the use of this service, it may not be possible to retain the contents of certain COs which are part of that linked structure.

17 Destructive break facility

17.1 References

ISO 9040, subclause 34.1.4 is extended by 17.2 of this amendment.

17.2 Destructive break facility

Insert the following paragraph before the paragraph beginning "As VT-BREAK is ... " *of ISO 9040, subclause 34.1.4.*

When a VT–BREAK collides with a VT–P–EXCEPTION, the underlying services resolve the collision. Due to the way in which the collision is resolved, the VT-user issuing the request may get a VT–P–EXCEPTION indication instead of the expected VT–BREAK confirm primitive."