



INTERNATIONAL STANDARD ISO 10303-104:2000
TECHNICAL CORRIGENDUM 2

Published 2014-07-01

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION

**Industrial automation systems and integration —
Product data representation and exchange —**

Part 104:

**Integrated generic resource:
Finite element analysis**

TECHNICAL CORRIGENDUM 2

Systèmes d'automatisation industrielle et intégration – Représentation et échange de données de produits

- Partie 104 Ressources génériques intégrées: Analyse par éléments finis

RECTIFICATIF TECHNIQUE 2

Technical Corrigendum 2 to International Standard ISO 10303-104:2000 was prepared by Technical Committee ISO/TC 184, *Automation systems and integration*, Subcommittee SC 4, *Industrial data*.

The purpose of the modifications to the text of ISO 10303-104:2000 is to correct a scoping issue in the FUNCTION required `jd_nodes` and the FUNCTION required `_3d_nodes`. In addition, the FUNCTION variable `_value_type` is corrected to return the correct value for a surface `_tensor2 _2d_variable`.

ICS 25.040.40

Ref. No. ISO 10303-104:2000/Cor.2:2014(E)

© ISO 2014 – All rights reserved

Published in Switzerland

Page 208, 5.15.3 required_2d_nodes

Fully qualify the references to “triangle” and “quadrilateral” in IF statements to avoid namespace clashes. Remove the current EXPRESS definition and replace with:

EXPRESS specification:

```
*)
FUNCTION required_2d_nodes
  (node_list: LIST [1:?] OF node_representation; element_shape:
  element_2d_shape; order: element_order) : BOOLEAN;
  LOCAL
    vertex_nodes          : INTEGER;
    edge_nodes            : INTEGER;
    edge_face_body_nodes  : INTEGER;
  END_LOCAL;

  IF (element_shape = element_2d_shape.triangle) THEN
    vertex_nodes := 3;
    IF (order = linear_order) THEN
      edge_nodes := 0;
      edge_face_body_nodes := 0;
    END_IF;
    IF (order = quadratic_order) THEN
      edge_nodes := 3;
      edge_face_body_nodes := 3;
    END_IF;
    IF (order = cubic_order) THEN
      edge_nodes := 6;
      edge_face_body_nodes := 7;
    END_IF;
  END_IF;

  IF (element_shape = element_2d_shape.quadrilateral) THEN
    vertex_nodes := 4;
    IF (order = linear_order) THEN
      edge_nodes := 0;
      edge_face_body_nodes := 0;
    END_IF;
    IF (order = quadratic_order) THEN
      edge_nodes := 4;
      edge_face_body_nodes := 5;
    END_IF;
    IF (order = cubic_order) THEN
      edge_nodes := 8;
      edge_face_body_nodes := 12;
    END_IF;
  END_IF;

  RETURN ((SIZEOF (node_list) = vertex_nodes + edge_nodes) OR
  (SIZEOF (node_list) = vertex_nodes + edge_face_body_nodes));
END_FUNCTION;
```

(*

Page 210, 5.15.3 required_3d_nodes

Fully qualify the references to “hexahedron”, “wedge”, “tetrahedron”, and “pyramid” in IF statements to avoid namespace clashes. Remove the current EXPRESS definition and replace with:

EXPRESS specification:

*)

```
FUNCTION required_3d_nodes
  (node_list: LIST [1:?] OF node_representation; element_shape:
  volume_3d_element_shape; order: element_order) : BOOLEAN;
  LOCAL
    vertex_nodes          : INTEGER;
    edge_nodes            : INTEGER;
    edge_face_body_nodes  : INTEGER;
  END_LOCAL;

  IF (element_shape = volume_3d_element_shape.hexahedron) THEN
    vertex_nodes := 8;
    IF (order = linear_order) THEN
      edge_nodes := 0;
      edge_face_body_nodes := 0;
    END_IF;
    IF (order = quadratic_order) THEN
      edge_nodes := 12;
      edge_face_body_nodes := 19;
    END_IF;
    IF (order = cubic_order) THEN
      edge_nodes := 24;
      edge_face_body_nodes := 56;
    END_IF;
  END_IF;

  IF (element_shape = volume_3d_element_shape.wedge) THEN
    vertex_nodes := 6;
    IF (order = linear_order) THEN
      edge_nodes := 0;
      edge_face_body_nodes := 0;
    END_IF;
    IF (order = quadratic_order) THEN
      edge_nodes := 9;
      edge_face_body_nodes := 12;
    END_IF;
    IF (order = cubic_order) THEN
      edge_nodes := 18;
      edge_face_body_nodes := 34;
    END_IF;
  END_IF;

  IF (element_shape = volume_3d_element_shape.tetrahedron) THEN
    vertex_nodes := 4;
    IF (order = linear_order) THEN
      edge_nodes := 0;
```

```

        edge_face_body_nodes := 0;
    END_IF;
    IF (order = quadratic_order) THEN
        edge_nodes := 6;
        edge_face_body_nodes := 6;
    END_IF;
    IF (order = cubic_order) THEN
        edge_nodes := 12;
        edge_face_body_nodes := 16;
    END_IF;
END_IF;

IF (element_shape = volume_3d_element_shape.pyramid) THEN
    vertex_nodes := 5;
    IF (order = linear_order) THEN
        edge_nodes := 0;
        edge_face_body_nodes := 0;
    END_IF;
    IF (order = quadratic_order) THEN
        edge_nodes := 8;
        edge_face_body_nodes := 9;
    END_IF;
    IF (order = cubic_order) THEN
        edge_nodes := 16;
        edge_face_body_nodes := 25;
    END_IF;
END_IF;

RETURN ((SIZEOF (node_list) = vertex_nodes + edge_nodes) OR
        (SIZEOF (node_list) = vertex_nodes + edge_face_body_nodes));
END_FUNCTION;

(*)

```

Page 394, 6.835 variable_value_type

Change the return value of SURFACE_TENSOR2_2D_VARIABLE and APPLICATION_DEFINED_TENSOR2_2D_VARIABLE to be SYMMETRIC_TENSOR2_2D instead of SYMMETRIC_TENSOR2_3D. Remove the current EXPRESS definition and replace with:

EXPRESS specification:

```

*)
FUNCTION variable_value_type
(variable : GENERIC) : STRING;

LOCAL
    svt : STRING;
    feacr : STRING;
    variable_typeof : SET [1:?] OF STRING;
END_LOCAL;

svt := 'FEA_SCALAR_VECTOR_TENSOR_SCHEMA.';
feacr := 'FINITE_ELEMENT_ANALYSIS_CONTROL_AND_RESULT_SCHEMA.';
variable_typeof := TYPEOF (variable);

```

```

IF SIZEOF ([ (feacr + 'CURVE_SCALAR_VARIABLE'),
              (feacr + 'SURFACE_SCALAR_VARIABLE'),
              (feacr + 'VOLUME_SCALAR_VARIABLE'),
              (feacr + 'BOUNDARY_CURVE_SCALAR_VARIABLE'),
              (feacr + 'BOUNDARY_SURFACE_SCALAR_VARIABLE'),
              (feacr + 'AGGREGATED_SCALAR_VARIABLE'),
              (feacr + 'VOLUME_ANGULAR_VARIABLE'),
              (feacr + 'AGGREGATED_ANGULAR_VARIABLE'),
              (feacr + 'APPLICATION_DEFINED_SCALAR_VARIABLE')] *
              variable_typeof ) = 1 THEN
    RETURN (svt + 'SCALAR');
END_IF;

IF SIZEOF ([ (feacr + 'CURVE_VECTOR_2D_VARIABLE'),
              (feacr + 'SURFACE_VECTOR_2D_VARIABLE'),
              (feacr + 'APPLICATION_DEFINED_VECTOR_2D_VARIABLE')] *
              variable_typeof ) = 1 THEN
    RETURN (svt + 'TENSOR1_2D');
END_IF;

IF SIZEOF ([ (feacr + 'CURVE_VECTOR_3D_VARIABLE'),
              (feacr + 'SURFACE_VECTOR_3D_VARIABLE'),
              (feacr + 'VOLUME_VECTOR_3D_VARIABLE'),
              (feacr + 'BOUNDARY_CURVE_VECTOR_3D_VARIABLE'),
              (feacr + 'BOUNDARY_SURFACE_VECTOR_3D_VARIABLE'),
              (feacr + 'AGGREGATED_VECTOR_3D_VARIABLE'),
              (feacr + 'APPLICATION_DEFINED_VECTOR_3D_VARIABLE')] *
              variable_typeof ) = 1 THEN
    RETURN (svt + 'TENSOR1_3D');
END_IF;

IF SIZEOF ([ (feacr + 'SURFACE_TENSOR2_2D_VARIABLE'),
              (feacr + 'APPLICATION_DEFINED_TENSOR2_2D_VARIABLE')] *
              variable_typeof ) = 1 THEN
    RETURN (svt + 'SYMMETRIC_TENSOR2_2D');
END_IF;

IF SIZEOF ([ (feacr + 'VOLUME_TENSOR2_3D_VARIABLE'),
              (feacr + 'AGGREGATED_TENSOR2_3D_VARIABLE'),
              (feacr + 'APPLICATION_DEFINED_TENSOR2_3D_VARIABLE')] *
              variable_typeof ) = 1 THEN
    RETURN (svt + 'SYMMETRIC_TENSOR2_3D');
END_IF;

RETURN ('NO_MATCH');

END_FUNCTION;

```

(*

Page 430, Annex B

With the changes identified in this Technical Corrigendum the document identifiers and the schema information object identifiers have changed. Delete the contents of clause B.1 and replace with the following text: