Reference number of working document: **ISO/TC 184/SC 1 N000**

Version description: **V2 - February 2003**

Date: 2003-03-04

Reference number of document: **ISO/DIS 14649-13/2**

Committee identification: ISO/TC 184/SC 1/WG 7

Secretariat: DIN

Industrial automation systems and integration

Physical device control

ISO 14649 Data model for Computerized Numerical Controllers

Part 13/1:PROCESS DATA FOR WIRE-EDM

Document type: Committee draft International standard
Document subtype: if applicable
Document stage: (20) Preparation
Document language: E

Information on technical or structural contents are available at the following addresses:

Convener:	Friedrich Glantschnig
	AMT Consulting
Address:	Höhenweg 33a
	CH-5417 Untersiggenthal
Telephone:	+41 (056) 288 2042
FAX;	+41 (056) 288 3942
E-mail:	fglantschnig@swissonline.ch

Owners of this part of documents,

	Gábor Erdős
	EPFL, LICP-Lausanne
Adress:	EPFL – DGM
	ME - Ecublens
	CH – 1015 Lausanne
Telephone:	+41 (21) 693 2930
FAX:	+41 (21) 693 3553
E-mail:	gabor.erdos@epfl.ch

Contents

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this part of ISO 14649 may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

International Standard ISO 14649-11/1 was prepared by Technical Committee ISO/TC 184, *Industrial automation systems and integration*, Subcommittee SC 1, *Physical device control*.

ISO 14649 consists of the following parts, under the general title

Data model for Computerized Numerical Controllers

— Part 1 *: Overview and fundamental principles, published as actual DIS Phase 1*

— Part 2: *Language bindings, Fundamentals, will be published as Phase 3*

— Part 3: *Language binding in Java, will be published as Phase 3*

— Part 9: *Glossary, will be published as Phase 3*

— Part 10: *General Process Data, published as actual DIS Phase 1*

— Part 11: *Process Data for Milling, published as actual DIS Phase 1*

— Part 11/1: *Tools for Milling, published as actual DIS Phase 1*

— Part 12: *Process Data for Turning, will be published as Phase 3*

— Part 13: *Process Data for EDM, will be published as Phase 3*

— Part 50: *AIM of General Process Data, will be published as Phase 2 as AP2xx*

— Part 51: *AIM of Process Data for Milling, will be published as Phase 2 as AP2xx*

— Part 52: *AIM of Process Data for Turning, will be published as Phase 3 as AP2xx*

— Part 53: *AIM of Process Data for EDM, will be published as Phase 3 as AP2xx*

Introduction

Part 10 of ISO 14649 describes the general process data for numerical controlled machining and includes its schema. The subject of this schema, which is called “machining_schema”, is the definition of data types, which are generally relevant for different technologies (e.g. milling, turning, wire-EDM). It includes the definition of the workpiece, a feature catalogue containing features, which might be referenced by several technologies, the general executables and the basis for an operation definition. Not included in this schema are geometric items and presentations, which are preferenced from ISO 10303’s generic resources, and the technology-specific definitions, which are defined in separate Parts of ISO 14649.

Part 10 of ISO 14649 cannot stand alone. An implementation needs in addition at least one technology-specific part (e.g Part11 for milling). Part13 for wire **E**lectrical **D**ischarging **M**achining (wire-EDM) is described in this paper. The subject is the definition of technology specific data types representing the machining process for wire-EDM.

Additionally, the schema uses machining features similar to ISO 10303-224. The description of process data is done using the EXPRESS language as defined in ISO 10303-11. The encoding of the data is done using ISO 10303-21.

Industrial automation systems and integration — Physical device control — Data model for Computerized Numerical Controllers — Part 13/1: Process data for wire-EDM

1 Scope

This part of ISO 14649 specifies the technology-specific data element needed as process data for wire-EDM. Together with the general process data described in Part 10 of this standard, it describes the interface between computerized numerical controller and the programming system (i.e. CAM system or shopfloor programming system) for wire-EDM. It can be used for wire-EDM operations on this kind of machine.

The scope of this part of ISO 14649 does not include tools for any other technologies, like turning, grinding. Tools for these technologies will be described in other parts of this standard.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO 14649. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO 14649 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO 10303 Part 11: Industrial automation systems and integration: Product data and exchange – Description methods: the EXPRESS Language Reference Manual (IS)

ISO 10303 Part 21: Industrial automation systems and integration: Product data and exchange – Implementation methods: Clear text encoding of exchange structure (IS)

ISO 10303 Part 224: Application protocol: Mechanical product definition for process planning using machining features (IS)

ISO 14649 Part 1: Data model for computerized numerical controllers – Overview and fundamental principles.

ISO 14649 Part 10: Data model for computerized numerical controllers – General process data (DIS).

3 Terms and definitions

For the purposes of this part of ISO 14649, the terms and definitions given in Part 10 of this standard and the followings apply.

3.1 Roughing

A machining operation used to cut a part. While the aim of roughing is to remove large quantities of material in a short time, the surface quality is usually not important. The roughing operation is usually followed by a finishing operation, cf. finishing.

3.2 Finishing

The finishing operation usually follows a roughing operation. The goal of finishing is to reach the tolerance of the feature required. The finishing operation is usually followed by a surface finishing operation.

3.3 Surface Finishing

The surface finishing operation usually follows a finishing operation. The goal of surface finishing is to reach the required surface quality.

4 Symbols (and abbreviated terms)

No symbols defined in this part.

5 Process data for wire-EDM

5.1 Header and references

The following listing gives the header and the list of entities which are referenced within this schema.

```
SCHEMA wire_edm_schema;
(*
Version 5 of Feb 28, 2003
Author: Gabor Erdos  <gabor.erdos@epfl.ch>
Modified by: Willy Maeder  <wmaeder@cadcamation.ch>
             Jacques Richard  <i-tech@eig.unige.ch>
*)

REFERENCE FROM machining_schema (*ISO   14649-10*)
        (bounded_curve,
        cartesian_point,
        direction,
        identifier,
        label,
        length_measure,
        machine_functions,
        machining_operation,
        machining_feature,
        machining_tool,
        material,
        pressure_measure,
        property_parameter,
        speed_measure,
        plane_angle_measure,
        technology,
        machining_strategy,
        toolpath_list
        );
```

5.2 Manufacturing features for wire-EDM

The wire-EDM features defined in this chapter are the features that are specific for wire-EDM technology, and are not defined in Part 10 of ISO 14649. The base class for all wire-EDM features is the *machining_feature*, defined in Part 10 of ISO 14649.

5.2.1 General_path

The most general 4-axis wire-EDM operation is a manufacturing feature described by a ruled surface. But in many cases curve based feature is sufficient. The *general_path* feature is characterised by the fact that the tool movements is curve driven:

The *general_path* feature can be specified in two ways:

i.) defined by one curve, an optional side angle and an optional transition type.

ii.) defined by two synchronised curves.

```
ENTITY general_path
SUPERTYPE OF (ONEOF(general_single_path, general_twin_path))
SUBTYPE OF (machining_feature);
END_ENTITY;
```

5.2.2 General_single_path

The *general_single_path* is defined by a general 2D or 3D curve with a slope angle specification and some specific parameters. The 2-axis wire-EDM operation, which is very usual, is a particular case of *general_single_path* where the slope angle is equal to 0 degree.

```
ENTITY general_single_path
SUBTYPE OF (general_path);
  feature_principal_boundary:       bounded_curve;
  slope:                            OPTIONAL plane_angle_measure;
  transition_types:                 OPTIONAL EDM_transition;
END_ENTITY;
```

feature_principal_boundary:

The outline or shape that forms the principal edge of the *general_path* . When travelling along the curve base as defined by its sense, the material lies on the left side of the curve according to the *axis2_placement_3d* orientation (i.e. when projecting the curve in the local xy plane). It is the *axis2_placement_3d* inherited from the *machining_feature* that defines the local z axis and the local xy plane.

slope:

Optional angle of the border of the *general_path* measured against the local z axis. Default is 0 degree. Implicitly a secondary curve boundary is defined : this is done by extending a line from each point on the principal boundary curve, at the specified angle, until it intersects the secondary local plane at the distance *depth* along the negative local z axis. This implicit secondary boundary has some importance in relation with the *transition_types*.

transition_types:

The type of transition between non-tangent segments.

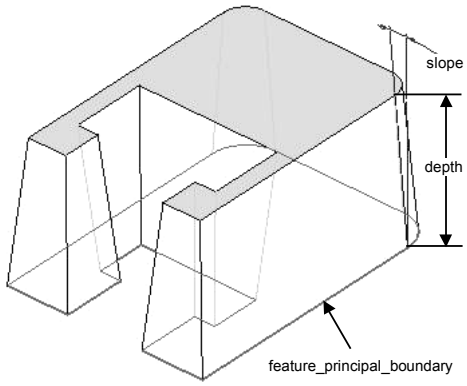


Fig. 1: General_single_path

5.2.3 General_twin_path

The *general_twin_path*, is defined by two general 2D or 3D curves. The two curves are synchronised by the curve parameterisation. This means that the side wall is defined by connecting with a line the points on the principal and the secondary boundary curves corresponding to the same parameter value. The *depth* attribute defined in the *machining_feature* is no more useful.

```
ENTITY general_twin_path
SUBTYPE OF (general_path);
   feature_principal_boundary:       bounded_curve;
   feature_secondary_boundary:       bounded_curve;
END_ENTITY;
```

feature_principal_boundary:

The outline or shape that forms the principal edge of the *general_path* . When travelling along the curve base as defined by its sense, the material lies on the left side of the curve according to the *axis2_placement_3d* defined in the *machining_feature*. (see Figure 2)

feature_secondary_boundary:

The outline or shape that forms the secondary edge of the *general_path*.

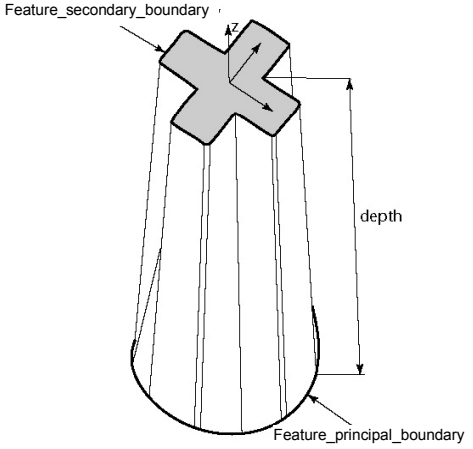


Fig. 2: General_twin_path

5.2.4 General_path_pocket

This is the abstract base class for wire-EDM pockets. Derived from this base class are closed pockets and open pockets. The geometry of the pocket is defined with a *general path*. The pocket may possess one or more bosses.

```
ENTITY general_path_pocket
ABSTRACT SUPERTYPE OF (ONEOF(general_path_closed_pocket,
  general_path_open_pocket))
SUBTYPE OF (machining_feature);

    its_hole: SET [0:?] OF general_path;

END_ENTITY;
```

its_hole:

Optional list of general_path entities which define the outline of the holes. This defines one or more parts of the pocket which are not cut during manufacturing of the pocket. When cutting the pocket the hole(s) is (are) cut simultaneously.

5.2.5 General_path_closed_pocket

Derived from the class general_path_pocket. A general_path_closed_pocket is a general_path_pocket that is surrounded by material everywhere along its circumference.

```
ENTITY general_path_closed_pocket
SUBTYPE OF (general_path_pocket);

    feature_boundary : general_path;

END_ENTITY;
```

feature_boundary:

The shape that describes the principal and secondary edges of the pocket. It is an enclosed area that has completely closed profile curves. The *general_path* entity specifies the volume required by a closed pocket.

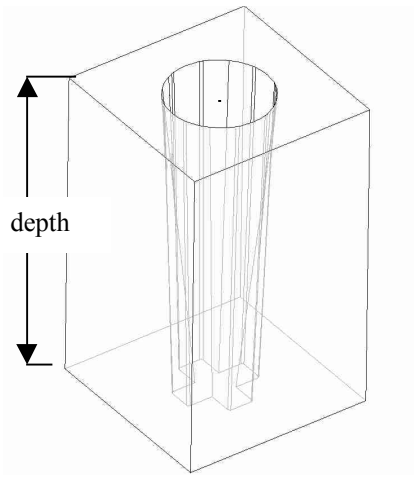


Fig. 3: General_path_closed_pocket

5.2.6 General_path_open_pocket

Derived from the class general_path_pocket class. A general_path_open_pocket is a general_path_pocket which is not a general_path_closed_pocket. The wall_boundary specifies the limit of the pocket from the open side.

```
ENTITY general_path_open_pocket
SUBTYPE OF (general_path_pocket);

    open_boundary : general_path;
    wall_boundary : OPTIONAL general_path;

END_ENTITY;
```

open_boundary:

The shape that describes the principal and secondary edges of the pocket. The *general_path* entity specifies the volume required by the pocket. When travelling along the curve base as defined by its sense, the material lies on the left side of the curve according to the *axis2_placement_3d* defined in the *machining_feature.*

wall_boundary:

Optional general_path entity which describes the shape of the pocket from the open side. Note that it is necessary to define this contour only if it differs from the side wall obtained by connecting the start and end points of the open_boundary with straight lines. See Figure 4.

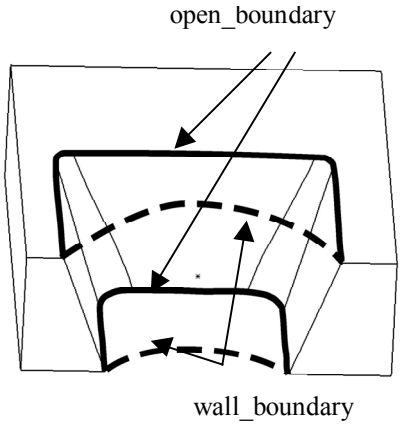


Fig. 4: General_path_open_pocket

5.2.7 Ruled surfaces

The description of the shape by ruled surfaces is used when normal vectors of the surfaces are necessary for the calculation of the wire offset, e.g. when the calculation of the offset in the two horizontal planes is not enough accurate.

The final shape of the cut is described by a list of ruled surface. The entity *region_surface_list,* which is a subtype of the entity *manufacturing_feature,* is used to describe the ordered and oriented list of surfaces. The material lies on the opposite side as the direction of the normal vector of the surface.

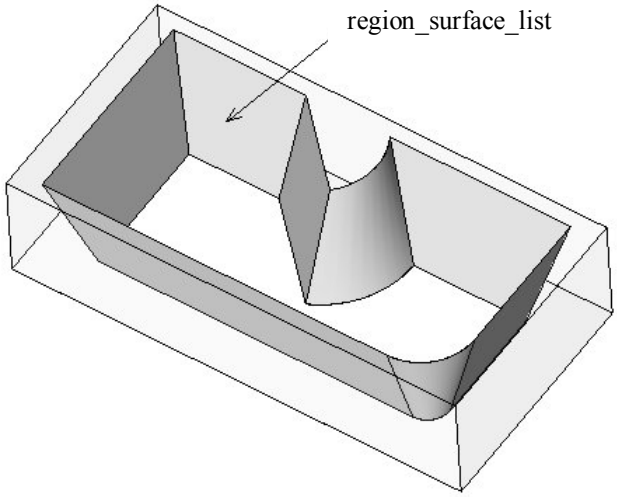


Fig. 5: Ruled surfaces

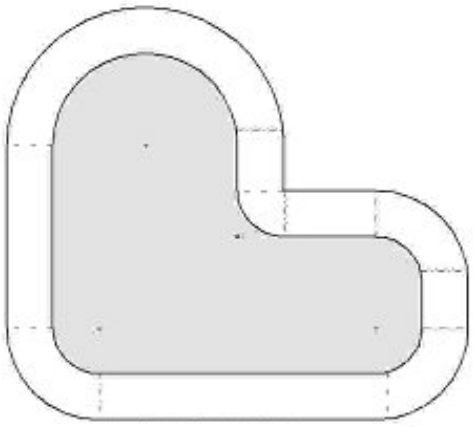
5.3 Additional types and entities

This section includes some further types and entities, which are used in the declaration of the machining features described above.

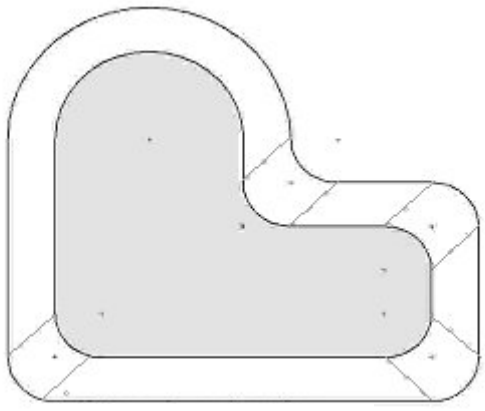
5.3.1 EDM_transition

The entity *EDM_transition* is used in the context of the *general_path* class declaration. In the case when the *general_path* class is defined by one curve (*feature_principal_boundary*) and a wall angle (*slope*) the transition has to be specified in order that the shapes of the angles are correctly defined.

```
TYPE EDM_transition = ENUMERATION OF(constant_radius, conical, sharp);
END_TYPE;
```



conical



constant_radius

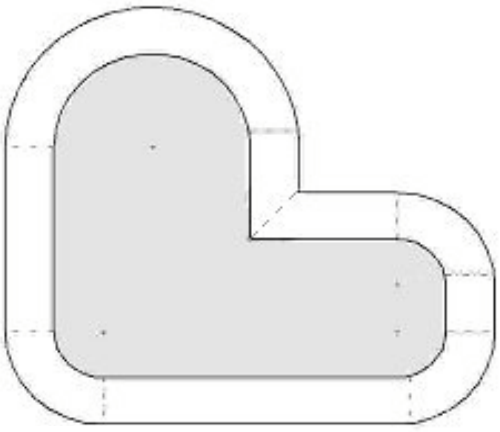


Fig. 6: EDM_transition types

The conical transition signifies that the secondary boundary curve is defined as the offset curve of the principal boundary curve. The constant radius transition defines the secondary curve by keeping the corner radius constant on the principal and secondary boundary curves (see Figure 6).

5.4 Machining operation for wire-EDM

In this section all machining operations and technology specific data needed for wire-EDM are introduced.

5.4.1 Wire_edm_machining_operation

The *wire_edm_machining_operation* classes define the machining process for a limited area of the workpiece, i.e. the contents of a machining workingstep. This entity is inherited by the *machining_workingstep* class defined in Part 10 of this standard. This class defines additional information needed by the wire-EDM machining. It is a subtype of entity *machining_operation* defined in Part 10 of this standard

```
ENTITY wire_edm_machining_operation
SUBTYPE OF (machining_operation);
    offset_length : OPTIONAL length_measure;
    approach:       OPTIONAL wire_edm_approach_retract_strategy;
    retract:        OPTIONAL  wire_edm_approach_retract_strategy;
    thread_point:   LIST OF [1:2] OF cartesian_point;
    cut_end_point:  OPTIONAL cartesian_point;
END_ENTITY;
```

offset_length:

Optional offset value defining the distance of the wire centre from the boundary curve of the feature. The controller based on the required tolerance and surface quality can determine this value. When travelling along the boundary curve of the feature based as defined by its sense, the offset curve lies on the left side of the curve. See Figure 7.

approach:

Defines the wire movement from the *thread_point* to the *cut_start_point*. If this is not defined the wire will start from the endpoint of the previous *machining_workingstep*.

retract:

Defines the wire movement from the *cut_end_point* to the *thread_point*. If this is not defined the wire will stop at the *cut_end_point*.

thread_point:

Defines the starting point of the cutting process. This point defines the threading hole's position where the wire can be threaded. In the case of four axis wire_EDM operation the inclined threading hole can be defined by two points. The first point is defined in the xy plane(z=0) while the second point is defined in the xy plane (z=-depth) of the feature coordinate system. See Figure 7.

cut_end_point:

Optional point defined on the principal boundary curve that specifies the end point of the cutting operation. If it is not defined the *cut_start_point* is considered as the end point. See Figure 6.

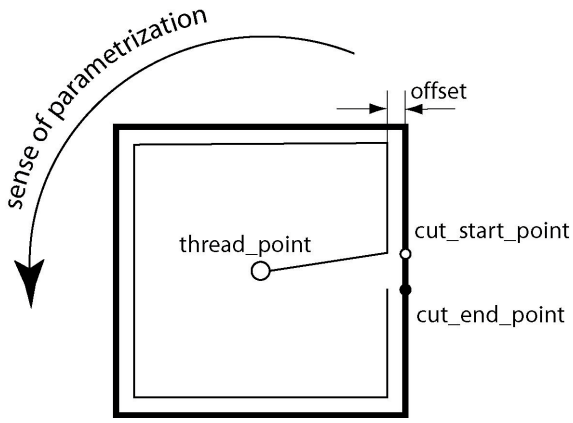


Fig. 7: wire_edm_machining_operation

5.4.2 Wire_edm_machining_strategy

The wire_edm_machining_strategy class specifies the strategy to be used when executing the operation. When it is specified it will modify the final offset toolpath generation method or gives a hint for an auxiliary operation like fixation or slug removal. It is a subtype of entity *machining_strategy* defined in Part 10 of this standard.

```
ENTITY wire_edm_machining_strategy
ABSTRACT SUPERTYPE OF (ONEOF (backmotion,cut_through,slug_removal))
SUBTYPE OF(machining_strategy);
END_ENTITY;
```

5.4.2.1 Backmotion

If the *backmotion* strategy is specified it defines that the wire tool should move backwards on the feature contour curve, starting from the *cut_end_point* towards the *cut_start_point* along the negative sense of parametrization. This strategy can be applied on features having closed contour curves. See Figure 8.

```
ENTITY backmotion
SUBTYPE OF(wire_edm_machining_strategy);
END_ENTITY;
```

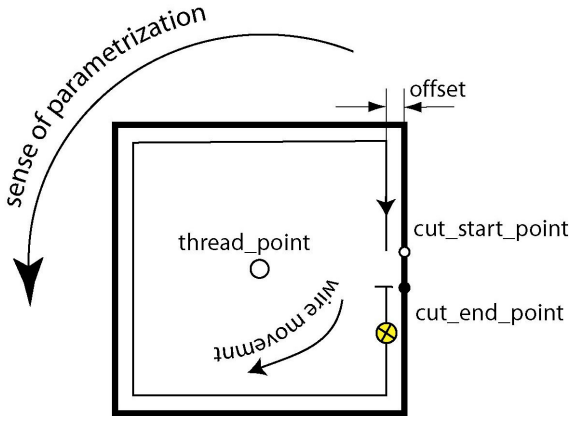


Fig. 8: backmotion strategy

5.4.2.2 Cut_through

The *cut_through* strategy specifies that the operation will cut a slug. This strategy signifies that the slug might have to be fixed before this operation starts in order to avoid the fall out of the slug causing a short circuit. The wire tool should start moving from the *cut_end_point* towards the *cut_start_point* along the positive sense of parametrization. This strategy can be applied to features having closed contour curves. See Figure 9.

```
ENTITY cut_through
SUBTYPE OF(wire_edm_machining_strategy);
END_ENTITY;
```

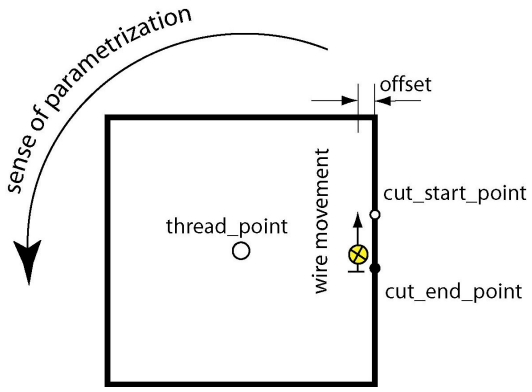


Fig. 9: cut_through strategy

5.4.2.3 Slug_removal

The *slug_removal* strategy defines that the operator should remove the slug. The machining_operation having *slug_removal* strategy usually follows an operation having *cut_through* strategy. The wire should not move during this operation. This strategy can be applied to features having closed contour curves. It might signify to the controller that the wire should be cut before this operation and re-threaded after.

```
ENTITY slug_removal
SUBTYPE OF(wire_edm_machining_strategy);
END_ENTITY;
```

5.4.3 Wire_edm_machine_functions

The entity describes the state of various functions of the machine, like coolant, etc. to be applied during the time span of an operation. It is a subtype of entity machine_functions defined in Part 10 of this standard.

```
ENTITY wire_edm_machine_functions
SUBTYPE OF (machine_functions);

    coolant :           BOOLEAN;
    coolant_pressure :  OPTIONAL pressure_measure;
    lower_nozzle :      BOOLEAN;
    upper_nozzle :      BOOLEAN;
    other_functions :   SET [0:?] OF property_parameter;

END_ENTITY;
```

coolant: If true, the coolant is activated.

coolant_pressure: Optional specification of the pressure of the coolant system. Only valid if coolant is true.

lower_nozzle: If true, lower nozzle is activated. Only valid if coolant is true.

upper_nozzle: If true, the upper nozzle is activated. Only valid if coolant is true.

other_functions: Optional list of other functions of generic type.

5.4.4 Wire_edm_technology

This entity defines the technological parameters of the wire EDM operation. It is a subtype of entity technology defined in Part 10 of this standard. Since the number of technology parameters are machine dependent the technology will be derived from independent parameters. Most of the wire EDM machines use an expert system or technology tables to define the spark generator parameters from the following data:

- material of the wire (*its_wire_material* of *wire_tool* entity)
  - diameter of the wire (*its_diameter* of *wire_tool* entity)
  - material of the workpiece (*its_material* of *workpiece* entity)
  - cutting height (*its_geometry* of *workpiece* entity)
  - final surface quality (roughness)

```
ENTITY wire_edm_technology
SUBTYPE OF (technology);
    small_corner_strategy :       OPTIONAL BOOLEAN;
    other_generator_parameters :  OPTIONAL SET [0:?] OF property_parameter;
END_ENTITY;
```

small_corner_strategy : If true, the special strategy to handle the small radius corners is activated.

other_generator_parameters: Set of other property parameters of the generator of generic type.

5.4.5 Wire_edm_approach_retract_strategy

Base class for the approach and retract strategy. All approach and retract strategies are defined relative to the start or end point of the cutting operation. The start point of the approach or end point of the retract movement are defined to be the thread point of the operation.

```
ENTITY wire_edm_approach_retract_strategy
ABSTRACT SUPERTYPE OF (ONEOF (along_path_strategy, linear_strategy,
  arc_strategy));
    its_technology: OPTIONAL wire_edm_technology;
    technology_switch_point: OPTIONAL LIST OF [1:2] OF cartesian_point;
END_ENTITY;
```

its_technology: Optional technology setting for the approach/retract path. If it is not set the technology specified for the main cutting is used.

technology_switch_point: Specifes the point where the technology specified for the approach/retract movement should switch to the technology of the main cut. Only valid if its_technology is specified.

5.4.5.1 Along_path_strategy

The along path strategy specifies the movement of the wire from the thread_point (to the thread_point) to the start point (from the end point) of the cutting operation with a list of tool paths. This class can describe an arbitrary approach or retract strategy.

```
ENTITY along_path_strategy
SUBTYPE OF (wire_edm_approach_retract_strategy);
    path:           toolpath_list;
END_ENTITY;
```

path: Specifies the path of the wire

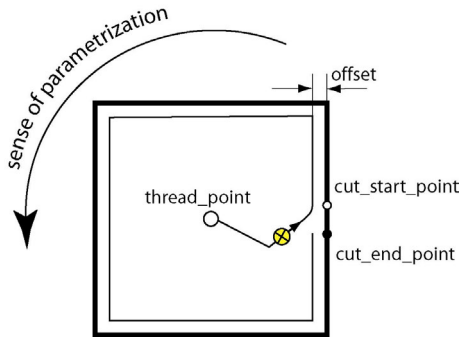


Fig. 10: along_path_strategy

5.4.5.2 Linear_strategy

The approach/ retract path connects the thread point with the start/end point of the cutting operation with a linear segment. See Figure 11.

```
ENTITY linear_strategy
SUBTYPE OF (wire_edm_approach_retract_strategy);
END_ENTITY;
```

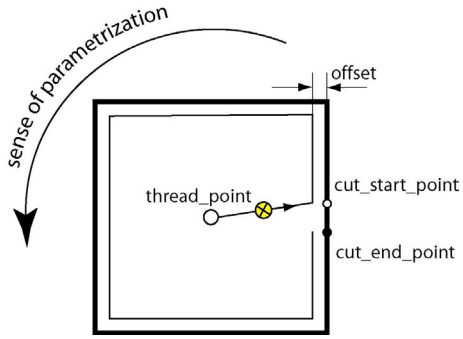


Fig. 11: linear_strategy

5.4.5.3 Arc_strategy

The approach/ retract path connects the thread point with the start/end point of the cutting operation with a linear-arc segment. The arc segment is tangential to the main cut's curve. See Figure 12.

```
ENTITY arc_strategy
SUBTYPE OF (wire_edm_approach_retract_strategy);

radius:     positive_length_measure;
END_ENTITY;
```

radius: The radius of the approach/retract movement.

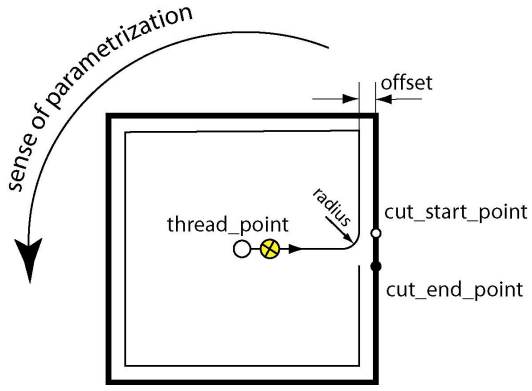


Fig. 12: arc_strategy

5.4.6 Wire_tool

This class describes the properties of the wire used in wire_EDM machining. It is a subtype of entity *machining_tool* defined in Part 10 of this standard.

```
ENTITY wire_tool
SUBTYPE OF (machining_tool);
    its_wire_material : material;
    its_diameter :      length_measure;
    its_tension :       OPTIONAL tension_measure;
    its_speed :         OPTIONAL speed_measure;
    other_parameters :  SET [0:?] OF property_parameter;
END_ENTITY;
```

its_wire_material: The material of the wire.

its_diameter: The diameter of the wire.

its_tension: Optional tension value applied to the wire .

its_speed: Optional speed value specifying the wire feeding velocity. Informative

other_parameters: Optional set of parameters of generic type.

Annex A
(normative)

EXPRESS listing

The following EXPRESS is the whole schema of Part13/1.Process Data for wire-EDM

```
SCHEMA wire_edm_schema;
(*
Version 5 of Feb 28, 2003
Author: Gabor Erdos  <gabor.erdos@epfl.ch>
Modified by: Willy Maeder  <wmaeder@cadcamation.ch>
             Jacques Richard  <i-tech@eig.unige.ch>
*)

REFERENCE FROM machining_schema (*ISO14649-10*)
        (bounded_curve,
        cartesian_point,
        direction,
        identifier,
        label,
        length_measure,
        machine_functions,
        machining_operation,
        machining_feature,
        machining_tool,
        material,
        pressure_measure,
        property_parameter,
        speed_measure,
        plane_angle_measure,
        technology,
        machining_strategy,
        toolpath_list
        );

(* ********************************************************** *)
(* Type definitions                                           *)
(* ********************************************************** *)

TYPE EDM_transition = ENUMERATION OF(constant_radius, conical, sharp);
END_TYPE;

TYPE tension_measure = REAL;
END_TYPE;

(* ********************************************************** *)
(* Machining feature                                          *)
(* ********************************************************** *)

ENTITY general_path
SUPERTYPE OF (ONEOF(general_single_path, general_twin_path))
SUBTYPE OF (machining_feature);
END_ENTITY;

ENTITY general_single_path
SUBTYPE OF (general_path);
```

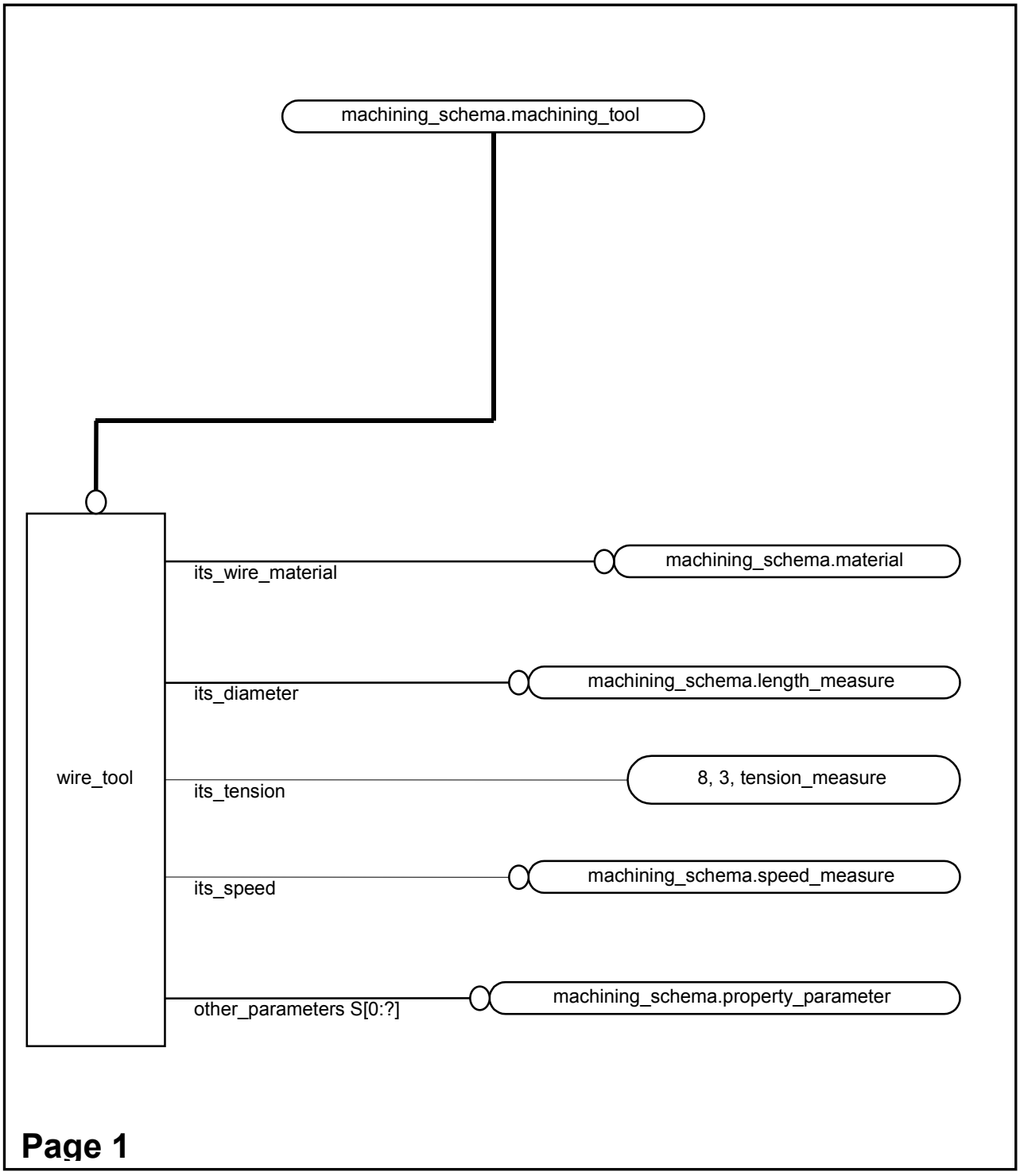
```
    feature_principal_boundary:       bounded_curve;
    slope:                            OPTIONAL plane_angle_measure;
    transition_types:                 OPTIONAL EDM_transition;
END_ENTITY;

ENTITY general_twin_path
SUBTYPE OF (general_path);
    feature_principal_boundary:       bounded_curve;
    feature_secondary_boundary:       bounded_curve;
END_ENTITY;

ENTITY general_path_pocket
ABSTRACT SUPERTYPE OF (ONEOF(general_path_closed_pocket, general_path_open_pocket))
SUBTYPE OF (machining_feature);

    its_hole : SET [0:?] OF general_path;

END_ENTITY;

ENTITY general_path_closed_pocket
SUBTYPE OF (general_path_pocket);

    feature_boundary : general_path;

END_ENTITY;

ENTITY general_path_open_pocket
SUBTYPE OF (general_path_pocket);

    open_boundary : general_path;
    wall_boundary : OPTIONAL general_path;

END_ENTITY;


(* ********************************************************** *)
(* Wire tool    *)
(* ********************************************************** *)
ENTITY wire_tool
SUBTYPE OF (machining_tool);
    its_wire_material : material;
    its_diameter :      length_measure;
    its_tension :       OPTIONAL tension_measure;
    its_speed :         OPTIONAL speed_measure;
    other_parameters :  SET [0:?] OF property_parameter;
END_ENTITY;

(* ********************************************************** *)
(* Wire EDM Machining strategy                                *)
(* ********************************************************** *)

ENTITY wire_edm_machining_strategy
ABSTRACT SUPERTYPE OF (ONEOF (backmotion,cut_through,slug_removal))
SUBTYPE OF(machining_strategy);
END_ENTITY;

ENTITY backmotion
SUBTYPE OF(wire_edm_machining_strategy);
END_ENTITY;
```

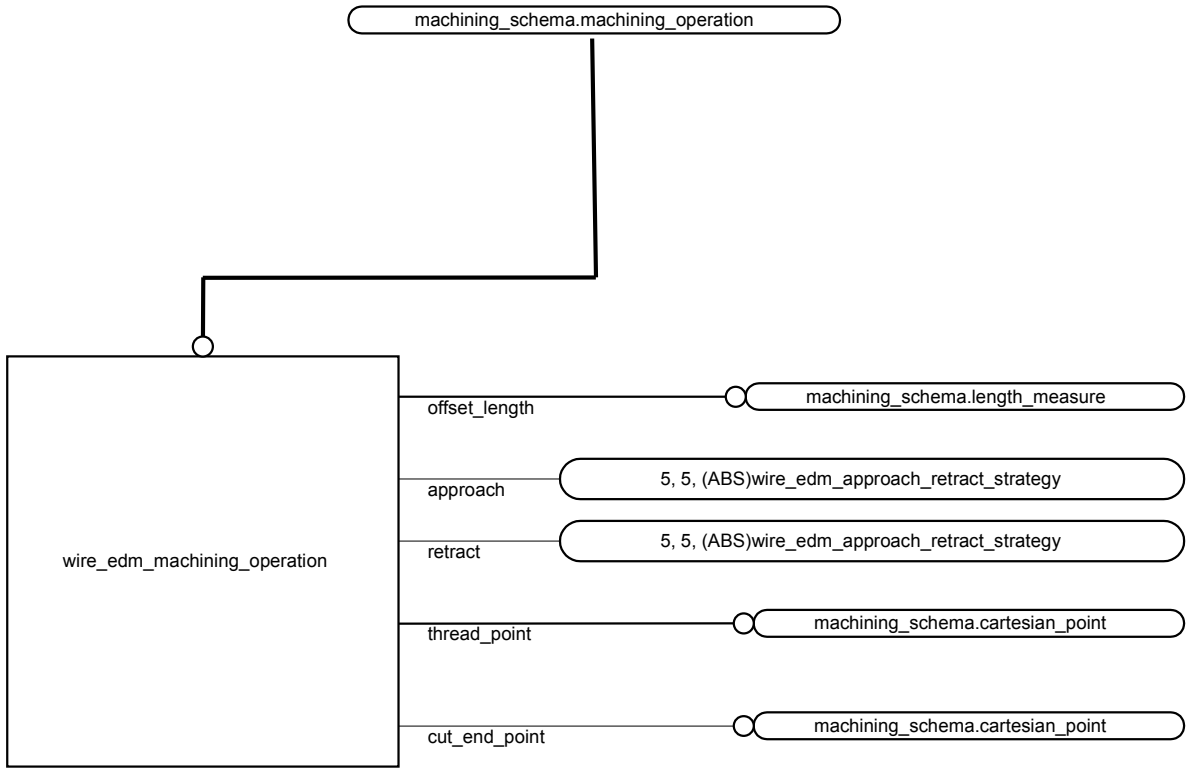
```
ENTITY cut_through
SUBTYPE OF(wire_edm_machining_strategy);
END_ENTITY;

ENTITY slug_removal
SUBTYPE OF(wire_edm_machining_strategy);
END_ENTITY;

(* ********************************************************** *)
(* wire_edm_approach_retract_strategy                         *)
(* ********************************************************** *)
ENTITY wire_edm_approach_retract_strategy
ABSTRACT SUPERTYPE OF (ONEOF (along_path_strategy, linear_strategy, arc_strategy));
    its_technology:          OPTIONAL wire_edm_technology;
    technology_switch_point: OPTIONAL LIST OF [1:2] OF cartesian_point;
END_ENTITY;

ENTITY along_path_strategy
SUBTYPE OF (wire_edm_approach_retract_strategy);
    path:           toolpath_list;
END_ENTITY;

ENTITY linear_strategy
SUBTYPE OF (wire_edm_approach_retract_strategy);
END_ENTITY;

ENTITY arc_strategy
SUBTYPE OF (wire_edm_approach_retract_strategy);
    radius:         positive_length_measure;
END_ENTITY;

(* ********************************************************** *)
(* Wire EDM operation   *)
(* ********************************************************** *)

ENTITY wire_edm_machining_operation
SUBTYPE OF (machining_operation);
    offset_length : length_measure;
    approach:       OPTIONAL wire_edm_approach_retract_strategy;
    retract:        OPTIONAL  wire_edm_approach_retract_strategy;
    thread_point:   cartesian_point;
    cut_end_point:  OPTIONAL cartesian_point;
END_ENTITY;


(* ********************************************************** *)
(* Wire EDM technology   *)
(* ********************************************************** *)

ENTITY wire_edm_technology
SUBTYPE OF (technology);
  small_corner_strategy :       OPTIONAL BOOLEAN;
  other_generator_parameters :  OPTIONAL SET [0:?] OF property_parameter;
END_ENTITY;

(* ********************************************************** *)
(* Wire EDM technology  machine functions *)
(* ********************************************************** *)

ENTITY wire_edm_machine_functions
SUBTYPE OF (machine_functions);
```

```
    coolant :           BOOLEAN;
    coolant_pressure :  OPTIONAL pressure_measure;
    lower_nozzle :      BOOLEAN;
    upper_nozzle :      BOOLEAN;
    other_functions :   SET [0:?] OF property_parameter;
END_ENTITY;

END_SCHEMA; (*wire_edm_schema *)
```

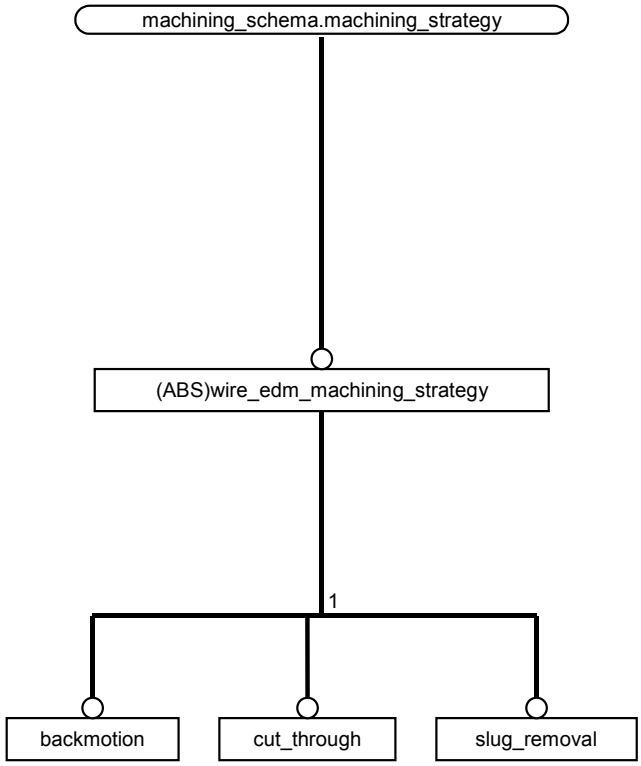
Annex B
(informative)

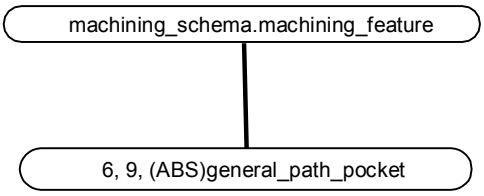
EXPRESS-G



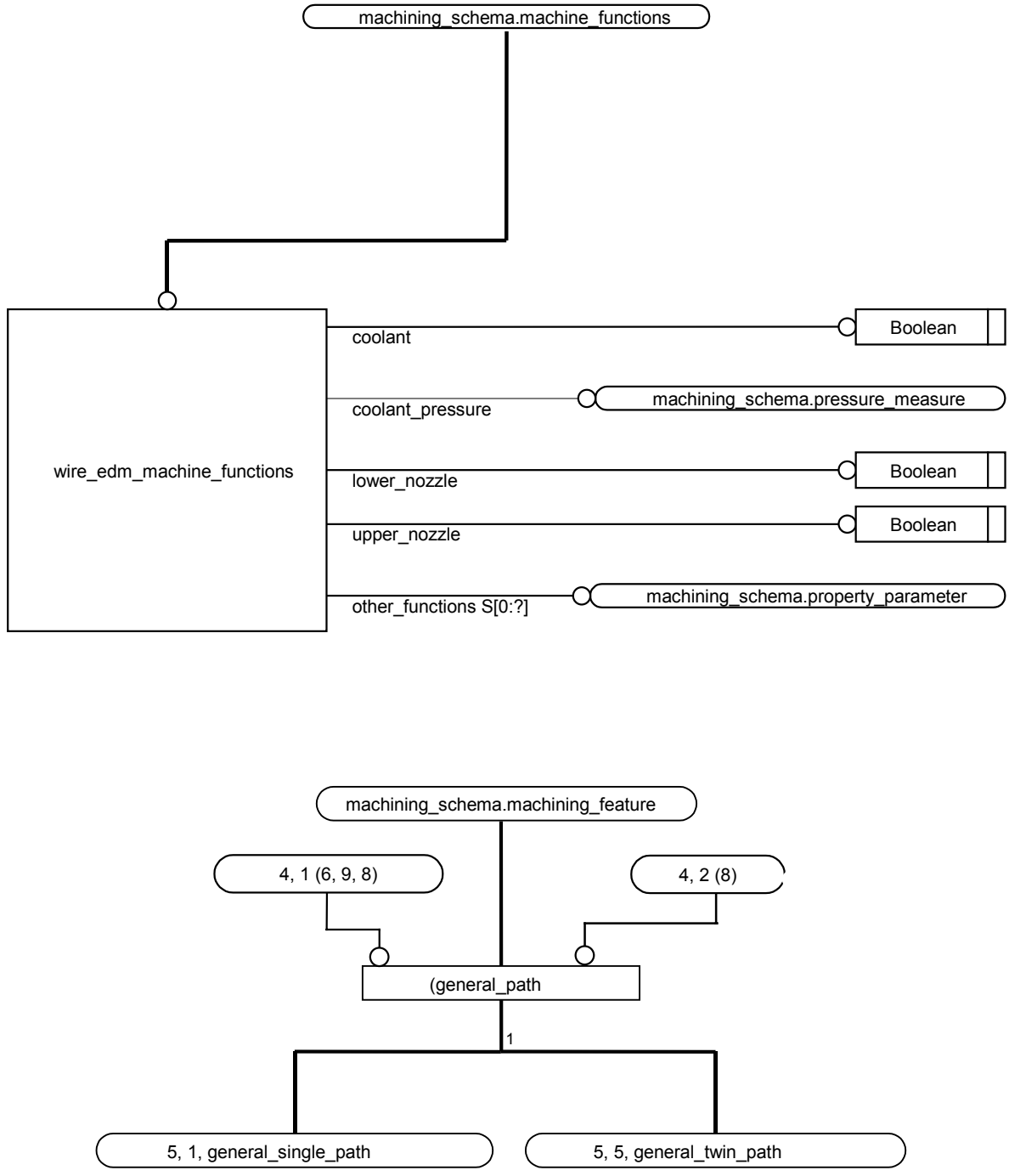


Page 2

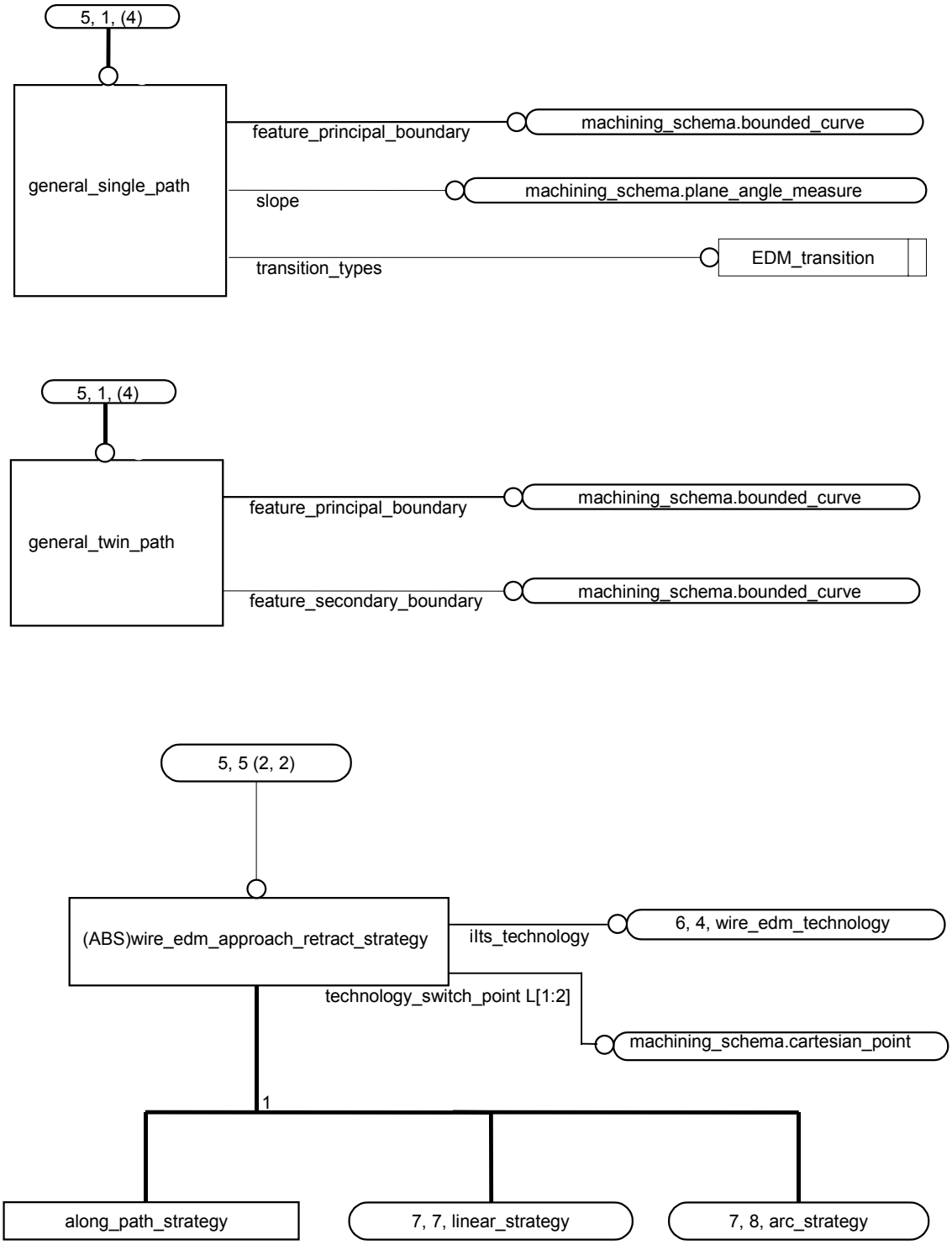


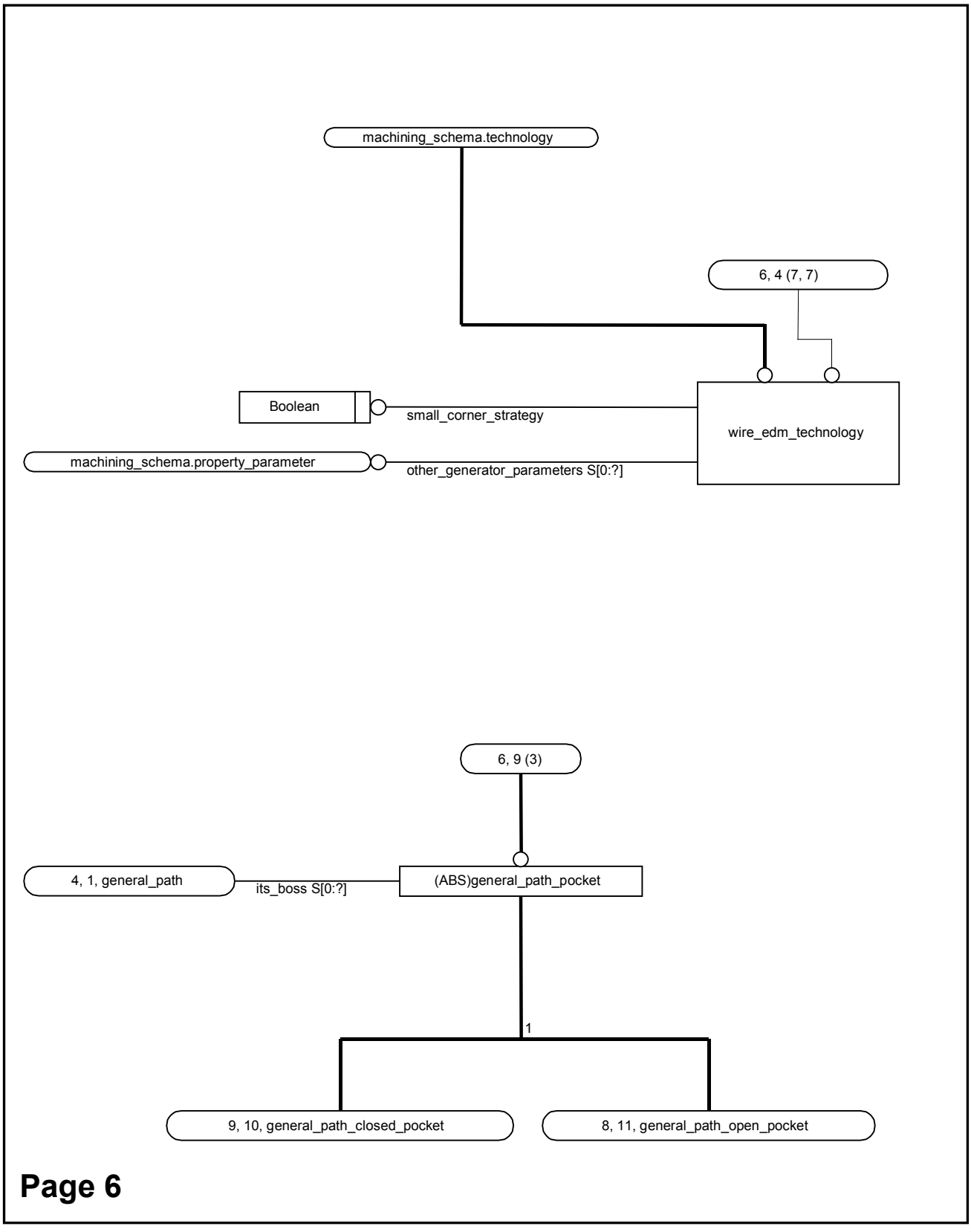


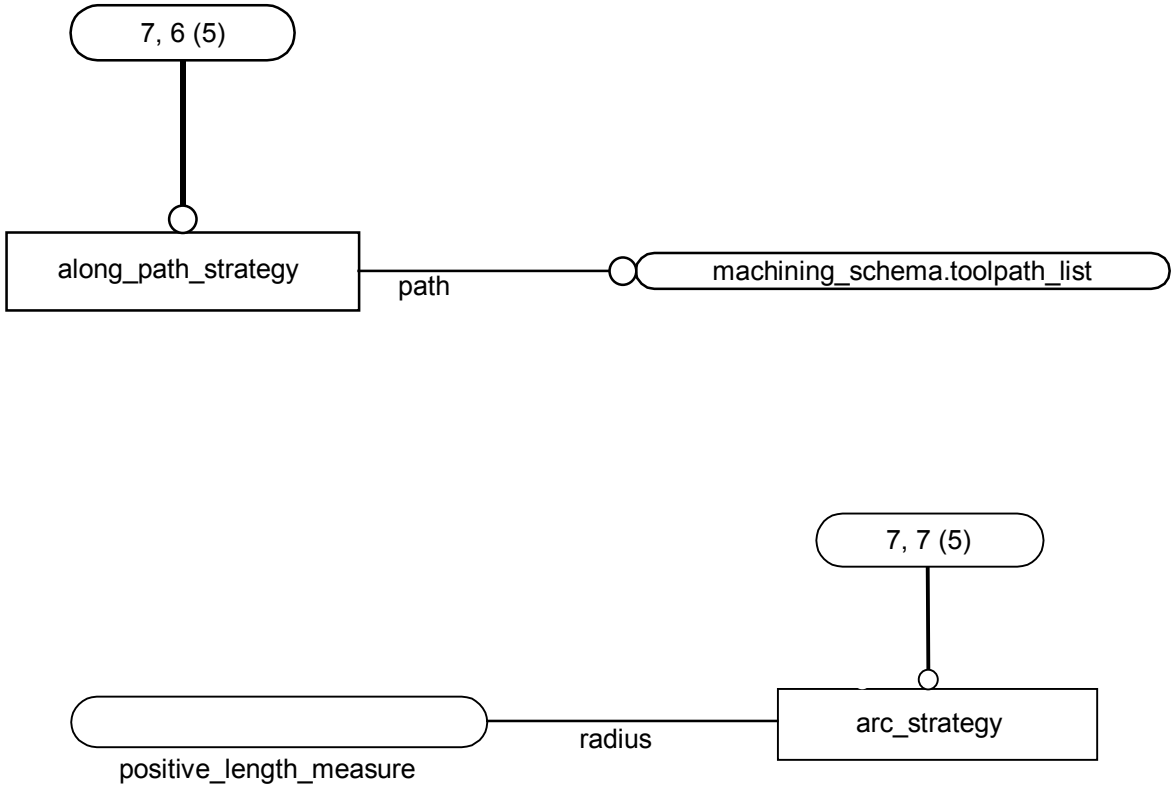
Page 3



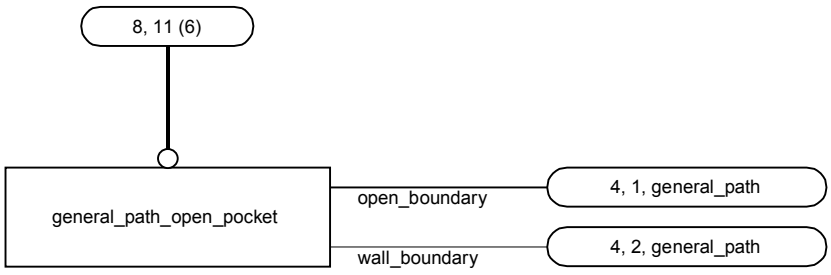
Page 4

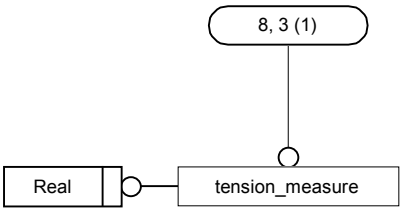




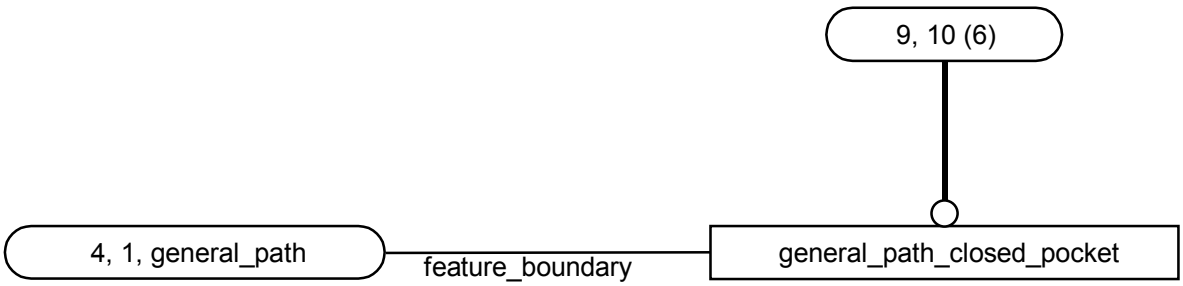


Page 7





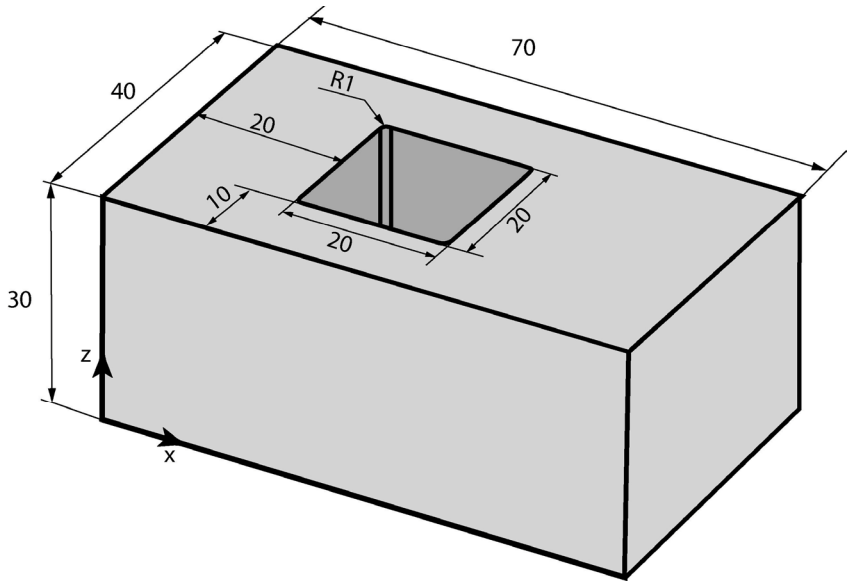
Page 8



Page 9

Annex C
(informative)

Simple wire-EDM example 1



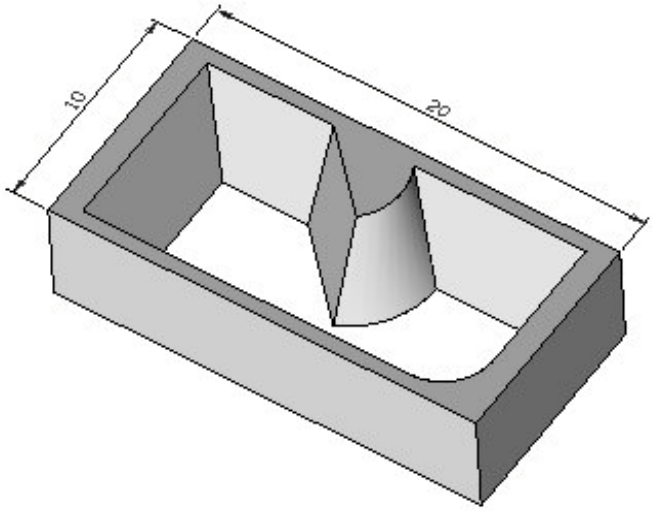

```
ISO-10303-21;
HEADER;

FILE_DESCRIPTION(
/* DESCRIPTION */ ('THIS FILE CONTAINS THE STEP-NC FILE OF A WIRE-EDM MANUFACTURING'),
/* IMPLEMENTATION_LEVEL */ '2;1');

FILE_NAME(
/* NAME */ 'WIRE_EDM_EXAMPLE',
/* TIME_STAMP */ '2003-02-28T14:17:26+01:00',
/* AUTHOR */ ('GABOR ERDOS'),
/* ORGANIZATION */ ('EPFL-ICAP-LICP, LAUSANNE SWITZERLAND'),

FILE_SCHEMA (('MACHINING_SCHEMA','WIRE_EDM_SCHEMA'));
ENDSEC;

DATA;
#10=ARC_STRATEGY($,$,1.);
#11=SLUG_REMOVAL();
#12=CUT_THROUGH();
#13=CIRCLE('CIRCLE_009',#83,1.);
#14=CIRCLE('CIRCLE_010',#84,1.);
#15=CIRCLE('CIRCLE_011',#85,1.);
#16=CIRCLE('CIRCLE_012',#86,1.);
#17=TRIMMED_CURVE('TRIMMED_CURVE_009',#13,(#55),(#56),.T.,$);
#18=TRIMMED_CURVE('TRIMMED_CURVE_010',#14,(#60),(#61),.T.,$);
#19=TRIMMED_CURVE('TRIMMED_CURVE_011',#15,(#65),(#66),.T.,$);
#20=TRIMMED_CURVE('TRIMMED_CURVE_012',#16,(#70),(#71),.T.,$);
```

```
#21=POLYLINE('Pline_009',(#52,#53));
#22=POLYLINE('Pline_010',(#57,#58));
#23=POLYLINE('Pline_011',(#62,#63));
#24=POLYLINE('Pline_012',(#67,#68));
#25=COMPOSITE_CURVE_SEGMENT($,.T.,#21);
#26=COMPOSITE_CURVE_SEGMENT($,.T.,#17);
#27=COMPOSITE_CURVE_SEGMENT($,.T.,#22);
#28=COMPOSITE_CURVE_SEGMENT($,.T.,#18);
#29=COMPOSITE_CURVE_SEGMENT($,.T.,#23);
#30=COMPOSITE_CURVE_SEGMENT($,.T.,#19);
#31=COMPOSITE_CURVE_SEGMENT($,.T.,#24);
#32=COMPOSITE_CURVE_SEGMENT($,.T.,#20);
#33=PROJECT('wire_edm_example',#39,(#34));
#34=WORKPIECE('workpiece',#35,0.005,$,$,#38,$);
#35=MATERIAL('ST221','Cold die Steel',$);
#36=MATERIAL('st234','Cobra Cut A',$);
#37=MATERIAL('','',$);
#38=BLOCK('Block_001',#82,40.,30.,70.);
#39=WORKPLAN('wp',(#40,#41,#42,#43,#44),#115,#117);
#40=MACHINING_WORKINGSTEP('ws_01',$,#89,#91);
#41=MACHINING_WORKINGSTEP('ws_02',$,#89,#92);
#42=MACHINING_WORKINGSTEP('ws_03',$,#89,#93);
#43=MACHINING_WORKINGSTEP('ws_04',$,#89,#94);
#44=MACHINING_WORKINGSTEP('ws_05',$,#89,#95);
#45=TOLERANCED_LENGTH_MEASURE(30.,$,0.,0.);
#46=TOLERANCED_LENGTH_MEASURE(1.,$,0.,0.);
#47=CARTESIAN_POINT('featOrigin',(20.,10.,30.));
#48=CARTESIAN_POINT('workpiece_bbBox_Location',(-10.,-15.,0.));
#49=CARTESIAN_POINT('cut_start_pt',(10.,0.,0.));
#50=CARTESIAN_POINT('thread_pt',(10.,10.,0.));
#51=CARTESIAN_POINT('cut_end_pt',(9.,0.,0.));
#52=CARTESIAN_POINT('Cartesian_Point_001_of_Pline_009',(1.,20.,0.));
#53=CARTESIAN_POINT('Cartesian_Point_002_of_Pline_009',(19.,20.,0.));
#54=CARTESIAN_POINT('Centre_Point_of_Circle_009',(19.,19.,0.));
#55=CARTESIAN_POINT('Start_Point_of_TrimmedCurve_009',(19.,20.,0.));
#56=CARTESIAN_POINT('End_Point_of_TrimmedCurve_009',(20.,19.,0.));
#57=CARTESIAN_POINT('Cartesian_Point_001_of_Pline_010',(20.,19.,0.));
#58=CARTESIAN_POINT('Cartesian_Point_002_of_Pline_010',(20.,1.,0.));
#59=CARTESIAN_POINT('Centre_Point_of_Circle_010',(19.,1.,0.));
#60=CARTESIAN_POINT('Start_Point_of_TrimmedCurve_010',(20.,1.,0.));
#61=CARTESIAN_POINT('End_Point_of_TrimmedCurve_010',(19.,0.,0.));
#62=CARTESIAN_POINT('Cartesian_Point_001_of_Pline_011',(19.,0.,0.));
#63=CARTESIAN_POINT('Cartesian_Point_002_of_Pline_011',(1.,0.,0.));
#64=CARTESIAN_POINT('Centre_Point_of_Circle_011',(1.,1.,0.));
#65=CARTESIAN_POINT('Start_Point_of_TrimmedCurve_011',(1.,0.,0.));
#66=CARTESIAN_POINT('End_Point_of_TrimmedCurve_011',(0.,1.,0.));
#67=CARTESIAN_POINT('Cartesian_Point_001_of_Pline_012',(0.,1.,0.));
#68=CARTESIAN_POINT('Cartesian_Point_002_of_Pline_012',(0.,19.,0.));
#69=CARTESIAN_POINT('Centre_Point_of_Circle_012',(1.,19.,0.));
#70=CARTESIAN_POINT('Start_Point_of_TrimmedCurve_012',(0.,19.,0.));
#71=CARTESIAN_POINT('End_Point_of_TrimmedCurve_012',(1.,20.,0.));
#72=CARTESIAN_POINT('SO1',(0.,0.,20.));
#73=DIRECTION('featZAxis',(0.,0.,1.));
#74=DIRECTION('featXAxis',(1.,0.,0.));
#75=DIRECTION('Axis_Z_of_Block_001',(1.,0,0));
#76=DIRECTION('Axis_X_of_Block_001',(0,1.,0.));
#77=DIRECTION('Z_axis_of_Circle_009',(0.,0.,-1.));
```

```
#78=DIRECTION('Z_AXIS_OF_CIRCLE_010',(0.,0.,-1.));
#79=DIRECTION('Z_AXIS_OF_CIRCLE_011',(0.,0.,-1.));
#80=DIRECTION('Z_AXIS_OF_CIRCLE_012',(0.,0.,-1.));
#81=AXIS2_PLACEMENT_3D('FEAT_FRAME',#47,#73,#74);
#82=AXIS2_PLACEMENT_3D('POSITION_OF_BLOCK_001',#48,#75,#76);
#83=AXIS2_PLACEMENT_3D('POSTION_OF_CIRCLE_009',#54,#77,$);
#84=AXIS2_PLACEMENT_3D('POSTION_OF_CIRCLE_010',#59,#78,$);
#85=AXIS2_PLACEMENT_3D('POSTION_OF_CIRCLE_011',#64,#79,$);
#86=AXIS2_PLACEMENT_3D('POSTION_OF_CIRCLE_012',#69,#80,$);
#87=AXIS2_PLACEMENT_3D('SECPLANE_FRAME',#72,$,$);
#88=COMPOSITE_CURVE('POCKET',(#25,#26,#27,#28,#29,#30,#31,#32),.F.);
#89=GENERAL_OUTSIDE_PROFILE('WORK1',#34,(#91,#92,#93,#94,#95),#81,#45,#46,$,$,$,#88,$,$);
#90=WIRE_TOOL('WIRE1',#36,0.1,0.2,160.,$);
#91=WIRE_EDM_MACHINING_OPERATION($,$,'ROUGHING',$,10.,#49,#90,#100,#114,0.01,#96,#97,#50,#51);
#92=WIRE_EDM_MACHINING_OPERATION($,$,'CUT_THROUGH',#12,10.,#49,#90,#99,$,0.01,$,$,#50,#51);
#93=WIRE_EDM_MACHINING_OPERATION($,$,'SLUG_REMOVAL',#11,10.,#49,#90,#99,$,1.,$,$,#50,#51);
#94=WIRE_EDM_MACHINING_OPERATION($,$,'FINISHING',$,10.,#49,#90,#98,#114,0.1,$,$,#50,#49);
#95=WIRE_EDM_MACHINING_OPERATION($,$,'SURFACE_FINISHING',$,10.,#49,#90,#101,#114,0.01,$,#10,#50,#49);
#96=LINEAR_STRATEGY($,$);
#97=LINEAR_STRATEGY($,$);
#98=WIRE_EDM_TECHNOLOGY(0.,.TCP.,$,$,#103,#108,#109,#110));
#99=WIRE_EDM_TECHNOLOGY(0.,.TCP.,$,$,());
#100=WIRE_EDM_TECHNOLOGY(0.,$,$,$,(#102,#105,#107,#106));
#101=WIRE_EDM_TECHNOLOGY(0.,.TCP.,$,$,(#104,#111,#112,#113));
#102=DESCRIPTIVE_PARAMETER('ID','Q2');
#103=DESCRIPTIVE_PARAMETER('ID','Q3');
#104=DESCRIPTIVE_PARAMETER('ID','Q8');
#105=NUMERIC_PARAMETER('RA',1.8,'MICM');
#106=NUMERIC_PARAMETER('TKM',10.,'MICM');
#107=NUMERIC_PARAMETER('TE',10.,'MICM');
#108=NUMERIC_PARAMETER('RA',0.8,'MICM');
#109=NUMERIC_PARAMETER('TKM',6.,'MICM');
#110=NUMERIC_PARAMETER('TE',6.,'MICM');
#111=NUMERIC_PARAMETER('RA',0.4,'MICM');
#112=NUMERIC_PARAMETER('TKM',3.,'MICM');
#113=NUMERIC_PARAMETER('TE',3.,'MICM');
#114=WIRE_EDM_MACHINE_FUNCTIONS(.T.,$,.T.,.F.,());
#115=CHANNEL('CHANEL_1');
#116=PLANE($,#87);
#117=SETUP('SETUP',$,#116,());
ENDSEC;
END-ISO-10303-21;
```

Annex D
(informative)

Simple wire-EDM example 2




```
ISO-10303-21;
HEADER;

FILE_DESCRIPTION(
/* DESCRIPTION */ (
'THIS FILE CONTAINS THE STEP-NC FILE OF A WIRE-EDM MANUFACTURING'),
/* IMPLEMENTATION_LEVEL */ '2;1');

FILE_NAME(
/* NAME */ 'SAMPLE WITH RULED SURFACESR',
/* TIME_STAMP */ '2003-02-28T13:09:21+02:00',
/* AUTHOR */ ('WILLY MAEDER'),
/* ORGANIZATION */ ('CADCAMATION SA, ONEX-GENEVA, SWITZERLAND'),
/* PREPROCESSOR_VERSION */ 'ST-DEVELOPER V8',
/* ORIGINATING_SYSTEM */ '',
/* AUTHORISATION */ '');

FILE_SCHEMA (('MACHINING_SCHEMA','WIRE_EDM_SCHEMA'));
ENDSEC;

DATA;
#2=CARTESIAN_POINT('',(-50.0,30.0,20.0));
#3=CARTESIAN_POINT('',(0.0,30.0,20.0));
#4=CARTESIAN_POINT('',(-46.0,26.0,0.0));
#5=CARTESIAN_POINT('',(-3.762857145299775,26.0,0.0));
#6=B_SPLINE_SURFACE_WITH_KNOTS('',1,1,((#2,#4),(#3,#5)),.UNSPECIFIED.,.F.,.F.,.U.,(2,2),(2,2),
(-50.0,0.0),(0.0,1.0),.UNSPECIFIED.);
#7=CARTESIAN_POINT('',(8.786796564403542,8.786796564403606,20.0));
#8=CARTESIAN_POINT('',(6.935595340170893,10.637997788636259,20.0));
#9=CARTESIAN_POINT('',(3.720779449950078,14.827639903211258,20.0));
#10=CARTESIAN_POINT('',(0.689306233719310,22.146263655806401,20.0));
```

```
#11=CARTESIAN_POINT('',(-2.645847E-015,27.382006122008541,20.0));
#12=CARTESIAN_POINT('',(0.0,30.0,20.0));
#13=CARTESIAN_POINT('',(8.955855216991068,3.295244424414630,0.0));
#14=CARTESIAN_POINT('',(6.625202790118829,5.131868128801239,0.0));
#15=CARTESIAN_POINT('',(2.447738076337918,9.418653244101920,0.0));
#16=CARTESIAN_POINT('',(-1.940637859416365,17.251877607654784,0.0));
#17=CARTESIAN_POINT('',(-3.414762067389234,23.053144355453618,0.0));
#18=CARTESIAN_POINT('',(-3.763871791715616,25.999879797732245,0.0));
#19=B_SPLINE_SURFACE_WITH_KNOTS('',3,1,((#7,#13),(#8,#14),(#9,#15),(#10,#16),(#11,#17),(#12,#18)),
.UNSPECIFIED.,.F.,.F.,.U.,(4,1,1,4),(2,2),(-3.926990816987240,-3.665191429188091,-3.403392041388941,
-3.141592653589792),(0.0,1.0),.UNSPECIFIED.);
#20=CARTESIAN_POINT('',(8.786796564403602,8.786796564403602,20.0));
#21=CARTESIAN_POINT('',(30.0,30.0,20.0));
#22=CARTESIAN_POINT('',(8.955855216991086,3.295244424414616,0.0));
#23=CARTESIAN_POINT('',(31.656854249492397,26.0,0.0));
#24=B_SPLINE_SURFACE_WITH_KNOTS('',1,1,((#20,#22),(#21,#23)),.UNSPECIFIED.,.F.,.F.,.U.,(2,2),(2,2),
(-30.0,0.0),(0.0,1.0),.UNSPECIFIED.);
#25=CARTESIAN_POINT('',(30.0,30.0,20.0));
#26=CARTESIAN_POINT('',(70.0,30.0,20.0));
#27=CARTESIAN_POINT('',(31.656854249492397,26.0,0.0));
#28=CARTESIAN_POINT('',(66.0,26.0,0.0));
#29=B_SPLINE_SURFACE_WITH_KNOTS('',1,1,((#25,#27),(#26,#28)),.UNSPECIFIED.,.F.,.F.,.U.,(2,2),(2,2),
(-40.0,0.0),(0.0,1.0),.UNSPECIFIED.);
#30=CARTESIAN_POINT('',(70.0,30.0,20.0));
#31=CARTESIAN_POINT('',(70.0,-50.0,20.0));
#32=CARTESIAN_POINT('',(66.0,26.0,0.0));
#33=CARTESIAN_POINT('',(66.0,-46.0,0.0));
#34=B_SPLINE_SURFACE_WITH_KNOTS('',1,1,((#30,#32),(#31,#33)),.UNSPECIFIED.,.F.,.F.,.U.,(2,2),(2,2),
(-80.0,0.0),(0.0,1.0),.UNSPECIFIED.);
#35=CARTESIAN_POINT('',(70.0,-50.0,20.0));
#36=CARTESIAN_POINT('',(-20.0,-50.0,20.0));
#37=CARTESIAN_POINT('',(66.0,-46.0,0.0));
#38=CARTESIAN_POINT('',(-20.0,-46.0,0.0));
#39=B_SPLINE_SURFACE_WITH_KNOTS('',1,1,((#35,#37),(#36,#38)),.UNSPECIFIED.,.F.,.F.,.U.,(2,2),(2,2),
(-90.0,0.0),(0.0,1.0),.UNSPECIFIED.);
#40=CARTESIAN_POINT('',(-50.0,-20.0,20.0));
#41=CARTESIAN_POINT('',(-50.0,30.0,20.0));
#42=CARTESIAN_POINT('',(-46.0,-20.0,0.0));
#43=CARTESIAN_POINT('',(-46.0,26.0,0.0));
#44=B_SPLINE_SURFACE_WITH_KNOTS('',1,1,((#40,#42),(#41,#43)),.UNSPECIFIED.,.F.,.F.,.U.,(2,2),(2,2),
(-50.0,0.0),(0.0,1.0),.UNSPECIFIED.);
#45=CARTESIAN_POINT('',(-50.0,-20.0,20.0));
#46=CARTESIAN_POINT('',(-50.0,-22.243994752564106,20.0));
#47=CARTESIAN_POINT('',(-49.494332216275893,-26.731874096003608,20.0));
#48=CARTESIAN_POINT('',(-47.256861435762637,-33.126216647277602,20.0));
#49=CARTESIAN_POINT('',(-43.652618263109133,-38.862332356046295,20.0));
#50=CARTESIAN_POINT('',(-38.862332356046380,-43.652618263109090,20.0));
#51=CARTESIAN_POINT('',(-33.126216647277673,-47.256861435762580,20.0));
#52=CARTESIAN_POINT('',(-26.731874096003700,-49.494332216275879,20.0));
#53=CARTESIAN_POINT('',(-22.243994752564198,-50.0,20.0));
#54=CARTESIAN_POINT('',(-20.0,-50.0,20.0));
#55=CARTESIAN_POINT('',(-46.0,-20.0,0.0));
#56=CARTESIAN_POINT('',(-46.0,-21.940717585790633,0.0));
#57=CARTESIAN_POINT('',(-45.561751785744534,-25.836183427878151,0.0));
#58=CARTESIAN_POINT('',(-43.622623050258596,-31.375547320925282,0.0));
#59=CARTESIAN_POINT('',(-40.498899405998614,-36.347490590759556,0.0));
#60=CARTESIAN_POINT('',(-36.347490590759676,-40.498899405998628,0.0));
```

```
#61=CARTESIAN_POINT('',(-31.375547320925357,-43.622623050258554,0.0));
#62=CARTESIAN_POINT('',(-25.836183427878254,-45.561751785744747,0.0));
#63=CARTESIAN_POINT('',(-21.940717585790722,-46.0,0.0));
#64=CARTESIAN_POINT('',(-20.0,-46.0,0.0));
#65=B_SPLINE_SURFACE_WITH_KNOTS('',3,1,((#45,#55),(#46,#56),(#47,#57),(#48,#58),(#49,#59),(#50,#60),
(#51,#61),(#52,#62),(#53,#63),(#54,#64)),.UNSPECIFIED.,.F.,.F.,.U.,(4,1,1,1,1,1,1,4),(2,2),
(3.141592653589792,3.365992128846206,3.590391604102619,3.814791079359032,4.039190554615447,
4.263590029871861,4.487989505128275,4.712388980384688),(0.0,1.0),.UNSPECIFIED.);
#110=DESCRIPTIVE_PARAMETER('TKM','+/-10 MICROM');
#111=DESCRIPTIVE_PARAMETER('TE','10-15 MICROM');
#112=WIRE_EDM_TECHNOLOGY(0.,$,$,$,(#113,#111,#110));
#113=NUMERIC_PARAMETER('RA',1.8,'MICROM');
#114=LINEAR_STRATEGY($,$);
#115=WIRE_EDM_MACHINE_FUNCTIONS(.T.,2.3,.F.,.F.,());
#116=WIRE_TOOL('WIRE TOOL',#120,0.25,0.,0.,$);
#117=PROJECT('SAMPLE 2',#121,(#118));
#118=WORKPIECE('WORKPIECE',#119,$,$,$,$,$);
#119=MATERIAL('','COLD DIE STEEL',$);
#120=MATERIAL('','COBRA CUT',$);
#121=WORKPLAN('WORKPLAN1',(#122),$,$);
#122=MACHINING_WORKINGSTEP('STEP1',$,#125,#163);
#125=REGION_SURFACE_LIST('REGION1',#18,(#163),$,(#6,#19,#24,#29,#34,#39,#44,#65));
#163=WIRE_EDM_MACHINING_OPERATION($,$,'OPER1',$,$,#94,#116,#112,#115,2.,$,#114,#95,#96);
ENDSEC;
END-ISO-10303-21;
```

Index