RESEARCH SERIES

NBSIR 88-3818

CONVERTING THE AMRF PART MODEL REPORT TO A PDES/STEP SUBSET

July 7, 1988

By:
Y. Tina Lee
Sanford P. Ressler

U.S. DEPARTMENT OF COMMERCE National Bureau of Standards Gaithersburg, Maryland

Converting the AMRF Part Model Report To a PDES/STEP Subset

(A Preliminary Implementation)

by

Y. Tina Lee
Sanford Ressler

National Bureau of Standards
Factory Automation Division
Machine Intelligence Group

July, 1988

Table of Contents

Converting the AMRF Part Model Report To a PDES/STEP Subset (A Preliminary Implementation)

by
Y. Tina Lee
Sanford Ressler
National Bureau of Standards
July, 1988

ABSTRACT

This paper identifies the process through which the Topology and Geometry of product data defined in the AMRF (Automated Manufacturing Research Facility) Part Model database report are converted to the PDES (Product Data Exchange Specification) / STEP (Standard for the Exchange of Product Model Data) physical file.

1 INTRODUCTION

The AMRF Part Model is a data format used to represent the information characterizing a part to be manufactured in the AMRF. The PDES/STEP standard is an international project to develop a mechanism for the exchange of complete product model data among a variety of different vendor CAD/CAM systems. Product data in the AMRF Part Model consists of topology, geometry, features, and functionality (tolerances). In this paper, only the two main resources of product data - topology and geometry - are discussed.

The AMRF Part Model evolved before the PDES/STEP file structure was available to fill a need in the AMRF. But the actual data contents of AMRF Part Model report and the PDES/STEP file appear to be quite similar. This paper presents the comparison and conversion between these two file structures.

It is assumed that the audience of this paper is familiar with the AMRF Part Model (refer to "AMRF Database Report Format : Part Model" [HOPP87]), the PDES/STEP exchange format (refer to "The Step File Structure" [ISO88]), and the resource entities of the Integrated Product Information Model (refer to "PDES/STEP Testing Draft - Washington Edition" [ISO88]). The PDES/STEP is an ongoing project. The PDES/STEP File Structure chosen for this development is the February, 1988 version.

2 OVERVIEW COMPARISON BETWEEN PART MODEL AND PDES/STEP FILE STRUCTURE

2.1 File Format

Both the Part Model report and the PDES/STEP file are sequential files. They are specified by a context-free grammar. There is no column-dependent information in either type of the files. However, an AMRF Part Model report is presented in a line-oriented or record-oriented manner, and the PDES/STEP file is a continuous stream of characters.

2.2 Organization

Both the Part Model report and PDES/STEP file are organized in a modular manner. They both consist of several sections. These sections comprise one or many entities. (The entries in all sections except the Part Model Header Section are named entities.) In the Topology and Geometry grammar of Part Model, there are three sections - Header Section, Topology Section, and Geometry section - in the Part Model report. But, there are only two sections - Header Section and Data Section - in the PDES/STEP file. Figure 1 shows the overview file structures of Part Model report and PDES/STEP file.

PART MODEL REPORT	PDES/STEP FILE
/PART_MODEL	STEP;
/HEADER	HEADER;
(header entries)	(header entities)
/END_HEADER	ENDSEC;
/TOPOLOGY	DATA;
(topology entities)	
/END_TOPOLOGY	(data entities)
/GEOMETRY	
(geometry entities)	
/END_GEOMETRY	ENDSEC;
/END_PART_MODEL	ENDSTEP;

Figure 1. Overview Comparison Between Part Model Report and PDES/STEP File

3 CONVERSION PROCEDURES

This section identifies the general rules and the detailed procedures for converting physical files from Part Model to PDES/STEP file structure.

3.1 General Rules of Conversion

There are four general rules to be followed in order to generate PDES/STEP files from Part Model reports.

(1) One of the requirements of PDES/STEP file structure is to minimize file size. Therefore, throughout every PDES/STEP file, there is no unnecessary space or line added to it. (Note : Throughout this paper, blank spaces and blank lines have been inserted into the PDES/STEP format records to aid readability. The reader should note that these blank spaces and blank lines do not appear in an actual PDES/STEP format file.)

(2) PDES/STEP comments may appear anywhere except within integers, real numbers, lists, entity identifiers, reserved words, key words, or delimiters. It is up to the programmer to add as many comments as they desire, however, PDES/STEP comments are intended to be read by humans only, they are not significant and should be ignored during lexical analysis. When converting the Part Model report, unnecessary Part Model blanks should be ignored, and Part Model comments may be ignored or may be converted to PDES/STEP comments.

(3) Throughout the Data Section of PDES/STEP file, entity occurrences must be defined before they may be referenced. Therefore, the order of the presentation entities is significant.

(4) An entity name in Part Model report file is a sequence of 1 to 16 alphanumeric ASCII characters, the first of which must be alphabetic. However, a PDES/STEP entity name is an integer, this integer may comprise any combination of 1 to 9 digits as long as uniqueness within the file is maintained. Therefore, entity name conversion (from ASCII to integers) is required when file conversion is performed. It is suggested to use a hash table for such conversion.

3.2 Detailed Syntax Conversion Procedures

This section provides a reference of the converting records from the format of the Part Model to the format of the PDES/STEP file structure. This section comprises several sub-sections. These subsections are devided by the types of file records. For convenience, reserved words or keywords are expressed in uppercase letters, and variable attributes are expressed in lowercase letters.

3.2.1 File Header and File Trailer

Part Model Report Constructs	PDES/STEP File Constructs
/PART_MODEL	STEP;
/END_PART_MODEL	ENDSTEP;

3.2.2 File Body

3.2.2.1 Header Section

Part Model Report Constructs	PDES/STEP File Constructs
/HEADER	HEADER;
PART_NAME = 'part_name'.	FILE_IDENTIFICATION('part_name', 'date', 'author', 'organization', 'step_version', 'preprocessor_version',

```
  'originating_system');
FILE_DESCRIPTION(
  'file_description');
IMP_LEVEL(
  'imp_level');
```

Algorithm:

All attributes in PDES/STEP Header Section, except attributes part_name and date , are supplied by programmer. Attribute part_name is read from Part Model report, and attribute date, the date and time of the file created, can be returned by a system call.

/END__HEADER	ENDSEC;

3.2.2.2 Data Section

(I). Header and Trailer of Data Section

Part Model Report Constructs	PDES/STEP File Constructs
/TOPOLOGY	(no action)
/END_TOPOLOGY	ENDSEC;
/GEOMETRY	DATA;
/END_GEOMETRY	(no action)

(II). Body of Data Section

As mentioned previously, entity occurrences in PDES/STEP Data Section must be defined before they may be referenced. This section has been organized by the presentation order of actual PDES/STEP physical file. However, other presentation orders may also be accepted. For example, point entities may be placed after unit vector entities.

The PDES/STEP entities defined in this section are representative of a common approach, other syntactical representations are possible.

Conversion of entity names between Part Model and PDES/STEP has been discussed previously. For convenience, Part Model entity names are expressed as: entity type_id or entity type_id_i and the corresponding PDES/STEP entity names are expressed as: entity type_integer_id or entity type_integer_id_i.

(a). Point Entities

Part Model Report Constructs	PDES/STEP File Constructs
/POINTS	(no action)
point_id;	@point_integer_id = CARTESIAN_THREE_COORDINATE
x,	(x,
y,	y,

z.	z);
/END_POINTS	(no action)

(b). Unit Vector Entities

Part Model Report Constructs	PDES/STEP File Constructs
/UNIT_VECTORS	(no action)
unit_vector_id;	@unit_vector_integer_id = THREE_SPACE_DIRECTION
x,	(x,
y,	y,
z.	z);
/END_UNIT_VECTORS	(no action)

(c). Curve Entities

Part Model Report Constructs	PDES/STEP File Constructs
/CURVES	(no action)
curve_id; LINE; point_id; unit_vector_id.	@curve_integer_id = LINE(#point_integer_id, #unit_vector_integer_id);
curve_id; CIRCLE; point_id_1; unit_vector_id; point_id_2.	@curve_integer_id = CIRCLE(radius, AXIS2_PLACEMENT(#point_integer_id_1, #unit_vector_integer_id, THREE_SPACE_DIRECTION (x, y, z)));

Algorithm:

v1 = vector (point_id_1)
v2 = vector (point_id_2)
radius = distance (v1, v2)
(x, y, z) = (v2 - v1) / radius

/END_CURVES	(no action)

(d). Surface Entities

Part Model Report Constructs	PDES/STEP File Constructs
/SURFACES	(no action)
surface_id; PLANE; unit_vector_id; number.	@surface_integer_id = PLANE(AXIS2_PLACEMENT(CARTESIAN_THREE_COORDINATE (x, y, z), #unit_vector_integer_id));

Algorithm:

v = vector (unit_vector_id)
(x, y, z) = number * v

surface_id; CYLINDER; point_id; unit_vector_id; number.	@surface_integer_id = CYLINDRICAL_SURFACE(number, AXIS2_PLACEMENT(#point_integer_id, #unit_vector_integer_id));
surface_id; SPHERE; point_id; number.	@surface_integer_id = SPHERICAL_SURFACE(number, AXIS2_PLACEMENT(

#point_integer_id));

surface_id;
CONE;
point_id;
unit_vector_id;
number.

@surface_integer_id =
CONICAL_SURFACE(
semi_angle,
0.0,
AXIS2_PLACEMENT(
#point_integer_id,
#unit_vector_integer_id));

Algorithm:
semi_angle = arccos (number) * 180.0 / 3.14159

/END_SURFACES (no action)

(e). Vertex Entities

Part Model Report Constructs	PDES/STEP File Constructs
/VERTICES	(no action)
vertex_id;	@vertex_integer_id = VERTEX

point_id.	(#point_integer_id);
/END_VERTICES	(no action)

(f). Edge Entities

Part Model Report Constructs	PDES/STEP File Constructs
/EDGES	(no action)
edge_id; vertex_id_1, vertex_id_2; curve_id sense.	@edge_integer_id = EDGE(#vertex_integer_id_1, #vertex_integer_id_2, CURVE_LOGICAL_STRUCTURE(#curve_integer_id, flag));

Algorithm:

If sense = "-" then flag = .F.
If sense = "+" then flag = .T.

/END_EDGES	(no action)

(g) Loop Entities

Part Model Report Constructs	PDES/STEP File Constructs
/LOOPS	(no action)
loop_id; edge_id_1 sense_1, edge_id_2 sense_2, . . edge_id_n sense_n;	@loop_integer_id = EDGE_LOOP((EDGE_LOGICAL_STRUCTURE (#edge_integer_id_1, flag_1), EDGE_LOGICAL_STRUCTURE (#edge_integer_id_2, flag_2), . . EDGE_LOGICAL_STRUCTURE (#edge_integer_id_n, flag_n)));

Algorithm:

```
If  sense_1 = "-"  then  flag_1 = .F.
If  sense_1 = "+"  then  flag_1 = .T.
If  sense_2 = "-"  then  flag_2 = .F.
If  sense_2 = "+"  then  flag_2 = .T.
                .
                .
If  sense_n = "-"  then  flag_n = .F.
If  sense_n = "+"  then  flag_n = .T.
```

/END_LOOPS	(no action)

(h). FACE Entities

Part Model Report Constructs	PDES/STEP File Constructs
/FACES	(no action)
loop_id; loop_id_1, loop_id_2, . . loop_id_n; surface_id sense.	@loop_integer_id = FACE(, (LOOP_LOGICAL_STRUCTURE (#loop_integer_id_1, .T.), LOOP_LOGICAL_STRUCTURE (#loop_integer_id_2, .T.), . . LOOP_LOGICAL_STRUCTURE (#loop_integer_id_n, .T.)), SURFACE_LOGICAL_STRUCTURE((#surface_integer_id, flag));

Algorithm:

If sense = "-" then flag = .F.
If sense = "+" then flag = .T.

/END_FACES	(no action)

(i). SHELL Entities

Part Model Report Constructs	PDES/STEP File Constructs
/SHELLS	(no action)
shell_id;	@shell_integer_id = CLOSED_SHELL((
face_id_1,	FACE_LOGICAL_STRUCTURE (face_integer_id_1, .T.),
face_id_2,	FACE_LOGICAL_STRUCTURE (face_integer_id_2, .T.),
. .	. .
face_id_n.	FACE_LOGICAL_STRUCTURE (face_integer_id_n, .T.)));
/END_SHELLS	(no action)

4 FILE CONVERSION SOFTWARE

Based on the previously described conversion procedures, an AMRF Part Model to PDES/STEP file structure translator has been implemented using Lex, YACC, and C on the Sun Microsystems Workstation. This translator interprets the Part Model report and transforms information data into a set of data structures by using the Part Model Parser [RESS87], it then selects and manipulates the information data from data structures, and produces a PDES/STEP physical file.

An example of a test run is provided for illustrating the actual usage of the translator. Appendix A contains the listing of the input Part Model report (part name is PIPECLAMP_FV), and Appendix B contains the listing of the output PDES/STEP physical file. Figure 2 presents the sketch of "PIPECLAMP_FV".

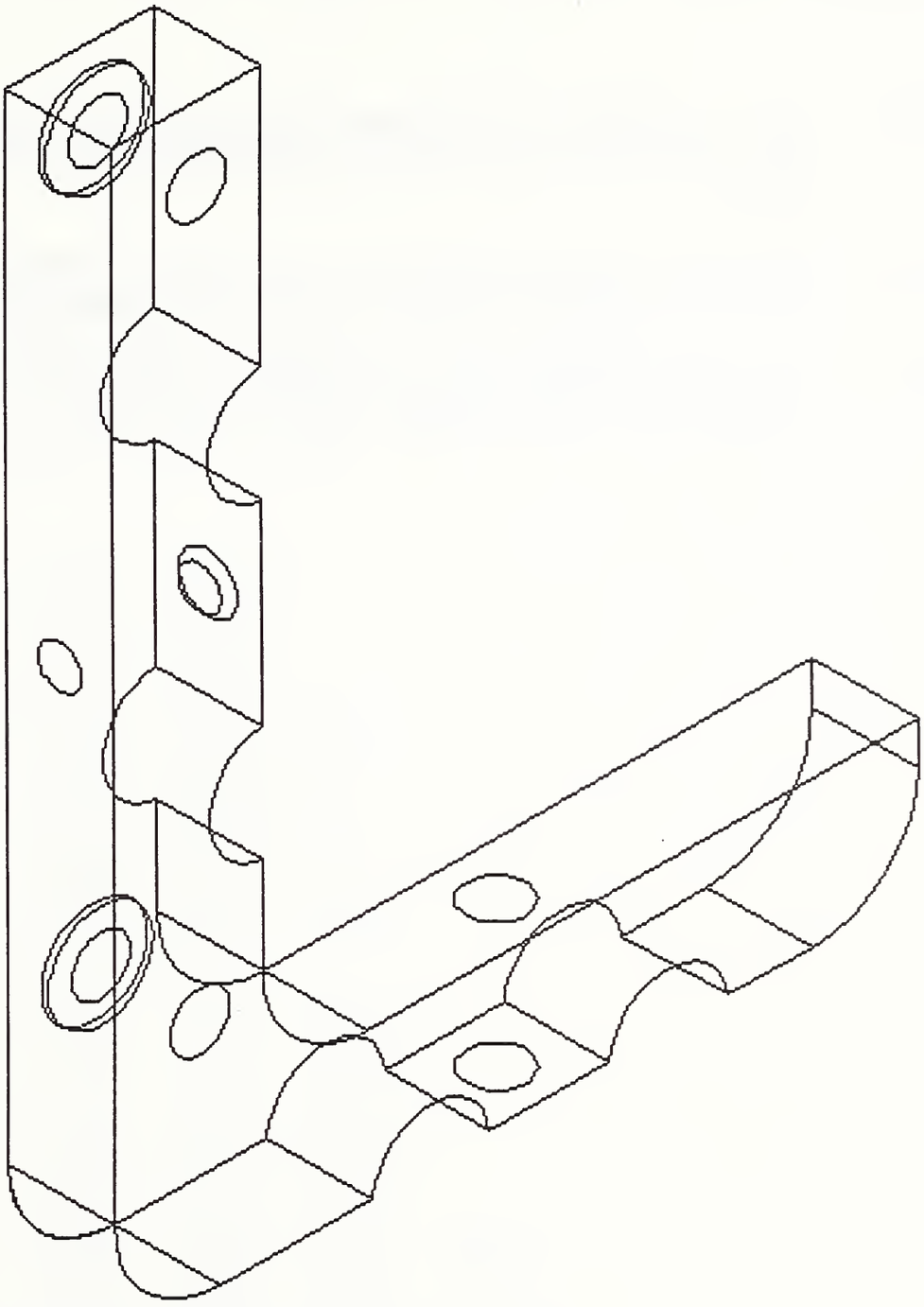

Figure 2. Sketch of "PIPECLAMP_FV"

5 REFERENCES

[HOPP87] Hopp, T., "AMRF Database Report Format : Part Model", NBSIR 87-3672, National Bureau of Standards, Gaithersburg, Maryland, September 1987.

[RESS87] Ressler, S., "Using the AMRF Part Model Report", NBSIR 87-3531, National Bureau of Standards, Gaithersburg, Maryland, February, 1987.

[ISO88] International Organization for Standardization Committee, "STEP/PDES Testing Draft", TC184/SC4/WG1, ISO, February, 1988

Appendix A: AMRF Part Model Report Example

```
/PART_MODEL
/HEADER
      PART_NAME = 'PIPECLAMP_FV' .
/END_HEADER
/TOPOLOGY
/SHELLS
      SHL001;
            FAC001,FAC002, FAC003, FAC004, FAC005,
            FAC006, FAC007, FAC008, FAC009, FAC010,
            FAC011,FAC012, FAC013, FAC014, FAC015,
            FAC016, FAC017, FAC018, FAC019, FAC020,
            FAC021,FAC022, FAC023, FAC024, FAC025,
            FAC026, FAC027, FAC028 .
/END_SHELLS
/FACES
      FAC001; LOP001; SUR001 - .
      FAC002; LOP002; SUR002 + .
      FAC003; LOP003; SUR003 + .
      FAC004; LOP004; SUR004 - .
      FAC005; LOP005; SUR005 + .
      FAC006; LOP006, LOP030; SUR006 + .
      FAC007; LOP007; SUR007 + .
      FAC008; LOP008; SUR008 + .
      FAC009; LOP009; SUR009 + .
      FAC010; LOP010; SUR010 + .
      FAC011; LOP011; SUR011 + .
      FAC012; LOP012, LOP032; SUR012 - .
      FAC013; LOP013; SUR013 - .
      FAC014; LOP014, LOP031; SUR014 + .
      FAC015; LOP015, LOP033; SUR015 + .
      FAC016; LOP016, LOP034; SUR016 + .
      FAC017; LOP017, LOP037; SUR017 + .
      FAC018; LOP018, LOP038; SUR018 + .
      FAC019; LOP019, LOP039; SUR019 + .
      FAC020; LOP020, LOP042; SUR020 + .
      FAC021; LOP021, LOP044; SUR021 + .
      FAC022; LOP022, LOP036, LOP041; SUR022 - .
      FAC023; LOP023, LOP040, LOP045; SUR023 + .
      FAC024; LOP024, LOP029; SUR001 - .
      FAC025; LOP025; SUR001 - .
      FAC026; LOP026, LOP035; SUR008 + .
      FAC027; LOP027; SUR008 + .
      FAC028; LOP028, LOP043; SUR018 + .
/END_FACES
/LOOPS
      LOP001; EDG001 +, EDG041 +, EDG025 -, EDG040 - .
      LOP002; EDG002 -, EDG042 +, EDG026 +, EDG041 - .
      LOP003; EDG003 -, EDG044 +, EDG027 +, EDG043 - .
      LOP004; EDG004 +, EDG046 +, EDG028 -, EDG045 - .
      LOP005; EDG005 +, EDG047 +, EDG029 -, EDG046 - .
```

```
      LOP006; EDG006 -, EDG048 +, EDG030 +, EDG047 - .
      LOP007; EDG007 -, EDG049 +, EDG031 +, EDG048 - .
      LOP008; EDG008 +, EDG050 +, EDG032 -, EDG049 - .
      LOP009; EDG009 -, EDG051 +, EDG033 +, EDG050 - .
      LOP010; EDG010 -, EDG053 +, EDG034 +, EDG052 - .
      LOP011; EDG011 -, EDG055 +, EDG035 +, EDG054 - .
      LOP012; EDG012 -, EDG056 +, EDG036 +, EDG055 - .
      LOP013; EDG013 +, EDG040 +, EDG037 -, EDG056 - .
      LOP014; EDG016 - .
      LOP015; EDG018 - .
      LOP016; EDG019 - .
      LOP017; EDG014 - .
      LOP018; EDG022 + .
      LOP019; EDG022 - .
      LOP020; EDG015 - .
      LOP021; EDG024 - .
      LOP022; EDG001 -, EDG013 -, EDG012 +, EDG011 +, EDG060 -,
            EDG010 +, EDG059 -, EDG009 +, EDG008 -, EDG007 +,
            EDG006 +, EDG005 -, EDG004 -, EDG058 -, EDG003 +,
            EDG057 -, EDG002 + .
      LOP023; EDG025 +, EDG026 -, EDG061 +, EDG027 -, EDG062 +,
            EDG028 +, EDG029 +, EDG030 -, EDG031 -, EDG032 +,
            EDG033 -, EDG063 +, EDG034 -, EDG064 +, EDG035 -,
            EDG036 -, EDG037 + .
      LOP024; EDG057 +, EDG043 +, EDG061 -, EDG042 - .
      LOP025; EDG058 +, EDG045 +, EDG062 -, EDG044 - .
      LOP026; EDG059 +, EDG052 +, EDG063 -, EDG051 - .
      LOP027; EDG060 +, EDG054 +, EDG064 -, EDG053 - .
      LOP028; EDG024 + .
      LOP029; EDG016 + .
      LOP030; EDG017 - .
      LOP031; EDG017 + .
      LOP032; EDG018 + .
      LOP033; EDG019 + .
      LOP034; EDG020 + .
      LOP035; EDG020 - .
      LOP036; EDG014 + .
      LOP037; EDG021 + .
      LOP038; EDG021 - .
      LOP039; EDG038 + .
      LOP040; EDG038 - .
      LOP041; EDG015 + .
      LOP042; EDG023 + .
      LOP043; EDG023 - .
      LOP044; EDG039 + .
      LOP045; EDG039 - .
/END_LOOPS
/EDGES
      EDG001; VTX001, VTX002; CRV001 + .
      EDG002; VTX004, VTX002; CRV002 + .
      EDG003; VTX007, VTX005; CRV003 + .
      EDG004; VTX008, VTX010; CRV004 + .
      EDG005; VTX010, VTX011; CRV005 + .
      EDG006; VTX012, VTX011; CRV006 + .
```

```
EDG007; VTX014, VTX012; CRV007 + .
EDG008; VTX014, VTX015; CRV008 + .
EDG009; VTX017, VTX015; CRV009 + .
EDG010; VTX020, VTX018; CRV010 + .
EDG011; VTX022, VTX021; CRV011 + .
EDG012; VTX023, VTX022; CRV012 + .
EDG013; VTX023, VTX001; CRV013 + .
EDG014; ; CRV014 + .
EDG015; ; CRV015 + .
EDG016; ; CRV016 + .
EDG017; ; CRV017 + .
EDG018; ; CRV018 + .
EDG019; ; CRV019 + .
EDG020; ; CRV020 + .
EDG021; ; CRV021 + .
EDG022; ; CRV022 + .
EDG023; ; CRV023 + .
EDG024; ; CRV024 + .
EDG025; VTX039, VTX040; CRV025 + .
EDG026; VTX042, VTX040; CRV026 + .
EDG027; VTX045, VTX043; CRV027 + .
EDG028; VTX046, VTX048; CRV028 + .
EDG029; VTX048, VTX049; CRV029 + .
EDG030; VTX050, VTX049; CRV030 + .
EDG031; VTX052, VTX050; CRV031 + .
EDG032; VTX052, VTX053; CRV032 + .
EDG033; VTX055, VTX053; CRV033 + .
EDG034; VTX058, VTX056; CRV034 + .
EDG035; VTX060, VTX059; CRV035 + .
EDG036; VTX061, VTX060; CRV036 + .
EDG037; VTX061, VTX039; CRV037 + .
EDG038; ; CRV038 + .
EDG039; ; CRV039 + .
EDG040; VTX001, VTX039; CRV040 + .
EDG041; VTX002, VTX040; CRV041 + .
EDG042; VTX004, VTX042; CRV042 + .
EDG043; VTX005, VTX043; CRV043 + .
EDG044; VTX007, VTX045; CRV044 + .
EDG045; VTX008, VTX046; CRV045 + .
EDG046; VTX010, VTX048; CRV046 + .
EDG047; VTX011, VTX049; CRV047 + .
EDG048; VTX012, VTX050; CRV048 + .
EDG049; VTX014, VTX052; CRV049 + .
EDG050; VTX015, VTX053; CRV050 + .
EDG051; VTX017, VTX055; CRV051 + .
EDG052; VTX018, VTX056; CRV052 + .
EDG053; VTX020, VTX058; CRV053 + .
EDG054; VTX021, VTX059; CRV054 + .
EDG055; VTX022, VTX060; CRV055 + .
EDG056; VTX023, VTX061; CRV056 + .
EDG057; VTX004, VTX005; CRV001 + .
EDG058; VTX007, VTX008; CRV001 + .
EDG059; VTX017, VTX018; CRV008 + .
EDG060; VTX020, VTX021; CRV008 + .
```

```
      EDG061; VTX042, VTX043; CRV025 + .
      EDG062; VTX045, VTX046; CRV025 + .
      EDG063; VTX055, VTX056; CRV032 + .
      EDG064; VTX058, VTX059; CRV032 + .
/END_EDGES
/VERTICES
      VTX001; PT001 .
      VTX002; PT002 .
      VTX004; PT004 .
      VTX005; PT005 .
      VTX007; PT007 .
      VTX008; PT008 .
      VTX010; PT010 .
      VTX011; PT011 .
      VTX012; PT012 .
      VTX014; PT014 .
      VTX015; PT015 .
      VTX017; PT017 .
      VTX018; PT018 .
      VTX020; PT020 .
      VTX021; PT021 .
      VTX022; PT022 .
      VTX023; PT023 .
      VTX026; PT026 .
      VTX028; PT028 .
      VTX030; PT030 .
      VTX032; PT032 .
      VTX034; PT034 .
      VTX036; PT036 .
      VTX038; PT038 .
      VTX039; PT039 .
      VTX040; PT040 .
      VTX042; PT042 .
      VTX043; PT043 .
      VTX045; PT045 .
      VTX046; PT046 .
      VTX048; PT048 .
      VTX049; PT049 .
      VTX050; PT050 .
      VTX052; PT052 .
      VTX053; PT053 .
      VTX055; PT055 .
      VTX056; PT056 .
      VTX058; PT058 .
      VTX059; PT059 .
      VTX060; PT060 .
      VTX061; PT061 .
      VTX064; PT064 .
      VTX066; PT066 .
      VTX068; PT068 .
      VTX069; PT069 .
      VTX071; PT071 .
      VTX072; PT072 .
/END_VERTICES
```

```
/END_TOPOLOGY
/GEOMETRY
/SURFACES
      SUR001; PLANE; UNV002; 0.0 .
      SUR002; CYLINDER; PT073; UNV003; 0.281 .
      SUR003; CYLINDER; PT074; UNV003; 0.281 .
      SUR004; CYLINDER; PT075; UNV003; 0.281 .
      SUR005; PLANE; UNV001; 3.812 .
      SUR006; PLANE; UNV002; 0.703 .
      SUR007; CYLINDER; PT076; UNV003; 0.500 .
      SUR008; PLANE; UNV001; 0.703 .
      SUR009; CYLINDER; PT077; UNV003; 0.281 .
      SUR010; CYLINDER; PT078; UNV003; 0.281 .
      SUR011; PLANE; UNV002; 5.000 .
      SUR012; PLANE; UNV001; 0.0 .
      SUR013; CYLINDER; PT079; UNV003; 0.500 .
      SUR014; CYLINDER; PT029; UNV002; 0.281 .
      SUR015; CYLINDER; PT033; UNV001; 0.203 .
      SUR016; CONE; PT080; UNV001; 0.707 .
      SUR017; CYLINDER; PT025; UNV003; 0.1405 .
      SUR018; PLANE; UNV003; 0.46875 .
      SUR019; CYLINDER; PT067; UNV003; 0.250 .
      SUR020; CYLINDER; PT027; UNV003; 0.1405 .
      SUR021; CYLINDER; PT070; UNV003; 0.250 .
      SUR022; PLANE; UNV003; 0.0 .
      SUR023; PLANE; UNV003; 0.500 .
/END_SURFACES
/CURVES
      CRV001; LINE; PT001; UNV001 .
      CRV002; CIRCLE; PT073; UNV003; PT002 .
      CRV003; CIRCLE; PT074; UNV003; PT005 .
      CRV004; CIRCLE; PT075; UNV003; PT010 .
      CRV005; LINE; PT010; UNV002 .
      CRV006; LINE; PT012; UNV001 .
      CRV007; CIRCLE; PT076; UNV003; PT012 .
      CRV008; LINE; PT014; UNV002 .
      CRV009; CIRCLE; PT077; UNV003; PT015 .
      CRV010; CIRCLE; PT078; UNV003; PT018 .
      CRV011; LINE; PT022; UNV001 .
      CRV012; LINE; PT023; UNV002 .
      CRV013; CIRCLE; PT079; UNV003; PT001 .
      CRV014; CIRCLE; PT025; UNV003; PT026 .
      CRV015; CIRCLE; PT027; UNV003; PT028 .
      CRV016; CIRCLE; PT029; UNV002; PT030 .
      CRV017; CIRCLE; PT031; UNV002; PT032 .
      CRV018; CIRCLE; PT033; UNV001; PT034 .
      CRV019; CIRCLE; PT035; UNV001; PT036 .
      CRV020; CIRCLE; PT037; UNV001; PT038 .
      CRV021; CIRCLE; PT067; UNV003; PT068 .
      CRV022; CIRCLE; PT067; UNV003; PT069 .
      CRV023; CIRCLE; PT070; UNV003; PT071 .
      CRV024; CIRCLE; PT070; UNV003; PT072 .
      CRV025; LINE; PT039; UNV001 .
      CRV026; CIRCLE; PT081; UNV003; PT040 .
```

```
      CRV027; CIRCLE; PT082; UNV003; PT043 .
      CRV028; CIRCLE; PT083; UNV003; PT048 .
      CRV029; LINE; PT048; UNV002 .
      CRV030; LINE; PT050; UNV001 .
      CRV031; CIRCLE; PT084; UNV003; PT050 .
      CRV032; LINE; PT052; UNV002 .
      CRV033; CIRCLE; PT085; UNV003; PT053 .
      CRV034; CIRCLE; PT086; UNV003; PT056 .
      CRV035; LINE; PT060; UNV001 .
      CRV036; LINE; PT061; UNV001 .
      CRV037; CIRCLE; PT087; UNV003; PT039 .
      CRV038; CIRCLE; PT063; UNV003; PT064 .
      CRV039; CIRCLE; PT065; UNV003; PT066 .
      CRV040; LINE; PT001; UNV003 .
      CRV041; LINE; PT002; UNV003 .
      CRV042; LINE; PT004; UNV003 .
      CRV043; LINE; PT005; UNV003 .
      CRV044; LINE; PT007; UNV003 .
      CRV045; LINE; PT008; UNV003 .
      CRV046; LINE; PT010; UNV003 .
      CRV047; LINE; PT011; UNV003 .
      CRV048; LINE; PT012; UNV003 .
      CRV049; LINE; PT014; UNV003 .
      CRV050; LINE; PT015; UNV003 .
      CRV051; LINE; PT017; UNV003 .
      CRV052; LINE; PT018; UNV003 .
      CRV053; LINE; PT020; UNV003 .
      CRV054; LINE; PT021; UNV003 .
      CRV055; LINE; PT022; UNV003 .
      CRV056; LINE; PT023; UNV003 .
/END_CURVES
/POINTS
      PT001; 0.5, 0.0, 0.0 .
      PT002; 1.22072, 0.0, 0.0 .
      PT004; 1.77928, 0.0, 0.0 .
      PT005; 2.34572, 0.0, 0.0 .
      PT007; 2.90428, 0.0, 0.0 .
      PT008; 3.312, 0.0, 0.0 .
      PT010; 3.812, 0.5, 0.0 .
      PT011; 3.812, 0.703, 0.0 .
      PT012; 1.203, 0.703, 0.0 .
      PT014; 0.703, 1.203, 0.0 .
      PT015; 0.703, 1.67372, 0.0 .
      PT017; 0.703, 2.23228, 0.0 .
      PT018; 0.703, 3.17372, 0.0 .
      PT020; 0.703, 3.73228, 0.0 .
      PT021; 0.703, 5.0, 0.0 .
      PT022; 0.0, 5.0, 0.0 .
      PT023; 0.0, 0.5, 0.0 .
      PT025; 0.406, 1.141, 0.0 .
      PT026; 0.5465, 1.141, 0.0 .
      PT027; 0.406, 4.641, 0.0 .
      PT028; 0.5465, 4.641, 0.0 .
      PT029; 2.062, 0.0, 0.25 .
```

```
      PT030; 2.2025, 0.0, 0.25 .
      PT031; 2.062, 0.703, 0.25 .
      PT032; 2.2025, 0.703, 0.25 .
      PT033; 0.0, 2.703, 0.25 .
      PT034; 0.0, 2.8045, 0.25 .
      PT035; 0.664, 2.703, 0.25 .
      PT036; 0.664, 2.8045, 0.25 .
      PT037; 0.703, 2.703, 0.25 .
      PT038; 0.703, 2.8435, 0.25 .
      PT039; 0.5, 0.0, 0.5 .
      PT040; 1.22072, 0.0, 0.5 .
      PT042; 1.77928, 0.0, 0.5 .
      PT043; 2.34572, 0.0, 0.5 .
      PT045; 2.90428, 0.0, 0.5 .
      PT046; 3.312, 0.0, 0.5 .
      PT048; 3.812, 0.5, 0.5 .
      PT049; 3.812, 0.703, 0.5 .
      PT050; 1.203, 0.703, 0.5 .
      PT052; 0.703, 1.203, 0.5 .
      PT053; 0.703, 1.67372, 0.5 .
      PT055; 0.703, 2.23228, 0.5 .
      PT056; 0.703, 3.17372, 0.5 .
      PT058; 0.703, 3.73228, 0.5 .
      PT059; 0.703, 5.0, 0.5 .
      PT060; 0.0, 5.0, 0.5 .
      PT061; 0.0, 0.5, 0.5 .
      PT063; 0.406, 1.141, 0.5 .
      PT064; 0.656, 1.141, 0.5 .
      PT065; 0.406, 4.641, 0.5 .
      PT066; 0.656, 4.641, 0.5 .
      PT067; 0.406, 1.141, 0.46875 .
      PT068; 0.5465, 1.141, 0.46875 .
      PT069; 0.656, 1.141, 0.46875 .
      PT070; 0.406, 4.641, 0.46875 .
      PT071; 0.5465, 4.641, 0.46875 .
      PT072; 0.656, 4.641, 0.46875 .
      PT073; 1.500, -0.031, 0.0 .
      PT074; 2.625, -0.031, 0.0 .
      PT075; 3.312, 0.500, 0.0 .
      PT076; 1.203, 1.203, 0.0 .
      PT077; 0.734, 1.953, 0.0 .
      PT078; 0.734, 3.453, 0.0 .
      PT079; 0.500, 0.500, 0.0 .
      PT080; 0.5625, 2.703, 0.250 .
      PT081; 1.500, -0.031, 0.5 .
      PT082; 2.625, -0.031, 0.5 .
      PT083; 3.312, 0.500, 0.5 .
      PT084; 1.203, 1.203, 0.5 .
      PT085; 0.734, 1.953, 0.5 .
      PT086; 0.734, 3.453, 0.5 .
      PT087; 0.500, 0.500, 0.5 .
/END_POINTS
/UNIT_VECTORS
      UNV001; 1.0, 0, 0 .
```

```
      UNV002; 0, 1.0, 0 .
      UNV003; 0, 0, 1.0 .
/END_UNIT_VECTORS
/END_GEOMETRY
/END_PART_MODEL
```

Appendix B: PDES/STEP Physical File Example

!* The file structure of this STEP file is based on the
Document 4.2.1 (Draft Paper) - Feb. 22, 1988, and
Testing Draft:PART 2 of IPIM (Washington) - Feb. 29, 1988.
This STEP file has been presented in a line-oriented or
record-oriented manner in order to aid readability.
Unnecessary spaces have also been added to aid readability.
Note that an ordinary STEP file is not aligned in this
manner, but is instead a continuous stream of characters.
*!

```
STEP;

HEADER;
 FILE_IDENTIFICATION(
 'PIPECLAMP_FV',
 '19880627.100730',
 'Tina Lee & Sandy Ressler, (301)-975-3550 or 3549',
 'National Bureau of Standards, Factory Automation Systems Division',
 'STEP VERSION 1.0',
 'AMRF Part Model to Step File Translator, Version 1.0',
 'AMRF PART MODEL');
 FILE_DESCRIPTION('THIS FILE IS TRANSLATED FROM AN AMRF PART MODEL FILE');
 IMP_LEVEL('1.0');
ENDSEC;

DATA;
@1 = CARTESIAN_THREE_COORDINATE( 0.500000, 0.000000, 0.000000);
@2 = CARTESIAN_THREE_COORDINATE( 1.220720, 0.000000, 0.000000);
@3 = CARTESIAN_THREE_COORDINATE( 1.779280, 0.000000, 0.000000);
@4 = CARTESIAN_THREE_COORDINATE( 2.345720, 0.000000, 0.000000);
@5 = CARTESIAN_THREE_COORDINATE( 2.904280, 0.000000, 0.000000);
@6 = CARTESIAN_THREE_COORDINATE( 3.312000, 0.000000, 0.000000);
@7 = CARTESIAN_THREE_COORDINATE( 3.812000, 0.500000, 0.000000);
@8 = CARTESIAN_THREE_COORDINATE( 3.812000, 0.703000, 0.000000);
@9 = CARTESIAN_THREE_COORDINATE( 1.203000, 0.703000, 0.000000);
@10 = CARTESIAN_THREE_COORDINATE( 0.703000, 1.203000, 0.000000);
@11 = CARTESIAN_THREE_COORDINATE( 0.703000, 1.673720, 0.000000);
@12 = CARTESIAN_THREE_COORDINATE( 0.703000, 2.232280, 0.000000);
@13 = CARTESIAN_THREE_COORDINATE( 0.703000, 3.173720, 0.000000);
@14 = CARTESIAN_THREE_COORDINATE( 0.703000, 3.732280, 0.000000);
@15 = CARTESIAN_THREE_COORDINATE( 0.703000, 5.000000, 0.000000);
@16 = CARTESIAN_THREE_COORDINATE( 0.000000, 5.000000, 0.000000);
@17 = CARTESIAN_THREE_COORDINATE( 0.000000, 0.500000, 0.000000);
@18 = CARTESIAN_THREE_COORDINATE( 0.406000, 1.141000, 0.000000);
@19 = CARTESIAN_THREE_COORDINATE( 0.546500, 1.141000, 0.000000);
@20 = CARTESIAN_THREE_COORDINATE( 0.406000, 4.641000, 0.000000);
@21 = CARTESIAN_THREE_COORDINATE( 0.546500, 4.641000, 0.000000);
@22 = CARTESIAN_THREE_COORDINATE( 2.062000, 0.000000, 0.250000);
@23 = CARTESIAN_THREE_COORDINATE( 2.202500, 0.000000, 0.250000);
```

```
@24 = CARTESIAN_THREE_COORDINATE( 2.062000, 0.703000, 0.250000);
@25 = CARTESIAN_THREE_COORDINATE( 2.202500, 0.703000, 0.250000);
@26 = CARTESIAN_THREE_COORDINATE( 0.000000, 2.703000, 0.250000);
@27 = CARTESIAN_THREE_COORDINATE( 0.000000, 2.804500, 0.250000);
@28 = CARTESIAN_THREE_COORDINATE( 0.664000, 2.703000, 0.250000);
@29 = CARTESIAN_THREE_COORDINATE( 0.664000, 2.804500, 0.250000);
@30 = CARTESIAN_THREE_COORDINATE( 0.703000, 2.703000, 0.250000);
@31 = CARTESIAN_THREE_COORDINATE( 0.703000, 2.843500, 0.250000);
@32 = CARTESIAN_THREE_COORDINATE( 0.500000, 0.000000, 0.500000);
@33 = CARTESIAN_THREE_COORDINATE( 1.220720, 0.000000, 0.500000);
@34 = CARTESIAN_THREE_COORDINATE( 1.779280, 0.000000, 0.500000);
@35 = CARTESIAN_THREE_COORDINATE( 2.345720, 0.000000, 0.500000);
@36 = CARTESIAN_THREE_COORDINATE( 2.904280, 0.000000, 0.500000);
@37 = CARTESIAN_THREE_COORDINATE( 3.312000, 0.000000, 0.500000);
@38 = CARTESIAN_THREE_COORDINATE( 3.812000, 0.500000, 0.500000);
@39 = CARTESIAN_THREE_COORDINATE( 3.812000, 0.703000, 0.500000);
@40 = CARTESIAN_THREE_COORDINATE( 1.203000, 0.703000, 0.500000);
@41 = CARTESIAN_THREE_COORDINATE( 0.703000, 1.203000, 0.500000);
@42 = CARTESIAN_THREE_COORDINATE( 0.703000, 1.673720, 0.500000);
@43 = CARTESIAN_THREE_COORDINATE( 0.703000, 2.232280, 0.500000);
@44 = CARTESIAN_THREE_COORDINATE( 0.703000, 3.173720, 0.500000);
@45 = CARTESIAN_THREE_COORDINATE( 0.703000, 3.732280, 0.500000);
@46 = CARTESIAN_THREE_COORDINATE( 0.703000, 5.000000, 0.500000);
@47 = CARTESIAN_THREE_COORDINATE( 0.000000, 5.000000, 0.500000);
@48 = CARTESIAN_THREE_COORDINATE( 0.000000, 0.500000, 0.500000);
@49 = CARTESIAN_THREE_COORDINATE( 0.406000, 1.141000, 0.500000);
@50 = CARTESIAN_THREE_COORDINATE( 0.656000, 1.141000, 0.500000);
@51 = CARTESIAN_THREE_COORDINATE( 0.406000, 4.641000, 0.500000);
@52 = CARTESIAN_THREE_COORDINATE( 0.656000, 4.641000, 0.500000);
@53 = CARTESIAN_THREE_COORDINATE( 0.406000, 1.141000, 0.468750);
@54 = CARTESIAN_THREE_COORDINATE( 0.546500, 1.141000, 0.468750);
@55 = CARTESIAN_THREE_COORDINATE( 0.656000, 1.141000, 0.468750);
@56 = CARTESIAN_THREE_COORDINATE( 0.406000, 4.641000, 0.468750);
@57 = CARTESIAN_THREE_COORDINATE( 0.546500, 4.641000, 0.468750);
@58 = CARTESIAN_THREE_COORDINATE( 0.656000, 4.641000, 0.468750);
@59 = CARTESIAN_THREE_COORDINATE( 1.500000, -0.031000, 0.000000);
@60 = CARTESIAN_THREE_COORDINATE( 2.625000, -0.031000, 0.000000);
@61 = CARTESIAN_THREE_COORDINATE( 3.312000, 0.500000, 0.000000);
@62 = CARTESIAN_THREE_COORDINATE( 1.203000, 1.203000, 0.000000);
@63 = CARTESIAN_THREE_COORDINATE( 0.734000, 1.953000, 0.000000);
@64 = CARTESIAN_THREE_COORDINATE( 0.734000, 3.453000, 0.000000);
@65 = CARTESIAN_THREE_COORDINATE( 0.500000, 0.500000, 0.000000);
@66 = CARTESIAN_THREE_COORDINATE( 0.562500, 2.703000, 0.250000);
@67 = CARTESIAN_THREE_COORDINATE( 1.500000, -0.031000, 0.500000);
@68 = CARTESIAN_THREE_COORDINATE( 2.625000, -0.031000, 0.500000);
@69 = CARTESIAN_THREE_COORDINATE( 3.312000, 0.500000, 0.500000);
@70 = CARTESIAN_THREE_COORDINATE( 1.203000, 1.203000, 0.500000);
@71 = CARTESIAN_THREE_COORDINATE( 0.734000, 1.953000, 0.500000);
@72 = CARTESIAN_THREE_COORDINATE( 0.734000, 3.453000, 0.500000);
@73 = CARTESIAN_THREE_COORDINATE( 0.500000, 0.500000, 0.500000);
!* Done : Points *!

@74 = THREE_SPACE_DIRECTION( 1.000000, 0.000000, 0.000000);
@75 = THREE_SPACE_DIRECTION( 0.000000, 1.000000, 0.000000);
```

```
@76 = THREE_SPACE_DIRECTION( 0.000000, 0.000000, 1.000000);
!* Done : UnitVectors *!

@77 = LINE( #1, #74);
@78 = CIRCLE( 0.280995, AXIS2_PLACEMENT(#59, #76,
       THREE_SPACE_DIRECTION(-0.993896, 0.110322, 0.000000)));
@79 = CIRCLE( 0.280995, AXIS2_PLACEMENT(#60, #76,
       THREE_SPACE_DIRECTION(-0.993896, 0.110322, 0.000000)));
@80 = CIRCLE( 0.500000, AXIS2_PLACEMENT(#61, #76,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@81 = LINE( #7, #75);
@82 = LINE( #9, #74);
@83 = CIRCLE( 0.500000, AXIS2_PLACEMENT(#62, #76,
       THREE_SPACE_DIRECTION(0.000000, -1.000000, 0.000000)));
@84 = LINE( #10, #75);
@85 = CIRCLE( 0.280995, AXIS2_PLACEMENT(#63, #76,
       THREE_SPACE_DIRECTION(-0.110322, -0.993896, 0.000000)));
@86 = CIRCLE( 0.280995, AXIS2_PLACEMENT(#64, #76,
       THREE_SPACE_DIRECTION(-0.110322, -0.993896, 0.000000)));
@87 = LINE( #16, #74);
@88 = LINE( #17, #75);
@89 = CIRCLE( 0.500000, AXIS2_PLACEMENT(#65, #76,
       THREE_SPACE_DIRECTION(0.000000, -1.000000, 0.000000)));
@90 = CIRCLE( 0.140500, AXIS2_PLACEMENT(#18, #76,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@91 = CIRCLE( 0.140500, AXIS2_PLACEMENT(#20, #76,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@92 = CIRCLE( 0.140500, AXIS2_PLACEMENT(#22, #75,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@93 = CIRCLE( 0.140500, AXIS2_PLACEMENT(#24, #75,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@94 = CIRCLE( 0.101500, AXIS2_PLACEMENT(#26, #74,
       THREE_SPACE_DIRECTION(0.000000, 1.000000, 0.000000)));
@95 = CIRCLE( 0.101500, AXIS2_PLACEMENT(#28, #74,
       THREE_SPACE_DIRECTION(0.000000, 1.000000, 0.000000)));
@96 = CIRCLE( 0.140500, AXIS2_PLACEMENT(#30, #74,
       THREE_SPACE_DIRECTION(0.000000, 1.000000, 0.000000)));
@97 = CIRCLE( 0.140500, AXIS2_PLACEMENT(#53, #76,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@98 = CIRCLE( 0.250000, AXIS2_PLACEMENT(#53, #76,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@99 = CIRCLE( 0.140500, AXIS2_PLACEMENT(#56, #76,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@100 = CIRCLE( 0.250000, AXIS2_PLACEMENT(#56, #76,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@101 = LINE( #32, #74);
@102 = CIRCLE( 0.280995, AXIS2_PLACEMENT(#67, #76,
       THREE_SPACE_DIRECTION(-0.993896, 0.110322, 0.000000)));
@103 = CIRCLE( 0.280995, AXIS2_PLACEMENT(#68, #76,
       THREE_SPACE_DIRECTION(-0.993896, 0.110322, 0.000000)));
@104 = CIRCLE( 0.500000, AXIS2_PLACEMENT(#69, #76,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@105 = LINE( #38, #75);
@106 = LINE( #40, #74);
```

```
@107 = CIRCLE( 0.500000, AXIS2_PLACEMENT(#70, #76,
       THREE_SPACE_DIRECTION(0.000000, -1.000000, 0.000000)));
@108 = LINE( #41, #75);
@109 = CIRCLE( 0.280995, AXIS2_PLACEMENT(#71, #76,
       THREE_SPACE_DIRECTION(-0.110322, -0.993896, 0.000000)));
@110 = CIRCLE( 0.280995, AXIS2_PLACEMENT(#72, #76,
       THREE_SPACE_DIRECTION(-0.110322, -0.993896, 0.000000)));
@111 = LINE( #47, #74);
@112 = LINE( #48, #74);
@113 = CIRCLE( 0.500000, AXIS2_PLACEMENT(#73, #76,
       THREE_SPACE_DIRECTION(0.000000, -1.000000, 0.000000)));
@114 = CIRCLE( 0.250000, AXIS2_PLACEMENT(#49, #76,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@115 = CIRCLE( 0.250000, AXIS2_PLACEMENT(#51, #76,
       THREE_SPACE_DIRECTION(1.000000, 0.000000, 0.000000)));
@116 = LINE( #1, #76);
@117 = LINE( #2, #76);
@118 = LINE( #3, #76);
@119 = LINE( #4, #76);
@120 = LINE( #5, #76);
@121 = LINE( #6, #76);
@122 = LINE( #7, #76);
@123 = LINE( #8, #76);
@124 = LINE( #9, #76);
@125 = LINE( #10, #76);
@126 = LINE( #11, #76);
@127 = LINE( #12, #76);
@128 = LINE( #13, #76);
@129 = LINE( #14, #76);
@130 = LINE( #15, #76);
@131 = LINE( #16, #76);
@132 = LINE( #17, #76);
!* Done : Curves *!

@133 = PLANE( AXIS2_PLACEMENT(
       CARTESIAN_THREE_COORDINATE( 0.000000, 0.000000, 0.000000), #75));
@134 = CYLINDRICAL_SURFACE(  0.281000, AXIS2_PLACEMENT(#59, #76));
@135 = CYLINDRICAL_SURFACE(  0.281000, AXIS2_PLACEMENT(#60, #76));
@136 = CYLINDRICAL_SURFACE(  0.281000, AXIS2_PLACEMENT(#61, #76));
@137 = PLANE( AXIS2_PLACEMENT(
       CARTESIAN_THREE_COORDINATE( 3.812000, 0.000000, 0.000000), #74));
@138 = PLANE( AXIS2_PLACEMENT(
       CARTESIAN_THREE_COORDINATE( 0.000000, 0.703000, 0.000000), #75));
@139 = CYLINDRICAL_SURFACE(  0.500000, AXIS2_PLACEMENT(#62, #76));
@140 = PLANE( AXIS2_PLACEMENT(
       CARTESIAN_THREE_COORDINATE( 0.703000, 0.000000, 0.000000), #74));
@141 = CYLINDRICAL_SURFACE(  0.281000, AXIS2_PLACEMENT(#63, #76));
@142 = CYLINDRICAL_SURFACE(  0.281000, AXIS2_PLACEMENT(#64, #76));
@143 = PLANE( AXIS2_PLACEMENT(
       CARTESIAN_THREE_COORDINATE( 0.000000, 5.000000, 0.000000), #75));
@144 = PLANE( AXIS2_PLACEMENT(
       CARTESIAN_THREE_COORDINATE( 0.000000, 0.000000, 0.000000), #74));
@145 = CYLINDRICAL_SURFACE(  0.500000, AXIS2_PLACEMENT(#65, #76));
@146 = CYLINDRICAL_SURFACE(  0.281000, AXIS2_PLACEMENT(#22, #75));
```

```
@147 = CYLINDRICAL_SURFACE(  0.203000, AXIS2_PLACEMENT(#26, #74));
@148 = CONICAL_SURFACE(  45.008690,  0.0, AXIS2_PLACEMENT(#66, #74));
@149 = CYLINDRICAL_SURFACE(  0.140500, AXIS2_PLACEMENT(#18, #76));
@150 = PLANE( AXIS2_PLACEMENT(
       CARTESIAN_THREE_COORDINATE( 0.000000, 0.000000, 0.468750), #76));
@151 = CYLINDRICAL_SURFACE(  0.250000, AXIS2_PLACEMENT(#53, #76));
@152 = CYLINDRICAL_SURFACE(  0.140500, AXIS2_PLACEMENT(#20, #76));
@153 = CYLINDRICAL_SURFACE(  0.250000, AXIS2_PLACEMENT(#56, #76));
@154 = PLANE( AXIS2_PLACEMENT(
       CARTESIAN_THREE_COORDINATE( 0.000000, 0.000000, 0.000000), #76));
@155 = PLANE( AXIS2_PLACEMENT(
       CARTESIAN_THREE_COORDINATE( 0.000000, 0.000000, 0.500000), #76));
!* Done : Surfaces *!

@156 = VERTEX(  #1);
@157 = VERTEX(  #2);
@158 = VERTEX(  #3);
@159 = VERTEX(  #4);
@160 = VERTEX(  #5);
@161 = VERTEX(  #6);
@162 = VERTEX(  #7);
@163 = VERTEX(  #8);
@164 = VERTEX(  #9);
@165 = VERTEX(  #10);
@166 = VERTEX(  #11);
@167 = VERTEX(  #12);
@168 = VERTEX(  #13);
@169 = VERTEX(  #14);
@170 = VERTEX(  #15);
@171 = VERTEX(  #16);
@172 = VERTEX(  #17);
@173 = VERTEX(  #19);
@174 = VERTEX(  #21);
@175 = VERTEX(  #23);
@176 = VERTEX(  #25);
@177 = VERTEX(  #27);
@178 = VERTEX(  #29);
@179 = VERTEX(  #31);
@180 = VERTEX(  #32);
@181 = VERTEX(  #33);
@182 = VERTEX(  #34);
@183 = VERTEX(  #35);
@184 = VERTEX(  #36);
@185 = VERTEX(  #37);
@186 = VERTEX(  #38);
@187 = VERTEX(  #39);
@188 = VERTEX(  #40);
@189 = VERTEX(  #41);
@190 = VERTEX(  #42);
@191 = VERTEX(  #43);
@192 = VERTEX(  #44);
@193 = VERTEX(  #45);
@194 = VERTEX(  #46);
@195 = VERTEX(  #47);
```

```
@196 = VERTEX(  #48);
@197 = VERTEX(  #50);
@198 = VERTEX(  #52);
@199 = VERTEX(  #54);
@200 = VERTEX(  #55);
@201 = VERTEX(  #57);
@202 = VERTEX(  #58);
!* Done : Vertices *!

@203 = EDGE( #156, #157, CURVE_LOGICAL_STRUCTURE(#77, .T.));
@204 = EDGE( #158, #157, CURVE_LOGICAL_STRUCTURE(#78, .T.));
@205 = EDGE( #160, #159, CURVE_LOGICAL_STRUCTURE(#79, .T.));
@206 = EDGE( #161, #162, CURVE_LOGICAL_STRUCTURE(#80, .T.));
@207 = EDGE( #162, #163, CURVE_LOGICAL_STRUCTURE(#81, .T.));
@208 = EDGE( #164, #163, CURVE_LOGICAL_STRUCTURE(#82, .T.));
@209 = EDGE( #165, #164, CURVE_LOGICAL_STRUCTURE(#83, .T.));
@210 = EDGE( #165, #166, CURVE_LOGICAL_STRUCTURE(#84, .T.));
@211 = EDGE( #167, #166, CURVE_LOGICAL_STRUCTURE(#85, .T.));
@212 = EDGE( #169, #168, CURVE_LOGICAL_STRUCTURE(#86, .T.));
@213 = EDGE( #171, #170, CURVE_LOGICAL_STRUCTURE(#87, .T.));
@214 = EDGE( #172, #171, CURVE_LOGICAL_STRUCTURE(#88, .T.));
@215 = EDGE( #172, #156, CURVE_LOGICAL_STRUCTURE(#89, .T.));
@216 = EDGE( #173, #173, CURVE_LOGICAL_STRUCTURE(#90, .T.));
@217 = EDGE( #174, #174, CURVE_LOGICAL_STRUCTURE(#91, .T.));
@218 = EDGE( #175, #175, CURVE_LOGICAL_STRUCTURE(#92, .T.));
@219 = EDGE( #176, #176, CURVE_LOGICAL_STRUCTURE(#93, .T.));
@220 = EDGE( #177, #177, CURVE_LOGICAL_STRUCTURE(#94, .T.));
@221 = EDGE( #178, #178, CURVE_LOGICAL_STRUCTURE(#95, .T.));
@222 = EDGE( #179, #179, CURVE_LOGICAL_STRUCTURE(#96, .T.));
@223 = EDGE( #199, #199, CURVE_LOGICAL_STRUCTURE(#97, .T.));
@224 = EDGE( #200, #200, CURVE_LOGICAL_STRUCTURE(#98, .T.));
@225 = EDGE( #201, #201, CURVE_LOGICAL_STRUCTURE(#99, .T.));
@226 = EDGE( #202, #202, CURVE_LOGICAL_STRUCTURE(#100, .T.));
@227 = EDGE( #180, #181, CURVE_LOGICAL_STRUCTURE(#101, .T.));
@228 = EDGE( #182, #181, CURVE_LOGICAL_STRUCTURE(#102, .T.));
@229 = EDGE( #184, #183, CURVE_LOGICAL_STRUCTURE(#103, .T.));
@230 = EDGE( #185, #186, CURVE_LOGICAL_STRUCTURE(#104, .T.));
@231 = EDGE( #186, #187, CURVE_LOGICAL_STRUCTURE(#105, .T.));
@232 = EDGE( #188, #187, CURVE_LOGICAL_STRUCTURE(#106, .T.));
@233 = EDGE( #189, #188, CURVE_LOGICAL_STRUCTURE(#107, .T.));
@234 = EDGE( #189, #190, CURVE_LOGICAL_STRUCTURE(#108, .T.));
@235 = EDGE( #191, #190, CURVE_LOGICAL_STRUCTURE(#109, .T.));
@236 = EDGE( #193, #192, CURVE_LOGICAL_STRUCTURE(#110, .T.));
@237 = EDGE( #195, #194, CURVE_LOGICAL_STRUCTURE(#111, .T.));
@238 = EDGE( #196, #195, CURVE_LOGICAL_STRUCTURE(#112, .T.));
@239 = EDGE( #196, #180, CURVE_LOGICAL_STRUCTURE(#113, .T.));
@240 = EDGE( #197, #197, CURVE_LOGICAL_STRUCTURE(#114, .T.));
@241 = EDGE( #198, #198, CURVE_LOGICAL_STRUCTURE(#115, .T.));
@242 = EDGE( #156, #180, CURVE_LOGICAL_STRUCTURE(#116, .T.));
@243 = EDGE( #157, #181, CURVE_LOGICAL_STRUCTURE(#117, .T.));
@244 = EDGE( #158, #182, CURVE_LOGICAL_STRUCTURE(#118, .T.));
@245 = EDGE( #159, #183, CURVE_LOGICAL_STRUCTURE(#119, .T.));
@246 = EDGE( #160, #184, CURVE_LOGICAL_STRUCTURE(#120, .T.));
@247 = EDGE( #161, #185, CURVE_LOGICAL_STRUCTURE(#121, .T.));
```

```
@248 = EDGE( #162, #186, CURVE_LOGICAL_STRUCTURE(#122, .T.));
@249 = EDGE( #163, #187, CURVE_LOGICAL_STRUCTURE(#123, .T.));
@250 = EDGE( #164, #188, CURVE_LOGICAL_STRUCTURE(#124, .T.));
@251 = EDGE( #165, #189, CURVE_LOGICAL_STRUCTURE(#125, .T.));
@252 = EDGE( #166, #190, CURVE_LOGICAL_STRUCTURE(#126, .T.));
@253 = EDGE( #167, #191, CURVE_LOGICAL_STRUCTURE(#127, .T.));
@254 = EDGE( #168, #192, CURVE_LOGICAL_STRUCTURE(#128, .T.));
@255 = EDGE( #169, #193, CURVE_LOGICAL_STRUCTURE(#129, .T.));
@256 = EDGE( #170, #194, CURVE_LOGICAL_STRUCTURE(#130, .T.));
@257 = EDGE( #171, #195, CURVE_LOGICAL_STRUCTURE(#131, .T.));
@258 = EDGE( #172, #196, CURVE_LOGICAL_STRUCTURE(#132, .T.));
@259 = EDGE( #158, #159, CURVE_LOGICAL_STRUCTURE(#77, .T.));
@260 = EDGE( #160, #161, CURVE_LOGICAL_STRUCTURE(#77, .T.));
@261 = EDGE( #167, #168, CURVE_LOGICAL_STRUCTURE(#84, .T.));
@262 = EDGE( #169, #170, CURVE_LOGICAL_STRUCTURE(#84, .T.));
@263 = EDGE( #182, #183, CURVE_LOGICAL_STRUCTURE(#101, .T.));
@264 = EDGE( #184, #185, CURVE_LOGICAL_STRUCTURE(#101, .T.));
@265 = EDGE( #191, #192, CURVE_LOGICAL_STRUCTURE(#108, .T.));
@266 = EDGE( #193, #194, CURVE_LOGICAL_STRUCTURE(#108, .T.));
!* Done : Edges *!

@267 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#203, .T.),
       EDGE_LOGICAL_STRUCTURE(#243, .T.),
       EDGE_LOGICAL_STRUCTURE(#227, .F.),
       EDGE_LOGICAL_STRUCTURE(#242, .F.) ));
@268 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#204, .F.),
       EDGE_LOGICAL_STRUCTURE(#244, .T.),
       EDGE_LOGICAL_STRUCTURE(#228, .T.),
       EDGE_LOGICAL_STRUCTURE(#243, .F.) ));
@269 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#205, .F.),
       EDGE_LOGICAL_STRUCTURE(#246, .T.),
       EDGE_LOGICAL_STRUCTURE(#229, .T.),
       EDGE_LOGICAL_STRUCTURE(#245, .F.) ));
@270 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#206, .T.),
       EDGE_LOGICAL_STRUCTURE(#248, .T.),
       EDGE_LOGICAL_STRUCTURE(#230, .F.),
       EDGE_LOGICAL_STRUCTURE(#247, .F.) ));
@271 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#207, .T.),
       EDGE_LOGICAL_STRUCTURE(#249, .T.),
       EDGE_LOGICAL_STRUCTURE(#231, .F.),
       EDGE_LOGICAL_STRUCTURE(#248, .F.) ));
@272 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#208, .F.),
       EDGE_LOGICAL_STRUCTURE(#250, .T.),
       EDGE_LOGICAL_STRUCTURE(#232, .T.),
       EDGE_LOGICAL_STRUCTURE(#249, .F.) ));
@273 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#209, .F.),
       EDGE_LOGICAL_STRUCTURE(#251, .T.),
```

```
        EDGE_LOGICAL_STRUCTURE(#233, .T.),
        EDGE_LOGICAL_STRUCTURE(#250, .F.) ));
@274 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#210, .T.),
        EDGE_LOGICAL_STRUCTURE(#252, .T.),
        EDGE_LOGICAL_STRUCTURE(#234, .F.),
        EDGE_LOGICAL_STRUCTURE(#251, .F.) ));
@275 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#211, .F.),
        EDGE_LOGICAL_STRUCTURE(#253, .T.),
        EDGE_LOGICAL_STRUCTURE(#235, .T.),
        EDGE_LOGICAL_STRUCTURE(#252, .F.) ));
@276 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#212, .F.),
        EDGE_LOGICAL_STRUCTURE(#255, .T.),
        EDGE_LOGICAL_STRUCTURE(#236, .T.),
        EDGE_LOGICAL_STRUCTURE(#254, .F.) ));
@277 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#213, .F.),
        EDGE_LOGICAL_STRUCTURE(#257, .T.),
        EDGE_LOGICAL_STRUCTURE(#237, .T.),
        EDGE_LOGICAL_STRUCTURE(#256, .F.) ));
@278 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#214, .F.),
        EDGE_LOGICAL_STRUCTURE(#258, .T.),
        EDGE_LOGICAL_STRUCTURE(#238, .T.),
        EDGE_LOGICAL_STRUCTURE(#257, .F.) ));
@279 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#215, .T.),
        EDGE_LOGICAL_STRUCTURE(#242, .T.),
        EDGE_LOGICAL_STRUCTURE(#239, .F.),
        EDGE_LOGICAL_STRUCTURE(#258, .F.) ));
@280 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#218, .F.) ));
@281 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#220, .F.) ));
@282 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#221, .F.) ));
@283 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#216, .F.) ));
@284 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#224, .T.) ));
@285 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#224, .F.) ));
@286 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#217, .F.) ));
@287 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#226, .F.) ));
@288 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#203, .F.),
        EDGE_LOGICAL_STRUCTURE(#215, .F.),
        EDGE_LOGICAL_STRUCTURE(#214, .T.),
        EDGE_LOGICAL_STRUCTURE(#213, .T.),
        EDGE_LOGICAL_STRUCTURE(#262, .F.),
```

```
        EDGE_LOGICAL_STRUCTURE(#212, .T.),
        EDGE_LOGICAL_STRUCTURE(#261, .F.),
        EDGE_LOGICAL_STRUCTURE(#211, .T.),
        EDGE_LOGICAL_STRUCTURE(#210, .F.),
        EDGE_LOGICAL_STRUCTURE(#209, .T.),
        EDGE_LOGICAL_STRUCTURE(#208, .T.),
        EDGE_LOGICAL_STRUCTURE(#207, .F.),
        EDGE_LOGICAL_STRUCTURE(#206, .F.),
        EDGE_LOGICAL_STRUCTURE(#260, .F.),
        EDGE_LOGICAL_STRUCTURE(#205, .T.),
        EDGE_LOGICAL_STRUCTURE(#259, .F.),
        EDGE_LOGICAL_STRUCTURE(#204, .T.) ));
@289 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#227, .T.),
        EDGE_LOGICAL_STRUCTURE(#228, .F.),
        EDGE_LOGICAL_STRUCTURE(#263, .T.),
        EDGE_LOGICAL_STRUCTURE(#229, .F.),
        EDGE_LOGICAL_STRUCTURE(#264, .T.),
        EDGE_LOGICAL_STRUCTURE(#230, .T.),
        EDGE_LOGICAL_STRUCTURE(#231, .T.),
        EDGE_LOGICAL_STRUCTURE(#232, .F.),
        EDGE_LOGICAL_STRUCTURE(#233, .F.),
        EDGE_LOGICAL_STRUCTURE(#234, .T.),
        EDGE_LOGICAL_STRUCTURE(#235, .F.),
        EDGE_LOGICAL_STRUCTURE(#265, .T.),
        EDGE_LOGICAL_STRUCTURE(#236, .F.),
        EDGE_LOGICAL_STRUCTURE(#266, .T.),
        EDGE_LOGICAL_STRUCTURE(#237, .F.),
        EDGE_LOGICAL_STRUCTURE(#238, .F.),
        EDGE_LOGICAL_STRUCTURE(#239, .T.) ));
@290 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#259, .T.),
        EDGE_LOGICAL_STRUCTURE(#245, .T.),
        EDGE_LOGICAL_STRUCTURE(#263, .F.),
        EDGE_LOGICAL_STRUCTURE(#244, .F.) ));
@291 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#260, .T.),
        EDGE_LOGICAL_STRUCTURE(#247, .T.),
        EDGE_LOGICAL_STRUCTURE(#264, .F.),
        EDGE_LOGICAL_STRUCTURE(#246, .F.) ));
@292 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#261, .T.),
        EDGE_LOGICAL_STRUCTURE(#254, .T.),
        EDGE_LOGICAL_STRUCTURE(#265, .F.),
        EDGE_LOGICAL_STRUCTURE(#253, .F.) ));
@293 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#262, .T.),
        EDGE_LOGICAL_STRUCTURE(#256, .T.),
        EDGE_LOGICAL_STRUCTURE(#266, .F.),
        EDGE_LOGICAL_STRUCTURE(#255, .F.) ));
@294 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#226, .T.) ));
@295 = EDGE_LOOP( (
        EDGE_LOGICAL_STRUCTURE(#218, .T.) ));
```

```
@296 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#219, .F.) ));
@297 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#219, .T.) ));
@298 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#220, .T.) ));
@299 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#221, .T.) ));
@300 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#222, .T.) ));
@301 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#222, .F.) ));
@302 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#216, .T.) ));
@303 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#223, .T.) ));
@304 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#223, .F.) ));
@305 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#240, .T.) ));
@306 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#240, .F.) ));
@307 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#217, .T.) ));
@308 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#225, .T.) ));
@309 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#225, .F.) ));
@310 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#241, .T.) ));
@311 = EDGE_LOOP( (
       EDGE_LOGICAL_STRUCTURE(#241, .F.) ));
!* Done : Loops *!

@312 = FACE( , (
       LOOP_LOGICAL_STRUCTURE(#267, .T.)  ),
       SURFACE_LOGICAL_STRUCTURE(#133, .F.));
@313 = FACE( , (
       LOOP_LOGICAL_STRUCTURE(#268, .T.)  ),
       SURFACE_LOGICAL_STRUCTURE(#134, .T.));
@314 = FACE( , (
       LOOP_LOGICAL_STRUCTURE(#269, .T.)  ),
       SURFACE_LOGICAL_STRUCTURE(#135, .T.));
@315 = FACE( , (
       LOOP_LOGICAL_STRUCTURE(#270, .T.)  ),
       SURFACE_LOGICAL_STRUCTURE(#136, .F.));
@316 = FACE( , (
       LOOP_LOGICAL_STRUCTURE(#271, .T.)  ),
       SURFACE_LOGICAL_STRUCTURE(#137, .T.));
@317 = FACE( , (
       LOOP_LOGICAL_STRUCTURE(#272, .T.),
       LOOP_LOGICAL_STRUCTURE(#296, .T.)  ),
       SURFACE_LOGICAL_STRUCTURE(#138, .T.));
@318 = FACE( , (
```

```
        LOOP_LOGICAL_STRUCTURE(#273, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#139, .T.));
@319 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#274, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#140, .T.));
@320 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#275, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#141, .T.));
@321 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#276, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#142, .T.));
@322 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#277, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#143, .T.));
@323 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#278, .T.),
        LOOP_LOGICAL_STRUCTURE(#298, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#144, .F.));
@324 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#279, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#145, .F.));
@325 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#280, .T.),
        LOOP_LOGICAL_STRUCTURE(#297, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#146, .T.));
@326 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#281, .T.),
        LOOP_LOGICAL_STRUCTURE(#299, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#147, .T.));
@327 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#282, .T.),
        LOOP_LOGICAL_STRUCTURE(#300, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#148, .T.));
@328 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#283, .T.),
        LOOP_LOGICAL_STRUCTURE(#303, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#149, .T.));
@329 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#284, .T.),
        LOOP_LOGICAL_STRUCTURE(#304, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#150, .T.));
@330 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#285, .T.),
        LOOP_LOGICAL_STRUCTURE(#305, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#151, .T.));
@331 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#286, .T.),
        LOOP_LOGICAL_STRUCTURE(#308, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#152, .T.));
@332 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#287, .T.),
        LOOP_LOGICAL_STRUCTURE(#310, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#153, .T.));
@333 = FACE( , (
```

```
        LOOP_LOGICAL_STRUCTURE(#288, .T.),
        LOOP_LOGICAL_STRUCTURE(#302, .T.),
        LOOP_LOGICAL_STRUCTURE(#307, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#154, .F.));
@334 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#289, .T.),
        LOOP_LOGICAL_STRUCTURE(#306, .T.),
        LOOP_LOGICAL_STRUCTURE(#311, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#155, .T.));
@335 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#290, .T.),
        LOOP_LOGICAL_STRUCTURE(#295, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#133, .F.));
@336 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#291, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#133, .F.));
@337 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#292, .T.),
        LOOP_LOGICAL_STRUCTURE(#301, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#140, .T.));
@338 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#293, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#140, .T.));
@339 = FACE( , (
        LOOP_LOGICAL_STRUCTURE(#294, .T.),
        LOOP_LOGICAL_STRUCTURE(#309, .T.)  ),
        SURFACE_LOGICAL_STRUCTURE(#150, .T.));
!* Done : Faces *!

@340 = CLOSED_SHELL( (
         FACE_LOGICAL_STRUCTURE(#312,.T.),
         FACE_LOGICAL_STRUCTURE(#313,.T.),
         FACE_LOGICAL_STRUCTURE(#314,.T.),
         FACE_LOGICAL_STRUCTURE(#315,.T.),
         FACE_LOGICAL_STRUCTURE(#316,.T.),
         FACE_LOGICAL_STRUCTURE(#317,.T.),
         FACE_LOGICAL_STRUCTURE(#318,.T.),
         FACE_LOGICAL_STRUCTURE(#319,.T.),
         FACE_LOGICAL_STRUCTURE(#320,.T.),
         FACE_LOGICAL_STRUCTURE(#321,.T.),
         FACE_LOGICAL_STRUCTURE(#322,.T.),
         FACE_LOGICAL_STRUCTURE(#323,.T.),
         FACE_LOGICAL_STRUCTURE(#324,.T.),
         FACE_LOGICAL_STRUCTURE(#325,.T.),
         FACE_LOGICAL_STRUCTURE(#326,.T.),
         FACE_LOGICAL_STRUCTURE(#327,.T.),
         FACE_LOGICAL_STRUCTURE(#328,.T.),
         FACE_LOGICAL_STRUCTURE(#329,.T.),
         FACE_LOGICAL_STRUCTURE(#330,.T.),
         FACE_LOGICAL_STRUCTURE(#331,.T.),
         FACE_LOGICAL_STRUCTURE(#332,.T.),
         FACE_LOGICAL_STRUCTURE(#333,.T.),
         FACE_LOGICAL_STRUCTURE(#334,.T.),
         FACE_LOGICAL_STRUCTURE(#335,.T.),
```

```
        FACE_LOGICAL_STRUCTURE(#336,.T.),
        FACE_LOGICAL_STRUCTURE(#337,.T.),
        FACE_LOGICAL_STRUCTURE(#338,.T.),
        FACE_LOGICAL_STRUCTURE(#339,.T.)    ));
!* Done : Shells *!

ENDSEC;

ENDSTEP;
```

NBS-114A (REV. 2-8C)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NBSIR 88-3818	2. Performing Organ. Report No.	3. Publication Date JULY 1988

4. TITLE AND SUBTITLE

Converting the AMRF Part Model Report to a PDES/STEP Subset

5. AUTHOR(S)

Y. Tina Lee and Sanford P. Ressler

6. PERFORMING ORGANIZATION (*If joint or other than NBS, see instructions*)

NATIONAL BUREAU OF STANDARDS
U.S. DEPARTMENT OF COMMERCE
GAITHERSBURG, MD 20899

7. Contract/Grant No.

8. Type of Report & Period Covered

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (*Street, City, State, ZIP*)

National Bureau of Standards
Gaithersburg, Maryland 20899

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (*A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here*)

This paper identifies the process through which the Topology and Geometry of product data defined in the AMRF (Automated Manufacturing Research Facility) Part Model database report are converted to the PDES (Product Data Exchange specification)/STEP (Standard for the Exchange of Product Model Data) physical file.

A file conversion program, which converts the AMRF Part Model report (topology and geometry only) to the PDES/STEP file, has been implemented. The PDES/STEP file can be generated 100% automatically by executing this program.

12. KEY WORDS (*Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons*)

AMRF, AMRF Part Model, CAD/CAM, data exchange, product data, PDES, STEP

13. AVAILABILITY

☒ Unlimited
☐ For Official Distribution. Do Not Release to NTIS
☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.
☒ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES 41

15. Price $11.95