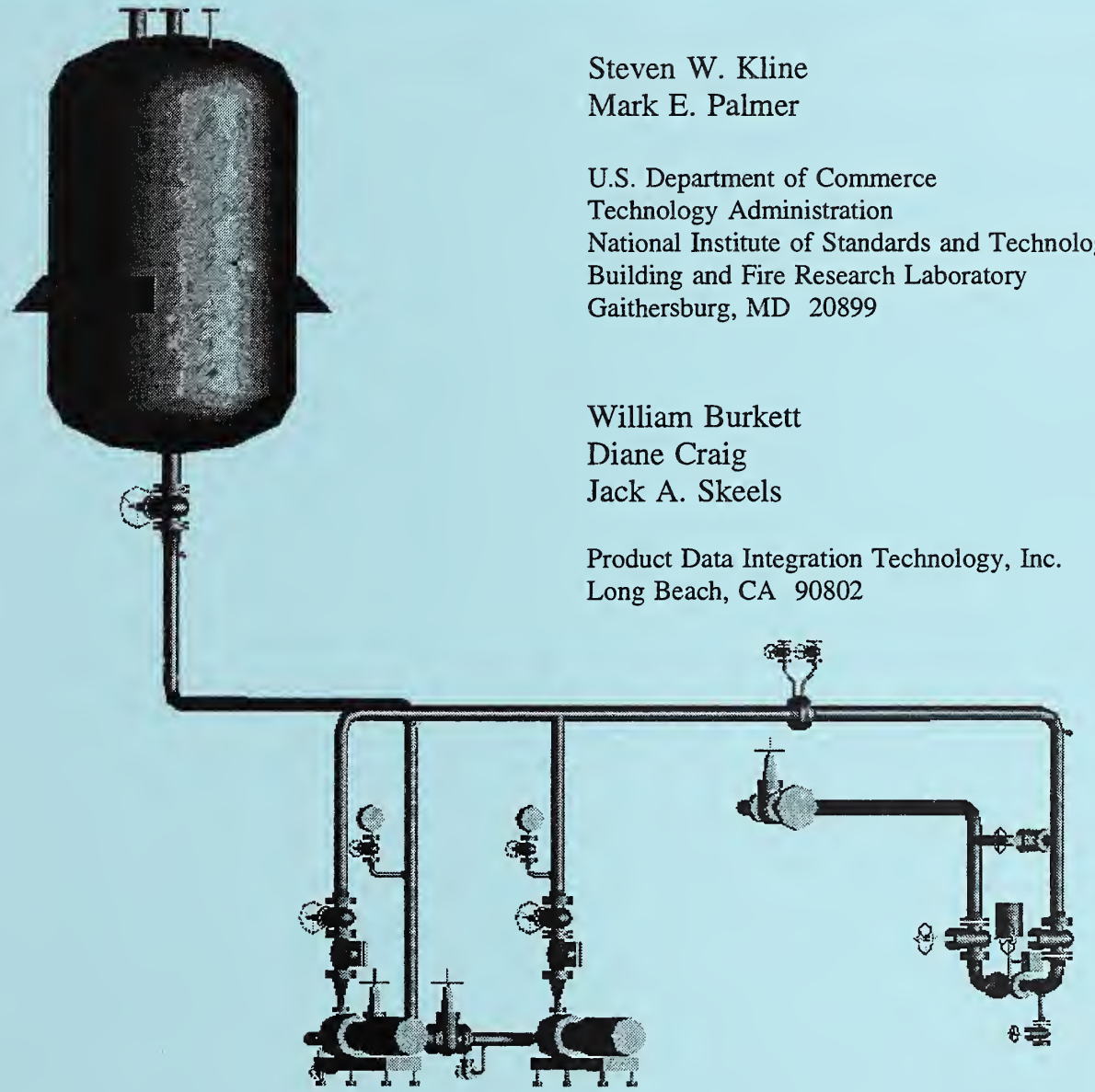
NISTIR 5812

Plant Spatial Configuration Application Protocol Version 1.0 — Volume 2



Steven W. Kline
Mark E. Palmer

U.S. Department of Commerce
Technology Administration
National Institute of Standards and Technology
Building and Fire Research Laboratory
Gaithersburg, MD 20899

William Burkett
Diane Craig
Jack A. Skeels

Product Data Integration Technology, Inc.
Long Beach, CA 90802

NIST

United States Department of Commerce
Technology Administration
National Institute of Standards and Technology

NISTIR 5812

Plant Spatial Configuration Application Protocol Version 1.0 — Volume 2

Steven W. Kline
Mark E. Palmer

U.S. Department of Commerce
Technology Administration
National Institute of Standards and Technology
Building and Fire Research Laboratory
Gaithersburg, MD 20899

William Burkett
Diane Craig
Jack A. Skeels

Product Data Integration Technology, Inc.
Long Beach, CA 90802

December 1995

U.S. Department of Commerce
Ronald H. Brown, *Secretary*

Technology Administration
Mary L. Good, *Under Secretary for Technology*

National Institute of Standards and Technology
Arati A. Prabhakar, *Director*

PREFACE

Industry and government require comprehensive and reliable information exchange mechanisms to effectively integrate computer-aided (CAx) systems and evolving information technologies. Subcommittee Four (Industrial data and global manufacturing programming languages) of the International Organization for Standardization (ISO) Technical Committee 184 (Industrial automation systems and integration), ISO TC184/SC4, is preparing ISO 10303, a set of international standards titled *Industrial automation systems and integration - Product data representation and exchange*. The set of proposed standards is informally known as STEP (STandard for the Exchange of Product model data).

ISO 10303 will provide a neutral mechanism for describing product data throughout the life cycle of a product, independent of any particular CAx system. ISO 10303 is suitable for file exchange and for implementing, sharing, and archiving product databases. The development of ISO 10303 is based upon the use of information models, a framework for product data modelling, formal data specification languages, and an architecture that separates information requirements from implementation methods.

A fundamental concept of STEP is the definition of application protocols (APs) as the mechanism for specifying information requirements and for ensuring reliable communication. An **application protocol** is a Part of ISO 10303 that defines the context, scope, and information requirements for designated application(s) and specifies the resource constructs used to satisfy these requirements. The scope of an AP is defined by the type of product, the supported stages in the life cycle of the product, the required types of product data, the uses of the product data, and the disciplines that use the product data. Additionally, an AP enumerates the conformance requirements for conformance testing of implementations of the AP.

Part 227 of ISO 10303, *Plant Spatial Configuration*, specifies an AP for the exchange of spatial configuration information of process plants. This document constitutes the Committee Draft (CD) documentation for AP 227. Volume 1 of this document contains clauses covering the AP 227 scope, normative references, definitions and abbreviations, information requirements, application interpreted model, and conformance requirements. Volume 2 of this document contains the normative and informative annexes.

The *Plant Spatial Configuration Application Protocol* has been submitted for international review and balloting as an ISO CD document.

Contents

Page

Volume 1

Volume 2

Annexes

Figures

Tables

Annex A
(normative)

AIM EXPRESS expanded listing

The following EXPRESS is the expanded form of the short form schema given in 5.2. In the event of any discrepancy between the short form and this expanded listing, the expanded listing shall be used.

```
*)
SCHEMA plant_spatial_configuration;

  TYPE ahead_or_behind = ENUMERATION OF
    (ahead,
     behind);
  END_TYPE; -- ahead_or_behind

  TYPE amount_of_substance_measure = REAL;
  END_TYPE; -- amount_of_substance_measure

  TYPE angle_relator = ENUMERATION OF
    (equal,
     large,
     small);
  END_TYPE; -- angle_relator

  TYPE approval_item = SELECT
    (change_action);
  END_TYPE; -- approval_item

  TYPE area_measure = REAL;
  END_TYPE; -- area_measure

  TYPE axis2_placement = SELECT
    (axis2_placement_2d,
     axis2_placement_3d);
  END_TYPE; -- axis2_placement

  TYPE b_spline_curve_form = ENUMERATION OF
    (polyline_form,
     circular_arc,
     elliptic_arc,
     parabolic_arc,
     hyperbolic_arc,
     unspecified);
  END_TYPE; -- b_spline_curve_form

  TYPE b_spline_surface_form = ENUMERATION OF
    (plane_surf,
     cylindrical_surf,
     conical_surf,
     spherical_surf,
     toroidal_surf,
     surf_of_revolution,
     ruled_surf,
     generalised_cone,
     quadric_surf,
```

```
    surf_of_linear_extrusion,
    unspecified);
END_TYPE; -- b_spline_surface_form

TYPE boolean_operand = SELECT
  (solid_model,
   csg_primitive,
   boolean_result);
END_TYPE; -- boolean_operand

TYPE boolean_operator = ENUMERATION OF
  (union,
   intersection,
   difference);
END_TYPE; -- boolean_operator

TYPE change_delta_item = SELECT
  (assembly_component_usage,
   line_branch_connection,
   line_plant_item_branch_connection,
   line_plant_item_connection,
   line_termination_connection,
   plant,
   product_definition_relationship);
END_TYPE; -- change_delta_item

TYPE change_item = SELECT
  (change_delta_item,
   electrical_system,
   externally_defined_plant_item_definition,
   hvac_system,
   instrumentation_and_control_system,
   material_property,
   piping_system,
   plant_item_connection,
   plant_item_connector,
   plant_item_definition,
   plant_line_definition,
   plant_line_segment_definition,
   plant_line_segment_termination,
   process_capability,
   product_definition,
   product_definition_shape,
   property_definition,
   reference_geometry,
   structural_system);
END_TYPE; -- change_item

TYPE characterized_definition = SELECT
  (characterized_object,
   characterized_product_definition,
   shape_definition);
END_TYPE; -- characterized_definition

TYPE characterized_material_property = SELECT
  (material_property);
END_TYPE; -- characterized_material_property
```

```
TYPE characterized_product_definition = SELECT
  (product_definition,
   product_definition_relationship);
END_TYPE; -- characterized_product_definition

TYPE classification_item = SELECT
  (plant_item_connection,
   plant_item_connector,
   piping_component_definition);
END_TYPE; -- classification_item

TYPE context_dependent_measure = REAL;
END_TYPE; -- context_dependent_measure

TYPE count_measure = NUMBER;
END_TYPE; -- count_measure

TYPE csg_primitive = SELECT
  (sphere,
   block,
   torus,
   right_circular_cone,
   right_circular_cylinder);
END_TYPE; -- csg_primitive

TYPE csg_select = SELECT
  (boolean_result,
   csg_primitive);
END_TYPE; -- csg_select

TYPE curve_on_surface = SELECT
  (pcurve,
   surface_curve,
   composite_curve_on_surface);
END_TYPE; -- curve_on_surface

TYPE date_time_select = SELECT
  (date,
   local_time,
   date_and_time);
END_TYPE; -- date_time_select

TYPE day_in_month_number = INTEGER;
END_TYPE; -- day_in_month_number

TYPE design_project_item = SELECT
  (plant_item_definition);
END_TYPE; -- design_project_item

TYPE dimension_count = INTEGER;
WHERE
  wr1: SELF > 0;
END_TYPE; -- dimension_count

TYPE dimensional_characteristic = SELECT
  (dimensional_location,
   dimensional_size);
END_TYPE; -- dimensional_characteristic
```

```
TYPE geometric_set_select = SELECT
  (point,
   curve,
   surface);
END_TYPE; -- geometric_set_select

TYPE hour_in_day = INTEGER;
WHERE
  wr1: (0 <= SELF) AND (SELF < 24);
END_TYPE; -- hour_in_day

TYPE identifier = STRING;
END_TYPE; -- identifier

TYPE knot_type = ENUMERATION OF
  (uniform_knots,
   unspecified,
   quasi_uniform_knots,
   piecewise_bezier_knots);
END_TYPE; -- knot_type

TYPE label = STRING;
END_TYPE; -- label

TYPE layered_item = SELECT
  (representation_item);
END_TYPE; -- layered_item

TYPE length_measure = REAL;
END_TYPE; -- length_measure

TYPE list_of_reversible_topology_item = LIST [0:?] OF
            reversible_topology_item;
END_TYPE; -- list_of_reversible_topology_item

TYPE mass_measure = REAL;
END_TYPE; -- mass_measure

TYPE measure_value = SELECT
  (length_measure,
   mass_measure,
   time_measure,
   thermodynamic_temperature_measure,
   amount_of_substance_measure,
   plane_angle_measure,
   solid_angle_measure,
   area_measure,
   volume_measure,
   ratio_measure,
   parameter_value,
   numeric_measure,
   context_dependent_measure,
   positive_length_measure,
   positive_plane_angle_measure,
   positive_ratio_measure,
   count_measure);
END_TYPE; -- measure_value
```

```
TYPE minute_in_hour = INTEGER;
WHERE
  wr1: (0 <= SELF) AND (SELF <= 59);
END_TYPE; -- minute_in_hour

TYPE month_in_year_number = INTEGER;
WHERE
  wr1: (1 <= SELF) AND (SELF <= 12);
END_TYPE; -- month_in_year_number

TYPE numeric_measure = NUMBER;
END_TYPE; -- numeric_measure

TYPE parameter_value = REAL;
END_TYPE; -- parameter_value

TYPE pcurve_or_surface = SELECT
  (pcurve,
   surface);
END_TYPE; -- pcurve_or_surface

TYPE person_organization_select = SELECT
  (person,
   organization,
   person_and_organization);
END_TYPE; -- person_organization_select

TYPE plane_angle_measure = REAL;
END_TYPE; -- plane_angle_measure

TYPE plant_spatial_configuration_action_request_item = SELECT
  (product);
END_TYPE; -- plant_spatial_configuration_action_request_item

TYPE plant_spatial_configuration_date_and_time_item = SELECT
  (product);
END_TYPE; -- plant_spatial_configuration_date_and_time_item

TYPE plant_spatial_configuration_dated_item = SELECT
  (change_action,
   action_directive,
   product,
   change_item);
END_TYPE; -- plant_spatial_configuration_dated_item

TYPE plant_spatial_configuration_document_item = SELECT
  (external_source,
   heat_tracing_representation,
   piping_component_definition,
   piping_system,
   plant_item_connector,
   plant_item_definition,
   plant_line_segment_definition,
   product,
   product_definition,
   product_definition_relationship,
   property_definition,
   representation,
   shape_dimension_representation,
```

```
    site);
END_TYPE; -- plant_spatial_configuration_document_item

TYPE plant_spatial_configuration_organization_item = SELECT
  (change_action,
   design_project,
   document,
   plant,
   product,
   product_definition_relationship,
   site);
END_TYPE; -- plant_spatial_configuration_organization_item

TYPE plant_spatial_configuration_person_and_organization_item = SELECT
  (change_item,
   plant,
   site);
END_TYPE; -- plant_spatial_configuration_person_and_organization_item

TYPE plant_spatial_configuration_person_item = SELECT
  (document,
   plant,
   product_definition_relationship,
   site);
END_TYPE; -- plant_spatial_configuration_person_item

TYPE positive_length_measure = length_measure;
WHERE
  wr1: SELF > 0;
END_TYPE; -- positive_length_measure

TYPE positive_plane_angle_measure = plane_angle_measure;
WHERE
  wr1: SELF > 0;
END_TYPE; -- positive_plane_angle_measure

TYPE positive_ratio_measure = ratio_measure;
WHERE
  wr1: SELF > 0;
END_TYPE; -- positive_ratio_measure

TYPE preferred_surface_curve_representation = ENUMERATION OF
  (curve_3d,
   pcurve_s1,
   pcurve_s2);
END_TYPE; -- preferred_surface_curve_representation

TYPE purchase_item = SELECT
  (product);
END_TYPE; -- purchase_item

TYPE ratio_measure = REAL;
END_TYPE; -- ratio_measure

TYPE reversible_topology = SELECT
  (reversible_topology_item,
   list_of_reversible_topology_item,
   set_of_reversible_topology_item);
END_TYPE; -- reversible_topology
```

```
TYPE reversible_topology_item = SELECT
  (edge,
   path,
   face,
   face_bound,
   closed_shell,
   open_shell);
END_TYPE; -- reversible_topology_item

TYPE second_in_minute = REAL;
WHERE
  wr1: (0 <= SELF) AND (SELF < 60);
END_TYPE; -- second_in_minute

TYPE set_of_reversible_topology_item = SET [0:?] OF
            reversible_topology_item;
END_TYPE; -- set_of_reversible_topology_item

TYPE shape_definition = SELECT
  (product_definition_shape,
   shape_aspect,
   shape_aspect_relationship);
END_TYPE; -- shape_definition

TYPE shell = SELECT
  (open_shell,
   closed_shell);
END_TYPE; -- shell

TYPE si_prefix = ENUMERATION OF
  (exa,
   peta,
   tera,
   giga,
   mega,
   kilo,
   hecto,
   deca,
   deci,
   centi,
   milli,
   micro,
   nano,
   pico,
   femto,
   atto);
END_TYPE; -- si_prefix

TYPE si_unit_name = ENUMERATION OF
  (metre,
   gram,
   second,
   ampere,
   kelvin,
   mole,
   candela,
   radian,
   steradian,
   hertz,
```

```
    newton,
    pascal,
    joule,
    watt,
    coulomb,
    volt,
    farad,
    ohm,
    siemens,
    weber,
    tesla,
    henry,
    degree_celsius,
    lumen,
    lux,
    becquerel,
    gray,
    sievert);
END_TYPE; -- si_unit_name

TYPE solid_angle_measure = REAL;
END_TYPE; -- solid_angle_measure

TYPE source = ENUMERATION OF
  (made,
   bought,
   not_known);
END_TYPE; -- source

TYPE source_item = SELECT
  (identifier);
END_TYPE; -- source_item

TYPE supported_item = SELECT
  (action_directive,
   action,
   action_method);
END_TYPE; -- supported_item

TYPE text = STRING;
END_TYPE; -- text

TYPE thermodynamic_temperature_measure = REAL;
END_TYPE; -- thermodynamic_temperature_measure

TYPE time_measure = REAL;
END_TYPE; -- time_measure

TYPE transformation = SELECT
  (functionally_defined_transformation);
END_TYPE; -- transformation

TYPE transition_code = ENUMERATION OF
  (discontinuous,
   continuous,
   cont_same_gradient,
   cont_same_gradient_same_curvature);
END_TYPE; -- transition_code
```

```
TYPE trimming_preference = ENUMERATION OF
  (cartesian,
   parameter,
   unspecified);
END_TYPE; -- trimming_preference

TYPE trimming_select = SELECT
  (cartesian_point,
   parameter_value);
END_TYPE; -- trimming_select

TYPE unit = SELECT
  (named_unit,
   derived_unit);
END_TYPE; -- unit

TYPE vector_or_direction = SELECT
  (vector,
   direction);
END_TYPE; -- vector_or_direction

TYPE volume_measure = REAL;
END_TYPE; -- volume_measure

TYPE year_number = INTEGER;
END_TYPE; -- year_number

ENTITY action;
    name          : label;
    description   : text;
    chosen_method : action_method;
END_ENTITY; -- action

ENTITY action_assignment
  ABSTRACT SUPERTYPE;
    assigned_action : action;
END_ENTITY; -- action_assignment

ENTITY action_directive;
    name        : label;
    description : text;
    analysis    : text;
    comment     : text;
    requests    : SET [1:?] OF versioned_action_request;
END_ENTITY; -- action_directive

ENTITY action_method;
    name        : label;
    description : text;
    consequence : text;
    purpose     : text;
END_ENTITY; -- action_method

ENTITY action_method_relationship;
    name            : label;
    description     : text;
    relating_method : action_method;
    related_method  : action_method;
END_ENTITY; -- action_method_relationship
```

```
ENTITY action_relationship;
    name            : label;
    description     : text;
    relating_action : action;
    related_action  : action;
END_ENTITY; -- action_relationship

ENTITY action_request_assignment
  ABSTRACT SUPERTYPE;
    assigned_action_request : versioned_action_request;
END_ENTITY; -- action_request_assignment

ENTITY action_request_solution;
    method  : action_method;
    request : versioned_action_request;
END_ENTITY; -- action_request_solution

ENTITY action_request_status;
    status           : label;
    assigned_request : versioned_action_request;
END_ENTITY; -- action_request_status

ENTITY action_status;
    status          : label;
    assigned_action : executed_action;
END_ENTITY; -- action_status

ENTITY advanced_csg_shape_representation
  SUBTYPE OF (shape_representation);
  WHERE
    wr1: rep_item_set_has_free_form(class_rep_item_in_set(SELF.items)) =
           FALSE;
    wr2: SIZEOF(QUERY ( i <* SELF\representation.items | (SIZEOF([
           'PLANT_SPATIAL_CONFIGURATION.BOOLEAN_RESULT',
           'PLANT_SPATIAL_CONFIGURATION.CSG_SOLID',
           'PLANT_SPATIAL_CONFIGURATION.TRUNCATED_PYRAMID',
           'PLANT_SPATIAL_CONFIGURATION.BLOCK',
           'PLANT_SPATIAL_CONFIGURATION.TORUS',
           'PLANT_SPATIAL_CONFIGURATION.RIGHT_CIRCULAR_CYLINDER',
           'PLANT_SPATIAL_CONFIGURATION.SPHERE',
           'PLANT_SPATIAL_CONFIGURATION.RIGHT_CIRCULAR_CONE',
           'PLANT_SPATIAL_CONFIGURATION.EXTRUDED_AREA_SOLID',
           'PLANT_SPATIAL_CONFIGURATION.REVOLVED_AREA_SOLID',
           'PLANT_SPATIAL_CONFIGURATION.AXIS2_PLACEMENT_3D',
           'PLANT_SPATIAL_CONFIGURATION.MEASURE_REPRESENTATION_ITEM',
           'PLANT_SPATIAL_CONFIGURATION.MAPPED_ITEM'] * TYPEOF(i)) = 0) ))
           = 0;
    wr3: SIZEOF(QUERY ( mi <* QUERY ( item <* SELF\representation.items
           | ('PLANT_SPATIAL_CONFIGURATION.MAPPED_ITEM' IN TYPEOF(item)))
           | (NOT (('PLANT_SPATIAL_CONFIGURATION.' +
         'ADVANCED_CSG_SHAPE_REPRESENTATION') IN TYPEOF(mi\
        mapped_item.mapping_source.mapped_representation))) )) = 0;
END_ENTITY; -- advanced_csg_shape_representation

ENTITY amount_of_substance_measure_with_unit
  SUBTYPE OF (measure_with_unit);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.AMOUNT_OF_SUBSTANCE_UNIT' IN
```

```
            TYPEOF(SELF\measure_with_unit.unit_component);
END_ENTITY; -- amount_of_substance_measure_with_unit

ENTITY amount_of_substance_unit
  SUBTYPE OF (named_unit);
  WHERE
    wr1: (SELF\named_unit.dimensions.length_exponent = 0) AND (SELF\
            named_unit.dimensions.mass_exponent = 0) AND (SELF\
            named_unit.dimensions.time_exponent = 0) AND (SELF\
            named_unit.dimensions.electric_current_exponent = 0) AND (
            SELF\named_unit.dimensions.
            thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
            .dimensions.amount_of_substance_exponent = 1) AND (SELF\
            named_unit.dimensions.luminous_intensity_exponent = 0);
END_ENTITY; -- amount_of_substance_unit

ENTITY angular_location
  SUBTYPE OF (dimensional_location);
    angle_selection : angle_relator;
END_ENTITY; -- angular_location

ENTITY application_context;
    application : text;
  INVERSE
    context_elements : SET [1:?] OF application_context_element FOR
                           frame_of_reference;
END_ENTITY; -- application_context

ENTITY application_context_element
  SUPERTYPE OF (ONEOF (product_context,product_definition_context));
    name                : label;
    frame_of_reference : application_context;
END_ENTITY; -- application_context_element

ENTITY application_protocol_definition;
    status                                    : label;
    application_interpreted_model_schema_name : label;
    application_protocol_year                 : year_number;
    application                               : application_context;
END_ENTITY; -- application_protocol_definition

ENTITY approval;
    status : approval_status;
    level  : label;
END_ENTITY; -- approval

ENTITY approval_assignment
  ABSTRACT SUPERTYPE;
    assigned_approval : approval;
END_ENTITY; -- approval_assignment

ENTITY approval_date_time;
    date_time      : date_time_select;
    dated_approval : approval;
END_ENTITY; -- approval_date_time

ENTITY approval_person_organization;
    person_organization : person_organization_select;
    authorized_approval : approval;
```

```
    role                : approval_role;
END_ENTITY; -- approval_person_organization

ENTITY approval_role;
    role : label;
END_ENTITY; -- approval_role

ENTITY approval_status;
    name : label;
END_ENTITY; -- approval_status

ENTITY area_measure_with_unit
  SUBTYPE OF (measure_with_unit);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.AREA_UNIT' IN TYPEOF(SELF\
              measure_with_unit.unit_component);
END_ENTITY; -- area_measure_with_unit

ENTITY area_unit
  SUBTYPE OF (named_unit);
  WHERE
    wr1: (SELF\named_unit.dimensions.length_exponent = 2) AND (SELF\
              named_unit.dimensions.mass_exponent = 0) AND (SELF\
              named_unit.dimensions.time_exponent = 0) AND (SELF\
              named_unit.dimensions.electric_current_exponent = 0) AND (
              SELF\named_unit.dimensions.
              thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
              .dimensions.amount_of_substance_exponent = 0) AND (SELF\
              named_unit.dimensions.luminous_intensity_exponent = 0);
END_ENTITY; -- area_unit

ENTITY assembly_component_usage
  SUBTYPE OF (product_definition_usage);
    reference_designator : OPTIONAL identifier;
END_ENTITY; -- assembly_component_usage

ENTITY axis1_placement
  SUBTYPE OF (placement);
    axis : OPTIONAL direction;
  DERIVE
    z : direction := NVL(normalise(axis),direction([0,0,1]));
  WHERE
    wr1: SELF\geometric_representation_item.dim = 3;
END_ENTITY; -- axis1_placement

ENTITY axis2_placement_2d
  SUBTYPE OF (placement);
    ref_direction : OPTIONAL direction;
  DERIVE
    p : LIST [2:2] OF direction := build_2axes(ref_direction);
  WHERE
    wr1: SELF\geometric_representation_item.dim = 2;
END_ENTITY; -- axis2_placement_2d

ENTITY axis2_placement_3d
  SUBTYPE OF (placement);
    axis          : OPTIONAL direction;
    ref_direction : OPTIONAL direction;
  DERIVE
```

```
    p : LIST [3:3] OF direction := build_axes(axis,ref_direction);
  WHERE
    wr1: SELF\placement.location.dim = 3;
    wr2: (NOT EXISTS(axis)) OR (axis.dim = 3);
    wr3: (NOT EXISTS(ref_direction)) OR (ref_direction.dim = 3);
    wr4: (NOT EXISTS(axis)) OR (NOT EXISTS(ref_direction)) OR (
            cross_product(axis,ref_direction).magnitude > 0);
END_ENTITY; -- axis2_placement_3d

ENTITY b_spline_curve
  SUPERTYPE OF (ONEOF (uniform_curve,b_spline_curve_with_knots,
     quasi_uniform_curve,bezier_curve) ANDOR rational_b_spline_curve)
  SUBTYPE OF (bounded_curve);
    degree              : INTEGER;
    control_points_list : LIST [2:?] OF cartesian_point;
    curve_form          : b_spline_curve_form;
    closed_curve        : LOGICAL;
    self_intersect      : LOGICAL;
  DERIVE
    upper_index_on_control_points : INTEGER := SIZEOF(
                                       control_points_list) - 1;
    control_points                : ARRAY [0:
                                       upper_index_on_control_points] OF
                                        cartesian_point := list_to_array(
                                       control_points_list,0,
                                       upper_index_on_control_points);
  WHERE
    wr1: ('PLANT_SPATIAL_CONFIGURATION.UNIFORM_CURVE' IN TYPEOF(SELF))
            OR ('PLANT_SPATIAL_CONFIGURATION.QUASI_UNIFORM_CURVE' IN
            TYPEOF(SELF)) OR ('PLANT_SPATIAL_CONFIGURATION.BEZIER_CURVE'
            IN TYPEOF(SELF)) OR (
            'PLANT_SPATIAL_CONFIGURATION.B_SPLINE_CURVE_WITH_KNOTS' IN
            TYPEOF(SELF));
END_ENTITY; -- b_spline_curve

ENTITY b_spline_curve_with_knots
  SUBTYPE OF (b_spline_curve);
    knot_multiplicities : LIST [2:?] OF INTEGER;
    knots               : LIST [2:?] OF parameter_value;
    knot_spec           : knot_type;
  DERIVE
    upper_index_on_knots : INTEGER := SIZEOF(knots);
  WHERE
    wr1: constraints_param_b_spline(degree,upper_index_on_knots,
            upper_index_on_control_points,knot_multiplicities,knots);
    wr2: SIZEOF(knot_multiplicities) = upper_index_on_knots;
END_ENTITY; -- b_spline_curve_with_knots

ENTITY b_spline_surface
  SUPERTYPE OF (ONEOF (b_spline_surface_with_knots,uniform_surface,
     quasi_uniform_surface,bezier_surface) ANDOR
     rational_b_spline_surface)
  SUBTYPE OF (bounded_surface);
    u_degree            : INTEGER;
    v_degree            : INTEGER;
    control_points_list : LIST [2:?] OF LIST [2:?] OF cartesian_point;
    surface_form        : b_spline_surface_form;
    u_closed            : LOGICAL;
    v_closed            : LOGICAL;
```

```
    self_intersect      : LOGICAL;
  DERIVE
    u_upper        : INTEGER := SIZEOF(control_points_list) - 1;
    v_upper        : INTEGER := SIZEOF(control_points_list[1]) - 1;
    control_points : ARRAY [0:u_upper] OF ARRAY [0:v_upper] OF
                         cartesian_point := make_array_of_array(
                        control_points_list,0,u_upper,0,v_upper);
  WHERE
    wr1: ('PLANT_SPATIAL_CONFIGURATION.UNIFORM_SURFACE' IN TYPEOF(SELF))
              OR ('PLANT_SPATIAL_CONFIGURATION.QUASI_UNIFORM_SURFACE' IN
              TYPEOF(SELF)) OR (
              'PLANT_SPATIAL_CONFIGURATION.BEZIER_SURFACE' IN
              TYPEOF(SELF)) OR (
              'PLANT_SPATIAL_CONFIGURATION.B_SPLINE_SURFACE_WITH_KNOTS' IN
               TYPEOF(SELF));
END_ENTITY; -- b_spline_surface

ENTITY b_spline_surface_with_knots
  SUBTYPE OF (b_spline_surface);
    u_multiplicities : LIST [2:?] OF INTEGER;
    v_multiplicities : LIST [2:?] OF INTEGER;
    u_knots          : LIST [2:?] OF parameter_value;
    v_knots          : LIST [2:?] OF parameter_value;
    knot_spec        : knot_type;
  DERIVE
    knot_u_upper : INTEGER := SIZEOF(u_knots);
    knot_v_upper : INTEGER := SIZEOF(v_knots);
  WHERE
    wr1: constraints_param_b_spline(SELF\b_spline_surface.u_degree,
              knot_u_upper,SELF\b_spline_surface.u_upper,u_multiplicities,
              u_knots);
    wr2: constraints_param_b_spline(SELF\b_spline_surface.v_degree,
              knot_v_upper,SELF\b_spline_surface.v_upper,v_multiplicities,
              v_knots);
    wr3: SIZEOF(u_multiplicities) = knot_u_upper;
    wr4: SIZEOF(v_multiplicities) = knot_v_upper;
END_ENTITY; -- b_spline_surface_with_knots

ENTITY bezier_curve
  SUBTYPE OF (b_spline_curve);
END_ENTITY; -- bezier_curve

ENTITY bezier_surface
  SUBTYPE OF (b_spline_surface);
END_ENTITY; -- bezier_surface

ENTITY blank_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- blank_fitting_classification

ENTITY block
  SUBTYPE OF (geometric_representation_item);
    position : axis2_placement_3d;
    x        : positive_length_measure;
    y        : positive_length_measure;
    z        : positive_length_measure;
END_ENTITY; -- block

ENTITY boolean_result
```

```
  SUBTYPE OF (geometric_representation_item);
    operator       : boolean_operator;
    first_operand  : boolean_operand;
    second_operand : boolean_operand;
END_ENTITY; -- boolean_result

ENTITY boundary_curve
  SUBTYPE OF (composite_curve_on_surface);
  WHERE
    wr1: SELF\composite_curve.closed_curve;
END_ENTITY; -- boundary_curve

ENTITY bounded_curve
  SUPERTYPE OF (ONEOF (polyline,b_spline_curve,trimmed_curve,
      bounded_pcurve,bounded_surface_curve,composite_curve))
  SUBTYPE OF (curve);
END_ENTITY; -- bounded_curve

ENTITY bounded_pcurve
  SUBTYPE OF (pcurve, bounded_curve);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.BOUNDED_CURVE' IN TYPEOF(SELF\
           pcurve.reference_to_curve.items[1]);
END_ENTITY; -- bounded_pcurve

ENTITY bounded_surface
  SUPERTYPE OF (ONEOF (b_spline_surface,rectangular_trimmed_surface,
      curve_bounded_surface,rectangular_composite_surface))
  SUBTYPE OF (surface);
END_ENTITY; -- bounded_surface

ENTITY bounded_surface_curve
  SUBTYPE OF (surface_curve, bounded_curve);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.BOUNDED_CURVE' IN TYPEOF(SELF\
           surface_curve.curve_3d);
END_ENTITY; -- bounded_surface_curve

ENTITY brep_with_voids
  SUBTYPE OF (manifold_solid_brep);
    voids : SET [1:?] OF oriented_closed_shell;
END_ENTITY; -- brep_with_voids

ENTITY bushing_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- bushing_fitting_classification

ENTITY calendar_date
  SUBTYPE OF (date);
    day_component   : day_in_month_number;
    month_component : month_in_year_number;
  WHERE
    wr1: valid_calendar_date(SELF);
END_ENTITY; -- calendar_date

ENTITY cartesian_point
  SUBTYPE OF (point);
```

```
    coordinates : LIST [1:3] OF length_measure;
END_ENTITY; -- cartesian_point

ENTITY cartesian_transformation_operator
  SUPERTYPE OF (cartesian_transformation_operator_3d)
  SUBTYPE OF (geometric_representation_item,
      functionally_defined_transformation);
    axis1        : OPTIONAL direction;
    axis2        : OPTIONAL direction;
    local_origin : cartesian_point;
    scale        : OPTIONAL REAL;
  DERIVE
    scl : REAL := NVL(scale,1);
  WHERE
    wr1: scl > 0;
END_ENTITY; -- cartesian_transformation_operator

ENTITY cartesian_transformation_operator_3d
  SUBTYPE OF (cartesian_transformation_operator);
    axis3 : OPTIONAL direction;
  DERIVE
    u : LIST [3:3] OF direction := base_axis(3,SELF\
          cartesian_transformation_operator.axis1,SELF\
          cartesian_transformation_operator.axis2,axis3);
  WHERE
    wr1: SELF\cartesian_transformation_operator.dim = 3;
END_ENTITY; -- cartesian_transformation_operator_3d

ENTITY catalogue_item
  SUBTYPE OF (externally_defined_plant_item_definition);
  WHERE
    wr1: SIZEOF(QUERY ( pscdr <* USEDIN(SELF,
              'PLANT_SPATIAL_CONFIGURATION.' +
              'PLANT_SPATIAL_CONFIGURATION_DOCUMENT_REFERENCE.ITEMS') | (
              SIZEOF(QUERY ( duc <* USEDIN(pscdr.assigned_document,
              'PLANT_SPATIAL_CONFIGURATION.DOCUMENT_USAGE_CONSTRAINT.SOURCE')
               | (duc.subject_element = 'item_definition') )) = 1) )) = 1;
    wr2: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
              'PLANT_SPATIAL_CONFIGURATION.' +
              'PRODUCT_DEFINITION_RELATIONSHIP.RELATING_PRODUCT_DEFINITION')
               | ('PLANT_SPATIAL_CONFIGURATION.' +
              'EXTERNALLY_DEFINED_PLANT_ITEM_DEFINITION'
              IN TYPEOF(pdr.related_product_definition)) )) = 1;
END_ENTITY; -- catalogue_item

ENTITY centre_of_symmetry
  SUBTYPE OF (derived_shape_aspect);
END_ENTITY; -- centre_of_symmetry

ENTITY change_action
  SUBTYPE OF (directed_action);
  WHERE
    wr1: SIZEOF(QUERY ( ca <* USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.'
              + 'ACTION_ASSIGNMENT.ASSIGNED_ACTION') | ((
              'PLANT_SPATIAL_CONFIGURATION.' +
              'PLANT_SPATIAL_CONFIGURATION_CHANGE_ASSIGNMENT') IN TYPEOF(
              ca)) )) >= 1;
    wr2: SIZEOF(QUERY ( ar <* SELF\directed_action.directive.requests |
              (NOT (SIZEOF(USEDIN(ar,'PLANT_SPATIAL_CONFIGURATION.' +
```

```
            'ACTION_REQUEST_SOLUTION.SOLUTION')) = 1)) )) = 0;
    wr3: SIZEOF(USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.ACTION_STATUS.'
           + 'ASSIGNED_ACTION')) = 1;
END_ENTITY; -- change_action

ENTITY change_from_assignment
  SUBTYPE OF (action_assignment);
    items : SET [1:?] OF change_delta_item;
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.CHANGE_ACTION' IN TYPEOF(SELF.
           assigned_action);
END_ENTITY; -- change_from_assignment

ENTITY change_item_id_assignment
  SUBTYPE OF (name_assignment);
    items : SET [1:?] OF change_item;
END_ENTITY; -- change_item_id_assignment

ENTITY change_to_assignment
  SUBTYPE OF (action_assignment);
    items : SET [1:?] OF change_delta_item;
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.CHANGE_ACTION' IN TYPEOF(SELF.
           assigned_action);
END_ENTITY; -- change_to_assignment

ENTITY characterized_object;
    name        : label;
    description : text;
END_ENTITY; -- characterized_object

ENTITY circle
  SUBTYPE OF (conic);
    radius : positive_length_measure;
END_ENTITY; -- circle

ENTITY classification_assignment
  SUBTYPE OF (group_assignment);
    items : SET [1:?] OF classification_item;
  WHERE
    wr1: (NOT (SIZEOF(QUERY ( item <* SELF.items | (NOT (
           'PLANT_SPATIAL_CONFIGURATION.PLANT_ITEM_CONNECTION' IN
           TYPEOF(item))) )) = 0)) XOR (SIZEOF(TYPEOF(SELF.
           assigned_group) * ['PLANT_SPATIAL_CONFIGURATION.' +
           'CONNECTION_FUNCTIONAL_CLASSIFICATION', 'PLANT_SPATIAL_' +
           'CONFIGURATION.CONNECTION_MOTION_CLASSIFICATION'])
           >= 1);
    wr2: (NOT (SIZEOF(QUERY ( item <* SELF.items | (NOT (
           'PLANT_SPATIAL_CONFIGURATION.PLANT_ITEM_CONNECTOR' IN
           TYPEOF(item))) )) = 0)) XOR (SIZEOF(TYPEOF(SELF.
           assigned_group) * ['PLANT_SPATIAL_CONFIGURATION.' +
           'CONNECTOR_END_TYPE_CLASSIFICATION',
           'PLANT_SPATIAL_CONFIGURATION.ELECTRICAL_CONNECTOR_' +
           'CLASSIFICATION','PLANT_SPATIAL_CONFIGURATION.' +
           'PIPING_CONNECTOR_CLASSIFICATION',
           'PLANT_SPATIAL_CONFIGURATION.' +
           'STRUCTURAL_LOAD_CONNECTOR_CLASSIFICATION']) >= 1);
    wr3: (NOT (SIZEOF(QUERY ( item <* SELF.items | (NOT (
           'PLANT_SPATIAL_CONFIGURATION.PIPING_COMPONENT_DEFINITION' IN
```

```
                TYPEOF(item))) )) = 0)) XOR (SIZEOF(TYPEOF(SELF.
                assigned_group) * ['PLANT_SPATIAL_CONFIGURATION.' +
               'BLANK_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.BUSHING_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.COUPLING_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.CROSS_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.ELBOW_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.FLANGE_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.' +
               'FLANGE_FITTING_NECK_TYPE_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.INSERT_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.' +
               'LAP_JOINT_STUB_END_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.LATERAL_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.OLET_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.ORIFICE_PLATE_FITTING_' +
               'CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.PIPE_CAP_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.PIPE_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.REDUCER_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.SPACER_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.SPECIALTY_ITEM_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.SWAGE_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.TEE_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.UNION_FITTING_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.VALVE_CLASSIFICATION',
               'PLANT_SPATIAL_CONFIGURATION.WYE_FITTING_CLASSIFICATION'])
              >= 1);
END_ENTITY; -- classification_assignment

ENTITY closed_shell
  SUBTYPE OF (connected_face_set);
END_ENTITY; -- closed_shell

ENTITY colour;
END_ENTITY; -- colour

ENTITY colour_rgb
  SUBTYPE OF (colour_specification);
    red   : REAL;
    green : REAL;
    blue  : REAL;
  WHERE
    wr1: (0 <= red) AND (red <= 1);
    wr2: (0 <= green) AND (green <= 1);
    wr3: (0 <= blue) AND (blue <= 1);
END_ENTITY; -- colour_rgb

ENTITY colour_specification
  SUBTYPE OF (colour);
    name : label;
END_ENTITY; -- colour_specification

ENTITY composite_curve
  SUBTYPE OF (bounded_curve);
    segments       : LIST [1:?] OF composite_curve_segment;
    self_intersect : LOGICAL;
  DERIVE
    n_segments   : INTEGER := SIZEOF(segments);
```

```
    closed_curve : LOGICAL := segments[n_segments].transition <>
                        discontinuous;
  WHERE
    wr1: ((NOT closed_curve) AND (SIZEOF(QUERY ( temp <* segments | (
              temp.transition = discontinuous) )) = 1)) OR (closed_curve
              AND (SIZEOF(QUERY ( temp <* segments | (temp.transition =
              discontinuous) )) = 0));
END_ENTITY; -- composite_curve

ENTITY composite_curve_on_surface
  SUPERTYPE OF (boundary_curve)
  SUBTYPE OF (composite_curve);
  DERIVE
    basis_surface : SET [0:2] OF surface := get_basis_surface(SELF);
  WHERE
    wr1: SIZEOF(basis_surface) > 0;
    wr2: constraints_composite_curve_on_surface(SELF);
END_ENTITY; -- composite_curve_on_surface

ENTITY composite_curve_segment;
    transition   : transition_code;
    same_sense   : BOOLEAN;
    parent_curve : curve;
  INVERSE
    using_curves : BAG [1:?] OF composite_curve FOR segments;
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.BOUNDED_CURVE' IN TYPEOF(
              parent_curve);
END_ENTITY; -- composite_curve_segment

ENTITY conic
  SUPERTYPE OF (ONEOF (circle,ellipse,hyperbola,parabola))
  SUBTYPE OF (curve);
    position : axis2_placement;
END_ENTITY; -- conic

ENTITY conical_surface
  SUBTYPE OF (elementary_surface);
    radius     : length_measure;
    semi_angle : plane_angle_measure;
  WHERE
    wr1: radius >= 0;
END_ENTITY; -- conical_surface

ENTITY connected_face_set
  SUPERTYPE OF (ONEOF (closed_shell,open_shell))
  SUBTYPE OF (topological_representation_item);
    cfs_faces : SET [1:?] OF face;
END_ENTITY; -- connected_face_set

ENTITY connection_functional_classification
  SUBTYPE OF (group);
END_ENTITY; -- connection_functional_classification

ENTITY connection_motion_classification
  SUBTYPE OF (group);
  WHERE
    wr1: SELF.name IN ['flexible','locked_orientation'];
END_ENTITY; -- connection_motion_classification
```

```
ENTITY connection_node
  SUBTYPE OF (shape_aspect);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.PIPING_SYSTEM' IN TYPEOF(SELF.
            of_shape.definition);
    wr2: SIZEOF(QUERY ( sar <* USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.SHAPE_ASPECT_RELATIONSHIP.' +
            'RELATING_SHAPE_ASPECT') | (
            'PLANT_SPATIAL_CONFIGURATION.LINE_TERMINATION_CONNECTION' IN
             TYPEOF(sar)) )) >= 3;
END_ENTITY; -- connection_node

ENTITY connector_end_type_classification
  SUBTYPE OF (group);
END_ENTITY; -- connector_end_type_classification

ENTITY context_dependent_unit
  SUBTYPE OF (named_unit);
    name : label;
END_ENTITY; -- context_dependent_unit

ENTITY conversion_based_unit
  SUBTYPE OF (named_unit);
    name              : label;
    conversion_factor : measure_with_unit;
END_ENTITY; -- conversion_based_unit

ENTITY coordinated_universal_time_offset;
    hour_offset   : hour_in_day;
    minute_offset : OPTIONAL minute_in_hour;
    sense         : ahead_or_behind;
END_ENTITY; -- coordinated_universal_time_offset

ENTITY coupling_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- coupling_fitting_classification

ENTITY cross_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- cross_fitting_classification

ENTITY csg_solid
  SUBTYPE OF (solid_model);
    tree_root_expression : csg_select;
END_ENTITY; -- csg_solid

ENTITY curve
  SUPERTYPE OF (ONEOF (line,conic,pcurve,surface_curve,offset_curve_2d,
      offset_curve_3d,curve_replica))
  SUBTYPE OF (geometric_representation_item);
END_ENTITY; -- curve

ENTITY curve_bounded_surface
  SUBTYPE OF (bounded_surface);
    basis_surface  : surface;
    boundaries     : SET [1:?] OF boundary_curve;
    implicit_outer : BOOLEAN;
  WHERE
    wr1: NOT (implicit_outer AND (
```

```
            'PLANT_SPATIAL_CONFIGURATION.OUTER_BOUNDARY_CURVE' IN
            TYPEOF(boundaries)));
    wr2: (NOT implicit_outer) OR (
            'PLANT_SPATIAL_CONFIGURATION.BOUNDED_SURFACE' IN TYPEOF(
            basis_surface));
    wr3: SIZEOF(QUERY ( temp <* boundaries | (
            'PLANT_SPATIAL_CONFIGURATION.OUTER_BOUNDARY_CURVE' IN
            TYPEOF(temp)) )) <= 1;
    wr4: SIZEOF(QUERY ( temp <* boundaries | (temp\
            composite_curve_on_surface.basis_surface[1] :<>: SELF.
            basis_surface) )) = 0;
END_ENTITY; -- curve_bounded_surface

ENTITY curve_replica
  SUBTYPE OF (curve);
    parent_curve   : curve;
    transformation : cartesian_transformation_operator;
  WHERE
    wr1: transformation.dim = parent_curve.dim;
    wr2: acyclic_curve_replica(SELF,parent_curve);
END_ENTITY; -- curve_replica

ENTITY cylindrical_surface
  SUBTYPE OF (elementary_surface);
    radius : positive_length_measure;
END_ENTITY; -- cylindrical_surface

ENTITY date
  SUPERTYPE OF (calendar_date);
    year_component : year_number;
END_ENTITY; -- date

ENTITY date_and_time;
    date_component : date;
    time_component : local_time;
END_ENTITY; -- date_and_time

ENTITY date_and_time_assignment
  ABSTRACT SUPERTYPE;
    assigned_date_and_time : date_and_time;
    role                   : date_time_role;
END_ENTITY; -- date_and_time_assignment

ENTITY date_assignment
  ABSTRACT SUPERTYPE;
    assigned_date : date;
    role          : date_role;
END_ENTITY; -- date_assignment

ENTITY date_role;
    name : label;
END_ENTITY; -- date_role

ENTITY date_time_role;
    name : label;
END_ENTITY; -- date_time_role

ENTITY definitional_representation
  SUBTYPE OF (representation);
```

```
    WHERE
      wr1: 'PLANT_SPATIAL_CONFIGURATION.PARAMETRIC_REPRESENTATION_CONTEXT'
              IN TYPEOF(SELF\representation.context_of_items);
  END_ENTITY; -- definitional_representation

  ENTITY degenerate_pcurve
    SUBTYPE OF (point);
      basis_surface      : surface;
      reference_to_curve : definitional_representation;
    WHERE
      wr1: SIZEOF(reference_to_curve\representation.items) = 1;
      wr2: 'PLANT_SPATIAL_CONFIGURATION.CURVE' IN TYPEOF(
              reference_to_curve\representation.items[1]);
      wr3: reference_to_curve\representation.items[1]\
              geometric_representation_item.dim = 2;
  END_ENTITY; -- degenerate_pcurve

  ENTITY degenerate_toroidal_surface
    SUBTYPE OF (toroidal_surface);
      select_outer : BOOLEAN;
    WHERE
      wr1: major_radius < minor_radius;
  END_ENTITY; -- degenerate_toroidal_surface

  ENTITY derived_shape_aspect
    SUPERTYPE OF (centre_of_symmetry)
    SUBTYPE OF (shape_aspect);
    INVERSE
      deriving_relationships : SET [1:?] OF
                                  shape_aspect_deriving_relationship FOR
                                relating_shape_aspect;
  END_ENTITY; -- derived_shape_aspect

  ENTITY derived_unit;
      elements : SET [1:?] OF derived_unit_element;
    WHERE
      wr1: (SIZEOF(elements) > 1) OR ((SIZEOF(elements) = 1) AND (elements
              [1].exponent <> 1));
  END_ENTITY; -- derived_unit

  ENTITY derived_unit_element;
      unit     : named_unit;
      exponent : REAL;
  END_ENTITY; -- derived_unit_element

  ENTITY descriptive_colour
    SUBTYPE OF (colour, descriptive_representation_item);
  END_ENTITY; -- descriptive_colour

  ENTITY descriptive_representation_item
    SUBTYPE OF (representation_item);
      description : text;
  END_ENTITY; -- descriptive_representation_item

  ENTITY design_project
    SUBTYPE OF (organization);
    WHERE
      wr1: SIZEOF(USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
```

```
            'ORGANIZATION_ASSIGNMENT.ASSIGNED_ORGANIZATION')) >= 1;
END_ENTITY; -- design_project

ENTITY design_project_assignment
  SUBTYPE OF (organization_assignment);
    items : SET [1:?] OF design_project_item;
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.DESIGN_PROJECT' IN TYPEOF(SELF.
           assigned_organization);
END_ENTITY; -- design_project_assignment

ENTITY dimensional_characteristic_representation;
    dimension      : dimensional_characteristic;
    representation : shape_dimension_representation;
END_ENTITY; -- dimensional_characteristic_representation

ENTITY dimensional_exponents;
    length_exponent                     : REAL;
    mass_exponent                       : REAL;
    time_exponent                       : REAL;
    electric_current_exponent           : REAL;
    thermodynamic_temperature_exponent : REAL;
    amount_of_substance_exponent        : REAL;
    luminous_intensity_exponent         : REAL;
END_ENTITY; -- dimensional_exponents

ENTITY dimensional_location
  SUPERTYPE OF (angular_location)
  SUBTYPE OF (shape_aspect_relationship);
END_ENTITY; -- dimensional_location

ENTITY dimensional_size;
    applies_to : shape_aspect;
    name       : label;
  WHERE
    wr1: applies_to.product_definitional = TRUE;
END_ENTITY; -- dimensional_size

ENTITY directed_action
  SUBTYPE OF (executed_action);
    directive : action_directive;
END_ENTITY; -- directed_action

ENTITY direction
  SUBTYPE OF (geometric_representation_item);
    direction_ratios : LIST [2:3] OF REAL;
  WHERE
    wr1: SIZEOF(QUERY ( tmp <* direction_ratios | (tmp <> 0) )) > 0;
END_ENTITY; -- direction

ENTITY document;
    id          : identifier;
    name        : label;
    description : text;
    kind        : document_type;
  UNIQUE
    ur1 : id;
END_ENTITY; -- document
```

```
ENTITY document_reference
  ABSTRACT SUPERTYPE;
    assigned_document : document;
    source            : label;
END_ENTITY; -- document_reference

ENTITY document_relationship;
    name              : label;
    description       : text;
    relating_document : document;
    related_document  : document;
END_ENTITY; -- document_relationship

ENTITY document_type;
    product_data_type : label;
END_ENTITY; -- document_type

ENTITY document_usage_constraint;
    source                : document;
    subject_element       : label;
    subject_element_value : text;
END_ENTITY; -- document_usage_constraint

ENTITY edge
  SUPERTYPE OF (ONEOF (edge_curve,oriented_edge))
  SUBTYPE OF (topological_representation_item);
    edge_start : vertex;
    edge_end   : vertex;
END_ENTITY; -- edge

ENTITY edge_curve
  SUBTYPE OF (edge, geometric_representation_item);
    edge_geometry : curve;
    same_sense    : BOOLEAN;
END_ENTITY; -- edge_curve

ENTITY edge_loop
  SUBTYPE OF (loop, path);
  DERIVE
    ne : INTEGER := SIZEOF(SELF\path.edge_list);
  WHERE
    wr1: SELF\path.edge_list[1].edge_start :=: SELF\path.edge_list[ne].
            edge_end;
END_ENTITY; -- edge_loop

ENTITY elbow_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- elbow_fitting_classification

ENTITY electrical_connector_classification
  SUBTYPE OF (group);
END_ENTITY; -- electrical_connector_classification

ENTITY electrical_system
  SUBTYPE OF (product_definition);
  WHERE
    wr1: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
              'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_' +
              'RELATIONSHIP.RELATED_PRODUCT_DEFINITION')
```

```
              | (('PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(pdr.
            relating_product_definition.formation.of_product)) AND (pdr.
            relating_product_definition.frame_of_reference.name =
            'functional_occurrence')) )) = 1;
END_ENTITY; -- electrical_system

ENTITY elementary_surface
  SUPERTYPE OF (ONEOF (plane,cylindrical_surface,conical_surface,
      spherical_surface,toroidal_surface))
  SUBTYPE OF (surface);
    position : axis2_placement_3d;
END_ENTITY; -- elementary_surface

ENTITY ellipse
  SUBTYPE OF (conic);
    semi_axis_1 : positive_length_measure;
    semi_axis_2 : positive_length_measure;
END_ENTITY; -- ellipse

ENTITY evaluated_degenerate_pcurve
  SUBTYPE OF (degenerate_pcurve);
    equivalent_point : cartesian_point;
END_ENTITY; -- evaluated_degenerate_pcurve

ENTITY executed_action
  SUBTYPE OF (action);
END_ENTITY; -- executed_action

ENTITY external_source;
    source_id : source_item;
END_ENTITY; -- external_source

ENTITY externally_defined_item;
    item_id : source_item;
    source  : external_source;
END_ENTITY; -- externally_defined_item

ENTITY externally_defined_plant_item_definition
  SUBTYPE OF (plant_item_definition, externally_defined_item);
END_ENTITY; -- externally_defined_plant_item_definition

ENTITY extruded_area_solid
  SUBTYPE OF (swept_area_solid);
    extruded_direction : direction;
    depth              : positive_length_measure;
  WHERE
    wr1: dot_product(SELF\swept_area_solid.swept_area.basis_surface\
              elementary_surface.position.p[3],extruded_direction) <> 0;
END_ENTITY; -- extruded_area_solid

ENTITY extruded_face_solid
  SUBTYPE OF (swept_face_solid);
    extruded_direction : direction;
    depth              : positive_length_measure;
  WHERE
    wr1: dot_product(SELF\swept_face_solid.swept_face.face_geometry\
              elementary_surface.position.p[3],extruded_direction) <> 0;
END_ENTITY; -- extruded_face_solid
```

```
ENTITY face
  SUPERTYPE OF (ONEOF (face_surface,oriented_face))
  SUBTYPE OF (topological_representation_item);
    bounds : SET [1:?] OF face_bound;
  WHERE
    wr1: NOT mixed_loop_type_set(list_to_set(list_face_loops(SELF)));
    wr2: SIZEOF(QUERY ( temp <* bounds | (
              'PLANT_SPATIAL_CONFIGURATION.FACE_OUTER_BOUND' IN TYPEOF(
              temp)) )) <= 1;
END_ENTITY; -- face

ENTITY face_bound
  SUBTYPE OF (topological_representation_item);
    bound       : loop;
    orientation : BOOLEAN;
END_ENTITY; -- face_bound

ENTITY face_surface
  SUBTYPE OF (face, geometric_representation_item);
    face_geometry : surface;
    same_sense    : BOOLEAN;
END_ENTITY; -- face_surface

ENTITY faceted_brep
  SUBTYPE OF (manifold_solid_brep);
END_ENTITY; -- faceted_brep

ENTITY flange_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- flange_fitting_classification

ENTITY flange_fitting_neck_type_classification
  SUBTYPE OF (group);
END_ENTITY; -- flange_fitting_neck_type_classification

ENTITY functionally_defined_transformation;
    name        : label;
    description : text;
END_ENTITY; -- functionally_defined_transformation

ENTITY geometric_curve_set
  SUBTYPE OF (geometric_set);
  WHERE
    wr1: SIZEOF(QUERY ( temp <* SELF\geometric_set.elements | (
              'PLANT_SPATIAL_CONFIGURATION.SURFACE' IN TYPEOF(temp)) )) =
              0;
END_ENTITY; -- geometric_curve_set

ENTITY geometric_representation_context
  SUBTYPE OF (representation_context);
    coordinate_space_dimension : dimension_count;
END_ENTITY; -- geometric_representation_context

ENTITY geometric_representation_item
  SUPERTYPE OF (ONEOF (point,direction,vector,placement,
      cartesian_transformation_operator,curve,surface,edge_curve,
      face_surface,poly_loop,solid_model,boolean_result,sphere,
      right_circular_cone,right_circular_cylinder,torus,block,
      geometric_set))
```

```
  SUBTYPE OF (representation_item);
  DERIVE
    dim : dimension_count := dimension_of(SELF);
  WHERE
    wr1: SIZEOF(QUERY ( using_rep <* using_representations(SELF) | (NOT
              ('PLANT_SPATIAL_CONFIGURATION.GEOMETRIC_REPRESENTATION_CONTEXT'

              IN TYPEOF(using_rep.context_of_items))) )) = 0;
END_ENTITY; -- geometric_representation_item

ENTITY geometric_set
  SUPERTYPE OF (ONEOF (geometric_curve_set,geometric_set_replica))
  SUBTYPE OF (geometric_representation_item);
    elements : SET [1:?] OF geometric_set_select;
END_ENTITY; -- geometric_set

ENTITY geometric_set_replica
  SUBTYPE OF (geometric_set);
    parent_set     : geometric_set;
    transformation : cartesian_transformation_operator;
  DERIVE
    SELF\geometric_set.elements : SET [1:?] OF geometric_set_select :=
                        build_transformed_set(transformation,
                                              parent_set);
  WHERE
    wr1: acyclic_set_replica(SELF,parent_set);
END_ENTITY; -- geometric_set_replica

ENTITY group;
    name        : label;
    description : text;
END_ENTITY; -- group

ENTITY group_assignment
  ABSTRACT SUPERTYPE;
    assigned_group : group;
END_ENTITY; -- group_assignment

ENTITY heat_tracing_representation
  SUBTYPE OF (representation);
END_ENTITY; -- heat_tracing_representation

ENTITY hvac_system
  SUBTYPE OF (product_definition);
  WHERE
    wr1: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
              'PLANT_SPATIAL_CONFIGURATION.' +
              'PRODUCT_DEFINITION_RELATIONSHIP.RELATED_PRODUCT_DEFINITION')
               | (('PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(pdr.
              relating_product_definition.formation.of_product)) AND (pdr.
              relating_product_definition.frame_of_reference.name =
              'functional_occurrence')) )) = 1;
END_ENTITY; -- hvac_system

ENTITY hybrid_shape_representation
  SUBTYPE OF (shape_representation);
  WHERE
    wr1: SIZEOF(QUERY ( i <* SELF\representation.items | (SIZEOF([
              'PLANT_SPATIAL_CONFIGURATION.BOOLEAN_RESULT',
```

```
              'PLANT_SPATIAL_CONFIGURATION.CSG_SOLID',
              'PLANT_SPATIAL_CONFIGURATION.TRUNCATED_PYRAMID',
              'PLANT_SPATIAL_CONFIGURATION.BLOCK',
              'PLANT_SPATIAL_CONFIGURATION.TORUS',
              'PLANT_SPATIAL_CONFIGURATION.RIGHT_CIRCULAR_CYLINDER',
              'PLANT_SPATIAL_CONFIGURATION.SPHERE',
              'PLANT_SPATIAL_CONFIGURATION.RIGHT_CIRCULAR_CONE',
              'PLANT_SPATIAL_CONFIGURATION.EXTRUDED_AREA_SOLID',
              'PLANT_SPATIAL_CONFIGURATION.REVOLVED_AREA_SOLID',
              'PLANT_SPATIAL_CONFIGURATION.AXIS2_PLACEMENT_3D',
              'PLANT_SPATIAL_CONFIGURATION.MANIFOLD_SOLID_BREP',
              'PLANT_SPATIAL_CONFIGURATION.CONIC',
              'PLANT_SPATIAL_CONFIGURATION.CURVE',
              'PLANT_SPATIAL_CONFIGURATION.B_SPLINE_CURVE',
              'PLANT_SPATIAL_CONFIGURATION.LINE',
              'PLANT_SPATIAL_CONFIGURATION.POINT',
              'PLANT_SPATIAL_CONFIGURATION.POLYLINE',
              'PLANT_SPATIAL_CONFIGURATION.SURFACE',
              'PLANT_SPATIAL_CONFIGURATION.VECTOR',
              'PLANT_SPATIAL_CONFIGURATION.' +
              'GEOMETRIC_REPRESENTATION_ITEM',
              'PLANT_SPATIAL_CONFIGURATION.MEASURE_REPRESENTATION_ITEM',
              'PLANT_SPATIAL_CONFIGURATION.MAPPED_ITEM'] * TYPEOF(i))
             =0))) = 0;
    wr2: SIZEOF(QUERY ( mi <* QUERY ( item <* SELF\representation.items
              | ('PLANT_SPATIAL_CONFIGURATION.MAPPED_ITEM' IN
             TYPEOF(item)) )
              | (NOT (SIZEOF(['PLANT_SPATIAL_CONFIGURATION.' +
             'ADVANCED_CSG_SHAPE_REPRESENTATION',
             'PLANT_SPATIAL_CONFIGURATION.' +
             'HYBRID_SHAPE_REPRESENTATION'] * TYPEOF(mi\mapped_item.
            mapping_source.mapped_representation)) = 1)) )) = 0;
END_ENTITY; -- hybrid_shape_representation

ENTITY hyperbola
  SUBTYPE OF (conic);
    semi_axis      : positive_length_measure;
    semi_imag_axis : positive_length_measure;
END_ENTITY; -- hyperbola

ENTITY inline_equipment
  SUBTYPE OF (piping_component_definition);
END_ENTITY; -- inline_equipment

ENTITY insert_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- insert_fitting_classification

ENTITY instrumentation_and_control_system
  SUBTYPE OF (product_definition);
  WHERE
    wr1: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
              'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_' +
              'RELATIONSHIP.RELATED_PRODUCT_DEFINITION')
               | (('PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(pdr.
             relating_product_definition.formation.of_product)) AND (pdr.
             relating_product_definition.frame_of_reference.name =
             'functional_occurrence')) )) = 1;
END_ENTITY; -- instrumentation_and_control_system
```

```
ENTITY interfering_shape_element
  SUBTYPE OF (shape_aspect, shape_aspect_relationship);
END_ENTITY; -- interfering_shape_element

ENTITY intersection_curve
  SUBTYPE OF (surface_curve);
  WHERE
    wr1: SIZEOF(SELF\surface_curve.associated_geometry) = 2;
    wr2: associated_surface(SELF\surface_curve.associated_geometry[1])
           <> associated_surface(SELF\surface_curve.associated_geometry
           [2]);
END_ENTITY; -- intersection_curve

ENTITY lap_joint_stub_end_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- lap_joint_stub_end_fitting_classification

ENTITY lateral_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- lateral_fitting_classification

ENTITY length_measure_with_unit
  SUBTYPE OF (measure_with_unit);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.LENGTH_UNIT' IN TYPEOF(SELF\
           measure_with_unit.unit_component);
END_ENTITY; -- length_measure_with_unit

ENTITY length_unit
  SUBTYPE OF (named_unit);
  WHERE
    wr1: (SELF\named_unit.dimensions.length_exponent = 1) AND (SELF\
           named_unit.dimensions.mass_exponent = 0) AND (SELF\
           named_unit.dimensions.time_exponent = 0) AND (SELF\
           named_unit.dimensions.electric_current_exponent = 0) AND (
           SELF\named_unit.dimensions.
           thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
           .dimensions.amount_of_substance_exponent = 0) AND (SELF\
           named_unit.dimensions.luminous_intensity_exponent = 0);
END_ENTITY; -- length_unit

ENTITY line
  SUBTYPE OF (curve);
    pnt : cartesian_point;
    dir : vector;
  WHERE
    wr1: dir.dim = pnt.dim;
END_ENTITY; -- line

ENTITY line_branch_connection
  SUBTYPE OF (shape_aspect_relationship);
  WHERE
    wr1: SELF.description = 'branch_location';
    wr2: 'PLANT_SPATIAL_CONFIGURATION.PLANT_LINE_SEGMENT_DEFINITION' IN
           TYPEOF(SELF.relating_shape_aspect.of_shape.definition);
    wr3: 'PLANT_SPATIAL_CONFIGURATION.PLANT_LINE_SEGMENT_TERMINATION' IN
           TYPEOF(SELF.related_shape_aspect);
END_ENTITY; -- line_branch_connection
```

```
ENTITY line_less_piping_system
  SUBTYPE OF (product_definition);
END_ENTITY; -- line_less_piping_system

ENTITY line_plant_item_branch_connection
  SUBTYPE OF (shape_aspect_relationship);
END_ENTITY; -- line_plant_item_branch_connection

ENTITY line_plant_item_connection
  SUBTYPE OF (shape_aspect_relationship);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.PLANT_LINE_SEGMENT_TERMINATION' IN
              TYPEOF(SELF.relating_shape_aspect);
    wr2: 'PLANT_SPATIAL_CONFIGURATION.PLANT_ITEM_CONNECTOR' IN TYPEOF(
             SELF.related_shape_aspect);
    wr3: 'PLANT_SPATIAL_CONFIGURATION.PLANT_ITEM_DEFINITION' IN TYPEOF(
             SELF.related_shape_aspect.of_shape.definition);
END_ENTITY; -- line_plant_item_connection

ENTITY line_termination_connection
  SUBTYPE OF (shape_aspect_relationship);
  WHERE
    wr1: SIZEOF(TYPEOF(SELF.relating_shape_aspect) * [
             'PLANT_SPATIAL_CONFIGURATION.CONNECTION_NODE',
             'PLANT_SPATIAL_CONFIGURATION.PLANT_LINE_SEGMENT_TERMINATION'])
             >= 1;
    wr2: 'PLANT_SPATIAL_CONFIGURATION.PLANT_LINE_SEGMENT_TERMINATION' IN
              TYPEOF(SELF.related_shape_aspect);
END_ENTITY; -- line_termination_connection

ENTITY local_time;
    hour_component   : hour_in_day;
    minute_component : OPTIONAL minute_in_hour;
    second_component : OPTIONAL second_in_minute;
    zone             : coordinated_universal_time_offset;
  WHERE
    wr1: valid_time(SELF);
END_ENTITY; -- local_time

ENTITY loop
  SUPERTYPE OF (ONEOF (edge_loop,poly_loop))
  SUBTYPE OF (topological_representation_item);
END_ENTITY; -- loop

ENTITY make_from_usage_option
  SUBTYPE OF (product_definition_usage);
    ranking           : INTEGER;
    ranking_rationale : text;
    quantity          : measure_with_unit;
  WHERE
    wr1: ranking > 0;
END_ENTITY; -- make_from_usage_option

ENTITY manifold_solid_brep
  SUBTYPE OF (solid_model);
    outer : closed_shell;
END_ENTITY; -- manifold_solid_brep
```

```
ENTITY mapped_item
  SUBTYPE OF (representation_item);
    mapping_source : representation_map;
    mapping_target : representation_item;
  WHERE
    wr1: acyclic_mapped_representation(
         using_representations(SELF),[SELF]);
END_ENTITY; -- mapped_item

ENTITY mass_measure_with_unit
  SUBTYPE OF (measure_with_unit);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.MASS_UNIT' IN TYPEOF(SELF\
            measure_with_unit.unit_component);
END_ENTITY; -- mass_measure_with_unit

ENTITY mass_unit
  SUBTYPE OF (named_unit);
  WHERE
    wr1: (SELF\named_unit.dimensions.length_exponent = 0) AND (SELF\
            named_unit.dimensions.mass_exponent = 1) AND (SELF\
            named_unit.dimensions.time_exponent = 0) AND (SELF\
            named_unit.dimensions.electric_current_exponent = 0) AND (
            SELF\named_unit.dimensions.
            thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
            .dimensions.amount_of_substance_exponent = 0) AND (SELF\
            named_unit.dimensions.luminous_intensity_exponent = 0);
END_ENTITY; -- mass_unit

ENTITY material_designation;
    name          : label;
    of_definition : characterized_definition;
END_ENTITY; -- material_designation

ENTITY material_designation_relationship;
    name        : label;
    description : text;
    designation : material_designation;
    property    : characterized_material_property;
END_ENTITY; -- material_designation_relationship

ENTITY material_property
  SUBTYPE OF (property_definition);
  UNIQUE
    ur1 : name, definition;
  WHERE
    wr1: SIZEOF(QUERY ( temp <* bag_to_set(USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.' +
            'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION')) | (((
            'MATERIAL_PROPERTY_REPRESENTATION_SCHEMA' +
            'MATERIAL_PROPERTY_REPRESENTATION') IN TYPEOF(temp)) AND ((
            'PLANT_SPATIAL_CONFIGURATION.' + 'CHARACTERIZED_OBJECT') IN
            TYPEOF(temp\property_definition.definition))) )) = 0;
END_ENTITY; -- material_property

ENTITY measure_representation_item
  SUBTYPE OF (representation_item, measure_with_unit);
END_ENTITY; -- measure_representation_item
```

```
ENTITY measure_with_unit
  SUPERTYPE OF (ONEOF (length_measure_with_unit,mass_measure_with_unit,
      time_measure_with_unit,thermodynamic_temperature_measure_with_unit,
      amount_of_substance_measure_with_unit,
      plane_angle_measure_with_unit,
      solid_angle_measure_with_unit,area_measure_with_unit,
      volume_measure_with_unit,ratio_measure_with_unit));
    value_component : measure_value;
    unit_component  : unit;
  WHERE
    wr1: valid_units(SELF);
END_ENTITY; -- measure_with_unit

ENTITY name_assignment
  ABSTRACT SUPERTYPE;
    assigned_name : label;
END_ENTITY; -- name_assignment

ENTITY named_unit
  SUPERTYPE OF (ONEOF (si_unit,conversion_based_unit,
      context_dependent_unit) ANDOR ONEOF (length_unit,mass_unit,
      time_unit,thermodynamic_temperature_unit,amount_of_substance_unit,
      plane_angle_unit,solid_angle_unit,area_unit,volume_unit,
      ratio_unit));
    dimensions : dimensional_exponents;
END_ENTITY; -- named_unit

ENTITY offset_curve_2d
  SUBTYPE OF (curve);
    basis_curve    : curve;
    distance       : length_measure;
    self_intersect : LOGICAL;
  WHERE
    wr1: basis_curve.dim = 2;
END_ENTITY; -- offset_curve_2d

ENTITY offset_curve_3d
  SUBTYPE OF (curve);
    basis_curve    : curve;
    distance       : length_measure;
    self_intersect : LOGICAL;
    ref_direction  : direction;
  WHERE
    wr1: (basis_curve.dim = 3) AND (ref_direction.dim = 3);
END_ENTITY; -- offset_curve_3d

ENTITY offset_surface
  SUBTYPE OF (surface);
    basis_surface  : surface;
    distance       : length_measure;
    self_intersect : LOGICAL;
END_ENTITY; -- offset_surface

ENTITY olet_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- olet_fitting_classification
```

```
ENTITY open_shell
  SUBTYPE OF (connected_face_set);
END_ENTITY; -- open_shell

ENTITY organization;
    id          : OPTIONAL identifier;
    name        : label;
    description : text;
END_ENTITY; -- organization

ENTITY organization_assignment
  ABSTRACT SUPERTYPE;
    assigned_organization : organization;
    role                  : organization_role;
END_ENTITY; -- organization_assignment

ENTITY organization_role;
    name : label;
END_ENTITY; -- organization_role

ENTITY organizational_project;
    name                       : label;
    description                : text;
    responsible_organizations : SET [1:?] OF organization;
END_ENTITY; -- organizational_project

ENTITY oriented_closed_shell
  SUBTYPE OF (closed_shell);
    closed_shell_element : closed_shell;
    orientation          : BOOLEAN;
  DERIVE
    SELF\connected_face_set.cfs_faces : SET [1:?] OF face :=
                            conditional_reverse(SELF.
                            orientation,SELF.
                            closed_shell_element.cfs_faces);
  WHERE
    wr1: NOT ('PLANT_SPATIAL_CONFIGURATION.ORIENTED_CLOSED_SHELL' IN
              TYPEOF(SELF.closed_shell_element));
END_ENTITY; -- oriented_closed_shell

ENTITY oriented_edge
  SUBTYPE OF (edge);
    edge_element : edge;
    orientation  : BOOLEAN;
  DERIVE
    SELF\edge.edge_start : vertex := boolean_choose(SELF.orientation,
                               SELF.edge_element.edge_start,SELF.
                               edge_element.edge_end);
    SELF\edge.edge_end   : vertex := boolean_choose(SELF.orientation,
                               SELF.edge_element.edge_end,SELF.
                               edge_element.edge_start);
  WHERE
    wr1: NOT ('PLANT_SPATIAL_CONFIGURATION.ORIENTED_EDGE' IN TYPEOF(SELF
              .edge_element));
END_ENTITY; -- oriented_edge

ENTITY oriented_face
  SUBTYPE OF (face);
    face_element : face;
```

```
    orientation  : BOOLEAN;
  DERIVE
    SELF\face.bounds : SET [1:?] OF face_bound := conditional_reverse(
                          SELF.orientation,SELF.face_element.bounds);
  WHERE
    wr1: NOT ('PLANT_SPATIAL_CONFIGURATION.ORIENTED_FACE' IN TYPEOF(SELF
           .face_element));
END_ENTITY; -- oriented_face

ENTITY oriented_open_shell
  SUBTYPE OF (open_shell);
    open_shell_element : open_shell;
    orientation        : BOOLEAN;
  DERIVE
    SELF\connected_face_set.cfs_faces : SET [1:?] OF face :=
                                          conditional_reverse(SELF.
                                          orientation,SELF.
                                          open_shell_element.cfs_faces);
  WHERE
    wr1: NOT ('PLANT_SPATIAL_CONFIGURATION.ORIENTED_OPEN_SHELL' IN
           TYPEOF(SELF.open_shell_element));
END_ENTITY; -- oriented_open_shell

ENTITY oriented_path
  SUBTYPE OF (path);
    path_element : path;
    orientation  : BOOLEAN;
  DERIVE
    SELF\path.edge_list : LIST [1:?] OF UNIQUE oriented_edge :=
                            conditional_reverse(SELF.orientation,SELF.
                            path_element.edge_list);
  WHERE
    wr1: NOT ('PLANT_SPATIAL_CONFIGURATION.ORIENTED_PATH' IN TYPEOF(SELF
           .path_element));
END_ENTITY; -- oriented_path

ENTITY orifice_plate_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- orifice_plate_fitting_classification

ENTITY parabola
  SUBTYPE OF (conic);
    focal_dist : length_measure;
  WHERE
    wr1: focal_dist <> 0;
END_ENTITY; -- parabola

ENTITY path
  SUPERTYPE OF (ONEOF (edge_loop,oriented_path))
  SUBTYPE OF (topological_representation_item);
    edge_list : LIST [1:?] OF UNIQUE oriented_edge;
  WHERE
    wr1: path_head_to_tail(SELF);
END_ENTITY; -- path

ENTITY pcurve
  SUBTYPE OF (curve);
    basis_surface      : surface;
    reference_to_curve : definitional_representation;
```

```
  WHERE
    wr1: SIZEOF(reference_to_curve\representation.items) = 1;
    wr2: 'PLANT_SPATIAL_CONFIGURATION.CURVE' IN TYPEOF(
              reference_to_curve\representation.items[1]);
    wr3: reference_to_curve\representation.items[1]\
              geometric_representation_item.dim = 2;
END_ENTITY; -- pcurve

ENTITY person;
    id            : identifier;
    last_name     : OPTIONAL label;
    first_name    : OPTIONAL label;
    middle_names  : OPTIONAL LIST [1:?] OF label;
    prefix_titles : OPTIONAL LIST [1:?] OF label;
    suffix_titles : OPTIONAL LIST [1:?] OF label;
  UNIQUE
    ur1 : id;
  WHERE
    wr1: EXISTS(last_name) OR EXISTS(first_name);
END_ENTITY; -- person

ENTITY person_and_organization;
    the_person       : person;
    the_organization : organization;
END_ENTITY; -- person_and_organization

ENTITY person_and_organization_assignment
  ABSTRACT SUPERTYPE;
    assigned_person_and_organization : person_and_organization;
    role                             : person_and_organization_role;
END_ENTITY; -- person_and_organization_assignment

ENTITY person_and_organization_role;
    name : label;
END_ENTITY; -- person_and_organization_role

ENTITY person_assignment
  ABSTRACT SUPERTYPE;
    assigned_person : person;
    role            : person_role;
END_ENTITY; -- person_assignment

ENTITY person_role;
    name : label;
END_ENTITY; -- person_role

ENTITY pipe_cap_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- pipe_cap_fitting_classification

ENTITY pipe_classification
  SUBTYPE OF (group);
END_ENTITY; -- pipe_classification

ENTITY piping_component_definition
  SUBTYPE OF (product_definition);
  WHERE
    wr1: (SIZEOF(USEDIN(SELF\product_definition.formation.of_product,
              'PLANT_SPATIAL_CONFIGURATION.' +
```

```
            'PRODUCT_RELATED_PRODUCT_CATEGORY.PRODUCTS')) = 1) AND (
            SIZEOF(QUERY ( prpc <* USEDIN(SELF\product_definition.
            formation.of_product,'PLANT_SPATIAL_CONFIGURATION.' +
            'PRODUCT_RELATED_PRODUCT_CATEGORY.PRODUCTS') | (prpc\
            product_category.name IN ['ducting_component','equipment',
            'fitting','instrument']) )) = 1);
    wr2: SIZEOF(USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
            'MATERIAL_DESIGNATION.OF_PRODUCT')) = 1;
    wr3: SIZEOF(QUERY ( pscdr <* USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.' +
            'PLANT_SPATIAL_CONFIGURATION_DOCUMENT_REFERENCE.ITEMS') | (
            pscdr\document_reference.assigned_document.kind.
            product_data_type = 'material_specification') )) >= 1;
    wr4: SIZEOF(QUERY ( prop_def <* USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.PROPERTY_DEFINITION.DEFINITION')
             | (SIZEOF(QUERY ( pdr <* USEDIN(prop_def,
            'PLANT_SPATIAL_CONFIGURATION.' +
            'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION') | (NOT
            (('PLANT_SPATIAL_CONFIGURATION.' +
            'SHAPE_DEFINITION_REPRESENTATION'
            IN TYPEOF(pdr)) AND ('PLANT_SPATIAL_CONFIGURATION.' +
            'SHAPE_DIMENSION_REPRESENTATION'
            IN TYPEOF(pdr.used_representation)) AND (pdr.
            used_representation.name = 'nominal_size') AND (SIZEOF(pdr.
            used_representation.items) = 1) AND (SIZEOF(QUERY ( it <*
            pdr.used_representation.items | (NOT (
            'PLANT_SPATIAL_CONFIGURATION.LENGTH_MEASURE_WITH_UNIT' IN
            TYPEOF(it))) )) = 0))) )) = 0) )) = 1;
    wr5: SIZEOF(QUERY ( prop_def <* USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.PROPERTY_DEFINITION.DEFINITION')
             | (SIZEOF(QUERY ( pdr <* USEDIN(prop_def,
            'PLANT_SPATIAL_CONFIGURATION.' +
            'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION') | (NOT ((
            pdr.used_representation.name = 'design_pressure_rating') AND
             (SIZEOF(pdr.used_representation.items) = 1) AND (
            'PLANT_SPATIAL_CONFIGURATION.MEASURE_WITH_UNIT' IN TYPEOF(
            pdr.used_representation.items[1]))))) )) = 0) )) > 0;
END_ENTITY; -- piping_component_definition

ENTITY piping_connector_classification
  SUBTYPE OF (group);
END_ENTITY; -- piping_connector_classification

ENTITY piping_design_csg_shape_representation
  SUBTYPE OF (shape_representation);
  WHERE
    wr1: SIZEOF (QUERY (i <* SELF\representation.items |
            SIZEOF (['PLANT_SPATIAL_CONFIGURATION.BOOLEAN_RESULT',
            'PLANT_SPATIAL_CONFIGURATION.CSG_SOLID',
            'PLANT_SPATIAL_CONFIGURATION.AXIS2_PLACEMENT_3D',
            'PLANT_SPATIAL_CONFIGURATION.MAPPED_ITEM'] *
            TYPEOF(i)) = 0 )) = 0;
    wr2: SIZEOF (QUERY (mi <* QUERY (item <* SELF\representation.items |
            ('PLANT_SPATIAL_CONFIGURATION.MAPPED_ITEM' IN
            TYPEOF(item))) | NOT ('PLANT_SPATIAL_CONFIGURATION.' +
            'PIPING_DESIGN_CSG_SHAPE_REPRESENTATION' IN
            TYPEOF(mi\mapped_item.mapping_source.mapped_representation))
```

```
            )) = 0;
END_ENTITY; -- piping_design_csg_shape_representation

ENTITY piping_system
  SUBTYPE OF (product_definition);
  WHERE
    wr1: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
              'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_' +
              'RELATIONSHIP.RELATED_PRODUCT_DEFINITION')
               | (('PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(pdr.
              relating_product_definition.formation.of_product)) AND (pdr.
              relating_product_definition.frame_of_reference.name =
              'functional_occurrence')) )) = 1;
END_ENTITY; -- piping_system

ENTITY placement
  SUPERTYPE OF (ONEOF (axis1_placement,axis2_placement_2d,
      axis2_placement_3d))
  SUBTYPE OF (geometric_representation_item);
    location : cartesian_point;
END_ENTITY; -- placement

ENTITY plane
  SUBTYPE OF (elementary_surface);
END_ENTITY; -- plane

ENTITY plane_angle_measure_with_unit
  SUBTYPE OF (measure_with_unit);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.PLANE_ANGLE_UNIT' IN TYPEOF(SELF\
            measure_with_unit.unit_component);
END_ENTITY; -- plane_angle_measure_with_unit

ENTITY plane_angle_unit
  SUBTYPE OF (named_unit);
  WHERE
    wr1: (SELF\named_unit.dimensions.length_exponent = 0) AND (SELF\
            named_unit.dimensions.mass_exponent = 0) AND (SELF\
            named_unit.dimensions.time_exponent = 0) AND (SELF\
            named_unit.dimensions.electric_current_exponent = 0) AND (
            SELF\named_unit.dimensions.
            thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
            .dimensions.amount_of_substance_exponent = 0) AND (SELF\
            named_unit.dimensions.luminous_intensity_exponent = 0);
END_ENTITY; -- plane_angle_unit

ENTITY plant
  SUBTYPE OF (product);
  WHERE
    wr1: (SIZEOF(QUERY ( pscoa <* USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.' +
            'PLANT_SPATIAL_CONFIGURATION_ORGANIZATION_ASSIGNMENT.ITEMS')
             | (pscoa\organization_assignment.role.name =
            'plant_operator') )) + SIZEOF(QUERY ( pscpaoa <*
            USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.PLANT_SPATIAL_' +
            'CONFIGURATION_PERSON_AND_ORGANIZATION_ASSIGNMENT.'
            + 'ITEMS') | (pscpaoa\person_and_organization_assignment.
            role.name = 'plant_operator') ))) <= 1;
```

```
      wr2: (SIZEOF(QUERY ( pscoa <* USEDIN(SELF,
                'PLANT_SPATIAL_CONFIGURATION.' +
                'PLANT_SPATIAL_CONFIGURATION_ORGANIZATION_ASSIGNMENT.ITEMS')
                 | (pscoa\organization_assignment.role.name = 'plant_owner'
                ))) + SIZEOF(QUERY ( pscpaoa <* USEDIN(SELF,
                'PLANT_SPATIAL_CONFIGURATION.PLANT_SPATIAL_' +
                'CONFIGURATION_PERSON_AND_ORGANIZATION_ASSIGNMENT.'
                + 'ITEMS') | (pscpaoa\person_and_organization_assignment.
                role.name = 'plant_owner') )) + SIZEOF(QUERY ( pscpa <*
                USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
                'PLANT_SPATIAL_CONFIGURATION_PERSON_ASSIGNMENT.ITEMS') | (
                pscpa\person_assignment.role.name = 'plant_owner') ))) >= 1;
      wr3: (SIZEOF(QUERY ( pscoa <* USEDIN(SELF,
                'PLANT_SPATIAL_CONFIGURATION.' +
                'PLANT_SPATIAL_CONFIGURATION_ORGANIZATION_ASSIGNMENT.ITEMS')
                 | (pscoa\organization_assignment.role.name =
                'plant_project_owner') )) + SIZEOF(QUERY ( pscpaoa <*
                USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.PLANT_SPATIAL' +
                'CONFIGURATION_PERSON_AND_ORGANIZATION_ASSIGNMENT.'
                + 'ITEMS') | (pscpaoa\person_and_organization_assignment.
                role.name = 'plant_project_owner') ))) >= 1;
      wr4: SIZEOF(QUERY ( pdf <* USEDIN(SELF,
                'PLANT_SPATIAL_CONFIGURATION.' +
                'PRODUCT_DEFINITION_FORMATION.OF_PRODUCT') | (NOT (SIZEOF(
                QUERY ( pd <* USEDIN(pdf,
                'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION.FORMATION')
                 | (pd.frame_of_reference.name = 'functional_occurrence') ))
                <= 1)) )) = 0;
  END_ENTITY; -- plant

  ENTITY plant_item_connection
    SUBTYPE OF (shape_aspect, shape_aspect_relationship);
    WHERE
      wr1: SELF\shape_aspect_relationship.name IN ['connection_definition',
                'connection_occurrence'];
      wr2: 'PLANT_SPATIAL_CONFIGURATION.PLANT_ITEM_CONNECTOR' IN TYPEOF(
                SELF\shape_aspect_relationship.relating_shape_aspect);
      wr3: 'PLANT_SPATIAL_CONFIGURATION.PLANT_ITEM_CONNECTOR' IN TYPEOF(
                SELF\shape_aspect_relationship.related_shape_aspect);
      wr4: SELF\shape_aspect.of_shape.definition.frame_of_reference.name
                IN ['functional_occurrence','physical_occurrence',
                'functional_definition','physical_definition'];
      wr5: SELF\shape_aspect_relationship.relating_shape_aspect.of_shape.
                definition.frame_of_reference.name = SELF\
                shape_aspect_relationship.related_shape_aspect.of_shape.
                definition.frame_of_reference.name;
      wr6: SIZEOF(USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
                'CLASSIFICATION_ASSIGNMENT.ITEMS')) > 0;
      wr7: SIZEOF(QUERY ( pscca <* USEDIN(SELF,'PLANT_SPATIAL_' +
                'CONFIGURATION.CLASSIFICATION_ASSIGNMENT.ITEMS')
                 | (NOT (SIZEOF(['PLANT_SPATIAL_CONFIGURATION.' +
                'CONNECTION_FUNCTIONAL_CLASSIFICATION',
                'PLANT_SPATIAL_CONFIGURATION.' +
                'CONNECTION_MOTION_CLASSIFICATION'] * TYPEOF(pscca\
                group_assignment.assigned_group)) > 1)) )) = 0;
      wr8: SIZEOF(QUERY ( pdr <* USEDIN(SELF\shape_aspect.of_shape.
                definition,'PLANT_SPATIAL_CONFIGURATION.' +
                'PRODUCT_DEFINITION_RELATIONSHIP.'
                + 'RELATED_PRODUCT_DEFINITION') | (pdr.name =
```

```
            'support_usage_connection') )) <= 1;
END_ENTITY; -- plant_item_connection

ENTITY plant_item_connector
  SUBTYPE OF (shape_aspect);
  WHERE
    wr1: SELF.of_shape.definition.frame_of_reference.name IN [
            'functional_definition','physical_definition',
            'functional_occurrence','physical_occurrence'];
    wr2: SIZEOF(QUERY ( pic <* (bag_to_set(USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.' +
            'SHAPE_ASPECT_RELATIONSHIP.RELATED_SHAPE_ASPECT')) +
            bag_to_set(USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
            'SHAPE_ASPECT_RELATIONSHIP.RELATING_SHAPE_ASPECT'))) | (
            'PLANT_SPATIAL_CONFIGURATION.PLANT_ITEM_CONNECTION' IN
            TYPEOF(pic)) )) <= 1;
END_ENTITY; -- plant_item_connector

ENTITY plant_item_definition
  SUBTYPE OF (product_definition);
  WHERE
    wr1: SELF.frame_of_reference\application_context_element.name IN [
            'functional_definition','physical_definition',
            'functional_occurrence','physical_occurrence',
            'fabrication_assembly'];
    wr2: (NOT (SELF.frame_of_reference.name = 'physical_occurrence')) OR
             (SIZEOF(QUERY ( pdr <* USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_' +
            'RELATIONSHIP.RELATED_PRODUCT_DEFINITION')
             | (SIZEOF(TYPEOF(pdr) * [
            'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_USAGE',
            'PLANT_SPATIAL_CONFIGURATION.MAKE_FROM_USAGE_OPTION',
            'PLANT_SPATIAL_CONFIGURATION.ASSEMBLY_COMPONENT_USAGE'])
            = 1) )) <= 1);
    wr3: (NOT (SELF.frame_of_reference.name = 'physical_definition')) OR
             (SIZEOF(QUERY ( pdr <* USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_' +
            'RELATIONSHIP.RELATING_PRODUCT_DEFINITION')
             | (pdr.related_product_definition.frame_of_reference.name =
             'physical_occurrence') )) <= 1);
    wr4: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_' +
            'RELATIONSHIP.RELATING_PRODUCT_DEFINITION')
             | (
            'PLANT_SPATIAL_CONFIGURATION.PLANT_ITEM_DEFINITION_CLASS' IN
             TYPEOF(pdr.related_product_definition)) )) <= 1;
END_ENTITY; -- plant_item_definition

ENTITY plant_item_definition_class
  SUBTYPE OF (product_definition_with_associated_documents);
END_ENTITY; -- plant_item_definition_class

ENTITY plant_item_interference
  SUBTYPE OF (product_definition_relationship);
END_ENTITY; -- plant_item_interference

ENTITY plant_item_route
  SUBTYPE OF (product_definition_shape);
  WHERE
```

```
    wr1: SELF\property_definition.definition.frame_of_reference\
         application_context_element.name = 'physical_occurrence';
    wr2: SIZEOF(TYPEOF(SELF\property_definition.definition) * [
         'PLANT_SPATIAL_CONFIGURATION.PLANT_LINE_DEFINITION',
         'PLANT_SPATIAL_CONFIGURATION.PLANT_LINE_SEGMENT_' +
         'DEFINITION'])
         = 1;
END_ENTITY; -- plant_item_route

ENTITY plant_item_weight_representation
  SUBTYPE OF (property_definition_representation);
  WHERE
    wr1: SELF.used_representation.name = 'item_weight';
    wr2: SIZEOF(SELF.used_representation.items) >= 2;
    wr3: SIZEOF(QUERY ( it <* SELF.used_representation.items | ((
         'PLANT_SPATIAL_CONFIGURATION.DESCRIPTIVE_REPRESENTATION_ITEM'
         IN TYPEOF(it)) AND (it.name = 'weight_state')) )) = 1;
    wr4: SIZEOF(QUERY ( it <* SELF.used_representation.items | ((
         'PLANT_SPATIAL_CONFIGURATION.MEASURE_REPRESENTATION_ITEM' IN
          TYPEOF(it)) AND (it.name IN ['weight_value',
         'mass_value'])))))
         = 1;
END_ENTITY; -- plant_item_weight_representation

ENTITY plant_line_definition
  SUBTYPE OF (product_definition_with_associated_documents);
  WHERE
    wr1: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_' +
         'RELATIONSHIP.RELATED_PRODUCT_DEFINITION')
          | ('PLANT_SPATIAL_CONFIGURATION.PIPING_SYSTEM' IN TYPEOF(
         pdr.relating_product_definition)) )) = 1;
    wr2: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_' +
         'RELATIONSHIP.RELATING_PRODUCT_DEFINITION')
          | (
         'PLANT_SPATIAL_CONFIGURATION.PLANT_LINE_SEGMENT_DEFINITION'
         IN TYPEOF(pdr.related_product_definition)) )) > 0;
    wr3: ((NOT (SIZEOF(QUERY ( pd <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.PROPERTY_DEFINITION.DEFINITION')
          | (SIZEOF(USEDIN(pd,'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION')) > 0) )) =
         0)) XOR (SIZEOF(QUERY ( pd <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.PROPERTY_DEFINITION.DEFINITION')
          | (SIZEOF(QUERY ( pdr <* USEDIN(pd,
         'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION') | (SIZEOF(
         QUERY ( rep <* USEDIN(pdr.used_representation.context_of_items,
         'PLANT_SPATIAL_CONFIGURATION.REPRESENTATION.CONTEXT_OF_ITEMS')
          | (SIZEOF(QUERY ( prop_def_rep <* USEDIN(rep,
         'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.USED_REPRESENTATION') |
         (SIZEOF(['PLANT_SPATIAL_CONFIGURATION.SITE',
         'PLANT_SPATIAL_CONFIGURATION.SITE_BUILDING'] * TYPEOF(
         prop_def_rep.definition)) = 0) AND (NOT (
         'PLANT_SPATIAL_CONFIGURATION.SITE' IN TYPEOF(prop_def_rep.
         definition.definition))) AND (NOT (
         'PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(prop_def_rep.
         definition.definition.formation.of_product))) )) > 0) )) >
```

```
            0) )) > 0) )) = 0));
    wr4: SELF.frame_of_reference.life_cycle_stage = 'design';
    wr5: SELF.frame_of_reference\application_context_element.name =
            'system';
END_ENTITY; -- plant_line_definition

ENTITY plant_line_segment_definition
  SUBTYPE OF (product_definition);
  WHERE
    wr1: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_'+
            'RELATIONSHIP.RELATED_PRODUCT_DEFINITION')
             | ('PLANT_SPATIAL_CONFIGURATION.PLANT_LINE_DEFINITION' IN
            TYPEOF(pdr.relating_product_definition)) )) > 0;
    wr2: SIZEOF(QUERY ( pd <* USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
            'PROPERTY_DEFINITION.DEFINITION')
             | ('PLANT_SPATIAL_CONFIGURATION.SHAPE_DEFINITION' IN
            TYPEOF(pd)) )) > 1;
    wr3: SELF.frame_of_reference\application_context_element.name =
            'functional_definition';
END_ENTITY; -- plant_line_segment_definition

ENTITY plant_line_segment_termination
  SUBTYPE OF (shape_aspect);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.PLANT_LINE_SEGMENT_DEFINITION' IN
            TYPEOF(SELF.of_shape.definition);
    wr2: SIZEOF(QUERY ( sar <* (USEDIN(SELF,
            'PLANT_SPATIAL_CONFIGURATION.' +
            'SHAPE_ASPECT_RELATIONSHIP.RELATING_SHAPE_ASPECT') + USEDIN(
            SELF,'PLANT_SPATIAL_CONFIGURATION.' +
            'SHAPE_ASPECT_RELATIONSHIP.RELATED_SHAPE_ASPECT')) | (NOT (
            SIZEOF(TYPEOF(sar) * [
            'PLANT_SPATIAL_CONFIGURATION.LINE_BRANCH_CONNECTION',
            'PLANT_SPATIAL_CONFIGURATION.LINE_PLANT_ITEM_CONNECTION',
            'PLANT_SPATIAL_CONFIGURATION.LINE_TERMINATION_CONNECTION'])
            > 0)) )) = 0;
END_ENTITY; -- plant_line_segment_termination

ENTITY plant_spatial_configuration_action_request_assignment
  SUBTYPE OF (action_request_assignment);
    items : SET [1:?] OF plant_spatial_configuration_action_request_item;
END_ENTITY; -- plant_spatial_configuration_action_request_assignment

ENTITY plant_spatial_configuration_approval_assignment
  SUBTYPE OF (approval_assignment);
    items : SET [1:?] OF approval_item;
END_ENTITY; -- plant_spatial_configuration_approval_assignment

ENTITY plant_spatial_configuration_change_assignment
  SUBTYPE OF (action_assignment);
    items : SET [1:?] OF change_item;
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.CHANGE_ACTION' IN TYPEOF(SELF.
            assigned_action);
END_ENTITY; -- plant_spatial_configuration_change_assignment

ENTITY plant_spatial_configuration_date_and_time_assignment
  SUBTYPE OF (date_and_time_assignment);
```

```
    items : SET [1:?] OF plant_spatial_configuration_date_and_time_item;
END_ENTITY; -- plant_spatial_configuration_date_and_time_assignment

ENTITY plant_spatial_configuration_date_assignment
  SUBTYPE OF (date_assignment);
    items : SET [1:?] OF plant_spatial_configuration_dated_item;
END_ENTITY; -- plant_spatial_configuration_date_assignment

ENTITY plant_spatial_configuration_document_reference
  SUBTYPE OF (document_reference);
    items : SET [1:?] OF plant_spatial_configuration_document_item;
END_ENTITY; -- plant_spatial_configuration_document_reference

ENTITY plant_spatial_configuration_organization_assignment
  SUBTYPE OF (organization_assignment);
    items : SET [1:?] OF plant_spatial_configuration_organization_item;
  WHERE
    wr1: plant_spatial_configuration_organization_correlation(SELF);
END_ENTITY; -- plant_spatial_configuration_organization_assignment

ENTITY plant_spatial_configuration_person_and_organization_assignment
  SUBTYPE OF (person_and_organization_assignment);
    items : SET [1:?] OF
                plant_spatial_configuration_person_and_organization_item;
  WHERE
    wr1: plant_spatial_configuration_person_and_organization_correlation(
             SELF);
END_ENTITY; --plant_spatial_configuration_person_and_organization_assignment

ENTITY plant_spatial_configuration_person_assignment
  SUBTYPE OF (person_assignment);
    items : SET [1:?] OF plant_spatial_configuration_person_item;
  WHERE
    wr1: plant_spatial_configuration_person_correlation(SELF);
END_ENTITY; -- plant_spatial_configuration_person_assignment

ENTITY point
  SUPERTYPE OF (ONEOF (cartesian_point,point_on_curve,point_on_surface,
      point_replica,degenerate_pcurve))
  SUBTYPE OF (geometric_representation_item);
END_ENTITY; -- point

ENTITY point_on_curve
  SUBTYPE OF (point);
    basis_curve     : curve;
    point_parameter : parameter_value;
END_ENTITY; -- point_on_curve

ENTITY point_on_surface
  SUBTYPE OF (point);
    basis_surface     : surface;
    point_parameter_u : parameter_value;
    point_parameter_v : parameter_value;
END_ENTITY; -- point_on_surface

ENTITY point_replica
  SUBTYPE OF (point);
    parent_pt      : point;
    transformation : cartesian_transformation_operator;
```

```
  WHERE
    wr1: transformation.dim = parent_pt.dim;
    wr2: acyclic_point_replica(SELF,parent_pt);
END_ENTITY; -- point_replica

ENTITY poly_loop
  SUBTYPE OF (loop, geometric_representation_item);
    polygon : LIST [3:?] OF UNIQUE cartesian_point;
END_ENTITY; -- poly_loop

ENTITY polyline
  SUBTYPE OF (bounded_curve);
    points : LIST [2:?] OF cartesian_point;
END_ENTITY; -- polyline

ENTITY presentation_layer_assignment;
    name           : label;
    description    : text;
    assigned_items : SET [1:?] OF layered_item;
END_ENTITY; -- presentation_layer_assignment

ENTITY process_capability
  SUBTYPE OF (property_definition);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(SELF.definition\
            product_definition.formation.of_product);
    wr2: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
              'PLANT_SPATIAL_CONFIGURATION.PROPERTY_DEFINITION_' +
              'REPRESENTATION.DEFINITION') |
              ((pdr.used_representation.name =
              'production_capacity') AND (NOT (SIZEOF(QUERY ( it <* pdr.
              used_representation.items | (('PLANT_SPATIAL_' +
              'CONFIGURATION.DESCRIPTIVE_REPRESENTATION_ITEM'
              IN TYPEOF(it)) AND (it.name = 'production_type')) )) = 1)))
              )) = 0;
END_ENTITY; -- process_capability

ENTITY product;
    id                 : identifier;
    name               : label;
    description        : text;
    frame_of_reference : SET [1:?] OF product_context;
  UNIQUE
    ur1 : id;
END_ENTITY; -- product

ENTITY product_category;
    name        : label;
    description : OPTIONAL text;
END_ENTITY; -- product_category

ENTITY product_category_relationship;
    name         : label;
    description  : text;
    category     : product_category;
    sub_category : product_category;
  WHERE
    wr1: acyclic_product_category_relationship(SELF,[SELF.sub_category]);
END_ENTITY; -- product_category_relationship
```

```
ENTITY product_context
  SUBTYPE OF (application_context_element);
    discipline_type : label;
END_ENTITY; -- product_context

ENTITY product_definition;
    id                  : identifier;
    description         : text;
    formation           : product_definition_formation;
    frame_of_reference : product_definition_context;
END_ENTITY; -- product_definition

ENTITY product_definition_context
  SUBTYPE OF (application_context_element);
    life_cycle_stage : label;
END_ENTITY; -- product_definition_context

ENTITY product_definition_formation;
    id          : identifier;
    description : text;
    of_product  : product;
  UNIQUE
    ur1 : id, of_product;
END_ENTITY; -- product_definition_formation

ENTITY product_definition_formation_relationship;
    id                                    : identifier;
    name                                  : label;
    description                           : text;
    relating_product_definition_formation : product_definition_formation;
    related_product_definition_formation  : product_definition_formation;
END_ENTITY; -- product_definition_formation_relationship

ENTITY product_definition_formation_with_specified_source
  SUBTYPE OF (product_definition_formation);
    make_or_buy : source;
END_ENTITY; -- product_definition_formation_with_specified_source

ENTITY product_definition_relationship;
    id                          : identifier;
    name                        : label;
    description                 : text;
    relating_product_definition : product_definition;
    related_product_definition  : product_definition;
END_ENTITY; -- product_definition_relationship

ENTITY product_definition_shape
  SUBTYPE OF (property_definition);
  UNIQUE
    ur1 : definition;
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.CHARACTERIZED_PRODUCT_DEFINITION'
            IN TYPEOF(SELF\property_definition.definition);
END_ENTITY; -- product_definition_shape

ENTITY product_definition_substitute;
    description           : text;
    context_relationship  : product_definition_relationship;
    substitute_definition : product_definition;
```

```
  WHERE
    wr1: context_relationship.related_product_definition :<>:
           substitute_definition;
END_ENTITY; -- product_definition_substitute

ENTITY product_definition_usage
  SUPERTYPE OF (ONEOF (make_from_usage_option,assembly_component_usage))
  SUBTYPE OF (product_definition_relationship);
  UNIQUE
    ur1 : id, relating_product_definition, related_product_definition;
  WHERE
    wr1: acyclic_product_definition_relationship(SELF,[SELF\
             product_definition_relationship.related_product_definition],
             'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_USAGE.' +
             'RELATED_PRODUCT_DEFINITION');
END_ENTITY; -- product_definition_usage

ENTITY product_definition_with_associated_documents
  SUBTYPE OF (product_definition);
    documentation_ids : SET [1:?] OF document;
END_ENTITY; -- product_definition_with_associated_documents

ENTITY product_related_product_category
  SUBTYPE OF (product_category);
    products : SET [1:?] OF product;
END_ENTITY; -- product_related_product_category

ENTITY property_definition;
    name        : label;
    description : text;
    definition  : characterized_definition;
END_ENTITY; -- property_definition

ENTITY property_definition_relationship;
    name                          : label;
    description                   : text;
    relating_property_definition : property_definition;
    related_property_definition  : property_definition;
END_ENTITY; -- property_definition_relationship

ENTITY property_definition_representation;
    definition          : property_definition;
    used_representation : representation;
END_ENTITY; -- property_definition_representation

ENTITY purchase_assignment
  SUBTYPE OF (action_assignment);
    items : SET [1:?] OF purchase_item;
END_ENTITY; -- purchase_assignment

ENTITY quasi_uniform_curve
  SUBTYPE OF (b_spline_curve);
END_ENTITY; -- quasi_uniform_curve

ENTITY quasi_uniform_surface
  SUBTYPE OF (b_spline_surface);
END_ENTITY; -- quasi_uniform_surface
```

```
ENTITY ratio_measure_with_unit
  SUBTYPE OF (measure_with_unit);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.RATIO_UNIT' IN TYPEOF(SELF\
              measure_with_unit.unit_component);
END_ENTITY; -- ratio_measure_with_unit

ENTITY ratio_unit
  SUBTYPE OF (named_unit);
  WHERE
    wr1: (SELF\named_unit.dimensions.length_exponent = 0) AND (SELF\
              named_unit.dimensions.mass_exponent = 0) AND (SELF\
              named_unit.dimensions.time_exponent = 0) AND (SELF\
              named_unit.dimensions.electric_current_exponent = 0) AND (
              SELF\named_unit.dimensions.
              thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
              .dimensions.amount_of_substance_exponent = 0) AND (SELF\
              named_unit.dimensions.luminous_intensity_exponent = 0);
END_ENTITY; -- ratio_unit

ENTITY rational_b_spline_curve
  SUBTYPE OF (b_spline_curve);
    weights_data : LIST [2:?] OF REAL;
  DERIVE
    weights : ARRAY [0:upper_index_on_control_points] OF REAL :=
                  list_to_array(weights_data,0,
                  upper_index_on_control_points);
  WHERE
    wr1: SIZEOF(weights_data) = SIZEOF(SELF\b_spline_curve.
              control_points_list);
    wr2: curve_weights_positive(SELF);
END_ENTITY; -- rational_b_spline_curve

ENTITY rational_b_spline_surface
  SUBTYPE OF (b_spline_surface);
    weights_data : LIST [2:?] OF LIST [2:?] OF REAL;
  DERIVE
    weights : ARRAY [0:u_upper] OF ARRAY [0:v_upper] OF REAL :=
                  make_array_of_array(weights_data,0,u_upper,0,v_upper);
  WHERE
    wr1: (SIZEOF(weights_data) = SIZEOF(SELF\b_spline_surface.
              control_points_list)) AND (SIZEOF(weights_data[1]) = SIZEOF(
              SELF\b_spline_surface.control_points_list[1]));
    wr2: surface_weights_positive(SELF);
END_ENTITY; -- rational_b_spline_surface

ENTITY rectangular_composite_surface
  SUBTYPE OF (bounded_surface);
    segments : LIST [1:?] OF LIST [1:?] OF surface_patch;
  DERIVE
    n_u : INTEGER := SIZEOF(segments);
    n_v : INTEGER := SIZEOF(segments[1]);
  WHERE
    wr1: [] = QUERY ( s <* segments | (n_v <> SIZEOF(s)) );
    wr2: constraints_rectangular_composite_surface(SELF);
END_ENTITY; -- rectangular_composite_surface

ENTITY rectangular_trimmed_surface
  SUBTYPE OF (bounded_surface);
```

```
    basis_surface : surface;
    u1            : parameter_value;
    u2            : parameter_value;
    v1            : parameter_value;
    v2            : parameter_value;
    usense        : BOOLEAN;
    vsense        : BOOLEAN;
  WHERE
    wr1: u1 <> u2;
    wr2: v1 <> v2;
    wr3: (('PLANT_SPATIAL_CONFIGURATION.ELEMENTARY_SURFACE' IN TYPEOF(
            basis_surface)) AND (NOT (
            'PLANT_SPATIAL_CONFIGURATION.PLANE' IN
            TYPEOF(basis_surface))))
            OR ('PLANT_SPATIAL_CONFIGURATION.SURFACE_OF_REVOLUTION' IN
            TYPEOF(basis_surface)) OR (usense = (u2 > u1));
    wr4: ('PLANT_SPATIAL_CONFIGURATION.SPHERICAL_SURFACE' IN TYPEOF(
            basis_surface)) OR (
            'PLANT_SPATIAL_CONFIGURATION.TOROIDAL_SURFACE' IN TYPEOF(
            basis_surface)) OR (vsense = (v2 > v1));
END_ENTITY; -- rectangular_trimmed_surface

ENTITY reducer_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- reducer_fitting_classification

ENTITY reference_geometry
  SUBTYPE OF (derived_shape_aspect);
  WHERE
    wr1: SIZEOF(QUERY ( pd <* USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.'
            + 'PROPERTY_DEFINITION.DEFINITION') | (NOT
            (SIZEOF(USEDIN(pd, 'PLANT_SPATIAL_CONFIGURATION.' +
            'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION')) >= 1)) ))
            = 0;
END_ENTITY; -- reference_geometry

ENTITY reparametrised_composite_curve_segment
  SUBTYPE OF (composite_curve_segment);
    param_length : parameter_value;
  WHERE
    wr1: param_length > 0;
END_ENTITY; -- reparametrised_composite_curve_segment

ENTITY representation;
    name             : label;
    items            : SET [1:?] OF representation_item;
    context_of_items : representation_context;
END_ENTITY; -- representation

ENTITY representation_context;
    context_identifier : identifier;
    context_type       : text;
  INVERSE
    representations_in_context : SET [1:?] OF representation FOR
                                    context_of_items;
END_ENTITY; -- representation_context

ENTITY representation_item;
    name : label;
```

```
    WHERE
      wr1: SIZEOF(using_representations(SELF)) > 0;
  END_ENTITY; -- representation_item

  ENTITY representation_map;
      mapping_origin        : representation_item;
      mapped_representation : representation;
    INVERSE
      map_usage : SET [1:?] OF mapped_item FOR mapping_source;
    WHERE
      wr1: item_in_context(SELF.mapping_origin,SELF.mapped_representation.
                context_of_items);
  END_ENTITY; -- representation_map

  ENTITY reserved_space
    SUBTYPE OF (shape_aspect);
    WHERE
      wr1: 'PLANT_SPATIAL_CONFIGURATION.PLANT_ITEM_DEFINITION' IN TYPEOF(
                SELF.of_shape.definition);
  END_ENTITY; -- reserved_space

  ENTITY revolved_area_solid
    SUBTYPE OF (swept_area_solid);
      axis  : axis1_placement;
      angle : plane_angle_measure;
    DERIVE
      axis_line : line := line(axis.location,vector(axis.z,1));
  END_ENTITY; -- revolved_area_solid

  ENTITY revolved_face_solid
    SUBTYPE OF (swept_face_solid);
      axis  : axis1_placement;
      angle : plane_angle_measure;
    DERIVE
      axis_line : line := line(axis.location,vector(axis.z,1));
  END_ENTITY; -- revolved_face_solid

  ENTITY right_circular_cone
    SUBTYPE OF (geometric_representation_item);
      position   : axis1_placement;
      height     : positive_length_measure;
      radius     : length_measure;
      semi_angle : plane_angle_measure;
    WHERE
      wr1: radius >= 0;
  END_ENTITY; -- right_circular_cone

  ENTITY right_circular_cylinder
    SUBTYPE OF (geometric_representation_item);
      position : axis1_placement;
      height   : positive_length_measure;
      radius   : positive_length_measure;
  END_ENTITY; -- right_circular_cylinder

  ENTITY seam_curve
    SUBTYPE OF (surface_curve);
    WHERE
      wr1: SIZEOF(SELF\surface_curve.associated_geometry) = 2;
      wr2: associated_surface(SELF\surface_curve.associated_geometry[1]) =
```

```
            associated_surface(SELF\surface_curve.associated_geometry[2]);
    wr3: 'PLANT_SPATIAL_CONFIGURATION.PCURVE' IN TYPEOF(SELF\
           surface_curve.associated_geometry[1]);
    wr4: 'PLANT_SPATIAL_CONFIGURATION.PCURVE' IN TYPEOF(SELF\
           surface_curve.associated_geometry[2]);
END_ENTITY; -- seam_curve

ENTITY shape_aspect;
    name                 : label;
    description          : text;
    of_shape             : product_definition_shape;
    product_definitional : LOGICAL;
END_ENTITY; -- shape_aspect

ENTITY shape_aspect_deriving_relationship
  SUBTYPE OF (shape_aspect_relationship);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.DERIVED_SHAPE_ASPECT' IN TYPEOF(
           SELF\shape_aspect_relationship.relating_shape_aspect);
END_ENTITY; -- shape_aspect_deriving_relationship

ENTITY shape_aspect_relationship;
    name                  : label;
    description           : text;
    relating_shape_aspect : shape_aspect;
    related_shape_aspect  : shape_aspect;
END_ENTITY; -- shape_aspect_relationship

ENTITY shape_definition_representation
  SUBTYPE OF (property_definition_representation);
  WHERE
    wr1: ('PLANT_SPATIAL_CONFIGURATION.SHAPE_DEFINITION' IN TYPEOF(SELF.
           definition.definition)) OR (
           'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_SHAPE' IN
           TYPEOF(SELF.definition));
    wr2: 'PLANT_SPATIAL_CONFIGURATION.SHAPE_REPRESENTATION' IN TYPEOF(
           SELF.used_representation);
END_ENTITY; -- shape_definition_representation

ENTITY shape_dimension_representation
  SUBTYPE OF (shape_representation);
  WHERE
    wr1: SIZEOF(QUERY ( temp <* SELF.items | ((NOT
           'PLANT_SPATIAL_CONFIGURATION.MEASURE_REPRESENTATION_ITEM')
           IN TYPEOF(temp)) )) = 0;
    wr2: SIZEOF(SELF.items) <= 2;
    wr3: SIZEOF(QUERY ( pos_mri <* QUERY ( real_mri <* SELF.items | (
           'REAL' IN TYPEOF(real_mri\measure_with_unit.
           value_component)) )
            | (NOT (pos_mri\measure_with_unit.value_component > 0)) ))
           = 0;
END_ENTITY; -- shape_dimension_representation

ENTITY shape_representation
  SUBTYPE OF (representation);
END_ENTITY; -- shape_representation

ENTITY si_unit
  SUBTYPE OF (named_unit);
```

```
    prefix : OPTIONAL si_prefix;
    name   : si_unit_name;
  DERIVE
    SELF\named_unit.dimensions : dimensional_exponents :=
                                 dimensions_for_si_unit(SELF.name);
END_ENTITY; -- si_unit

ENTITY site
  SUBTYPE OF (characterized_object, property_definition);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(SELF\
         property_definition.definition\product_definition.formation.
         of_product);
END_ENTITY; -- site

ENTITY site_building
  SUBTYPE OF (property_definition);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.SITE' IN TYPEOF(SELF.definition);
    wr2: SIZEOF(USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION')) > 2;
    wr3: coordinated_representation_names_and_item_types(SELF);
END_ENTITY; -- site_building

ENTITY site_feature
  SUBTYPE OF (property_definition);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.SITE' IN TYPEOF(SELF.definition);
    wr2: SIZEOF(USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION')) = 3;
    wr3: coordinated_representation_names_and_item_types(SELF);
END_ENTITY; -- site_feature

ENTITY site_representation
  SUBTYPE OF (shape_representation);
  WHERE
    wr1: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.USED_REPRESENTATION') |
         (NOT ('PLANT_SPATIAL_CONFIGURATION.SITE' IN TYPEOF(pdr.
         definition.definition))) )) = 0;
    wr2: ((SIZEOF(QUERY ( item <* SELF\representation.items | (NOT (
         'PLANT_SPATIAL_CONFIGURATION.CONNECTED_FACE_SET' IN TYPEOF(
         item))) )) = 0) XOR (SIZEOF(QUERY ( item <* SELF\
         representation.items | (NOT (
         'PLANT_SPATIAL_CONFIGURATION.GEOMETRIC_CURVE_SET' IN TYPEOF(
         item))) )) = 0));
    wr3: SIZEOF(QUERY ( i <* SELF\representation.items | (NOT ((NOT (
         'PLANT_SPATIAL_CONFIGURATION.CONNECTED_FACE_SET' IN
         TYPEOF(i)))
         OR ((rep_item_set_has_analytic(connected_face_set_class_set(
         i)) = FALSE) AND (rep_item_set_has_free_form(
         connected_face_set_class_set(i)) = FALSE)))) )) = 0;
    wr4: SIZEOF(QUERY ( i <* SELF\representation.items | (NOT ((NOT (
         'PLANT_SPATIAL_CONFIGURATION.CONNECTED_FACE_SET' IN
         TYPEOF(i)))
         OR (SIZEOF(QUERY ( f <* i\connected_face_set.cfs_faces | (
         NOT ((face_edge_count(f) = 3) OR (face_edge_count(f) = 4)))
         )) = 0))) )) = 0;
```

```
    wr5: SIZEOF(QUERY ( i <* SELF\representation.items | (NOT ((NOT (
              'PLANT_SPATIAL_CONFIGURATION.GEOMETRIC_CURVE_SET' IN TYPEOF(
              i))) OR (SIZEOF(QUERY ( e <* i\geometric_set.elements | ((
              'PLANT_SPATIAL_CONFIGURATION.CURVE' IN TYPEOF(e)) AND (NOT (
              'PLANT_SPATIAL_CONFIGURATION.POLYLINE' IN TYPEOF(e)))) )) =
              0))) )) = 0;
END_ENTITY; -- site_representation

ENTITY solid_angle_measure_with_unit
  SUBTYPE OF (measure_with_unit);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.SOLID_ANGLE_UNIT' IN TYPEOF(SELF\
            measure_with_unit.unit_component);
END_ENTITY; -- solid_angle_measure_with_unit

ENTITY solid_angle_unit
  SUBTYPE OF (named_unit);
  WHERE
    wr1: (SELF\named_unit.dimensions.length_exponent = 0) AND (SELF\
            named_unit.dimensions.mass_exponent = 0) AND (SELF\
            named_unit.dimensions.time_exponent = 0) AND (SELF\
            named_unit.dimensions.electric_current_exponent = 0) AND (
            SELF\named_unit.dimensions.
            thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
            .dimensions.amount_of_substance_exponent = 0) AND (SELF\
            named_unit.dimensions.luminous_intensity_exponent = 0);
END_ENTITY; -- solid_angle_unit

ENTITY solid_model
  SUPERTYPE OF (ONEOF (csg_solid,manifold_solid_brep,swept_face_solid,
      swept_area_solid))
  SUBTYPE OF (geometric_representation_item);
END_ENTITY; -- solid_model

ENTITY spacer_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- spacer_fitting_classification

ENTITY specialty_item_classification
  SUBTYPE OF (group);
END_ENTITY; -- specialty_item_classification

ENTITY sphere
  SUBTYPE OF (geometric_representation_item);
    radius : positive_length_measure;
    centre : point;
END_ENTITY; -- sphere

ENTITY spherical_surface
  SUBTYPE OF (elementary_surface);
    radius : positive_length_measure;
END_ENTITY; -- spherical_surface

ENTITY stream_design_case
  SUBTYPE OF (property_definition, characterized_object);
  WHERE
    wr1: SIZEOF(USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
            'PROPERTY_DEFINITION.DEFINITION')) >= 1;
```

```
    wr2: SIZEOF(USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION')) >= 2;
    wr3: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION') | (pdr.
         used_representation.name = 'flow_rate') )) = 1;
    wr4: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION') | (pdr.
         used_representation.name = 'pressure') )) = 1;
END_ENTITY; -- stream_design_case

ENTITY stream_phase
  SUBTYPE OF (property_definition);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.STREAM_DESIGN_CASE' IN TYPEOF(SELF
         .definition);
    wr2: SIZEOF(USEDIN(SELF,'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION')) >= 3;
    wr3: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION') | (pdr.
         used_representation.name = 'constituent_mole_fraction') )) =
          1;
    wr4: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION') | (pdr.
         used_representation.name = 'constituents') )) = 1;
    wr5: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.' +
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION') | (pdr.
         used_representation.name = 'temperature') )) = 1;
END_ENTITY; -- stream_phase

ENTITY structural_load_connector_classification
  SUBTYPE OF (group);
END_ENTITY; -- structural_load_connector_classification

ENTITY structural_system
  SUBTYPE OF (product_definition);
  WHERE
    wr1: SIZEOF(QUERY ( pdr <* USEDIN(SELF,
         'PLANT_SPATIAL_CONFIGURATION.PRODUCT_DEFINITION_' +
         'RELATIONSHIP.RELATED_PRODUCT_DEFINITION')
          | (('PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(pdr.
         relating_product_definition.formation.of_product)) AND (pdr.
         relating_product_definition.frame_of_reference.name =
         'functional_occurrence')) )) = 1;
END_ENTITY; -- structural_system

ENTITY support_constraint_representation
  SUBTYPE OF (representation);
  WHERE
    wr1: SIZEOF(SELF.items) >= 3;
    wr2: SIZEOF(QUERY ( it <* SELF.items | ((
         'PLANT_SPATIAL_CONFIGURATION.MEASURE_REPRESENTATION_ITEM' IN
          TYPEOF(it)) AND (it.name IN ['negative_x','positive_x',
         'negative_y','positive_y','negative_z','positive_z',
         'negative_x_rotation','positive_x_rotation',
```

```
            'negative_y_rotation','positive_y_rotation',
            'negative_z_rotation','positive_z_rotation'])) )) = 1;
    wr3: SIZEOF(QUERY ( it <* SELF.items | (
            'PLANT_SPATIAL_CONFIGURATION.RATIO_MEASURE_WITH_UNIT' IN
            TYPEOF(it)) )) = 1;
    wr4: SIZEOF(QUERY ( it <* SELF.items | (
         'PLANT_SPATIAL_CONFIGURATION.DESCRIPTIVE_REPRESENTATION_ITEM'
            IN TYPEOF(it)) )) = 1;
END_ENTITY; -- support_constraint_representation

ENTITY surface
  SUPERTYPE OF (ONEOF (elementary_surface,swept_surface,bounded_surface,
      offset_surface,surface_replica))
  SUBTYPE OF (geometric_representation_item);
END_ENTITY; -- surface

ENTITY surface_curve
  SUPERTYPE OF (ONEOF (intersection_curve,seam_curve) ANDOR
      bounded_surface_curve)
  SUBTYPE OF (curve);
    curve_3d              : curve;
    associated_geometry   : LIST [1:2] OF pcurve_or_surface;
    master_representation : preferred_surface_curve_representation;
  DERIVE
    basis_surface : SET [1:2] OF surface := get_basis_surface(SELF);
  WHERE
    wr1: curve_3d.dim = 3;
    wr2: ('PLANT_SPATIAL_CONFIGURATION.PCURVE' IN TYPEOF(
            associated_geometry[1])) OR (master_representation <>
            pcurve_s1);
    wr3: ('PLANT_SPATIAL_CONFIGURATION.PCURVE' IN TYPEOF(
            associated_geometry[2])) OR (master_representation <>
            pcurve_s2);
    wr4: NOT ('PLANT_SPATIAL_CONFIGURATION.PCURVE' IN TYPEOF(curve_3d));
END_ENTITY; -- surface_curve

ENTITY surface_of_linear_extrusion
  SUBTYPE OF (swept_surface);
    extrusion_axis : vector;
END_ENTITY; -- surface_of_linear_extrusion

ENTITY surface_of_revolution
  SUBTYPE OF (swept_surface);
    axis_position : axis1_placement;
  DERIVE
    axis_line : line := line(axis_position.location,vector(axis_position
                  .z,1));
END_ENTITY; -- surface_of_revolution

ENTITY surface_patch;
    parent_surface : bounded_surface;
    u_transition   : transition_code;
    v_transition   : transition_code;
    u_sense        : BOOLEAN;
    v_sense        : BOOLEAN;
  INVERSE
    using_surfaces : BAG [1:?] OF rectangular_composite_surface FOR
                        segments;
  WHERE
```

```
      wr1: NOT ('PLANT_SPATIAL_CONFIGURATION.CURVE_BOUNDED_SURFACE' IN
                TYPEOF(parent_surface));
  END_ENTITY; -- surface_patch

  ENTITY surface_replica
    SUBTYPE OF (surface);
      parent_surface : surface;
      transformation : cartesian_transformation_operator_3d;
    WHERE
      wr1: acyclic_surface_replica(SELF,parent_surface);
  END_ENTITY; -- surface_replica

  ENTITY swage_fitting_classification
    SUBTYPE OF (group);
  END_ENTITY; -- swage_fitting_classification

  ENTITY swept_area_solid
    SUPERTYPE OF (ONEOF (revolved_area_solid,extruded_area_solid))
    SUBTYPE OF (solid_model);
      swept_area : curve_bounded_surface;
    WHERE
      wr1: 'PLANT_SPATIAL_CONFIGURATION.PLANE' IN TYPEOF(swept_area.
                basis_surface);
  END_ENTITY; -- swept_area_solid

  ENTITY swept_face_solid
    SUPERTYPE OF (ONEOF (extruded_face_solid, revolved_face_solid))
    SUBTYPE OF (solid_model);
      swept_face : face_surface;
    WHERE
      wr1: 'PLANT_SPATIAL_CONFIGURATION.PLANE' IN TYPEOF(swept_face.
                face_geometry);
  END_ENTITY; -- swept_face_solid

  ENTITY swept_surface
    SUPERTYPE OF (ONEOF (surface_of_linear_extrusion,
             surface_of_revolution))
    SUBTYPE OF (surface);
      swept_curve : curve;
  END_ENTITY; -- swept_surface

  ENTITY system_space
    SUBTYPE OF (product_definition_shape);
    WHERE
      wr1: SIZEOF(TYPEOF(SELF.definition) * [
                'PLANT_SPATIAL_CONFIGURATION.ELECTRICAL_SYSTEM',
                'PLANT_SPATIAL_CONFIGURATION.HVAC_SYSTEM',
                'PLANT_SPATIAL_CONFIGURATION.' +
                'INSTRUMENTATION_AND_CONTROL_SYSTEM',
                'PLANT_SPATIAL_CONFIGURATION.PIPING_SYSTEM',
                'PLANT_SPATIAL_CONFIGURATION.STRUCTURAL_SYSTEM']) = 1;
  END_ENTITY; -- system_space

  ENTITY tee_fitting_classification
    SUBTYPE OF (group);
  END_ENTITY; -- tee_fitting_classification

  ENTITY thermodynamic_temperature_measure_with_unit
    SUBTYPE OF (measure_with_unit);
```

```
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.THERMODYNAMIC_TEMPERATURE_UNIT' IN
              TYPEOF(SELF\measure_with_unit.unit_component);
END_ENTITY; -- thermodynamic_temperature_measure_with_unit

ENTITY thermodynamic_temperature_unit
  SUBTYPE OF (named_unit);
  WHERE
    wr1: (SELF\named_unit.dimensions.length_exponent = 0) AND (SELF\
              named_unit.dimensions.mass_exponent = 0) AND (SELF\
              named_unit.dimensions.time_exponent = 0) AND (SELF\
              named_unit.dimensions.electric_current_exponent = 0) AND (
              SELF\named_unit.dimensions.
              thermodynamic_temperature_exponent = 1) AND (SELF\named_unit
              .dimensions.amount_of_substance_exponent = 0) AND (SELF\
              named_unit.dimensions.luminous_intensity_exponent = 0);
END_ENTITY; -- thermodynamic_temperature_unit

ENTITY time_measure_with_unit
  SUBTYPE OF (measure_with_unit);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.TIME_UNIT' IN TYPEOF(SELF\
              measure_with_unit.unit_component);
END_ENTITY; -- time_measure_with_unit

ENTITY time_unit
  SUBTYPE OF (named_unit);
  WHERE
    wr1: (SELF\named_unit.dimensions.length_exponent = 0) AND (SELF\
              named_unit.dimensions.mass_exponent = 0) AND (SELF\
              named_unit.dimensions.time_exponent = 1) AND (SELF\
              named_unit.dimensions.electric_current_exponent = 0) AND (
              SELF\named_unit.dimensions.
              thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
              .dimensions.amount_of_substance_exponent = 0) AND (SELF\
              named_unit.dimensions.luminous_intensity_exponent = 0);
END_ENTITY; -- time_unit

ENTITY topological_representation_item
  SUPERTYPE OF (ONEOF (vertex,edge,face_bound,face,connected_face_set,
      loop ANDOR path))
  SUBTYPE OF (representation_item);
END_ENTITY; -- topological_representation_item

ENTITY toroidal_surface
  SUBTYPE OF (elementary_surface);
    major_radius : positive_length_measure;
    minor_radius : positive_length_measure;
END_ENTITY; -- toroidal_surface

ENTITY torus
  SUBTYPE OF (geometric_representation_item);
    position     : axis1_placement;
    major_radius : positive_length_measure;
    minor_radius : positive_length_measure;
  WHERE
    wr1: major_radius > minor_radius;
END_ENTITY; -- torus
```

```
ENTITY trimmed_curve
  SUBTYPE OF (bounded_curve);
    basis_curve           : curve;
    trim_1                : SET [1:2] OF trimming_select;
    trim_2                : SET [1:2] OF trimming_select;
    sense_agreement       : BOOLEAN;
    master_representation : trimming_preference;
  WHERE
    wr1: (HIINDEX(trim_1) = 1) XOR (TYPEOF(trim_1[1]) <>
                TYPEOF(trim_1[2]));
    wr2: (HIINDEX(trim_2) = 1) XOR (TYPEOF(trim_2[1]) <>
                TYPEOF(trim_2[2]));
END_ENTITY; -- trimmed_curve

ENTITY truncated_pyramid
  SUBTYPE OF (boolean_result);
END_ENTITY; -- truncated_pyramid

ENTITY uniform_curve
  SUBTYPE OF (b_spline_curve);
END_ENTITY; -- uniform_curve

ENTITY uniform_surface
  SUBTYPE OF (b_spline_surface);
END_ENTITY; -- uniform_surface

ENTITY union_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- union_fitting_classification

ENTITY valve_classification
  SUBTYPE OF (group);
END_ENTITY; -- valve_classification

ENTITY vector
  SUBTYPE OF (geometric_representation_item);
    orientation : direction;
    magnitude   : length_measure;
  WHERE
    wr1: magnitude >= 0;
END_ENTITY; -- vector

ENTITY versioned_action_request;
    id          : identifier;
    version     : label;
    purpose     : text;
    description : text;
END_ENTITY; -- versioned_action_request

ENTITY vertex
  SUBTYPE OF (topological_representation_item);
END_ENTITY; -- vertex

ENTITY volume_measure_with_unit
  SUBTYPE OF (measure_with_unit);
  WHERE
    wr1: 'PLANT_SPATIAL_CONFIGURATION.VOLUME_UNIT' IN TYPEOF(SELF\
            measure_with_unit.unit_component);
END_ENTITY; -- volume_measure_with_unit
```

```
ENTITY volume_unit
  SUBTYPE OF (named_unit);
  WHERE
    wr1: (SELF\named_unit.dimensions.length_exponent = 3) AND (SELF\
            named_unit.dimensions.mass_exponent = 0) AND (SELF\
            named_unit.dimensions.time_exponent = 0) AND (SELF\
            named_unit.dimensions.electric_current_exponent = 0) AND (
            SELF\named_unit.dimensions.
            thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
            .dimensions.amount_of_substance_exponent = 0) AND (SELF\
            named_unit.dimensions.luminous_intensity_exponent = 0);
END_ENTITY; -- volume_unit

ENTITY wye_fitting_classification
  SUBTYPE OF (group);
END_ENTITY; -- wye_fitting_classification

RULE approval_requires_approval_date_time FOR (approval_date_time,
      approval);

WHERE
  wr1: SIZEOF(QUERY ( app <* approval | (NOT (SIZEOF(QUERY ( adt <*
          approval_date_time | (app :=: adt.dated_approval) )) = 1)) ))
          = 0;

END_RULE; -- approval_requires_approval_date_time

RULE approval_requires_approval_person_organization FOR (
          approval_person_organization, approval);

WHERE
  wr1: SIZEOF(QUERY ( app <* approval | (NOT (SIZEOF(QUERY ( apo <*
          approval_person_organization | (app :=: apo.
          authorized_approval) )) = 1)) )) = 0;

END_RULE; -- approval_requires_approval_person_organization

RULE catalogue_requires_vendor FOR (document_usage_constraint,
          plant_spatial_configuration_organization_assignment,
          plant_spatial_configuration_person_assignment);

WHERE
  wr1: SIZEOF(QUERY ( duc <* document_usage_constraint | ((duc.
          subject_element = 'item_definition') AND (NOT ((SIZEOF(
          QUERY ( pscoa <*
          plant_spatial_configuration_organization_assignment | ((duc.
          source IN pscoa.items) AND (pscoa.role.name = 'vendor')) )) =
          1) XOR (SIZEOF(QUERY ( pscpa <*
          plant_spatial_configuration_person_assignment | ((duc.source
          IN pscpa.items) AND (pscpa.role.name = 'vendor')) )) = 1))))
          )) = 0;

END_RULE; -- catalogue_requires_vendor

RULE change_action_requires_date FOR (change_action,
          plant_spatial_configuration_date_assignment);

WHERE
  wr1: SIZEOF(QUERY ( ca <* change_action | (NOT (SIZEOF(
```

```
            QUERY ( pscda <* plant_spatial_configuration_date_assignment
              | (ca IN pscda.items) )) = 1)) )) = 0;

END_RULE; -- change_action_requires_date

RULE change_item_requires_creation_date FOR (
            plant_spatial_configuration_change_assignment,
            plant_spatial_configuration_date_assignment);

WHERE
  wr1: SIZEOF(QUERY ( pscca <*
            plant_spatial_configuration_change_assignment | (NOT (SIZEOF(
            QUERY ( ch_it <* pscca.items | (NOT (SIZEOF(QUERY ( pscda <*
            plant_spatial_configuration_date_assignment | ((NOT (ch_it IN
            pscda.items)) OR (pscda.role.name = 'creation_date')) )) = 1))
            )) = 0)) )) = 0;

END_RULE; -- change_item_requires_creation_date

RULE change_item_requires_id FOR (
            plant_spatial_configuration_change_assignment,
            change_item_id_assignment);

WHERE
  wr1: SIZEOF(QUERY ( pscca <*
            plant_spatial_configuration_change_assignment | (NOT (SIZEOF(
            QUERY ( ch_it <* pscca.items | (NOT (SIZEOF(QUERY ( ciia <*
            change_item_id_assignment | (ch_it IN ciia.items) )) = 1)) ))
            = 0)) )) = 0;

END_RULE; -- change_item_requires_id

RULE change_life_cycle_stage_usage_requires_approval FOR (
            versioned_action_request,
            plant_spatial_configuration_approval_assignment);

WHERE
  wr1: SIZEOF(QUERY ( vareq <* versioned_action_request | ((NOT SIZEOF(
            QUERY ( pscaa <*
            plant_spatial_configuration_approval_assignment | (vareq IN
            pscaa.items) ))) = 1) )) = 0;

END_RULE; -- change_life_cycle_stage_usage_requires_approval

RULE change_life_cycle_stage_usage_requires_stage FOR (
            versioned_action_request, action_request_status);

WHERE
  wr1: SIZEOF(QUERY ( vareq <* versioned_action_request | ((NOT SIZEOF(
            QUERY ( ars <* action_request_status | (vareq :=: ars.
            assigned_request) ))) = 1) )) = 0;

END_RULE; -- change_life_cycle_stage_usage_requires_stage

RULE compatible_dimension FOR (cartesian_point, direction,
            representation_context, geometric_representation_context);

WHERE
  wr1: SIZEOF(QUERY ( x <* cartesian_point | (SIZEOF(QUERY ( y <*
```

```
            geometric_representation_context | (item_in_context(x,y) AND (
            HIINDEX(x.coordinates) <> y.coordinate_space_dimension)) )) >
            0) )) = 0;
  wr2: SIZEOF(QUERY ( x <* direction | (SIZEOF(QUERY ( y <*
            geometric_representation_context | (item_in_context(x,y) AND (
            HIINDEX(x.direction_ratios) <> y.coordinate_space_dimension))
            )) > 0) )) = 0;

END_RULE; -- compatible_dimension

RULE versioned_action_request_requires_change_action FOR (change_action,
            versioned_action_request);

WHERE
  wr1: SIZEOF(QUERY ( vareq <* versioned_action_request | (NOT (SIZEOF(
            QUERY ( ca <* change_action | (vareq IN ca.directive.requests)
            )) = 1)) )) = 0;

END_RULE; -- versioned_action_request_requires_change_action

FUNCTION acyclic_curve_replica(
              rep: curve_replica;
              parent: curve
    ): BOOLEAN;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.CURVE_REPLICA' IN TYPEOF(parent))
       THEN
    RETURN(TRUE);
  END_IF;
  IF parent :=: rep THEN
    RETURN(FALSE);
  ELSE
    RETURN(acyclic_curve_replica(rep,parent\curve_replica.parent_curve));
  END_IF;

END_FUNCTION; -- acyclic_curve_replica

FUNCTION acyclic_mapped_representation(
              parent_set: SET OF representation;
              children_set: SET OF representation_item
    ): BOOLEAN;

  LOCAL
    i : INTEGER;
    x : SET OF representation_item;
    y : SET OF representation_item;
  END_LOCAL;
  x := QUERY ( z <* children_set | (
      'PLANT_SPATIAL_CONFIGURATION.MAPPED_ITEM' IN TYPEOF(z)) );
  IF SIZEOF(x) > 0 THEN
    REPEAT i := 1 TO HIINDEX(x) BY 1;
      IF x[i]\mapped_item.mapping_source.mapped_representation IN
          parent_set THEN
        RETURN(FALSE);
      END_IF;
      IF NOT acyclic_mapped_representation(parent_set + x[i]\mapped_item
          .mapping_source.mapped_representation,x[i]\mapped_item.
          mapping_source.mapped_representation.items) THEN
        RETURN(FALSE);
      END_IF;
```

```
      END_REPEAT;
    END_IF;
    x := children_set - x;
    IF SIZEOF(x) > 0 THEN
      REPEAT i := 1 TO HIINDEX(x) BY 1;
        y := QUERY ( z <* bag_to_set(USEDIN(x[i],'')) | (
             'PLANT_SPATIAL_CONFIGURATION.REPRESENTATION_ITEM' IN
              TYPEOF(z)) );
        IF NOT acyclic_mapped_representation(parent_set,y) THEN
          RETURN(FALSE);
        END_IF;
      END_REPEAT;
    END_IF;
    RETURN(TRUE);

  END_FUNCTION; -- acyclic_mapped_representation

  FUNCTION acyclic_point_replica(
                rep: point_replica;
                parent: point
      ): BOOLEAN;
    IF NOT ('PLANT_SPATIAL_CONFIGURATION.POINT_REPLICA' IN TYPEOF(parent))
         THEN
      RETURN(TRUE);
    END_IF;
    IF parent :=: rep THEN
      RETURN(FALSE);
    ELSE
      RETURN(acyclic_point_replica(rep,parent\point_replica.parent_pt));
    END_IF;

  END_FUNCTION; -- acyclic_point_replica

  FUNCTION acyclic_product_category_relationship(
                relation: product_category_relationship;
                children: SET OF product_category
      ): LOGICAL;

    LOCAL
      i              : INTEGER;
      x              : SET OF product_category_relationship;
      local_children : SET OF product_category;
    END_LOCAL;
    REPEAT i := 1 TO HIINDEX(children) BY 1;
      IF relation.category :=: children[i] THEN
        RETURN(FALSE);
      END_IF;
    END_REPEAT;
    x := bag_to_set(USEDIN(relation.category,
        'PLANT_SPATIAL_CONFIGURATION.' +
        'PRODUCT_CATEGORY_RELATIONSHIP.SUB_CATEGORY'));
    local_children := children + relation.category;
    IF SIZEOF(x) > 0 THEN
      REPEAT i := 1 TO HIINDEX(x) BY 1;
        IF NOT acyclic_product_category_relationship(x[i],local_children)
             THEN
          RETURN(FALSE);
        END_IF;
      END_REPEAT;
```

```
  END_IF;
  RETURN(TRUE);

END_FUNCTION; -- acyclic_product_category_relationship

FUNCTION acyclic_product_definition_relationship(
             relation: product_definition_relationship;
             relatives: SET OF product_definition;
             specific_relation: STRING
    ): LOGICAL;

  LOCAL
    i               : INTEGER;
    x               : SET OF product_definition_relationship;
    local_relatives : SET OF product_definition;
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(relatives) BY 1;
    IF relation.relating_product_definition :=: relatives[i] THEN
      RETURN(FALSE);
    END_IF;
  END_REPEAT;
  x := bag_to_set(USEDIN(relation.relating_product_definition,
      specific_relation));
  local_relatives := relatives + relation.relating_product_definition;
  IF SIZEOF(x) > 0 THEN
    REPEAT i := 1 TO HIINDEX(x) BY 1;
      IF NOT acyclic_product_definition_relationship(x[i],
          local_relatives,specific_relation) THEN
        RETURN(FALSE);
      END_IF;
    END_REPEAT;
  END_IF;
  RETURN(TRUE);

END_FUNCTION; -- acyclic_product_definition_relationship

FUNCTION acyclic_set_replica(
             rep: geometric_set_replica;
             parent: geometric_set
    ): BOOLEAN;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.GEOMETRIC_SET_REPLICA' IN TYPEOF(
      parent)) THEN
    RETURN(TRUE);
  END_IF;
  IF parent :=: rep THEN
    RETURN(FALSE);
  ELSE
    RETURN(acyclic_set_replica(rep,parent\geometric_set_replica.
        parent_set));
  END_IF;

END_FUNCTION; -- acyclic_set_replica

FUNCTION acyclic_surface_replica(
             rep: surface_replica;
             parent: surface
    ): BOOLEAN;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.SURFACE_REPLICA' IN
           TYPEOF(parent))
```

```
        THEN
      RETURN(TRUE);
    END_IF;
    IF parent :=: rep THEN
      RETURN(FALSE);
    ELSE
      RETURN(acyclic_surface_replica(rep,parent\surface_replica.
          parent_surface));
    END_IF;

  END_FUNCTION; -- acyclic_surface_replica

  FUNCTION associated_surface(
              arg: pcurve_or_surface
      ): surface;

    LOCAL
      surf : surface;
    END_LOCAL;
    IF 'PLANT_SPATIAL_CONFIGURATION.PCURVE' IN TYPEOF(arg) THEN
      surf := arg.basis_surface;
    ELSE
      surf := arg;
    END_IF;
    RETURN(surf);

  END_FUNCTION; -- associated_surface

  FUNCTION bag_to_set(
              the_bag: BAG OF GENERIC:intype
      ): SET OF GENERIC:intype;

    LOCAL
      i       : INTEGER;
      the_set : SET OF GENERIC:intype := [];
    END_LOCAL;
    IF SIZEOF(the_bag) > 0 THEN
      REPEAT i := 1 TO HIINDEX(the_bag) BY 1;
        the_set := the_set + the_bag[i];
      END_REPEAT;
    END_IF;
    RETURN(the_set);

  END_FUNCTION; -- bag_to_set

  FUNCTION base_axis(
              dim: INTEGER;
              axis1, axis2, axis3: direction
      ): LIST [2:3] OF direction;

    LOCAL
      u      : LIST [2:3] OF direction;
      vec    : direction;
      factor : REAL;
    END_LOCAL;
    IF dim = 3 THEN
      u[3] := NVL(normalise(axis3),direction([0,0,1]));
      u[1] := first_proj_axis(u[3],axis1);
      u[2] := second_proj_axis(u[3],u[1],axis2);
```

```
  ELSE
    u[3] := ?;
    IF EXISTS(axis1) THEN
      u[1] := normalise(axis1);
      u[2] := orthogonal_complement(u[1]);
      IF EXISTS(axis2) THEN
        factor := dot_product(axis2,u[2]);
        IF factor < 0 THEN
          u[2].direction_ratios[1] := -u[2].direction_ratios[1];
          u[2].direction_ratios[2] := -u[2].direction_ratios[2];
        END_IF;
      END_IF;
    ELSE
      IF EXISTS(axis2) THEN
        u[2] := normalise(axis2);
        u[1] := orthogonal_complement(u[2]);
        u[1].direction_ratios[1] := -u[1].direction_ratios[1];
        u[1].direction_ratios[2] := -u[1].direction_ratios[2];
      ELSE
        u[1].direction_ratios[1] := 1;
        u[1].direction_ratios[2] := 0;
        u[2].direction_ratios[1] := 0;
        u[2].direction_ratios[2] := 1;
      END_IF;
    END_IF;
  END_IF;
  RETURN(u);

END_FUNCTION; -- base_axis

FUNCTION boolean_choose(
              b: BOOLEAN;
              choice1, choice2: GENERIC:item
    ): GENERIC:item;
  IF b THEN
    RETURN(choice1);
  ELSE
    RETURN(choice2);
  END_IF;

END_FUNCTION; -- boolean_choose

FUNCTION boolean_result_class_set(
              rep_item: representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.BOOLEAN_RESULT' IN TYPEOF(
      rep_item)) THEN
    RETURN(return_set);
  END_IF;
  return_set := return_set + boolean_result_class_set(rep_item\
      boolean_result.first_operand) + boolean_result_class_set(rep_item\
      boolean_result.second_operand) + csg_primitive_class_set(rep_item\
      boolean_result.first_operand) + csg_primitive_class_set(rep_item\
      boolean_result.second_operand) + csg_solid_class_set(rep_item\
      boolean_result.first_operand) + csg_solid_class_set(rep_item\
```

```
      boolean_result.second_operand) + manifold_solid_brep_class_set(
      rep_item\boolean_result.first_operand) +
      manifold_solid_brep_class_set(rep_item\boolean_result.
      second_operand) + swept_area_solid_class_set(rep_item\
      boolean_result.first_operand) + swept_area_solid_class_set(rep_item
      \boolean_result.second_operand);
  RETURN(return_set);

END_FUNCTION; -- boolean_result_class_set

FUNCTION build_2axes(
            ref_direction: direction
    ): LIST [2:2] OF direction;

  LOCAL
    u : LIST [2:2] OF direction;
  END_LOCAL;
  u[1] := NVL(normalise(ref_direction),direction([1,0]));
  u[2] := orthogonal_complement(u[1]);
  RETURN(u);

END_FUNCTION; -- build_2axes

FUNCTION build_axes(
            axis, ref_direction: direction
    ): LIST [3:3] OF direction;

  LOCAL
    u : LIST [3:3] OF direction;
  END_LOCAL;
  u[3] := NVL(normalise(axis),direction([0,0,1]));
  u[1] := first_proj_axis(u[3],ref_direction);
  u[2] := normalise(cross_product(u[3],u[1])).orientation;
  RETURN(u);

END_FUNCTION; -- build_axes

FUNCTION build_transformed_set(
            tr: cartesian_transformation_operator;
            gset: geometric_set
    ): SET [0:?] OF geometric_set_select;

  LOCAL
    trcurve : curve;
    s       : SET [1:?] OF geometric_set_select := gset.elements;
    trpoint : point;
    trset   : SET [0:?] OF geometric_set_select := [];
    trsurf  : surface;
  END_LOCAL;
  REPEAT j := 1 TO SIZEOF(s) BY 1;
    IF 'PLANT_SPATIAL_CONFIGURATION.CURVE' IN TYPEOF(s[j]) THEN
      trset := trset + curve_replica(s[j],tr);
    ELSE
      IF 'PLANT_SPATIAL_CONFIGURATION.POINT' IN TYPEOF(s[j]) THEN
        trset := trset + point_replica(s[j],tr);
      ELSE
        IF 'PLANT_SPATIAL_CONFIGURATION.SURFACE' IN TYPEOF(s[j]) THEN
          trset := trset + surface_replica(s[j],tr ||
              cartesian_transformation_operator_3d(?));
```

```
        END_IF;
      END_IF;
    END_IF;
  END_REPEAT;
  RETURN(trset);

END_FUNCTION; -- build_transformed_set

FUNCTION class_rep_item_in_set(
            rep_item_set: SET [1:?] OF representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    i          : INTEGER;
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF SIZEOF(rep_item_set) = 0 THEN
    RETURN(FALSE);
  END_IF;
  REPEAT i := 1 TO HIINDEX(rep_item_set) BY 1;
    return_set := return_set + manifold_solid_brep_class_set(
        rep_item_set[i]) + face_class_set(rep_item_set[i]) +
        connected_face_set_class_set(rep_item_set[i]) + curve_class_set(
        rep_item_set[i]) + surface_class_set(rep_item_set[i]) +
        swept_area_solid_class_set(rep_item_set[i]) +
        boolean_result_class_set(rep_item_set[i]) + csg_solid_class_set(
        rep_item_set[i]) + csg_primitive_class_set(rep_item_set[i]) +
        geometric_set_class_set(rep_item_set[i]);
  END_REPEAT;
  RETURN(return_set);

END_FUNCTION; -- class_rep_item_in_set

FUNCTION conditional_reverse(
            p: BOOLEAN;
            an_item: reversible_topology
    ): reversible_topology;
  IF p THEN
    RETURN(an_item);
  ELSE
    RETURN(topology_reversed(an_item));
  END_IF;

END_FUNCTION; -- conditional_reverse

FUNCTION connected_face_set_class_set(
            rep_item: representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    i          : INTEGER;
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.CONNECTED_FACE_SET' IN TYPEOF(
      rep_item)) THEN
    RETURN(return_set);
  END_IF;
  REPEAT i := 1 TO HIINDEX(rep_item\connected_face_set.cfs_faces) BY 1;
    return_set := return_set + face_class_set(rep_item\
```

```
        connected_face_set.cfs_faces[i]);
    END_REPEAT;
    IF 'PLANT_SPATIAL_CONFIGURATION.ORIENTED_CLOSED_SHELL' IN TYPEOF(
        rep_item) THEN
      return_set := return_set + connected_face_set(rep_item\
        oriented_closed_shell.closed_shell_element);
    END_IF;
    IF 'PLANT_SPATIAL_CONFIGURATION.ORIENTED_OPEN_SHELL' IN TYPEOF(
        rep_item) THEN
      return_set := return_set + connected_face_set(rep_item\
        oriented_open_shell.open_shell_element);
    END_IF;
    RETURN(return_set);

  END_FUNCTION; -- connected_face_set_class_set

  FUNCTION constraints_composite_curve_on_surface(
              c: composite_curve_on_surface
      ): BOOLEAN;

    LOCAL
      n_segments : INTEGER := SIZEOF(c.segments);
    END_LOCAL;
    REPEAT k := 1 TO n_segments BY 1;
      IF (NOT ('PLANT_SPATIAL_CONFIGURATION.PCURVE' IN TYPEOF(c\
          composite_curve.segments[k].parent_curve))) AND (NOT (
          'PLANT_SPATIAL_CONFIGURATION.SURFACE_CURVE' IN TYPEOF(c\
          composite_curve.segments[k].parent_curve))) AND (NOT (
          'PLANT_SPATIAL_CONFIGURATION.COMPOSITE_CURVE_ON_SURFACE' IN
          TYPEOF(c\composite_curve.segments[k].parent_curve))) THEN
        RETURN(FALSE);
      END_IF;
    END_REPEAT;
    RETURN(TRUE);

  END_FUNCTION; -- constraints_composite_curve_on_surface

  FUNCTION constraints_param_b_spline(
              degree, up_knots, up_cp: INTEGER;
              knot_mult: LIST OF INTEGER;
              knots: LIST OF parameter_value
      ): BOOLEAN;

    LOCAL
      k      : INTEGER;
      l      : INTEGER;
      sum    : INTEGER;
      result : BOOLEAN := TRUE;
    END_LOCAL;
    sum := knot_mult[1];
    REPEAT i := 2 TO up_knots BY 1;
      sum := sum + knot_mult[i];
    END_REPEAT;
    IF (degree < 1) OR (up_knots < 2) OR (up_cp < degree) OR (sum <> (
        degree + up_cp + 2)) THEN
      result := FALSE;
      RETURN(result);
    END_IF;
    k := knot_mult[1];
```

```
  IF (k < 1) OR (k > (degree + 1)) THEN
    result := FALSE;
    RETURN(result);
  END_IF;
  REPEAT i := 2 TO up_knots BY 1;
    IF (knot_mult[i] < 1) OR (knots[i] <= knots[i - 1]) THEN
      result := FALSE;
      RETURN(result);
    END_IF;
    k := knot_mult[i];
    IF (i < up_knots) AND (k > degree) THEN
      result := FALSE;
      RETURN(result);
    END_IF;
    IF (i = up_knots) AND (k > (degree + 1)) THEN
      result := FALSE;
      RETURN(result);
    END_IF;
  END_REPEAT;
  RETURN(result);

END_FUNCTION; -- constraints_param_b_spline

FUNCTION constraints_rectangular_composite_surface(
            s: rectangular_composite_surface
    ): BOOLEAN;
  REPEAT i := 1 TO s.n_u BY 1;
    REPEAT j := 1 TO s.n_v BY 1;
      IF NOT (('PLANT_SPATIAL_CONFIGURATION.B_SPLINE_SURFACE' IN TYPEOF(
          s.segments[i][j].parent_surface)) OR (
          'PLANT_SPATIAL_CONFIGURATION.RECTANGULAR_TRIMMED_SURFACE' IN
          TYPEOF(s.segments[i][j].parent_surface))) THEN
        RETURN(FALSE);
      END_IF;
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO s.n_u - 1 BY 1;
    REPEAT j := 1 TO s.n_v BY 1;
      IF s.segments[i][j].u_transition = discontinuous THEN
        RETURN(FALSE);
      END_IF;
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO s.n_u BY 1;
    REPEAT j := 1 TO s.n_v - 1 BY 1;
      IF s.segments[i][j].v_transition = discontinuous THEN
        RETURN(FALSE);
      END_IF;
    END_REPEAT;
  END_REPEAT;
  RETURN(TRUE);

END_FUNCTION; -- constraints_rectangular_composite_surface

FUNCTION coordinated_representation_names_and_item_types(
            input_entity: property_definition
    ): BOOLEAN;
```

```
LOCAL
  type_set          : SET [1:?] OF STRING;
  i                 : INTEGER;
  rep_set           : SET [0:?] OF representation;
  prop_def_rep_set  : SET [0:?] OF property_definition_representation;
  rep               : representation;
END_LOCAL;
prop_def_rep_set := USEDIN(input_entity,'PLANT_SPATIAL_CONFIGURATION.'
    + 'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION');
rep_set := [];
REPEAT i := 1 TO HIINDEX(prop_def_rep_set) BY 1;
  rep_set := rep_set + prop_def_rep_set[i].used_representation;
END_REPEAT;
type_set := TYPEOF(input_entity);
IF 'PLANT_SPATIAL_CONFIGURATION.SITE_PROPERTY_DEFINITION' IN type_set
     THEN
  REPEAT i := 1 TO HIINDEX(rep_set) BY 1;
    IF ((rep_set[i].name = 'site_address') AND (SIZEOF(QUERY ( item <*
        rep_set[i].items | (NOT (('PLANT_SPATIAL_CONFIGURATION.' +
        'DESCRIPTIVE_REPRESENTATION_ITEM') IN TYPEOF(item))) )) > 0))
         THEN
      RETURN(FALSE);
    END_IF;
    IF ((rep_set[i].name = 'site_coordinate') AND (SIZEOF(
        QUERY ( item <* rep_set[i].items | (NOT (SIZEOF([
        'PLANT_SPATIAL_CONFIGURATION.MEASURE_REPRESENTATION_ITEM',
        'PLANT_SPATIAL_CONFIGURATION.PLANE_ANGLE_MEASURE'] *
        TYPEOF(item)) = 2)) )) > 0)) THEN
      RETURN(FALSE);
    END_IF;
    IF ((rep_set[i].name = 'site_elevation') AND (SIZEOF(
        QUERY ( item <* rep_set[i].items | (NOT (SIZEOF([
        'PLANT_SPATIAL_CONFIGURATION.MEASURE_REPRESENTATION_ITEM',
        'PLANT_SPATIAL_CONFIGURATION.LENGTH_ANGLE_MEASURE'] *
        TYPEOF(item)) = 2)) )) > 0)) THEN
      RETURN(FALSE);
    END_IF;
    IF ((rep_set[i].name = 'site_locality') AND (SIZEOF(QUERY ( item <*
        rep_set[i].items | (NOT (('PLANT_SPATIAL_CONFIGURATION.' +
        'DESCRIPTIVE_REPRESENTATION_ITEM') IN TYPEOF(item))) )) > 0))
         THEN
      RETURN(FALSE);
    END_IF;
    IF ((rep_set[i].name = 'site_shape') AND (NOT (
        'PLANT_SPATIAL_CONFIGURATION.SHAPE_REPRESENTATION' IN
        TYPEOF(rep_set[i])))) THEN
      RETURN(FALSE);
    END_IF;
  END_REPEAT;
  RETURN(TRUE);
ELSE
  IF 'PLANT_SPATIAL_CONFIGURATION.SITE_BUILDING' IN type_set THEN
    REPEAT i := 1 TO HIINDEX(rep_set) BY 1;
      IF ((rep_set[i].name = 'building_number') AND (SIZEOF(
          QUERY ( item <* rep_set[i].items | (NOT ((
          'PLANT_SPATIAL_CONFIGURATION.' +
          'DESCRIPTIVE_REPRESENTATION_ITEM') IN TYPEOF(item))) )) > 0))
           THEN
        RETURN(FALSE);
```

```
    END_IF;
    IF ((rep_set[i].name = 'building_location_and_orientation') AND (
       NOT ((SIZEOF(QUERY ( item <* rep_set[i].items | (SIZEOF([
        'PLANT_SPATIAL_CONFIGURATION.AXIS2_PLACEMENT_2D',
        'PLANT_SPATIAL_CONFIGURATION.AXIS2_PLACEMENT_3D'] *
        TYPEOF(item)) = 1) )) = 1) AND (SIZEOF(QUERY ( item <* rep_set
        [i].items | (SIZEOF(['PLANT_SPATIAL_CONFIGURATION.' +
        'LENGTH_MEASURE_WITH_UNIT','PLANT_SPATIAL_CONFIGURATION.' +
        'MEASURE_REPRESENTATION_ITEM'] * TYPEOF(item)) = 2) )) > 0)
        AND (SIZEOF(QUERY ( item <* rep_set[i].items | (SIZEOF([
        'PLANT_SPATIAL_CONFIGURATION.LENGTH_MEASURE_WITH_UNIT',
        'PLANT_SPATIAL_CONFIGURATION.' +
        'MEASURE_REPRESENTATION_ITEM'] * TYPEOF(item)) = 2) OR
        (SIZEOF(['PLANT_SPATIAL_CONFIGURATION.AXIS2_PLACEMENT_2D',
        'PLANT_SPATIAL_CONFIGURATION.AXIS2_PLACEMENT_3D'] *
        TYPEOF(item)) = 1) )) = 0)))) THEN
      RETURN(FALSE);
    END_IF;
    IF ((rep_set[i].name = 'building_shape') AND (NOT (
        'PLANT_SPATIAL_CONFIGURATION.SHAPE_REPRESENTATION' IN
        TYPEOF(rep_set[i])))) THEN
      RETURN(FALSE);
    END_IF;
  END_REPEAT;
  RETURN(TRUE);
ELSE
  IF 'PLANT_SPATIAL_CONFIGURATION.SITE_FEATURE' IN type_set THEN
    REPEAT i := 1 TO HIINDEX(rep_set) BY 1;
      IF (rep_set[i].name = 'site_feature_type') AND (SIZEOF(
          QUERY ( item <* rep_set[i].items | (NOT ((
          'PLANT_SPATIAL_CONFIGURATION.' +
          'DESCRIPTIVE_REPRESENTATION_ITEM') IN TYPEOF(item))) )) >
          0)
           THEN
        RETURN(FALSE);
      END_IF;
      IF (rep_set[i].name = 'site_feature_location_and_orientation')
          AND (SIZEOF(QUERY ( item <* rep_set[i].items | (NOT (
          SIZEOF(['PLANT_SPATIAL_CONFIGURATION.' +
          'AXIS2_PLACEMENT_2D','PLANT_SPATIAL_CONFIGURATION.' +
          'AXIS2_PLACEMENT_3D'] * TYPEOF(item)) = 1)) )) > 0) THEN
        RETURN(FALSE);
      END_IF;
      IF (rep_set[i].name = 'site_feature_shape') AND (NOT ((
          'PLANT_SPATIAL_CONFIGURATION.' + 'SHAPE_REPRESENTATION') IN
           TYPEOF(rep_set[i]))) THEN
        RETURN(FALSE);
      END_IF;
      RETURN(TRUE);
    END_REPEAT;
  ELSE
    IF ('PLANT_SPATIAL_CONFIGURATION.PLANT_ITEM_' +
        'CONNECTOR_PROPERTY_DEFINITION') IN type_set THEN
      REPEAT i := 1 TO HIINDEX(rep_set) BY 1;
        IF (rep_set[i].name = 'point') AND (SIZEOF(QUERY ( item <*
            rep_set[i].items | (NOT (('PLANT_SPATIAL_CONFIGURATION.'
            + 'POINT') IN TYPEOF(item))) )) > 0) THEN
          RETURN(FALSE);
        END_IF;
```

```
            IF (rep_set[i].name = 'direction') AND (SIZEOF(
                QUERY ( item <* rep_set[i].items | (NOT ((
                'PLANT_SPATIAL_CONFIGURATION.' + 'DIRECTION') IN TYPEOF(
                item))) )) > 0) THEN
              RETURN(FALSE);
            END_IF;
            IF (rep_set[i].name = 'connector_location_and_orientation')
                AND (SIZEOF(QUERY ( item <* rep_set[i].items | (NOT (
                SIZEOF(['PLANT_SPATIAL_CONFIGURATION.' +
                'AXIS2_PLACEMENT_2D','PLANT_SPATIAL_CONFIGURATION.' +
                'AXIS2_PLACEMENT_3D'] * TYPEOF(item)) = 1)) )) > 0) THEN
              RETURN(FALSE);
            END_IF;
            RETURN(TRUE);
          END_REPEAT;
        END_IF;
      END_IF;
    END_IF;
  END_IF;
  RETURN(TRUE);

END_FUNCTION; -- coordinated_representation_names_and_item_types

FUNCTION cross_product(
            arg1, arg2: direction
    ): vector;

  LOCAL
    v2     : LIST [3:3] OF REAL;
    v1     : LIST [3:3] OF REAL;
    mag    : REAL;
    res    : direction;
    result : vector;
  END_LOCAL;
  IF (NOT EXISTS(arg1)) OR (arg1.dim = 2) OR (NOT EXISTS(arg2)) OR (arg2
      .dim = 2) THEN
    RETURN(?);
  ELSE
    BEGIN
      v1 := normalise(arg1).direction_ratios;
      v2 := normalise(arg2).direction_ratios;
      res.direction_ratios[1] := (v1[2] * v2[3]) - (v1[3] * v2[2]);
      res.direction_ratios[2] := (v1[3] * v2[1]) - (v1[1] * v2[3]);
      res.direction_ratios[3] := (v1[1] * v2[2]) - (v1[2] * v2[1]);
      mag := 0;
      REPEAT i := 1 TO 3 BY 1;
        mag := mag + (res.direction_ratios[i] * res.direction_ratios[i]);
      END_REPEAT;
      IF mag > 0 THEN
        result.orientation := res;
        result.magnitude := SQRT(mag);
      ELSE
        result.orientation := arg1;
        result.magnitude := 0;
      END_IF;
      RETURN(result);
    END;
  END_IF;
```

```
END_FUNCTION; -- cross_product

FUNCTION csg_primitive_class_set(
            rep_item: representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF NOT (SIZEOF(['PLANT_SPATIAL_CONFIGURATION.TORUS' +
      'PLANT_SPATIAL_CONFIGURATION.RIGHT_CIRCULAR_CYLINDER' +
      'PLANT_SPATIAL_CONFIGURATION.SPHERE' +
      'PLANT_SPATIAL_CONFIGURATION.BLOCK' +
      'PLANT_SPATIAL_CONFIGURATION.RIGHT_CIRCULAR_CONE'] * TYPEOF(
      rep_item)) > 0) THEN
    RETURN(return_set);
  END_IF;
  return_set := return_set + rep_item;
  RETURN(return_set);

END_FUNCTION; -- csg_primitive_class_set

FUNCTION csg_solid_class_set(
            rep_item: representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.CSG_SOLID' IN TYPEOF(rep_item))
      THEN
    RETURN(return_set);
  END_IF;
  return_set := return_set + boolean_result_class_set(rep_item\csg_solid
      .tree_root_expression) + csg_primitive_class_set(rep_item\csg_solid
      .tree_root_expression);
  RETURN(return_set);

END_FUNCTION; -- csg_solid_class_set

FUNCTION curve_bounded_surface_class_set(
            rep_item: representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    i          : INTEGER;
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.CURVE_BOUNDED_SURFACE' IN TYPEOF(
      rep_item)) THEN
    RETURN(return_set);
  END_IF;
  return_set := return_set + surface_class_set(rep_item\
      curve_bounded_surface.basis_surface);
  REPEAT i := 1 TO HIINDEX(rep_item\curve_bounded_surface.boundaries)
      BY 1;
    return_set := return_set + curve_class_set(rep_item\
        curve_bounded_surface.boundaries[i]);
```

```
    END_REPEAT;
    RETURN(return_set);

  END_FUNCTION; -- curve_bounded_surface_class_set

  FUNCTION curve_class_set(
              rep_item: representation_item
      ): SET [0:?] OF representation_item;

    LOCAL
      i          : INTEGER;
      return_set : SET [0:?] OF representation_item := [];
    END_LOCAL;
    IF NOT ('PLANT_SPATIAL_CONFIGURATION.CURVE' IN TYPEOF(rep_item)) THEN
      RETURN(return_set);
    END_IF;
    IF SIZEOF(['PLANT_SPATIAL_CONFIGURATION.POLYLINE',
        'PLANT_SPATIAL_CONFIGURATION.B_SPLINE_CURVE',
        'PLANT_SPATIAL_CONFIGURATION.CIRCLE',
        'PLANT_SPATIAL_CONFIGURATION.ELLIPSE',
        'PLANT_SPATIAL_CONFIGURATION.PARABOLA',
        'PLANT_SPATIAL_CONFIGURATION.HYPERBOLA',
        'PLANT_SPATIAL_CONFIGURATION.LINE'] * TYPEOF(rep_item)) > 0 THEN
      return_set := return_set + rep_item;
    END_IF;
    IF 'PLANT_SPATIAL_CONFIGURATION.COMPOSITE_CURVE' IN TYPEOF(rep_item)
        THEN
      REPEAT i := 1 TO HIINDEX(rep_item\composite_curve.segments) BY 1;
        return_set := return_set + curve_class_set(rep_item\
            composite_curve.segments[i].parent_curve);
      END_REPEAT;
    END_IF;
    IF 'PLANT_SPATIAL_CONFIGURATION.TRIMMED_CURVE' IN TYPEOF(rep_item)
        THEN
      return_set := return_set + curve_class_set(rep_item\trimmed_curve.
          basis_curve);
    END_IF;
    IF 'PLANT_SPATIAL_CONFIGURATION.SURFACE_CURVE' IN TYPEOF(rep_item)
        THEN
      return_set := return_set + curve_class_set(rep_item\surface_curve.
          curve_3d);
      REPEAT i := 1 TO HIINDEX(rep_item\surface_curve.associated_geometry)
          BY 1;
        return_set := return_set + surface_class_set(rep_item\
            surface_curve.associated_geometry[i]);
      END_REPEAT;
    END_IF;
    RETURN(return_set);

  END_FUNCTION; -- curve_class_set

  FUNCTION curve_weights_positive(
              b: rational_b_spline_curve
      ): BOOLEAN;

    LOCAL
      result : BOOLEAN := TRUE;
    END_LOCAL;
    REPEAT i := 0 TO b.upper_index_on_control_points BY 1;
```

```
    IF b.weights[i] <= 0 THEN
      result := FALSE;
      RETURN(result);
    END_IF;
  END_REPEAT;
  RETURN(result);

END_FUNCTION; -- curve_weights_positive

FUNCTION derive_dimensional_exponents(
             x: unit
    ): dimensional_exponents;

  LOCAL
    i      : INTEGER;
    result : dimensional_exponents := dimensional_exponents(0,0,0,0,0,0,
              0);
  END_LOCAL;
  IF 'PLANT_SPATIAL_CONFIGURATION.DERIVED_UNIT' IN TYPEOF(x) THEN
    REPEAT i := LOINDEX(x.elements) TO HIINDEX(x.elements) BY 1;
      result.length_exponent := result.length_exponent + (x.elements[i].
          exponent * x.elements[i].unit.dimensions.length_exponent);
      result.mass_exponent := result.mass_exponent + (x.elements[i].
          exponent * x.elements[i].unit.dimensions.mass_exponent);
      result.time_exponent := result.time_exponent + (x.elements[i].
          exponent * x.elements[i].unit.dimensions.time_exponent);
      result.electric_current_exponent := result.
          electric_current_exponent + (x.elements[i].exponent * x.
          elements[i].unit.dimensions.electric_current_exponent);
      result.thermodynamic_temperature_exponent := result.
          thermodynamic_temperature_exponent + (x.elements[i].exponent *
          x.elements[i].unit.dimensions.
          thermodynamic_temperature_exponent);
      result.amount_of_substance_exponent := result.
          amount_of_substance_exponent + (x.elements[i].exponent * x.
          elements[i].unit.dimensions.amount_of_substance_exponent);
      result.luminous_intensity_exponent := result.
          luminous_intensity_exponent + (x.elements[i].exponent * x.
          elements[i].unit.dimensions.luminous_intensity_exponent);
    END_REPEAT;
  ELSE
    result := x.dimensions;
  END_IF;
  RETURN(result);

END_FUNCTION; -- derive_dimensional_exponents

FUNCTION dimension_of(
             item: geometric_representation_item
    ): dimension_count;

  LOCAL
    x : SET OF representation;
    y : representation_context;
  END_LOCAL;
  x := using_representations(item);
  y := x[1].context_of_items;
  RETURN(y\geometric_representation_context.coordinate_space_dimension);
```

```
END_FUNCTION; -- dimension_of

FUNCTION dimensions_for_si_unit(
            n: si_unit_name
    ): dimensional_exponents;
  CASE n OF
    metre           :     RETURN(dimensional_exponents(1,0,0,0,0,0,0));
    gram            :     RETURN(dimensional_exponents(0,1,0,0,0,0,0));
    second          :     RETURN(dimensional_exponents(0,0,1,0,0,0,0));
    ampere          :     RETURN(dimensional_exponents(0,0,0,1,0,0,0));
    kelvin          :     RETURN(dimensional_exponents(0,0,0,0,1,0,0));
    mole            :     RETURN(dimensional_exponents(0,0,0,0,0,1,0));
    candela         :     RETURN(dimensional_exponents(0,0,0,0,0,0,1));
    radian          :     RETURN(dimensional_exponents(0,0,0,0,0,0,0));
    steradian       :     RETURN(dimensional_exponents(0,0,0,0,0,0,0));
    hertz           :     RETURN(dimensional_exponents(0,0,-1,0,0,0,0));
    newton          :     RETURN(dimensional_exponents(1,1,-2,0,0,0,0));
    pascal          :     RETURN(dimensional_exponents(-1,1,-2,0,0,0,0));
    joule           :     RETURN(dimensional_exponents(2,1,-2,0,0,0,0));
    watt            :     RETURN(dimensional_exponents(2,1,-3,0,0,0,0));
    coulomb         :     RETURN(dimensional_exponents(0,0,1,1,0,0,0));
    volt            :     RETURN(dimensional_exponents(2,1,-3,-1,0,0,0));
    farad           :     RETURN(dimensional_exponents(-2,-1,4,1,0,0,0));
    ohm             :     RETURN(dimensional_exponents(2,1,-3,-2,0,0,0));
    siemens         :     RETURN(dimensional_exponents(-2,-1,3,2,0,0,0));
    weber           :     RETURN(dimensional_exponents(2,1,-2,-1,0,0,0));
    tesla           :     RETURN(dimensional_exponents(0,1,-2,-1,0,0,0));
    henry           :     RETURN(dimensional_exponents(2,1,-2,-2,0,0,0));
    degree_celsius  :     RETURN(dimensional_exponents(0,0,0,0,1,0,0));
    lumen           :     RETURN(dimensional_exponents(0,0,0,0,0,0,1));
    lux             :     RETURN(dimensional_exponents(-2,0,0,0,0,0,1));
    becquerel       :     RETURN(dimensional_exponents(0,0,-1,0,0,0,0));
    gray            :     RETURN(dimensional_exponents(2,0,-2,0,0,0,0));
    sievert         :     RETURN(dimensional_exponents(2,0,-2,0,0,0,0));
    END_CASE;

END_FUNCTION; -- dimensions_for_si_unit

FUNCTION dot_product(
            arg1, arg2: direction
    ): REAL;

  LOCAL
    ndim   : INTEGER;
    scalar : REAL;
    vec1   : direction;
    vec2   : direction;
  END_LOCAL;
  IF (NOT EXISTS(arg1)) OR (NOT EXISTS(arg2)) THEN
    scalar := ?;
  ELSE
    IF arg1.dim <> arg2.dim THEN
      scalar := ?;
    ELSE
      BEGIN
        vec1 := normalise(arg1);
        vec2 := normalise(arg2);
        ndim := arg1.dim;
        scalar := 0;
```

```
        REPEAT i := 1 TO ndim BY 1;
          scalar := scalar + (vec1.direction_ratios[i] * vec2.
            direction_ratios[i]);
        END_REPEAT;
      END;
    END_IF;
  END_IF;
  RETURN(scalar);

END_FUNCTION; -- dot_product

FUNCTION edge_reversed(
             an_edge: edge
    ): edge;

  LOCAL
    the_reverse : edge;
  END_LOCAL;
  IF 'PLANT_SPATIAL_CONFIGURATION.ORIENTED_EDGE' IN TYPEOF(an_edge)
       THEN
    the_reverse := oriented_edge(an_edge\oriented_edge.edge_element,NOT
        an_edge\oriented_edge.orientation);
  ELSE
    the_reverse := oriented_edge(an_edge,FALSE);
  END_IF;
  RETURN(the_reverse);

END_FUNCTION; -- edge_reversed

FUNCTION elementary_surface_class_set(
             rep_item: representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.ELEMENTARY_SURFACE' IN TYPEOF(
      rep_item)) THEN
    RETURN(return_set);
  END_IF;
  return_set := return_set + rep_item;
  RETURN(return_set);

END_FUNCTION; -- elementary_surface_class_set

FUNCTION face_bound_reversed(
             a_face_bound: face_bound
    ): face_bound;

  LOCAL
    the_reverse : face_bound;
  END_LOCAL;
  IF 'PLANT_SPATIAL_CONFIGURATION.FACE_OUTER_BOUND' IN TYPEOF(
      a_face_bound) THEN
    the_reverse := face_bound(a_face_bound\face_bound.bound,NOT
        a_face_bound\face_bound.orientation);
  ELSE
    the_reverse := face_bound(a_face_bound.bound,NOT a_face_bound.
        orientation);
```

```
    END_IF;
    RETURN(the_reverse);

  END_FUNCTION; -- face_bound_reversed

  FUNCTION face_class_set(
              rep_item: representation_item
      ): SET [0:?] OF representation_item;

    LOCAL
      local_loop  : loop;
      i           : INTEGER;
      j           : INTEGER;
      local_oedge : oriented_edge;
      return_set  : SET [0:?] OF representation_item := [];
    END_LOCAL;
    IF NOT ('PLANT_SPATIAL_CONFIGURATION.FACE' IN TYPEOF(rep_item)) THEN
      RETURN(return_set);
    END_IF;
    REPEAT i := 1 TO HIINDEX(rep_item\face.bounds) BY 1;
      local_loop := rep_item\face.bounds[i].bound;
      IF 'PLANT_SPATIAL_CONFIGURATION.POLY_LOOP' IN TYPEOF(local_loop)
           THEN
        return_set := return_set + local_loop;
      END_IF;
      IF 'PLANT_SPATIAL_CONFIGURATION.EDGE_LOOP' IN TYPEOF(local_loop)
           THEN
        REPEAT j := 1 TO HIINDEX(local_loop\path.edge_list) BY 1;
          local_oedge := local_loop\path.edge_list[j];
          return_set := return_set + local_oedge.edge_element\edge_curve.
              edge_geometry;
        END_REPEAT;
      END_IF;
    END_REPEAT;
    IF 'PLANT_SPATIAL_CONFIGURATION.FACE_SURFACE' IN TYPEOF(rep_item)
         THEN
      return_set := return_set + surface_class_set(rep_item\face_surface.
          face_geometry);
    END_IF;
    RETURN(return_set);

  END_FUNCTION; -- face_class_set

  FUNCTION face_edge_count(
              face_with_edges: face
      ): INTEGER;

    LOCAL
      local_loop : loop;
      i          : INTEGER;
      edge_count : INTEGER := 0;
    END_LOCAL;
    IF NOT ('PLANT_SPATIAL_CONFIGURATION.FACE' IN TYPEOF(face_with_edges))
        THEN
      RETURN(0);
    END_IF;
    REPEAT i := 1 TO HIINDEX(face_with_edges\face.bounds) BY 1;
      local_loop := face_with_edges\face.bounds[i].bound;
      IF 'PLANT_SPATIAL_CONFIGURATION.EDGE_LOOP' IN TYPEOF(local_loop)
```

```
        THEN
      edge_count := edge_count + HIINDEX(local_loop\path.edge_list);
    END_IF;
    IF 'PLANT_SPATIAL_CONFIGURATION.POLY_LOOP' IN TYPEOF(local_loop)
        THEN
      edge_count := edge_count + HIINDEX(local_loop\poly_loop.polygon);
    END_IF;
  END_REPEAT;
  RETURN(edge_count);

END_FUNCTION; -- face_edge_count

FUNCTION face_reversed(
            a_face: face
    ): face;

  LOCAL
    the_reverse : face;
  END_LOCAL;
  IF 'PLANT_SPATIAL_CONFIGURATION.ORIENTED_FACE' IN TYPEOF(a_face) THEN
    the_reverse := oriented_face(a_face\oriented_face.face_element,NOT
        a_face\oriented_face.orientation);
  ELSE
    the_reverse := oriented_face(a_face,FALSE);
  END_IF;
  RETURN(the_reverse);

END_FUNCTION; -- face_reversed

FUNCTION first_proj_axis(
            z_axis, arg: direction
    ): direction;

  LOCAL
    x_vec  : vector;
    v      : direction;
    z      : direction;
    x_axis : direction;
  END_LOCAL;
  IF (NOT EXISTS(z_axis)) OR (NOT EXISTS(arg)) OR (arg.dim <> 3) THEN
    x_axis := ?;
  ELSE
    z_axis := normalise(z_axis);
    IF NOT EXISTS(arg) THEN
      IF z_axis <> direction([1,0,0]) THEN
        v := direction([1,0,0]);
      ELSE
        v := direction([0,1,0]);
      END_IF;
    ELSE
      IF cross_product(arg,z).magnitude = 0 THEN
        RETURN(?);
      ELSE
        v := normalise(arg);
      END_IF;
    END_IF;
    x_vec := scalar_times_vector(dot_product(v,z),z_axis);
    x_axis := vector_difference(v,x_vec).orientation;
    x_axis := normalise(x_axis);
```

```
    END_IF;
    RETURN(x_axis);

  END_FUNCTION; -- first_proj_axis

  FUNCTION geometric_set_class_set(
              rep_item: representation_item
      ): SET [0:?] OF representation_item;

    LOCAL
      i          : INTEGER;
      return_set : SET [0:?] OF representation_item := [];
    END_LOCAL;
    IF NOT ('PLANT_SPATIAL_CONFIGURATION.GEOMETRIC_SET' IN
         TYPEOF(rep_item))
         THEN
      RETURN(return_set);
    END_IF;
    REPEAT i := 1 TO HIINDEX(rep_item\geometric_set.elements) BY 1;
      return_set := return_set + curve_class_set(rep_item\geometric_set.
          elements[i]);
    END_REPEAT;
    RETURN(return_set);

  END_FUNCTION; -- geometric_set_class_set

  FUNCTION get_basis_surface(
              c: curve_on_surface
      ): SET [0:2] OF surface;

    LOCAL
      surfs : SET [0:2] OF surface;
      n     : INTEGER;
    END_LOCAL;
    surfs := [];
    IF 'PLANT_SPATIAL_CONFIGURATION.PCURVE' IN TYPEOF(c) THEN
      surfs := [c\pcurve.basis_surface];
    ELSE
      IF 'PLANT_SPATIAL_CONFIGURATION.SURFACE_CURVE' IN TYPEOF(c) THEN
        n := SIZEOF(c\surface_curve.associated_geometry);
        REPEAT i := 1 TO n BY 1;
          surfs := surfs + associated_surface(c\surface_curve.
              associated_geometry[i]);
        END_REPEAT;
      END_IF;
    END_IF;
    IF 'PLANT_SPATIAL_CONFIGURATION.COMPOSITE_CURVE_ON_SURFACE' IN TYPEOF(
        c) THEN
      n := SIZEOF(c\composite_curve_on_surface.segments);
      surfs := get_basis_surface(c\composite_curve_on_surface.segments[1].
          parent_curve);
      IF n > 1 THEN
        REPEAT i := 2 TO n BY 1;
          surfs := surfs * get_basis_surface(c\composite_curve_on_surface.
              segments[i].parent_curve);
        END_REPEAT;
      END_IF;
    END_IF;
    RETURN(surfs);
```

```
END_FUNCTION; -- get_basis_surface

FUNCTION item_in_context(
             item: representation_item;
             cntxt: representation_context
    ): BOOLEAN;

  LOCAL
    i : INTEGER;
    y : BAG OF representation_item;
  END_LOCAL;
  IF SIZEOF(USEDIN(item,
      'PLANT_SPATIAL_CONFIGURATION.REPRESENTATION.ITEMS') * cntxt.
      representations_in_context) > 0 THEN
    RETURN(TRUE);
  ELSE
    y := QUERY ( z <* USEDIN(item,'') | (
        'PLANT_SPATIAL_CONFIGURATION.REPRESENTATION_ITEM' IN TYPEOF(z))
        );
    IF SIZEOF(y) > 0 THEN
      REPEAT i := 1 TO HIINDEX(y) BY 1;
        IF item_in_context(y[i],cntxt) THEN
          RETURN(TRUE);
        END_IF;
      END_REPEAT;
    END_IF;
  END_IF;
  RETURN(FALSE);

END_FUNCTION; -- item_in_context

FUNCTION leap_year(
             year: year_number
    ): BOOLEAN;
  IF (((year MOD 4) = 0) AND ((year MOD 100) <> 0)) OR ((year MOD 400) =
      0) THEN
    RETURN(TRUE);
  ELSE
    RETURN(FALSE);
  END_IF;

END_FUNCTION; -- leap_year

FUNCTION list_face_loops(
             f: face
    ): LIST [0:?] OF loop;

  LOCAL
    loops : LIST [0:?] OF loop := [];
  END_LOCAL;
  REPEAT i := 1 TO SIZEOF(f.bounds) BY 1;
    loops := loops + f.bounds[i].bound;
  END_REPEAT;
  RETURN(loops);

END_FUNCTION; -- list_face_loops

FUNCTION list_of_topology_reversed(
             a_list: list_of_reversible_topology_item
```

```
    ): list_of_reversible_topology_item;

  LOCAL
    the_reverse : list_of_reversible_topology_item;
  END_LOCAL;
  the_reverse := [];
  REPEAT i := 1 TO SIZEOF(a_list) BY 1;
    the_reverse := topology_reversed(a_list[i]) + the_reverse;
  END_REPEAT;
  RETURN(the_reverse);

END_FUNCTION; -- list_of_topology_reversed

FUNCTION list_to_array(
            lis: LIST [0:?] OF GENERIC:t;
            low, u: INTEGER
    ): ARRAY [low:u] OF GENERIC:t;

  LOCAL
    n   : INTEGER;
    res : ARRAY [low:u] OF GENERIC:t;
  END_LOCAL;
  n := SIZEOF(lis);
  IF n <> ((u - low) + 1) THEN
    RETURN(?);
  ELSE
    REPEAT i := 1 TO n BY 1;
      res[(low + i) - 1] := lis[i];
    END_REPEAT;
    RETURN(res);
  END_IF;

END_FUNCTION; -- list_to_array

FUNCTION list_to_set(
            l: LIST [0:?] OF GENERIC:t
    ): SET OF GENERIC:t;

  LOCAL
    s : SET OF GENERIC:t := [];
  END_LOCAL;
  REPEAT i := 1 TO SIZEOF(l) BY 1;
    s := s + l[i];
  END_REPEAT;
  RETURN(s);

END_FUNCTION; -- list_to_set

FUNCTION make_array_of_array(
            lis: LIST [1:?] OF LIST [1:?] OF GENERIC:t;
            low1, u1, low2, u2: INTEGER
    ): ARRAY [low1:u1] OF ARRAY [low2:u2] OF GENERIC:t;

  LOCAL
    n2   : INTEGER;
    n1   : INTEGER;
    res  : ARRAY [low1:u1] OF ARRAY [low2:u2] OF GENERIC:t;
    res1 : LIST [1:?] OF ARRAY [low2:u2] OF GENERIC:t;
  END_LOCAL;
```

```
  n1 := SIZEOF(lis);
  n2 := SIZEOF(lis[1]);
  IF (n1 <> ((u1 - low1) + 1)) AND (n2 <> ((u2 - low2) + 1)) THEN
    RETURN(?);
  END_IF;
  REPEAT i := 1 TO n1 BY 1;
    IF SIZEOF(lis[i]) <> n2 THEN
      RETURN(?);
    END_IF;
  END_REPEAT;
  REPEAT i := 1 TO n1 BY 1;
    resl[i] := list_to_array(lis[i],low2,u2);
  END_REPEAT;
  res := list_to_array(resl,low1,u1);
  RETURN(res);

END_FUNCTION; -- make_array_of_array

FUNCTION manifold_solid_brep_class_set(
              rep_item: representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    i          : INTEGER;
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.MANIFOLD_SOLID_BREP' IN TYPEOF(
      rep_item)) THEN
    RETURN(return_set);
  END_IF;
  return_set := return_set + connected_face_set_class_set(rep_item\
      manifold_solid_brep.outer);
  IF 'PLANT_SPATIAL_CONFIGURATION.BREP_WITH_VOIDS' IN TYPEOF(rep_item)
       THEN
    REPEAT i := 1 TO HIINDEX(rep_item\brep_with_voids.voids) BY 1;
      return_set := return_set + connected_face_set_class_set(rep_item\
          brep_with_voids.voids[i]);
    END_REPEAT;
  END_IF;
  RETURN(return_set);

END_FUNCTION; -- manifold_solid_brep_class_set

FUNCTION mixed_loop_type_set(
              l: SET [0:?] OF loop
    ): LOGICAL;

  LOCAL
    i              : INTEGER;
    poly_loop_type : LOGICAL;
  END_LOCAL;
  IF SIZEOF(l) <= 1 THEN
    RETURN(FALSE);
  END_IF;
  poly_loop_type := 'PLANT_SPATIAL_CONFIGURATION.POLY_LOOP' IN TYPEOF(l[
      1]);
  REPEAT i := 2 TO SIZEOF(l) BY 1;
    IF ('PLANT_SPATIAL_CONFIGURATION.POLY_LOOP' IN TYPEOF(l[i])) <>
        poly_loop_type THEN
```

```
      RETURN(TRUE);
    END_IF;
  END_REPEAT;
  RETURN(FALSE);

END_FUNCTION; -- mixed_loop_type_set

FUNCTION normalise(
            arg: vector_or_direction
  ): vector_or_direction;

  LOCAL
    ndim   : INTEGER;
    v      : direction;
    vec    : vector;
    mag    : REAL;
    result : vector_or_direction;
  END_LOCAL;
  IF NOT EXISTS(arg) THEN
    result := ?;
  ELSE
    ndim := arg.dim;
    IF 'PLANT_SPATIAL_CONFIGURATION.VECTOR' IN TYPEOF(arg) THEN
      BEGIN
        vec := arg;
        v := arg.orientation;
        IF arg.magnitude = 0 THEN
          RETURN(?);
        ELSE
          vec.magnitude := 1;
        END_IF;
      END;
    ELSE
      v := arg;
    END_IF;
    mag := 0;
    REPEAT i := 1 TO ndim BY 1;
      mag := mag + (v.direction_ratios[i] * v.direction_ratios[i]);
    END_REPEAT;
    IF mag > 0 THEN
      mag := SQRT(mag);
      REPEAT i := 1 TO ndim BY 1;
        v.direction_ratios[i] := v.direction_ratios[i] / mag;
      END_REPEAT;
      IF 'PLANT_SPATIAL_CONFIGURATION.VECTOR' IN TYPEOF(arg) THEN
        vec.orientation := v;
        result := vec;
      ELSE
        result := v;
      END_IF;
    ELSE
      RETURN(?);
    END_IF;
  END_IF;
  RETURN(result);

END_FUNCTION; -- normalise
```

```
FUNCTION orthogonal_complement(
            vec: direction
    ): direction;

  LOCAL
    result : direction;
  END_LOCAL;
  IF (vec.dim <> 2) OR (NOT EXISTS(vec)) THEN
    RETURN(?);
  ELSE
    result.direction_ratios[1] := -vec.direction_ratios[2];
    result.direction_ratios[2] := vec.direction_ratios[1];
    RETURN(result);
  END_IF;

END_FUNCTION; -- orthogonal_complement

FUNCTION path_head_to_tail(
            a_path: path
    ): LOGICAL;

  LOCAL
    n : INTEGER;
    p : BOOLEAN := TRUE;
  END_LOCAL;
  n := SIZEOF(a_path.edge_list);
  REPEAT i := 2 TO n BY 1;
    p := p AND (a_path.edge_list[i - 1].edge_end :=: a_path.edge_list[i]
        .edge_start);
  END_REPEAT;
  RETURN(p);

END_FUNCTION; -- path_head_to_tail

FUNCTION path_reversed(
            a_path: path
    ): path;

  LOCAL
    the_reverse : path;
  END_LOCAL;
  IF 'PLANT_SPATIAL_CONFIGURATION.ORIENTED_PATH' IN TYPEOF(a_path) THEN
    the_reverse := oriented_path(a_path\oriented_path.path_element,NOT
        a_path\oriented_path.orientation);
  ELSE
    the_reverse := oriented_path(a_path,FALSE);
  END_IF;
  RETURN(the_reverse);

END_FUNCTION; -- path_reversed

FUNCTION plant_spatial_configuration_organization_correlation(
            e: plant_spatial_configuration_organization_assignment
    ): BOOLEAN;

  LOCAL
    o_role : STRING;
  END_LOCAL;
  o_role := e\organization_assignment.role.name;
```

```
    CASE o_role OF
      'vendor'          :          IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <*
            e.items | (SIZEOF(['PLANT_SPATIAL_CONFIGURATION.PRODUCT',
            'PLANT_SPATIAL_CONFIGURATION.DOCUMENT'] * TYPEOF(x)) = 1) ))
             THEN
          RETURN(FALSE);
        END_IF;
      'owner'           :          IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <*
            e.items | (SIZEOF(['PLANT_SPATIAL_CONFIGURATION.SITE',
            'PLANT_SPATIAL_CONFIGURATION.DOCUMENT'] * TYPEOF(x)) = 1) ))
             THEN
          RETURN(FALSE);
        END_IF;
      'plant_operator'  :          IF SIZEOF(e.items) <> SIZEOF(
            QUERY ( x <* e.items | ('PLANT_SPATIAL_CONFIGURATION.PLANT' IN
            TYPEOF(x)) )) THEN
          RETURN(FALSE);
        END_IF;
      'plant_owner'     :          IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <*
            e.items | ('PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(x)) ))
             THEN
          RETURN(FALSE);
        END_IF;
      'project_owner'   :          IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <*
            e.items | ('PLANT_SPATIAL_CONFIGURATION.DESIGN_PROJECT' IN
            TYPEOF(x)) )) THEN
          RETURN(FALSE);
        END_IF;
      'assessor'        :          IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <*
            e.items | (('PLANT_SPATIAL_CONFIGURATION.' +
            'PRODUCT_DEFINITION_RELATIONSHIP') IN TYPEOF(x)) )) THEN
          RETURN(FALSE);
        END_IF;
      END_CASE;
    RETURN(TRUE);

  END_FUNCTION; -- plant_spatial_configuration_organization_correlation

  FUNCTION plant_spatial_configuration_person_and_organization_correlation(
          e: plant_spatial_configuration_person_and_organization_assignment
      ): BOOLEAN;

    LOCAL
      po_role : STRING;
    END_LOCAL;
    po_role := e\person_and_organization_assignment.role.name;
    CASE po_role OF
      'owner'           :          IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <*
            e.items | (SIZEOF(['PLANT_SPATIAL_CONFIGURATION.SITE',
            'PLANT_SPATIAL_CONFIGURATION.' + 'CHANGE_ITEM'] * TYPEOF(x)) =
            1) )) THEN
          RETURN(FALSE);
        END_IF;
      'plant_owner'     :          IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <*
            e.items | ('PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(x)) ))
             THEN
          RETURN(FALSE);
        END_IF;
      'plant_operator'  :          IF SIZEOF(e.items) <> SIZEOF(
```

```
          QUERY ( x <* e.items | ('PLANT_SPATIAL_CONFIGURATION.PLANT' IN
          TYPEOF(x)) )) THEN
        RETURN(FALSE);
      END_IF;
    END_CASE;
  RETURN(TRUE);

END_FUNCTION; --
plant_spatial_configuration_person_and_organization_correlation

FUNCTION plant_spatial_configuration_person_correlation(
            e: plant_spatial_configuration_person_assignment
    ): BOOLEAN;

  LOCAL
    p_role : STRING;
  END_LOCAL;
  p_role := e\person_assignment.role.name;
  CASE p_role OF
    'vendor'     :         IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <* e.
          items | ('PLANT_SPATIAL_CONFIGURATION.DOCUMENT' IN TYPEOF(x))
          ))
           THEN
      RETURN(FALSE);
    END_IF;
    'owner'      :         IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <* e.
          items | (SIZEOF(['PLANT_SPATIAL_CONFIGURATION.SITE',
          'PLANT_SPATIAL_CONFIGURATION.DOCUMENT'] * TYPEOF(x)) = 1) ))
           THEN
      RETURN(FALSE);
    END_IF;
    'plant_owner'  :          IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <* e
          .items | ('PLANT_SPATIAL_CONFIGURATION.PLANT' IN TYPEOF(x)) ))
           THEN
      RETURN(FALSE);
    END_IF;
    'assessor'   :         IF SIZEOF(e.items) <> SIZEOF(QUERY ( x <* e.
          items | (('PLANT_SPATIAL_CONFIGURATION.' +
          'PRODUCT_DEFINITION_RELATIONSHIP') IN TYPEOF(x)) )) THEN
      RETURN(FALSE);
    END_IF;
  END_CASE;
  RETURN(TRUE);

END_FUNCTION; -- plant_spatial_configuration_person_correlation

FUNCTION rectangular_composite_surface_class_set(
            rep_item: representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    i          : INTEGER;
    patch      : surface_patch;
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.RECTANGULAR_COMPOSITE_SURFACE' IN
      TYPEOF(rep_item)) THEN
    RETURN(return_set);
  END_IF;
```

```
    REPEAT i := 1 TO HIINDEX(rep_item\rectangular_composite_surface.
        segments) BY 1;
      patch := rep_item\rectangular_composite_surface.segments[i];
      return_set := return_set + surface_class_set(patch.parent_surface);
    END_REPEAT;
    RETURN(return_set);

  END_FUNCTION; -- rectangular_composite_surface_class_set

  FUNCTION rectangular_trimmed_surface_class_set(
              rep_item: representation_item
      ): SET [0:?] OF representation_item;

    LOCAL
      return_set : SET [0:?] OF representation_item := [];
    END_LOCAL;
    IF NOT ('PLANT_SPATIAL_CONFIGURATION.RECTANGULAR_TRIMMED_SURFACE' IN
        TYPEOF(rep_item)) THEN
      RETURN(return_set);
    END_IF;
    return_set := return_set + surface_class_set(rep_item\
        rectangular_trimmed_surface.basis_surface);
    RETURN(return_set);

  END_FUNCTION; -- rectangular_trimmed_surface_class_set

  FUNCTION rep_item_set_has_analytic(
              rep_item_set: SET [0:?] OF representation_item
      ): BOOLEAN;

    LOCAL
      set_iterator : INTEGER;
    END_LOCAL;
    IF SIZEOF(rep_item_set) = 0 THEN
      RETURN(FALSE);
    END_IF;
    REPEAT set_iterator := 1 TO HIINDEX(rep_item_set) BY 1;
      IF SIZEOF((['PLANT_SPATIAL_CONFIGURATION.CIRCLE',
          'PLANT_SPATIAL_CONFIGURATION.ELLIPSE',
          'PLANT_SPATIAL_CONFIGURATION.PARABOLA',
          'PLANT_SPATIAL_CONFIGURATION.HYPERBOLA',
          'PLANT_SPATIAL_CONFIGURATION.CONICAL_SURFACE',
          'PLANT_SPATIAL_CONFIGURATION.CYLINDRICAL_SURFACE',
          'PLANT_SPATIAL_CONFIGURATION.SPHERICAL_SURFACE',
          'PLANT_SPATIAL_CONFIGURATION.TOROIDAL_SURFACE'] * TYPEOF(
          rep_item_set[set_iterator])) > 0) THEN
        RETURN(TRUE);
      END_IF;
    END_REPEAT;
    RETURN(FALSE);

  END_FUNCTION; -- rep_item_set_has_analytic

  FUNCTION rep_item_set_has_free_form(
              rep_item_set: SET [0:?] OF representation_item
      ): BOOLEAN;

    LOCAL
      set_iterator : INTEGER;
```

```
  END_LOCAL;
  IF SIZEOF(rep_item_set) = 0 THEN
    RETURN(FALSE);
  END_IF;
  REPEAT set_iterator := 1 TO HIINDEX(rep_item_set) BY 1;
    IF SIZEOF((['PLANT_SPATIAL_CONFIGURATION.B_SPLINE_CURVE',
        'PLANT_SPATIAL_CONFIGURATION.B_SPLINE_SURFACE'] * TYPEOF(
        rep_item_set[set_iterator])) > 0) THEN
      RETURN(TRUE);
    END_IF;
  END_REPEAT;
  RETURN(FALSE);

END_FUNCTION; -- rep_item_set_has_free_form

FUNCTION scalar_times_vector(
             scalar: REAL;
             vec: vector_or_direction
    ): vector;

  LOCAL
    v      : direction;
    mag    : REAL;
    result : vector;
  END_LOCAL;
  IF (NOT EXISTS(scalar)) OR (NOT EXISTS(vec)) THEN
    result := ?;
  ELSE
    IF 'PLANT_SPATIAL_CONFIGURATION.VECTOR' IN TYPEOF(vec) THEN
      v := vec.orientation;
      mag := scalar * vec.magnitude;
    ELSE
      v := vec;
      mag := scalar;
    END_IF;
    IF mag < 0 THEN
      REPEAT i := 1 TO SIZEOF(v.direction_ratios) BY 1;
        v.direction_ratios[i] := -v.direction_ratios[i];
      END_REPEAT;
      mag := -mag;
    END_IF;
    result.orientation := normalise(v);
    result.magnitude := mag;
  END_IF;
  RETURN(result);

END_FUNCTION; -- scalar_times_vector

FUNCTION second_proj_axis(
             z_axis, x_axis, arg: direction
    ): direction;

  LOCAL
    temp   : vector;
    v      : direction;
    y_axis : vector;
  END_LOCAL;
  IF NOT EXISTS(arg) THEN
    v := direction([0,1,0]);
```

```
    ELSE
      v := arg;
    END_IF;
    temp := scalar_times_vector(dot_product(v,z_axis),z_axis);
    y_axis := vector_difference(v,temp);
    temp := scalar_times_vector(dot_product(v,x_axis),x_axis);
    y_axis := vector_difference(y_axis,temp);
    y_axis := normalise(y_axis);
    RETURN(y_axis.orientation);

  END_FUNCTION; -- second_proj_axis

  FUNCTION set_of_topology_reversed(
              a_set: set_of_reversible_topology_item
      ): set_of_reversible_topology_item;

    LOCAL
      the_reverse : set_of_reversible_topology_item;
    END_LOCAL;
    the_reverse := [];
    REPEAT i := 1 TO SIZEOF(a_set) BY 1;
      the_reverse := the_reverse + topology_reversed(a_set[i]);
    END_REPEAT;
    RETURN(the_reverse);

  END_FUNCTION; -- set_of_topology_reversed

  FUNCTION shell_reversed(
              a_shell: shell
      ): shell;

    LOCAL
      the_reverse : shell;
    END_LOCAL;
    IF 'PLANT_SPATIAL_CONFIGURATION.ORIENTED_OPEN_SHELL' IN TYPEOF(a_shell)
        THEN
      the_reverse := oriented_open_shell(a_shell\oriented_open_shell.
          open_shell_element,NOT a_shell\oriented_open_shell.orientation);
    ELSE
      IF 'PLANT_SPATIAL_CONFIGURATION.OPEN_SHELL' IN TYPEOF(a_shell) THEN
        the_reverse := oriented_open_shell(a_shell,FALSE);
      ELSE
        IF 'PLANT_SPATIAL_CONFIGURATION.ORIENTED_CLOSED_SHELL' IN TYPEOF(
            a_shell) THEN
          the_reverse := oriented_closed_shell(a_shell\
              oriented_closed_shell.closed_shell_element,NOT a_shell\
              oriented_closed_shell.orientation);
        ELSE
          IF 'PLANT_SPATIAL_CONFIGURATION.CLOSED_SHELL' IN TYPEOF(a_shell)
              THEN
            the_reverse := oriented_closed_shell(a_shell,FALSE);
          ELSE
            the_reverse := ?;
          END_IF;
        END_IF;
      END_IF;
    END_IF;
    RETURN(the_reverse);
```

```
END_FUNCTION; -- shell_reversed

FUNCTION surface_class_set(
            rep_item: representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    i          : INTEGER;
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.SURFACE' IN TYPEOF(rep_item))
      THEN
    RETURN(return_set);
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.B_SPLINE_SURFACE' IN TYPEOF(rep_item)
      THEN
    return_set := return_set + rep_item;
  END_IF;
  return_set := return_set + curve_bounded_surface_class_set(rep_item) +
      rectangular_composite_surface_class_set(rep_item) +
     rectangular_trimmed_surface_class_set(rep_item) +
     elementary_surface_class_set(rep_item);
  RETURN(return_set);

END_FUNCTION; -- surface_class_set

FUNCTION surface_weights_positive(
            b: rational_b_spline_surface
    ): BOOLEAN;

  LOCAL
    result : BOOLEAN := TRUE;
  END_LOCAL;
  REPEAT i := 0 TO b.u_upper BY 1;
    REPEAT j := 0 TO b.v_upper BY 1;
      IF b.weights[i][j] <= 0 THEN
        result := FALSE;
        RETURN(result);
      END_IF;
    END_REPEAT;
  END_REPEAT;
  RETURN(result);

END_FUNCTION; -- surface_weights_positive

FUNCTION swept_area_solid_class_set(
            rep_item: representation_item
    ): SET [0:?] OF representation_item;

  LOCAL
    return_set : SET [0:?] OF representation_item := [];
  END_LOCAL;
  IF NOT ('PLANT_SPATIAL_CONFIGURATION.SWEPT_AREA_SOLID' IN TYPEOF(
      rep_item)) THEN
    RETURN(return_set);
  END_IF;
  return_set := return_set + rep_item + curve_bounded_surface_class_set(
      rep_item\swept_area_solid.swept_area);
  RETURN(return_set);
```

```
END_FUNCTION; -- swept_area_solid_class_set

FUNCTION topology_reversed(
             an_item: reversible_topology
    ): reversible_topology;
  IF 'PLANT_SPATIAL_CONFIGURATION.EDGE' IN TYPEOF(an_item) THEN
    RETURN(edge_reversed(an_item));
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.PATH' IN TYPEOF(an_item) THEN
    RETURN(path_reversed(an_item));
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.FACE_BOUND' IN TYPEOF(an_item) THEN
    RETURN(face_bound_reversed(an_item));
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.FACE' IN TYPEOF(an_item) THEN
    RETURN(face_reversed(an_item));
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.SHELL' IN TYPEOF(an_item) THEN
    RETURN(shell_reversed(an_item));
  END_IF;
  IF 'SET' IN TYPEOF(an_item) THEN
    RETURN(set_of_topology_reversed(an_item));
  END_IF;
  IF 'LIST' IN TYPEOF(an_item) THEN
    RETURN(list_of_topology_reversed(an_item));
  END_IF;
  RETURN(?);

END_FUNCTION; -- topology_reversed

FUNCTION using_representations(
             item: representation_item
    ): SET OF representation;

  LOCAL
    results             : SET OF representation;
    i                   : INTEGER;
    intermediate_items  : SET OF representation_item;
    result_bag          : BAG OF representation;
  END_LOCAL;
  result_bag := USEDIN(item,
      'PLANT_SPATIAL_CONFIGURATION.REPRESENTATION.ITEMS');
  IF SIZEOF(result_bag) > 0 THEN
    REPEAT i := 1 TO HIINDEX(result_bag) BY 1;
      results := results + result_bag[i];
    END_REPEAT;
  END_IF;
  intermediate_items := QUERY ( z <* bag_to_set(USEDIN(item,'')) | (
      'PLANT_SPATIAL_CONFIGURATION.REPRESENTATION_ITEM' IN TYPEOF(z)) );
  IF SIZEOF(intermediate_items) > 0 THEN
    REPEAT i := 1 TO HIINDEX(intermediate_items) BY 1;
      results := results + using_representations(intermediate_items[i]);
    END_REPEAT;
  END_IF;
  RETURN(results);

END_FUNCTION; -- using_representations
```

```
FUNCTION valid_calendar_date(
             date: calendar_date
    ): LOGICAL;
  IF NOT ((1 <= date.day_component) AND (date.day_component <= 31))
       THEN
    RETURN(FALSE);
  END_IF;
  CASE date.month_component OF
    4 :          RETURN((1 <= date.day_component) AND (date.day_component
          <= 30));
    6 :          RETURN((1 <= date.day_component) AND (date.day_component
          <= 30));
    9 :          RETURN((1 <= date.day_component) AND (date.day_component
          <= 30));
    11  :          RETURN((1 <= date.day_component) AND (date.
          day_component <= 30));
    2 :          BEGIN
        IF leap_year(date.year_component) THEN
          RETURN((1 <= date.day_component) AND (date.day_component
          <= 29));
        ELSE
          RETURN((1 <= date.day_component) AND (date.day_component
          <= 28));
        END_IF;
      END;
    END_CASE;

END_FUNCTION; -- valid_calendar_date

FUNCTION valid_time(
             time: local_time
    ): BOOLEAN;
  IF EXISTS(time.second_component) THEN
    RETURN(EXISTS(time.minute_component));
  ELSE
    RETURN(TRUE);
  END_IF;

END_FUNCTION; -- valid_time

FUNCTION valid_units(
             m: measure_with_unit
    ): BOOLEAN;
  IF 'PLANT_SPATIAL_CONFIGURATION.LENGTH_MEASURE' IN TYPEOF(m.
      value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(1,0,0,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.MASS_MEASURE' IN TYPEOF(m.
      value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(0,1,0,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.TIME_MEASURE' IN TYPEOF(m.
      value_component) THEN
```

```
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(0,0,1,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.ELECTRIC_CURRENT_MEASURE' IN TYPEOF(m.
      value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(0,0,0,1,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.THERMODYNAMIC_TEMPERATURE_MEASURE' IN
      TYPEOF(m.value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(0,0,0,0,1,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.AMOUNT_OF_SUBSTANCE_MEASURE' IN
      TYPEOF(m.value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(0,0,0,0,0,1,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.LUMINOUS_INTENSITY_MEASURE' IN TYPEOF(
      m.value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(0,0,0,0,0,0,1) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.PLANE_ANGLE_MEASURE' IN TYPEOF(m.
      value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(0,0,0,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.SOLID_ANGLE_MEASURE' IN TYPEOF(m.
      value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(0,0,0,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.AREA_MEASURE' IN TYPEOF(m.
      value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(2,0,0,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.VOLUME_MEASURE' IN TYPEOF(m.
      value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(3,0,0,0,0,0,0) THEN
      RETURN(FALSE);
```

```
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.RATIO_MEASURE' IN TYPEOF(m.
      value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(0,0,0,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.POSITIVE_LENGTH_MEASURE' IN TYPEOF(m.
      value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(1,0,0,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'PLANT_SPATIAL_CONFIGURATION.POSITIVE_PLANE_ANGLE_MEASURE' IN
      TYPEOF(m.value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
        dimensional_exponents(0,0,0,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  RETURN(TRUE);

END_FUNCTION; -- valid_units

FUNCTION vector_difference(
            arg1, arg2: vector_or_direction
    ): vector;

  LOCAL
    ndim   : INTEGER;
    mag2   : REAL;
    mag1   : REAL;
    mag    : REAL;
    res    : direction;
    vec1   : direction;
    vec2   : direction;
    result : vector;
  END_LOCAL;
  IF (NOT EXISTS(arg1)) OR (NOT EXISTS(arg2)) OR (arg1.dim <> arg2.dim)
      THEN
    result := ?;
  ELSE
    BEGIN
      IF 'PLANT_SPATIAL_CONFIGURATION.VECTOR' IN TYPEOF(arg1) THEN
        mag1 := arg1.magnitude;
        vec1 := arg1.orientation;
      ELSE
        mag1 := 1;
        vec1 := arg1;
      END_IF;
      IF 'PLANT_SPATIAL_CONFIGURATION.VECTOR' IN TYPEOF(arg2) THEN
        mag2 := arg2.magnitude;
        vec2 := arg2.orientation;
      ELSE
        mag2 := 1;
        vec2 := arg2;
```

```
        END_IF;
        vec1 := normalise(vec1);
        vec2 := normalise(vec2);
        ndim := SIZEOF(vec1.direction_ratios);
        mag := 0;
        REPEAT i := 1 TO ndim BY 1;
          res.direction_ratios[i] := (mag1 * vec1.direction_ratios[i]) - (
              mag2 * vec2.direction_ratios[i]);
          mag := mag + (res.direction_ratios[i] * res.direction_ratios[i]);
        END_REPEAT;
        IF mag > 0 THEN
          result.magnitude := SQRT(mag);
          result.orientation := res;
        ELSE
          result.magnitude := 0;
          result.orientation := vec1;
        END_IF;
      END;
    END_IF;
    RETURN(result);

  END_FUNCTION; -- vector_difference

END_SCHEMA; -- plant_spatial_configuration
(*
```

Annex B
(normative)

AIM short names

Table B.1 provides the short names of entities specified in the AIM of this part of ISO 10303. Requirements on the use of the short names are found in the implementation methods included in ISO 10303.

Table B.1 - Short names of entities

Entity names	Short names
ACTION	ACTION
ACTION_ASSIGNMENT	ACTASS
ACTION_DIRECTIVE	ACTDRC
ACTION_METHOD	ACTMTH
ACTION_METHOD_RELATIONSHIP	ACMTRL
ACTION_RELATIONSHIP	ACTRLT
ACTION_REQUEST_ASSIGNMENT	ACRQAS
ACTION_REQUEST_SOLUTION	ACRQSL
ACTION_REQUEST_STATUS	ACRQST
ACTION_STATUS	ACTSTT
ADVANCED_CSG_SHAPE_REPRESENTATION	ACSR
AMOUNT_OF_SUBSTANCE_MEASURE_WITH_UNIT	AOSMWU
AMOUNT_OF_SUBSTANCE_UNIT	AOSU
ANGULAR_LOCATION	ANGLCT
APPLICATION_CONTEXT	APPCNT
APPLICATION_CONTEXT_ELEMENT	APCNEL
APPLICATION_PROTOCOL_DEFINITION	APPRDF
APPROVAL	APPRVL
APPROVAL_ASSIGNMENT	APPASS
APPROVAL_DATE_TIME	APDTTM

Table B.1 - Short names of entities (continued)

Entity names	Short names
APPROVAL_PERSON_ORGANIZATION	APPROR
APPROVAL_ROLE	APPRL
APPROVAL_STATUS	APPSTT
AREA_MEASURE_WITH_UNIT	AMWU
AREA_UNIT	ARUNT
ASSEMBLY_COMPONENT_USAGE	ASCMUS
AXIS1_PLACEMENT	AX1PLC
AXIS2_PLACEMENT_2D	A2PL2D
AXIS2_PLACEMENT_3D	A2PL3D
BEZIER_CURVE	BZRCRV
BEZIER_SURFACE	BZRSRF
BLANK_FITTING_CLASSIFICATION	BLFTCL
BLOCK	BLOCK
BOOLEAN_RESULT	BLNRSL
BOUNDARY_CURVE	BNDCR
BOUNDED_CURVE	BNDCRV
BOUNDED_PCURVE	BNDPCR
BOUNDED_SURFACE	BNDSRF
BOUNDED_SURFACE_CURVE	BNSRCR
BREP_WITH_VOIDS	BRWTVD
BUSHING_FITTING_CLASSIFICATION	BSFTCL
B_SPLINE_CURVE	BSPCR
B_SPLINE_CURVE_WITH_KNOTS	BSCWK
B_SPLINE_SURFACE	BSPSR
B_SPLINE_SURFACE_WITH_KNOTS	BSSWK

Table B.1 - Short names of entities (continued)

Entity names	Short names
CALENDAR_DATE	CLNDT
CARTESIAN_POINT	CRTPNT
CARTESIAN_TRANSFORMATION_OPERATOR	CRTROP
CARTESIAN_TRANSFORMATION_OPERATOR_3D	CTO3
CATALOGUE_ITEM	CTLITM
CENTRE_OF_SYMMETRY	CNOFSY
CHANGE_ACTION	CHNACT
CHANGE_FROM_ASSIGNMENT	CHFRAS
CHANGE_ITEM_ID_ASSIGNMENT	CIIA
CHANGE_TO_ASSIGNMENT	CHTAS
CHARACTERIZED_OBJECT	CHROBJ
CIRCLE	CIRCLE
CLASSIFICATION_ASSIGNMENT	CLSASS
CLOSED_SHELL	CLSSHL
COLOUR	COLOUR
COLOUR_RGB	CLRRGB
COLOUR_SPECIFICATION	CLRSPC
COMPOSITE_CURVE	CMPCRV
COMPOSITE_CURVE_ON_SURFACE	CCOS
COMPOSITE_CURVE_SEGMENT	CMCRSG
CONIC	CONIC
CONICAL_SURFACE	CNCSRF
CONNECTED_FACE_SET	CNFCST
CONNECTION_FUNCTIONAL_CLASSIFICATION	CNFNCL
CONNECTION_MOTION_CLASSIFICATION	CNMTCL

Table B.1 - Short names of entities (continued)

Entity names	Short names
CONNECTION_NODE	CNNND
CONNECTOR_END_TYPE_CLASSIFICATION	CETC
CONTEXT_DEPENDENT_UNIT	CNDPUN
CONVERSION_BASED_UNIT	CNBSUN
COORDINATED_UNIVERSAL_TIME_OFFSET	CUTO
COUPLING_FITTING_CLASSIFICATION	CPFTCL
CROSS_FITTING_CLASSIFICATION	CRFTCL
CSG_SOLID	CSGSLD
CURVE	CURVE
CURVE_BOUNDED_SURFACE	CRBNSR
CURVE_REPLICA	CRVRPL
CYLINDRICAL_SURFACE	CYLSRF
DATE	DATE
DATE_AND_TIME	DTANTM
DATE_AND_TIME_ASSIGNMENT	DATA
DATE_ASSIGNMENT	DTASS
DATE_ROLE	DTRL
DATE_TIME_ROLE	DTTMRL
DEFINITIONAL_REPRESENTATION	DFNRPR
DEGENERATE_PCURVE	DGNPCR
DEGENERATE_TOROIDAL_SURFACE	DGTRSR
DERIVED_SHAPE_ASPECT	DRSHAS
DERIVED_UNIT	DRVUNT
DERIVED_UNIT_ELEMENT	DRUNEL
DESCRIPTIVE_COLOUR	DSCCLR

Table B.1 - Short names of entities (continued)

Entity names	Short names
DESCRIPTIVE_REPRESENTATION_ITEM	DSRPIT
DESIGN_PROJECT	DSGPRJ
DESIGN_PROJECT_ASSIGNMENT	DSPRAS
DIMENSIONAL_CHARACTERISTIC_REPRESENTATION	DMCHRP
DIMENSIONAL_EXPONENTS	DMNEXP
DIMENSIONAL_LOCATION	DMNLCT
DIMENSIONAL_SIZE	DMNSZ
DIRECTED_ACTION	DRCACT
DIRECTION	DRCTN
DOCUMENT	DCMNT
DOCUMENT_REFERENCE	DCMRFR
DOCUMENT_RELATIONSHIP	DCMRLT
DOCUMENT_TYPE	DCMTYP
DOCUMENT_USAGE_CONSTRAINT	DCUSCN
EDGE	EDGE
EDGE_CURVE	EDGCRV
EDGE_LOOP	EDGLP
ELBOW_FITTING_CLASSIFICATION	ELFTCL
ELECTRICAL_CONNECTOR_CLASSIFICATION	ELCNCL
ELECTRICAL_SYSTEM	ELCSYS
ELEMENTARY_SURFACE	ELMSRF
ELLIPSE	ELLPS
EVALUATED_DEGENERATE_PCURVE	EVDGPC
EXECUTED_ACTION	EXCACT
EXTERNALLY_DEFINED_ITEM	EXDFIT

Table B.1 - Short names of entities (continued)

Entity names	Short names
EXTERNALLY_DEFINED_PLANT_ITEM_DEFINITION	EDPID
EXTERNAL_SOURCE	EXTSRC
EXTRUDED_AREA_SOLID	EXARSL
EXTRUDED_FACE_SOLID	EXFCSL
FACE	FACE
FACETED_BREP	FCTBR
FACE_BOUND	FCBND
FACE_SURFACE	FCSRF
FLANGE_FITTING_CLASSIFICATION	FLFTCL
FLANGE_FITTING_NECK_TYPE_CLASSIFICATION	FFNTC
FUNCTIONALLY_DEFINED_TRANSFORMATION	FNDFTR
GEOMETRIC_CURVE_SET	GMCRST
GEOMETRIC_REPRESENTATION_CONTEXT	GMRPCN
GEOMETRIC_REPRESENTATION_ITEM	GMRPIT
GEOMETRIC_SET	GMTST
GEOMETRIC_SET_REPLICA	GMSTRP
GROUP	GROUP
GROUP_ASSIGNMENT	GRPASS
HEAT_TRACING_REPRESENTATION	HTTRRP
HVAC_SYSTEM	HVCSYS
HYBRID_SHAPE_REPRESENTATION	HYSHRP
HYPERBOLA	HYPRBL
INLINE_EQUIPMENT	INLEQP
INSERT_FITTING_CLASSIFICATION	INFTCL
INSTRUMENTATION_AND_CONTROL_SYSTEM	IACS

Table B.1 - Short names of entities (continued)

Entity names	Short names
INTERFERING_SHAPE_ELEMENT	INSHEL
INTERSECTION_CURVE	INTCRV
LAP_JOINT_STUB_END_FITTING_CLASSIFICATION	LJSEFC
LATERAL_FITTING_CLASSIFICATION	LTFTCL
LENGTH_MEASURE_WITH_UNIT	LMWU
LENGTH_UNIT	LNGUNT
LINE	LINE
LINE_BRANCH_CONNECTION	LNBRCN
LINE_LESS_PIPING_SYSTEM	LLPS
LINE_PLANT_ITEM_BRANCH_CONNECTION	LPIBC
LINE_PLANT_ITEM_CONNECTION	LPIC
LINE_TERMINATION_CONNECTION	LNTRCN
LOCAL_TIME	LCLTM
LOOP	LOOP
MAKE_FROM_USAGE_OPTION	MFUO
MANIFOLD_SOLID_BREP	MNSLBR
MAPPED_ITEM	MPPITM
MASS_MEASURE_WITH_UNIT	MMWU
MASS_UNIT	MSSUNT
MATERIAL_DESIGNATION	MTRDSG
MATERIAL_DESIGNATION_RELATIONSHIP	MTDSRL
MATERIAL_PROPERTY	MTRPRP
MEASURE_REPRESENTATION_ITEM	MSRPIT
MEASURE_WITH_UNIT	MSWTUN
NAMED_UNIT	NMDUNT

Table B.1 - Short names of entities (continued)

Entity names	Short names
NAME_ASSIGNMENT	NMASS
OFFSET_CURVE_2D	OFCR2D
OFFSET_CURVE_3D	OFCR3D
OFFSET_SURFACE	OFFSRF
OLET_FITTING_CLASSIFICATION	OLFTCL
OPEN_SHELL	OPNSHL
ORGANIZATION	ORGNZT
ORGANIZATIONAL_PROJECT	ORGPRJ
ORGANIZATION_ASSIGNMENT	ORGASS
ORGANIZATION_ROLE	ORGRL
ORIENTED_CLOSED_SHELL	ORCLSH
ORIENTED_EDGE	ORNEDG
ORIENTED_FACE	ORNFC
ORIENTED_OPEN_SHELL	OROPSH
ORIENTED_PATH	ORNPTH
ORIFICE_PLATE_FITTING_CLASSIFICATION	OPFC
PARABOLA	PRBL
PATH	PATH
PCURVE	PCURVE
PERSON	PERSON
PERSON_AND_ORGANIZATION	PRANOR
PERSON_AND_ORGANIZATION_ASSIGNMENT	PAOA
PERSON_AND_ORGANIZATION_ROLE	PAOR
PERSON_ASSIGNMENT	PRSASS
PERSON_ROLE	PRSRL

Table B.1 - Short names of entities (continued)

Entity names	Short names
PIPE_CAP_FITTING_CLASSIFICATION	PCFC
PIPE_CLASSIFICATION	PPCLS
PIPING_COMPONENT_DEFINITION	PPCMDF
PIPING_CONNECTOR_CLASSIFICATION	PPCNCL
PIPING_DESIGN_CSG_SHAPE_REPRESENTATION	PDCSR
PIPING_SYSTEM	PPNSYS
PLACEMENT	PLCMNT
PLANE	PLANE
PLANE_ANGLE_MEASURE_WITH_UNIT	PAMWU
PLANE_ANGLE_UNIT	PLANUN
PLANT	PLANT
PLANT_ITEM_CONNECTION	PLITCN
PLANT_ITEM_CONNECTOR	PLTCN
PLANT_ITEM_DEFINITION	PLITDF
PLANT_ITEM_DEFINITION_CLASS	PIDC
PLANT_ITEM_INTERFERENCE	PLITIN
PLANT_ITEM_ROUTE	PLITRT
PLANT_ITEM_WEIGHT_REPRESENTATION	PIWR
PLANT_LINE_DEFINITION	PLLNDF
PLANT_LINE_SEGMENT_DEFINITION	PLSD
PLANT_LINE_SEGMENT_TERMINATION	PLST
PLANT_SPATIAL_CONFIGURATION_ACTION_REQUEST_ASSIGNMENT	PSCARA
PLANT_SPATIAL_CONFIGURATION_APPROVAL_ASSIGNMENT	PSCAA
PLANT_SPATIAL_CONFIGURATION_CHANGE_ASSIGNMENT	PSCCA
PLANT_SPATIAL_CONFIGURATION_DATE_AND_TIME_ASSIGNMENT	PSCDAT

Table B.1 - Short names of entities (continued)

Entity names	Short names
PLANT_SPATIAL_CONFIGURATION_DATE_ASSIGNMENT	PSCDA
PLANT_SPATIAL_CONFIGURATION_DOCUMENT_REFERENCE	PSCDR
PLANT_SPATIAL_CONFIGURATION_ORGANIZATION_ASSIGNMENT	PSCOA
PLANT_SPATIAL_CONFIGURATION_PERSON_AND_ORGANIZATION_-ASSIGNMENT	PSCPAO
PLANT_SPATIAL_CONFIGURATION_PERSON_ASSIGNMENT	PSCPA
POINT	POINT
POINT_ON_CURVE	PNONCR
POINT_ON_SURFACE	PNONSR
POINT_REPLICA	PNTRPL
POLYLINE	PLYLN
POLY_LOOP	PLYLP
PRESENTATION_LAYER_ASSIGNMENT	PRLYAS
PROCESS_CAPABILITY	PRCCPB
PRODUCT	PRDCT
PRODUCT_CATEGORY	PRDCTG
PRODUCT_CATEGORY_RELATIONSHIP	PRCTRL
PRODUCT_CONTEXT	PRDCNT
PRODUCT_DEFINITION	PRDDFN
PRODUCT_DEFINITION_CONTEXT	PRDFCN
PRODUCT_DEFINITION_FORMATION	PRDFFR
PRODUCT_DEFINITION_FORMATION_RELATIONSHIP	PDFR
PRODUCT_DEFINITION_FORMATION_WITH_SPECIFIED_SOURCE	PDFWSS
PRODUCT_DEFINITION_RELATIONSHIP	PRDFRL
PRODUCT_DEFINITION_SHAPE	PRDFSH

Table B.1 - Short names of entities (continued)

Entity names	Short names
PRODUCT_DEFINITION_SUBSTITUTE	PRDFSB
PRODUCT_DEFINITION_USAGE	PRDFUS
PRODUCT_DEFINITION_WITH_ASSOCIATED_DOCUMENTS	PDWAD
PRODUCT_RELATED_PRODUCT_CATEGORY	PRPC
PROPERTY_DEFINITION	PRPDFN
PROPERTY_DEFINITION_RELATIONSHIP	PRDFR
PROPERTY_DEFINITION_REPRESENTATION	PRDFRP
PURCHASE_ASSIGNMENT	PRCASS
QUASI_UNIFORM_CURVE	QSUNCR
QUASI_UNIFORM_SURFACE	QSUNSR
RATIONAL_B_SPLINE_CURVE	RBSC
RATIONAL_B_SPLINE_SURFACE	RBSS
RATIO_MEASURE_WITH_UNIT	RMWU
RATIO_UNIT	RTUNT
RECTANGULAR_COMPOSITE_SURFACE	RCCMSR
RECTANGULAR_TRIMMED_SURFACE	RCTRSR
REDUCER_FITTING_CLASSIFICATION	RDFTCL
REFERENCE_GEOMETRY	RFRGMT
REPARAMETRISED_COMPOSITE_CURVE_SEGMENT	RCCS
REPRESENTATION	RPRSNT
REPRESENTATION_CONTEXT	RPRCNT
REPRESENTATION_ITEM	RPRITM
REPRESENTATION_MAP	RPRMP
RESERVED_SPACE	RSRSPC
REVOLVED_AREA_SOLID	RVARSL

Table B.1 - Short names of entities (continued)

Entity names	Short names
REVOLVED_FACE_SOLID	RVFCSL
RIGHT_CIRCULAR_CONE	RGCRCN
RIGHT_CIRCULAR_CYLINDER	RGCRCY
SEAM_CURVE	SMCRV
SHAPE_ASPECT	SHPASP
SHAPE_ASPECT_DERIVING_RELATIONSHIP	SADR
SHAPE_ASPECT_RELATIONSHIP	SHASRL
SHAPE_DEFINITION_REPRESENTATION	SHDFRP
SHAPE_DIMENSION_REPRESENTATION	SHDMRP
SHAPE_REPRESENTATION	SHPRPR
SITE	SITE
SITE_BUILDING	STBLD
SITE_FEATURE	STFTR
SITE_REPRESENTATION	STRPR
SI_UNIT	SUNT
SOLID_ANGLE_MEASURE_WITH_UNIT	SAMWU
SOLID_ANGLE_UNIT	SLANUN
SOLID_MODEL	SLDMDL
SPACER_FITTING_CLASSIFICATION	SPFTCL
SPECIALTY_ITEM_CLASSIFICATION	SPITCL
SPHERE	SPHERE
SPHERICAL_SURFACE	SPHSRF
STREAM_DESIGN_CASE	STDSCS
STREAM_PHASE	STRPHS
STRUCTURAL_LOAD_CONNECTOR_CLASSIFICATION	SLCC

Table B.1 - Short names of entities (continued)

Entity names	Short names
STRUCTURAL_SYSTEM	STRSYS
SUPPORT_CONSTRAINT_REPRESENTATION	SPCNRP
SURFACE	SRFC
SURFACE_CURVE	SRFCRV
SURFACE_OF_LINEAR_EXTRUSION	SL
SURFACE_OF_REVOLUTION	SROFRV
SURFACE_PATCH	SRFPTC
SURFACE_REPLICA	SRFRPL
SWAGE_FITTING_CLASSIFICATION	SWFTCL
SWEPT_AREA_SOLID	SWARSL
SWEPT_FACE_SOLID	SWFCSL
SWEPT_SURFACE	SWPSRF
SYSTEM_SPACE	SYSSPC
TEE_FITTING_CLASSIFICATION	TFTCL
THERMODYNAMIC_TEMPERATURE_MEASURE_WITH_UNIT	TTMWU
THERMODYNAMIC_TEMPERATURE_UNIT	THTMUN
TIME_MEASURE_WITH_UNIT	TMWU
TIME_UNIT	TMUNT
TOPOLOGICAL_REPRESENTATION_ITEM	TPRPIT
TOROIDAL_SURFACE	TRDSRF
TORUS	TORUS
TRIMMED_CURVE	TRMCRV
TRUNCATED_PYRAMID	TRNPYR
UNIFORM_CURVE	UNFCRV
UNIFORM_SURFACE	UNFSRF

Table B.1 - Short names of entities (concluded)

Entity names	Short names
UNION_FITTING_CLASSIFICATION	UNFTCL
VALVE_CLASSIFICATION	VLVCLS
VECTOR	VECTOR
VERSIONED_ACTION_REQUEST	VRACRQ
VERTEX	VERTEX
VOLUME_MEASURE_WITH_UNIT	VMWU
VOLUME_UNIT	VLMUNT
WYE_FITTING_CLASSIFICATION	WYFTCL

Annex C
(normative)

Implementation method specific requirements

The implementation method defines what types of exchange behaviour are required with respect to this part of ISO 10303. Conformance to this part of ISO 10303 shall be realized in an exchange structure. The file format shall be encoded according to the syntax and EXPRESS language mapping defined in ISO 10303-21 and the AIM defined in annex A of this part of ISO 10303. The header of the exchange structure shall identify the use of this part of ISO 10303 by the schema name 'plant_spatial_configuration'.

Annex D
(normative)

Protocol Information Conformance Statement (PICS) proforma

The PICS proforma is supplied for completion by the person or organization (the client) requesting conformance testing. Its purpose is to ascertain the scope of claimed conformance to a particular application protocol by an implementation under test (IUT) using a defined implementation method. Through the completion of this form, the PICS Proforma becomes a PICS.

The information contained in the PICS is used to configure an appropriate executable test suite for use by the client.

Ten conformance classes are identified in this part of ISO 10303. A conforming implementation shall support at least one conformance class. Each class specifies a subset of ISO 10303-227 AIM constructs. These classes are detailed in clause 6 of ISO 10303-227.

Questions:

1. Please provide an identifier for the product or system for which conformance is claimed:

   Product name and current version number:_____

2. Please indicate the implementation method chosen:

   — ISO 10303-21 Exchange Structure -- preprocessor
   Preprocessor name and current version number:_____

   — ISO 10303-21 Exchange Structure -- postprocessor
   Postprocessor name and current version number:_____

3. Please indicate the classes for which conformance is claimed:

   — Class 1:_____

   — Class 2:_____

   — Class 3:_____

   — Class 4:_____

   — Class 5:_____

   — Class 6:_____

— Class 7:_____

— Class 8:_____

— Class 9:_____

— Class 10:_____

Annex E
(normative)

Information object registration

E.1 Document identification

In order to provide for unambiguous identification of an information object in an open system, the object identifier

{ iso standard 10303 part(227) version(-1) }

is assigned to this part of ISO 10303. The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

E.2 Schema identification

In order to provide for unambiguous identification of the schema specifications given in this application protocol in an open information system, object identifiers are assigned as follows:

{ iso standard 10303 part(227) version(-1) object(1) plant-spatial-configuration(1) }

is assigned to the **plant_spatial_configuration** expanded schema (see annex A).

{ iso standard 10303 part(227) version(-1) object(1) plant-spatial-configuration-schema(2) }

is assigned to the **plant_spatial_configuration** short form schema (see 5.2).

The meaning of these values is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Annex F
(informative)

Application activity model

The AAM is provided to aid the understanding of the scope and information requirements defined in this application protocol. The model is presented as a set of definitions of the activities and the data, and a set of activity figures. The viewpoint of the AAM is the users of plant spatial configuration information, including architect, engineer, and builder.

F.1 Application activity model definitions and abbreviations

The following terms are used in the application activity model. Terms marked with an asterisk are outside the scope of this application protocol.

The definitions given in this annex do not supersede the definitions given in the main body of the text.

F.1.1 Analyze Final Plant Design (AAM A345): Examine all aspects of final design for compliance to performance criteria and generate any necessary changes required to meet these criteria.

F.1.2 Authorization Plan*: High level plan, justification, and forecast for design and construction of a Plant. The authorization plan describes how funds, people, and resources are to be allocated for the plant project. It is a document used internally; it is sometimes called a "white paper".

F.1.3 As-built Documents*: Site plans, detailed equipment descriptions, electrical instrument diagrams, and P&I diagrams that record the as-built conditions of a Plant. They aid in meeting government documentation and safety requirements. Frequently, they are simply corrections or mods to existing design documents delivered to construction.

F.1.4 Bids*: Commercial proposal by vendor or supplier for provision of equipment, supplies, or services.

F.1.5 Calculate Heat/Mass Balance (A312): Calculations performed based on Design Basis, Unit operations, Fuel and/or Materials in the process, along with associated chemical properties to optimize Plant and operational cost.

F.1.6 Capital Appropriation*: Authorization of funding for capital project or expenditure.

F.1.7 Change Request: A request made by user of data (other than engineering user) for changes to original process plant design due to errors, omissions, and or other reasons (e.g., new requirements). A request is followed by review, analysis, and approval. Change requests are tracked in terms of cost and schedule (a kind of mini-project within project). Change requests may originate from construction or operations, for example.

Changes requests may be made against:

— supplier list;

— process;

— plant;

— procedure;

— design basis.

F.1.8 Chem Props*: Important chemical data needed by the Process Engineer during design. This data will typically include, but is not limited to:

— Boiling Point

— Critical Pressure

— Critical Temperature

— Density

— Enthalpy

— Entropy

— Flash Point

— Heat of Vapourization

— Melting Point

— Molecular Weight

— Specific Heat

— Thermal Conductivity

— Viscosity

F.1.9 Codes: A widely recognized, accepted, and sometimes legally mandated set of rules that apply during the life-cycle of the Plant. These rules govern life-cycle activities such as design, fabrication, and operation and characteristics such as safety. Codes are a consensus documents/specifications and are sometimes a subset of regulatory requirements. The ANSI 31.x series of codes are an example.

The design basis data specified will guide the code application, i.e., how the code is applied, e.g., 31.3 overpressurizing is allowed for short periods of times to accommodate specific design basis scenarios.)

F.1.10 Commission Plant (AAM A56): Test the functionality of the completed Plant prior to operation. Develop final operating and maintenance procedures. Obtain final regulatory approval to operate the Plant.

F.1.11 Commissioned Plant*: A Plant that has been proven to be operational through commissioning procedures.

F.1.12 Commissioning Procedures*: Step-by-step explanation of start-up actions required to commission Plant.

F.1.13 Confirm Safety and Regulatory Compliance (AAM A344): Establish that the final design of the Plant meets specified safety and regulatory criteria.

F.1.14 Construct Plant (AAM A5): The process of building a physical Plant, ready for use, using plans and building materials. The layout drawings and material requirements are used to establish the physical arrangement and to procure the materials required. A plan for erecting the Plant is determined from material schedules, heavy equipment schedules, labor schedules, and environmental conditions (such as weather). Temporary erection material is procured as needed (such as scaffolding). Regulatory requirements and client requirements are used to plan and erect the Plant, and for the final testing and certification for operation.

F.1.15 Construction Plan: A plan for building or fabricating a Plant or Plant Item.

F.1.16 Construction Services*: Constructor activities/ability to be used to construct the Plant.

F.1.17 Control and Approve Activities (AAM A11): Prepare plans, check conformance to plan, and arrange for corrective action.

F.1.18 Control Requirements: Requirements/criteria specified in operating procedures and safety requirements imposed on the mechanisms/systems that monitor and control Plant operation.

F.1.19 Corporate Standards: Procedures, instructions, or specifications that may be used in the execution of a plant project and are standardized within an organization; they are not project specific, but may be used (and possibly customized) by a variety of projects. Corporate standards are developed over a long period of time as standard, recommended, or best practice.

Kinds of standards include:

— safety;

— design;

— maintenance.

F.1.20 Define Plant Operating Philosophy (AAM A331): Ascertain and confirm those plant operating characteristics and activities necessary to achieve Plant owner's operational goals such as Methods of Production, Technology, Plant Safety, or Plant Availability. Includes selecting types and sequences of unit operations and processing steps so that the Plant production objectives can be achieved. Specify alternate or abnormal operating conditions and procedures such as startup and shutdowns.

F.1.21 Define Procedures, Standards, Guidelines, Specifications, and Codes (AAM A13): Define the engineering policies to be used and determine appropriate procedures, codes, standards, guidelines, and specifications that may apply.

F.1.22 Delivery Dates*: Scheduled date for delivery of procured items to meet or support construction.

F.1.23 Deploy Component/Service (AAM A44): The process by which the part/service is delivered and the acquisition agreement is fulfilled.

F.1.24 Design Basis: A document provided by the Plant owner or developed by the AEC contractors which establishes or defines the information/data on which Plant engineering is to be based. It consists of guidelines/requirements, corporate standards, codes, references to regulatory agreements, form of deliverables, plant/production capacity.

The design basis will specify:

— Type of plant;

— Plant product or output (type and capacity);

— Plant inputs (e.g., fuel, feedstock);

— Plant operating requirements;

— Plant process requirements;

— Site parameters (geographical, meteorological, soils, hydrological);

— Environmental requirements;

— Design Safety Philosophy;

— Plant license/permit requirements.

It also addresses performance objectives for the Plant such as:

— Safety and health;

— Environmental;

— Investment and project economics;

— Schedule;

— Capacity;

— Product and raw material storage;

— Product and plant quality;

— Engineering quality;

— Project execution;

— Technology.

Performance objectives usually take the form of a written document owned and maintained by the project team (consisting of members from the business, engineering, construction, and Plant site).

NOTE - This definition is from an owners perspective.

F.1.25 Design Plant (AAM A3): The activity of extending a concept into a detailed Plant definition suitable to support component purchase and Plant construction.

F.1.26 Design Strategy: The "how-to" for the design. It encompasses building technology, mechanical technology, utility technology, automation technology, schedules, scope, standards and regulations, process definition, control philosophies, costs, benefits and timings, and project approach (AE, CM, internal).

F.1.27 Dispose of Plant (AAM A64): Plan and complete the activities required to dispose of the Plant.

F.1.28 Engineer/Design Equipment (AAM A323): Preliminary sizing of all major pieces of equipment required is made at this time with enough detail to obtain budget quotes. This should include equipment such as refrigeration machines, purchase power substations, secondary substations, switch gear, compressors, and boilers, as well as the major process equipment specific to the Plant.

F.1.29 Engineer/Design Plant Systems (AAM A322): At the conceptual level, this is a preliminary scoping of all major modifications or additions to major Plant distribution systems such as steam, compressed air, electrical power, refrigeration, water, firewater, and sewers (storm, process, sanitary). Impact on distribution pipe lines and feeder systems quantity and size are also included. A consideration of system operation in terms of reliability, uptime, planned maintenance, and winter/summer operations are vital to proper planning.

F.1.30 Environmental Impact Assessment*: Evaluation of project/plant's affect on the environment. A report is usually required by an environmental regulatory agency before construction can begin.

F.1.31 Equipment Characteristics: Attributes of an item that have a name and measurable value. Equipment characteristics may be subdivided into physical and functional/performance characteristics.

Physical equipment characteristics describe or specify the size, material, or shape of the equipment. They are items of information that describe the shape, position, and orientation of equipment. For a pump, such information might include primitive elements (e.g., boxes and cylinders) to describe the shape, the position of suction and discharge nozzles and flange ratings of the nozzles.

Functional/performance characteristics describe or specify the process requirements for the equipment: what it is supposed to do and how much it is supposed to do it. They are items of information that describe: (1) the effect that equipment has on the process, (2) the service provided to the process by the equipment, or (3) other operational information. For a pump, such information might include flow rate, total developed head, efficiency, and a descriptive name or title, such as 1502-B Condensate Return Pump B.

Equipment Process Requirements are a subset of Equipment Functional Data that describe the contribution to the process desired from equipment. Such data is specified prior to the actual selection of specific equipment to fulfill the purpose.

F.1.32 Equipment ID: An identifier assigned to a piece of Equipment.

F.1.33 Equipment List: A list of equipment in the process plant. List is comprised of, but not limited to, name, identifier (e.g., tag), location (e.g., building, elevation, area, column row), contract numbers (e.g., purchase, install), spare requirements, service requirements (e.g., air, water, structural base, electrical power, control circuitry), drawing references (e.g., P&I diagrams, plant arrangements) and if applicable, electrical load and type. The list may not include all equipment; it does not include miscellaneous equipment/devices (e.g., y-pattern strainers, inline flow meters, instruments) or valves.

F.1.34 Erect Plant (AAM A53): Utilizing plans, materials, services and labor, build the physical Plant which conforms with the detailed design. This process begins with the site preparation; grading and primary foundations poured, and temporary roads and rails created. Utility services are provided and temporary warehouses are built. Major equipment is moved, often in pieces, and installed on foundations with supporting steel. Site permanent buildings are built, as are pipe racks and other permanent steel. Pipe runs and pipe spools are put in place with valves and miscellaneous equipment and welded or joined. Other items such as ducting, electrical, instrumentation are installed.

F.1.35 Establish Plant Design Basis (AAM A23): The activity of collecting a complete and consistent set of constraints, requirements, and guidelines for subsequent engineering activities. This activity results in the development of the design basis documents. See Design Basis.

F.1.36 Establish Initial Process Control Logic (AAM A314) : Document philosophical and operational requirements between instrumentation, equipment and process.

F.1.37 Evaluate Bids and Negotiate Purchase (AAM A43): The process by which bid packages are evaluated, a vendor is selected, and an agreement is entered into for the acquisition of the part.

F.1.38 Facilities*: Resources such as structures and infrastructures which are allocated/committed for a Plant project.

F.1.39 Finalize Layout/Arrangement/Spatial Design (AAM A342): Develop the spatial design of the Plant to its final approved-for-construction state; the primary input is the finalized system design.

F.1.40 Finalize System Design (AAM A341): Develop the system design, expressed by flow and control information and equipment performance data, to its final state. The resulting design serves as a basis for detailed Plant design.

F.1.41 Guidelines/Requirements: Specifications, instructions, and mandates specified by management that shall be followed in Plant project. They may be project specific. Guidelines are more generally applicable than requirements, which tend to be specific in terms of what must be done.

Originators and users of guidelines and requirements are:

— A&E;

— E&C;

— EPC;

— Owner;

— Constructor;

— CM;

— Contractor (basic practices).

These categories not mutually exclusive.

F.1.42 Heat/Mass Calculations: Calculations performed based on Design Basis, Unit operations, Fuel and/or Materials in the process, along with associated chemical properties to optimize Plant and operational cost.

F.1.43 Identify/Analyze Safety Requirements/Hazards (AAM A315): Review Design Basis, Unit Operations, Heat/Mass Balances, Materials, Identified Equipment, Control Logic and Process Flow Diagram against federal,state and/or local regulations, codes and/or standards to determine compliance and produce an analysis of results.

F.1.44 Identify/Define Unit Operations (AAM 311): Incorporate Design Basis and Owner Requirements to define and document basis for conceptual process design. If the design activity is related to an existing Plant, then existing operations are incorporated into the conceptual process design. This activity also results in estimated time and cost expenditures.

F.1.45 Identify Plant Performance Requirements and Establish Design Strategy (AAM A321): Plant Performance Requirements are a quantitative description of the quantity and quality of a product to be produced by the Plant in a yearly time period. Usually stated as "units of product/unit time". Additional qualifications are made regarding the quality of the Plant. Examples include: time between major shutdowns for continuous processes, percent uptime required, and expected yield.

Design strategy is a description of major steps required to complete enough design to obtain a budget estimate for business calculations as well as begin the identification of process unknowns which may or may not require piloting.

F.1.46 Identify/Size Equipment (AAM A313): Identify equipment requirements based on Design Basis and Unit Operations. Sizing of the equipment is based on the Heat/Mass Balance calculations and Unit Operations.

F.1.47 Inspection plan*: The description of anticipated activities necessary for surveillance of vendors, fabricators and assemblers to verify compliance to contractual specifications, codes and good practice. The plan usually lists the items to be inspected, the place at which the inspections are expected to occur, anticipated frequency of inspection and type of activity to be undertaken at each inspection. The reporting procedure for the surveillance results is usually included in the developed inspection plan.

F.1.48 Inspection Results*: Reports that result from inspection and vendor surveillance activities.

F.1.49 Layout Plant (AAM A324): A general arrangement of the Plant in plan view, showing all the major components of the distribution systems affected by the Plant and the location of the Plant. A general arrangement of all major equipment within the battery limits of the Plant is also included.

F.1.50 Line Sched/list: A subset of information presented on the P&I diagram (and possibly the Heat/Mass balance) that describes the characteristics of pipelines required for a given process. This information is used by the Piping Designer during the Detail Design. Analogous to equipment list.

F.1.51 Maintain Plant (AAM A63): Conduct and monitor the activities required to maintain the Plant.

F.1.52 Maintain Suppliers List (AAM A41): The process by which a list of accepted/approved suppliers is kept up to date.

F.1.53 Manage, Operate, Maintain, and Dispose of Plant (AAM A6): The activities required to manage, operate, maintain, and dispose of the Plant safely, efficiently, and according to operating procedures and regulations.

F.1.54 Manage Plant (AAM A61): Direct and administrate the operations, maintenance, and disposal of the Plant.

F.1.55 Manage Project (AAM A1): Provide sufficient resources to execute the project and check that the execution is done in accordance with the plans and regulations.

F.1.56 Management Authorizations/Controls*: Management authorization, imperatives, directives, and procedures for initiating and executing plant management activities.

F.1.57 Material Requirements: The specification of materials, material options, and/or material parameters for a Plant item. Includes references to applicable specifications and standards.

F.1.58 Money*: Funds available or committed for Plant project.

F.1.59 Obtain Agreements with Regulatory Bodies (AAM A22): The activity of confirming that the intended design, construction, commissioning, operation, and decommissioning of the proposed project will comply with requirements of the regulatory body. This confirmation is recorded by formal documentation such as written agreements and safety compliance reports.

F.1.60 Obtain Construction Services (AAM A52): Use the construction plan as a requirements list for outside services needed and their schedule. Negotiate contracts and agreements with sub-contractors, equipment suppliers, and labor unions as needed for the erection of the Plant. Adjust schedule of construction plan to allow for availability of sub-contractors, equipment and labor.

F.1.61 Operate Plant (AAM A62): Conduct and monitor the activities required to operate the Plant.

F.1.62 Operating Philosophy: The Plant owner's operational goals, such as Methods of Production, Technology, Plant Safety, and Plant Availability.

F.1.63 Operating Procedures*: Documentation necessary to run Plant safely; it covers many different phases and aspects of Plant operation.

F.1.64 Optimize for Environment (AAM A336): Evaluate the Plant design against the applicable environmental regulations (Federal, State, and local) and modify where required. These regulations influence many of the activities in Plant and process design such as operating procedures, Plant and process control strategies, specification and design of piping, instrumentation, and equipment, as well as site selection. Broad regulatory interpretations often mean that conservative measures are incorporated in Plant design.

F.1.65 Owner Requirements*: An initial statement of Plant requirements provided by the owner. It is an aggregation of items such as design requirements, specifications, standards, and client general specifications. The owner requirement may be provided at any level of abstraction from very general to very specific. Leads to design_basis.

F.1.66 Personnel*: A body of persons, usually employed in a factory, office, or organization.

F.1.67 Piping and Instrumentation (P&I) Diagram: A schematic diagram which shows engineering details of the equipment, instruments, pipes, valves, and their connectivity and sequence.
The following are kinds of P&I diagrams which reflect the evolutionary cycle of design:

— Preliminary: Conveys the flow of the fluids from equipment to equipment in the system. It shows the valves which are used to control the flow. The major fluid containing lines have been sized.

— Approved for Design (AFD): Process definition firm, needs instrumentation.

— Design: Instrumentation details are included. All lines and valves have been sized. All valving, vents and drains are included. Instrumentation and loops are indicated but final instrumentation may not have been selected.

— Final: Complete P&I diagram; approved for release by engineering for construction; a sealed document. It is a last version of the design P&I diagram. It contains all changes which were incorporated during the physical design of the systems. It reflects the Plant as it was, or will be, constructed.

F.1.68 Plan/Analyze Project Finances (AAM A21): The activity of anticipating and estimating the financial resource requirements for a project. This activity establishes the expected financial performance for the project and the project financial plan.

F.1.69 Plan Plant Project (AAM A2): The activity that established a detailed technical plan and a financial plan that are consistent with the engineering, construction, and commissioning activities required to fulfill the project objectives.

F.1.70 Plant: Produces chemicals, pharmaceutical, electrical power, petroleum, and similar products (i.e., produces "stuff" rather than "things").

F.1.71 Plant Items: A Plant Item is an item or piece of Equipment that may be used as a component of the Plant.

F.1.72 Pre-commission Plant (AAM A55): Resolve any differences between the detailed design and the as-built Plant. Changes occur primarily to the documentation of the Plant. Perform all testing required by regulatory agencies and the client. Resolve any problems which were discovered during testing. Obtain regulatory permission to start-up the Plant for functional testing.

F.1.73 Plant Performance Requirements: Plant Performance Requirements are a quantitative description of the quantity and quality of a product to be produced by the Plant in a yearly time period. Usually stated as "units of product/unit time". Additional qualifications are made regarding the quality of the Plant. Examples include: time between major shutdowns for continuous processes, percent uptime required, and expected yield.

F.1.74 Pre-commissioned Plant*: A Plant that is completed and ready for check out. Mechanical complete; transfer of ownership, operation remain.

F.1.75 Prepare Bid Packages and Solicit Bids (AAM A42): The process by which the technical and commercial requirements for a part are compiled and sent out for pricing by multiple vendors.

F.1.76 Process Control Logic*: Prose/diagrammatic explanation of mechanisms/systems that monitor and control a process.

F.1.77 Process Flow Diagram (PFD): A diagram of the Plant process flow scheme and related information. The PROCESS is defined as the chemical, physical or other schemes which are used to produce a resulting product. The FLOW is defined as the relationship of operations, reactions and other steps that are required to produce the product. It also may specify the stream connections, stream flow rates and compositions, and operating conditions.

The diagram usually contains simplified representation of basic instrumentation, operating conditions and material information necessary to provide a basic understanding of the process flow.

PFDs evolve through stages like P&I diagrams and other design documentation.

F.1.78 Procure Components (AAM A4): The process by which needed parts, equipment, or services are purchased or acquired.

F.1.79 Procured item*: Plant item that has been obtained from vendor or supplier for incorporation into the Plant.

States or status of procured items include:

— In_fabrication;

— Accepted;

— Shipped;

— Delivered_to_site.

F.1.80 Produce As-built Surveys (AAM A54): The completed Plant is given a physical inspection to determine where the Plant no longer conforms to the detailed design. The detail design drawings and other documents are updated to reflect these changes. Remove construction materials, temporary buildings, and equipment from the completed Plant. Place surplus Plant material in the spare parts warehouse or otherwise dispose of.

F.1.81 Produce Conceptual Plant Design (AAM A32): The activity of extending the conceptual process design into a preliminary Plant spatial arrangement.

F.1.82 Produce Conceptual Process Design (AAM A31): The activity of defining the basic parameters of Plant flow scheme. Parameters include relationship of operations necessary to transform feed stock into a product.

F.1.83 Produce Conceptual Safety Engineering Designs (AAM A325): Designs which specifically address how the major hazards associated with the new Plant are to be dealt with to ensure the safety of all personnel working in the Plant as well as the general site and surrounding neighborhood population. These hazards will have been identified during the preliminary screening reviews to identify chemical, fire and health hazards associated with the specific chemical and/or mechanical characteristics of the process.

F.1.84 Produce Construction Plans (AAM A51): Using site drawings, layout drawings, and other documents, determine the desired sequence to building the Plant which will meet contract budget and schedule. Determine those services which will be needed from sub-contractors for the site preparation, erection, and testing of the Plant. Develop detailed erection drawings and schedules for each section of the Plant and for temporary structures.

F.1.85 Produce Final Plant Design (AAM A34): Activity of producing Plant definition based on final process design and conceptual process design.

F.1.86 Produce Final Process Design (AAM A33): Integration of conceptual process and Plant designs to fully define parameters of Plant flow scheme.

F.1.87 Produce Process Flow Diagram (AAM A316): Production of a schematic showing basic process flow developed from the cumulative results of Unit Operations, Equipment Sizing, Initial Logic and Safety Requirements along with related chemical properties. If design activity is related to a modification or addition to an existing Plant then the existing Plant information is reflected in the developed Process Flow Diagram.

F.1.88 Product*: Marketable output of Plant.

F.1.89 Project Authorizations/Controls*: Management authorization, imperatives, directives, and procedures for initiating and executing project activities.

F.1.90 Project Financial Plan*: Plan that states how much the Plant will cost to construct, how it is to be paid for, and when payments are to be made. It is a general financing and cash flow plan.

F.1.91 Project Specific Procs/Stds/Gdlns/Specs/Codes: Procedures, Standards, Guidelines, Specifications, and Codes created specifically for the Plant project. They may call out and add to, modify, or tailor a standard. Portions are derived from design basis.

Project specific procedures, standards, guidelines, specifications, and codes evolve through stages like P&I diagrams and other design documentation.

F.1.92 Provide Resources (AAM A12): Acquire and deploy personnel, facilities, and funding to perform the project activities.

F.1.93 PSM Report (AAM A63): Plant safety management report.

F.1.94 Purchase Agreement*: Contract between two parties to provide a service or item for designated payment.

F.1.95 Qualified Construction Firms*: List of construction firms that are capable and acceptable to construct Plant.

F.1.96 Regulatory Agreements*: Mutual agreement between owner/operators and regulatory agencies

F.1.97 Regulatory Authorizations*: Approval from regulatory agencies to initiate activities.

F.1.98 Regulations/Regulatory requirements*: Federal, state, or local laws, codes, or standards that impact various activities related to the process Plant. Regulatory requirements may apply to, but are not limited to, permitting, engineering, construction, operations and decommissioning.

F.1.99 Resources*: Technology, people, facilities.

F.1.100 Restored Site*: Remediation of the site to environmentally acceptable conditions.

F.1.101 Safety Compliance Reports*: Documentation of analyses and evaluations of Plant with respect to safety considerations.

F.1.102 Safety Hazop Analysis*: The results of the analysis of the Plant design with respect to safety and hazardous operations. Identifies possible causes of faults and consequences, and recommends remedies.

F.1.103 Safety System Spec: Job specific specifications related to Plant safety. May be developed from applying the design safety philosophy to Plant design.

F.1.104 Satisfy Safety Requirements (HazOps) (AAM A335): Perform a formal Plant process design, operation, and control review (HazOps or Hazardous Operations review). Plant safety requirements involve issues such as source terms for spill scenarios, vapour dispersion for combustible and toxic releases, reliability of metallurgy and other materials, component failure rates, operator response and error, fail-safe instrumentation, equipment spacing, number and size of equipment trains, radiation from fires, relief system design philosophies, deflagration test results, thermal runaways and associated vent sizing, detonations and resulting shock waves. A formal HAZOPS review is performed against Plant designs which affect these issues.

F.1.105 Schedules*: A time-based list of project tasks that describes:

— what is supposed to happen;

— when it is supposed to happen;

— task sequence and dependencies;

— restraint and constraints;

— float;

— critical path.

Kinds of schedules include:

— reqd on site;

— planned.

F.1.106 Schematic Diagrams: A physically non-dimensional, 2D graphical representation of the logical/functional design of a system that do not (necessarily) encompass physical information.

Examples include:

— PFD;

— P&I diagram;

— electrical single line;

— motor control;

— control loops;

— HVAC;

— plumbing;

— I/O.

Schematic diagrams evolve through stages like P&I diagrams and other design documentation.

F.1.107 Site Information*: Information about the physical location where the Plant will be constructed and the conditions of any Plant on the site (if one exists). It includes:

— geological data, such as before and after terrain contours, and subterranean structure, and seismic activity.

— meteorological data such as seasonal wind profile, precipitation, snowfall, ambient temperature, and such.

— Road data;

— cadastre (property lines) zones;

— utilities.

A kind of site information is verified field dimensions. They are parameters which specify the physical and spatial characteristics of an existing item or component in a Plant which have been verified by

measurements taken by a second, independent agency. For example, field dimensions provided by the Plant owner for piping tie-in locations (coordinate locations and sizes) are considered as VERIFIED when duplicated by the AEC representative. If discrepancies are discovered during verification of the field dimensions, the initial and verification measurement processes must be repeated to assure verified dimensions.

Existing Plant Conditions are the characteristics of existing Plant relevant to Plant project. Used with revamps, retrofits, or expansion.

F.1.108 Specifications & Standards: Consensus or mandated technical descriptions of Plant hardware or systems that control the design or construction of a Plant.

F.1.109 Specify Building and Plant Services (AAM A343): Establish utility and other service needs for the building and Plant based on owner requirements, final system design, and final spatial design.

F.1.110 Specify Control Requirements (AAM A334): Instrumentation and control system characteristics required to fulfill requirements for Plant operation are defined by the operating procedures and safety requirements. Control systems are used to help maintain Plant safety, ensure product quality, and to safeguard equipment. These systems are used to control areas such as process reactions, flows, temperatures, pressures, and levels. They operate automatically, or provide indications to Plant personnel. Control Requirements are generally defined in the operating procedures and specified on the P&I diagram and in the instrument list.

F.1.111 Specify Equipment Functional Characteristics (AAM A333): The Functional Characteristics of each major item of equipment are defined based on confirmed Plant operating requirements, process technologies, and process optimization. Functional characteristics include equipment type, process stream inputs/outputs/capacities/conditions, equipment metallurgy, piping and instrumentation, power requirements, and auxiliary systems.

F.1.112 Specify Piping and Instrumentation (AAM A332): Piping and instrumentation functional requirements are developed based on Plant production capacities, process type/technology, control methodology, chemical content of process streams, and equipment layout. Alternative operating conditions, maintenance requirements, Plant operating and personal safety are issues considered. The results of this activity are detailed on P&I diagrams, line lists, equipment lists, and instrument lists.

F.1.113 Status*: A report of the current state of a task, design, action, or schedule. It is a quality assurance feedback mechanism.

F.1.114 Stream Data: Chemical composition, physical state, and mass quantities of process flows. Part of a PFD.

F.1.115 Suppliers List*: A list of vendors that provide commodities or services to an organization.

Kinds or statuses of supplier lists include:

— Approved;

— Recommended;

— Partnered.

F.1.116 System Layout/Design: The definitions/representation of the system sufficient for construction. This definition results from the use of the system design basis, P&I diagrams, specifications, and other documentation/information to specify the physical components/items and spatial configuration of the process Plant.

Note - The definition of "system" is broader than common usage, e.g., it encompasses structural systems.

System layout/designs can be viewed or categorized according to three different breakdowns:

1. Evolutionary Phase

— Initial;

— Design;

— Final.

2. System Type

— Piping;

— HVAC;

— Electrical

— I&C;

— Structural/Civil;

— Architecture;

— Safety.

3. Functional Views

— conceptual arrangement;

— Spatial information;

— Schematic diagram;

— P&I diagram (includes piping connectivity/sequencing).

For example, the final HVAC spatial information system design/layout will specify the definition, physical dimensions, location coordinates, and characteristics for all HVAC components which occupy space in the Plant. Only those physical dimensions, location coordinates, and characteristics required to specify the spatial instance of each component are included in this definition.

F.1.117 Technologies: Resources created through the practical application of scientific methods.

F.1.118 Time/Cost Estimates*: Projected or forecasted cost and length of time to:

— design, produce, or procure a Plant Item;

— obtain a service;

— achieve some goal.

F.1.119 Unit Operations: Design Basis and Owner Requirements that define basis for conceptual process design.

F.1.120 Vendor data: Documentation received from vendor concerning procured_items; drawings, manuals, calculations, anything necessary to define their deliverable items and services.

Kinds of vendor data include:

— Preliminary: in process design information;

— Certified: information from the supplier of equipment or other components which is warranted to correctly describe the as-delivered functional and/or physical data;

— released for fabrication/construction.

F.2 Application activity model diagrams

The application activity model is given in figures F.2 through F.13. The diagrams use IDEF0 function modelling. Figure 1 describes the basic notation used in IDEF0 modelling. Each activity may be decomposed to provide more detail. If an activity has been decomposed, a separate diagram is included.

As with any IDEF0 model, the AAM is dependent on a particular viewpoint and purpose. The purpose of the AAM is to describe the exchange of process plant spatial configuration information and design, fabrication, and maintenance information for process plant piping systems. Activities and data flows that are out of scope are marked with asterisks.

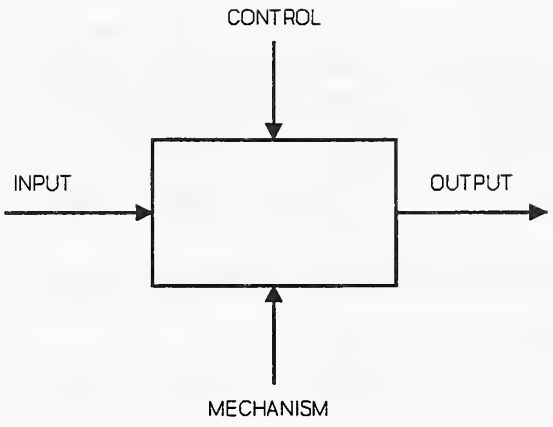


Figure F.1 - IDEF0 basic notation

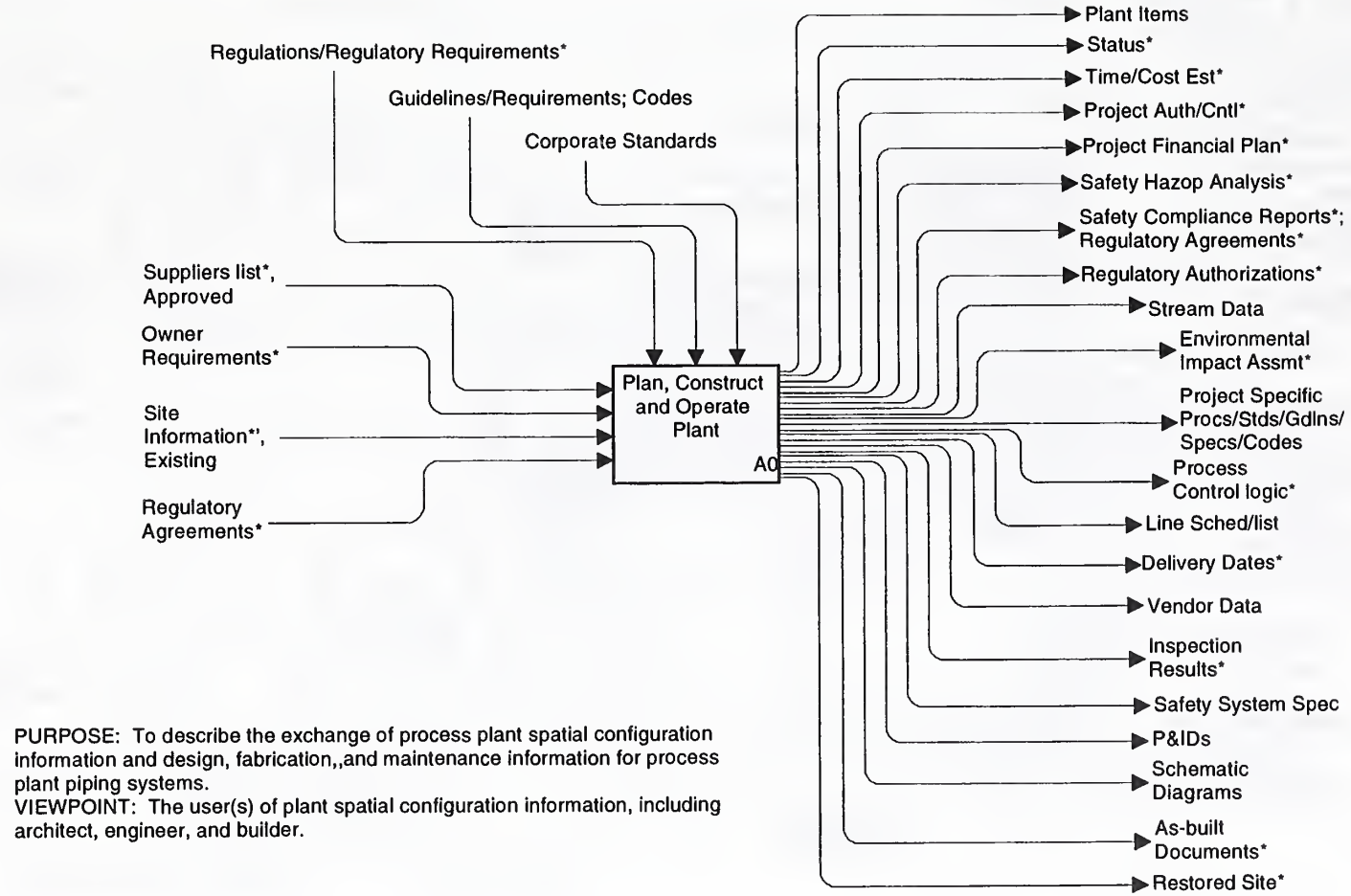
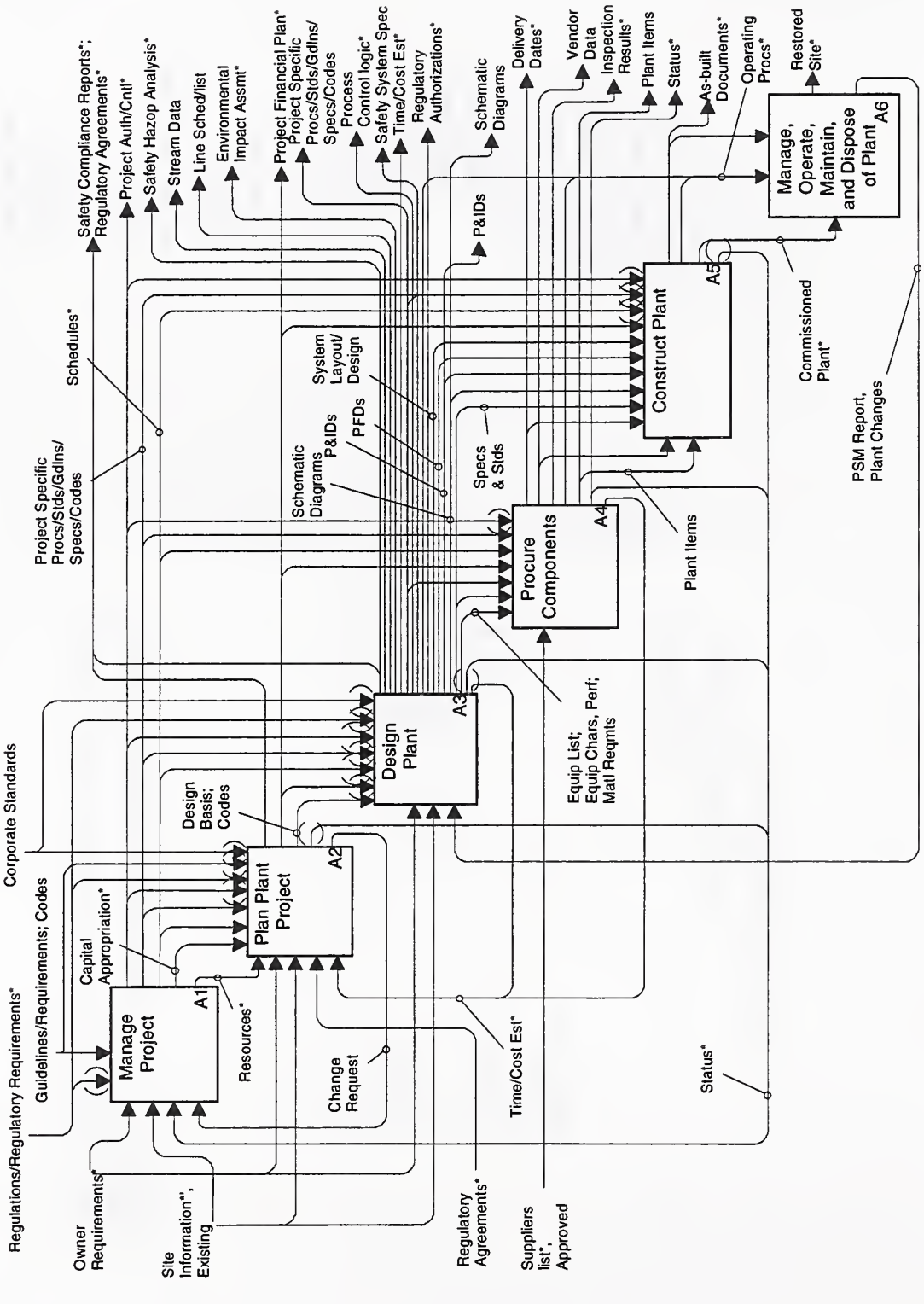
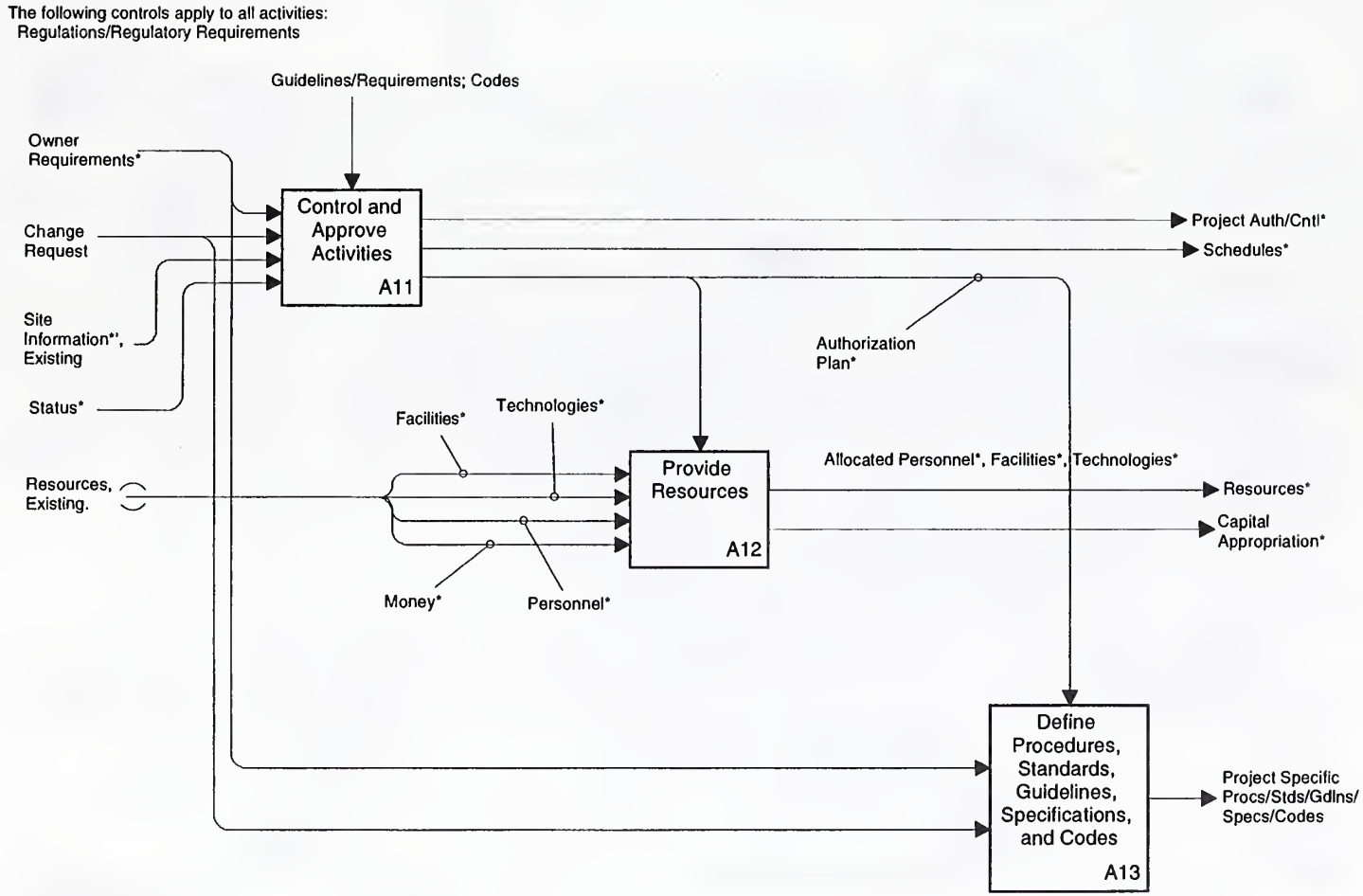


Figure F.2 - A-0: Process Plants in IDEF0



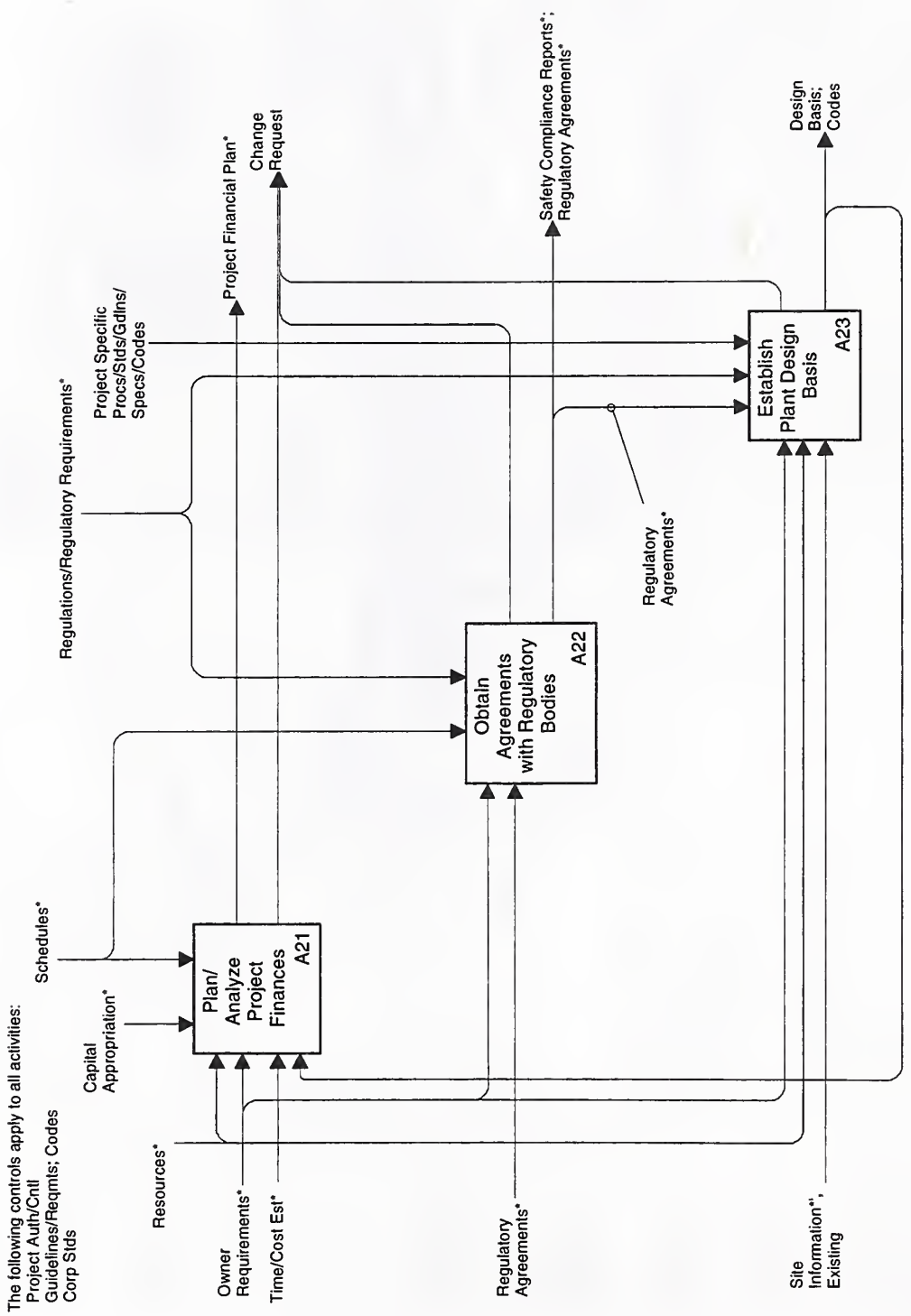
A0 - Plan, Construct, and Operate Plant

Figure F.3 - A0: Plan, Construct, and Operate Plant in IDEF0



A1 - Manage Project

Figure F.4 - A1: Manage Project in IDEF0



A2 - Plan Plant Project

Figure F.5 - A2: Plan Plant Project in IDEF0

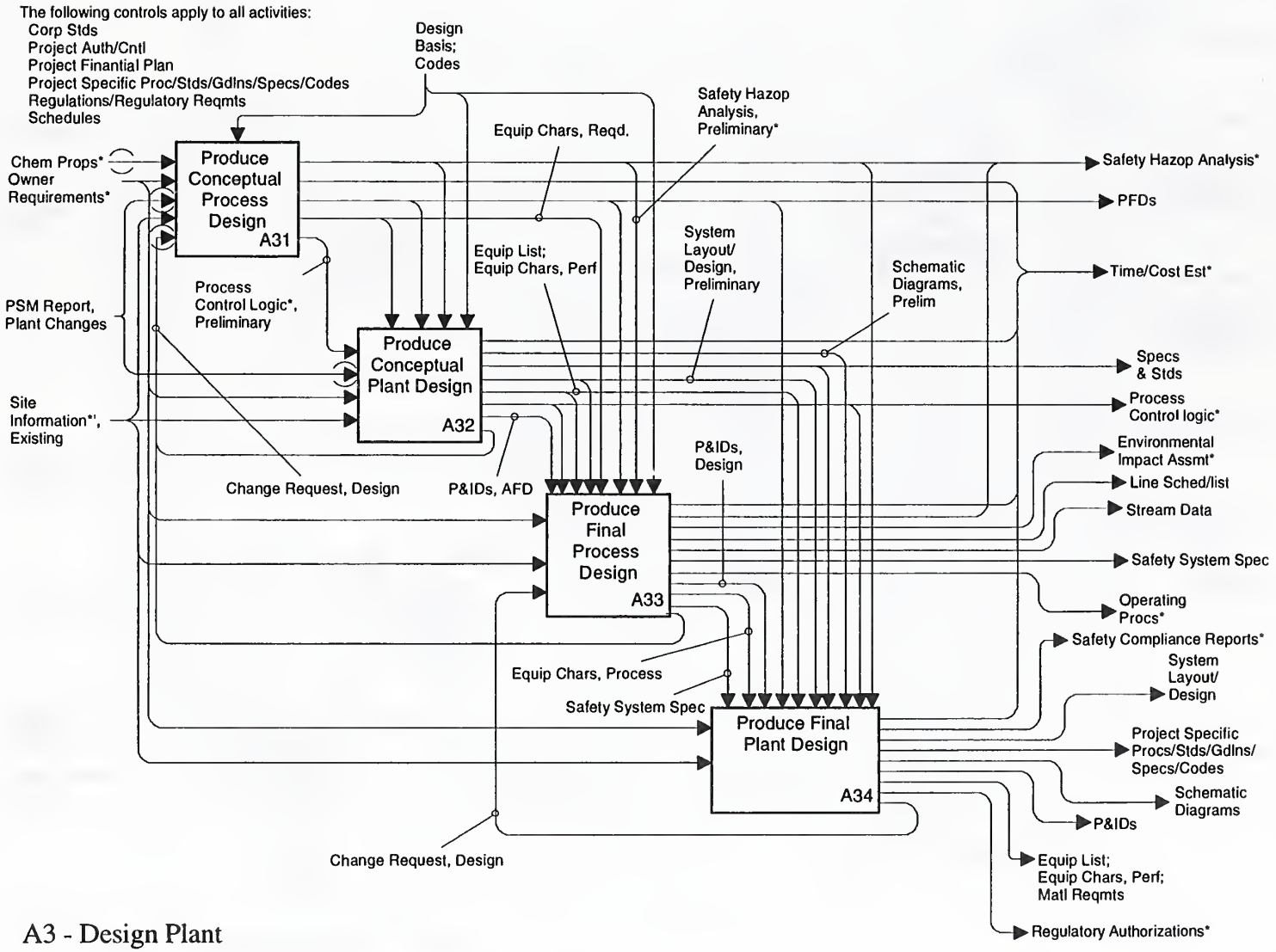


Figure F.6 - A3: Design Plant in IDEF0

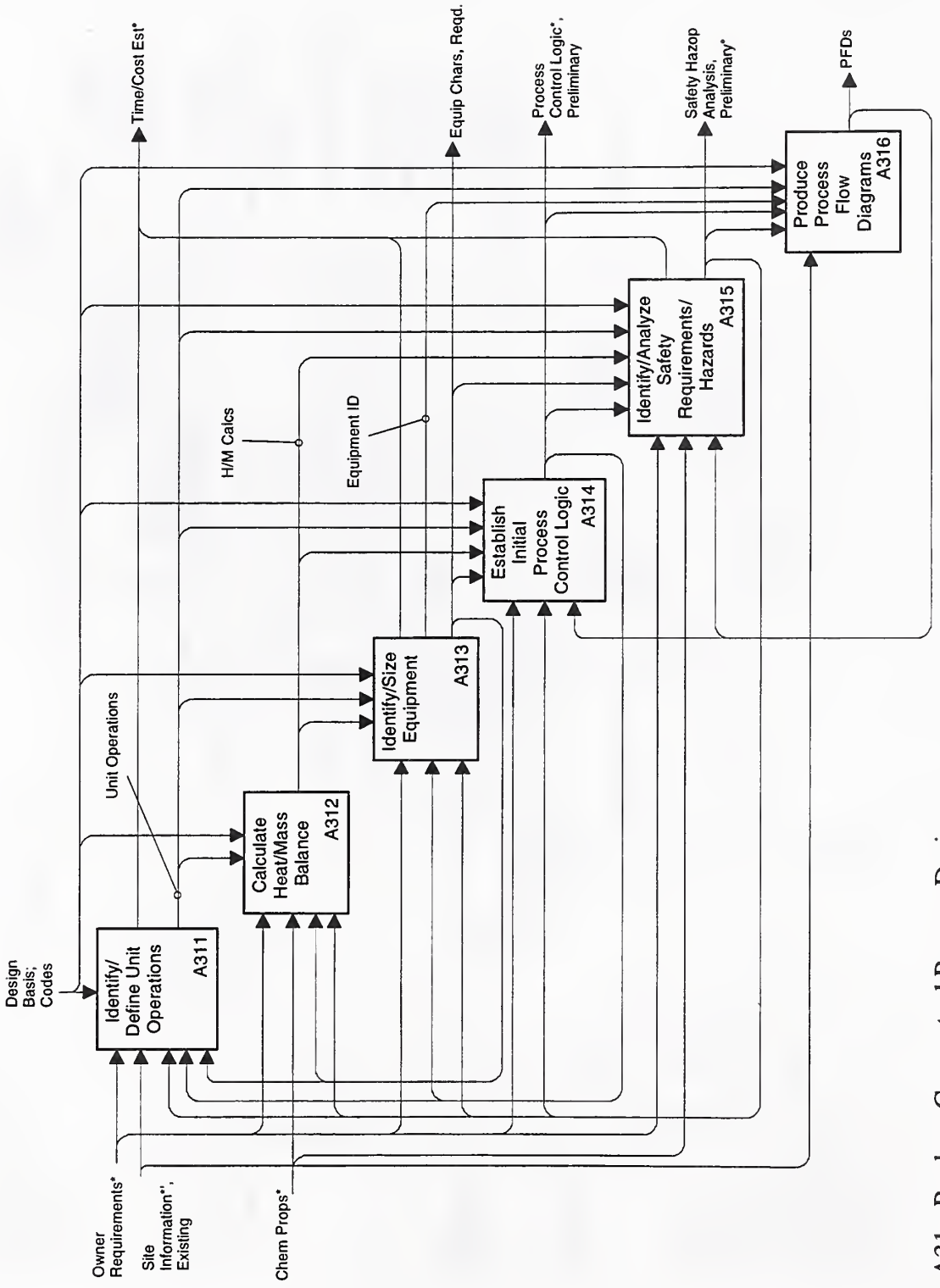
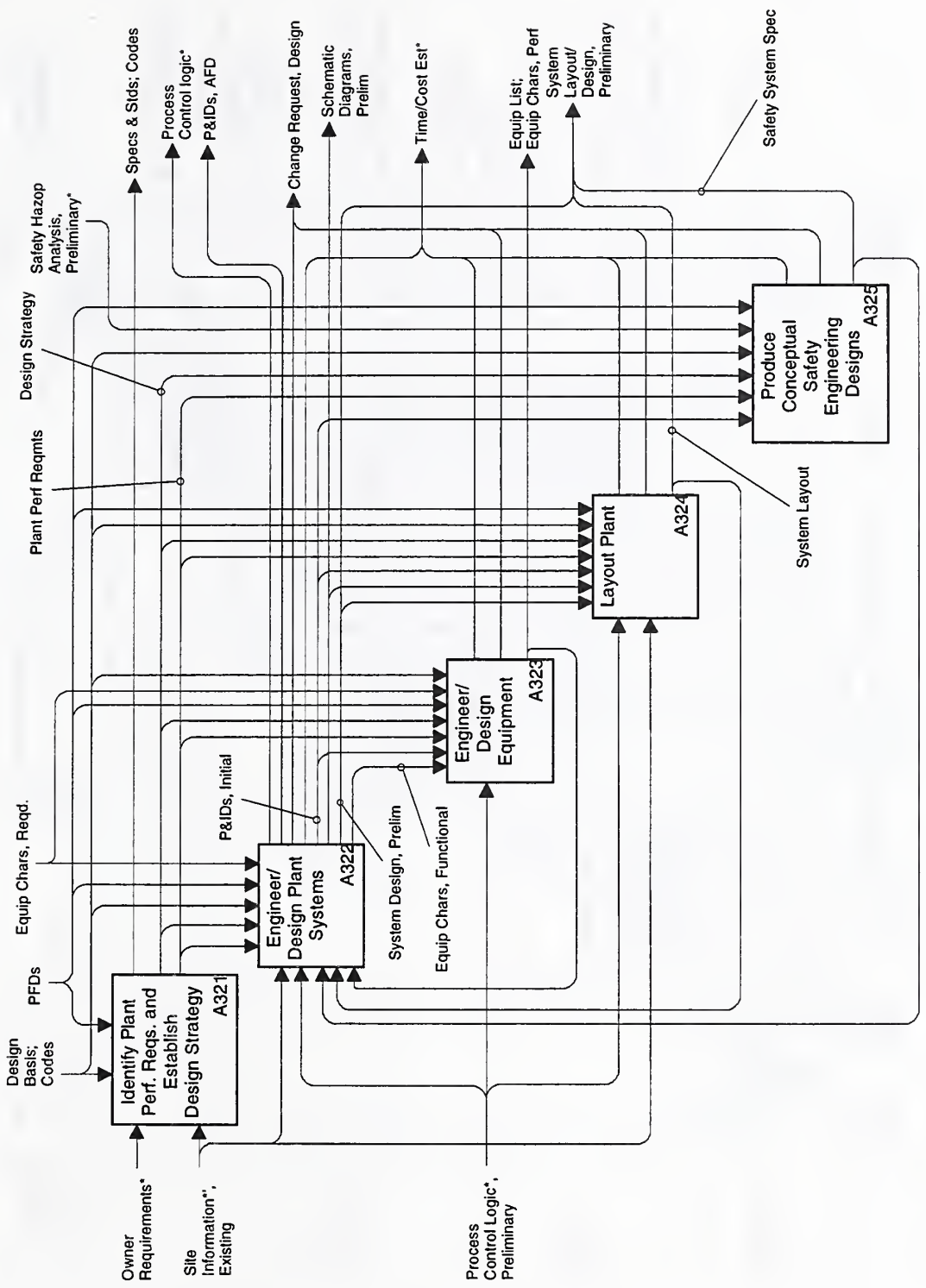


Figure F.7 - A31: Produce Conceptual Process Design in IDEF0



A32 - Produce Conceptual Plant Design

Figure F.8 - A32: Produce Conceptual Plant Design in IDEF0

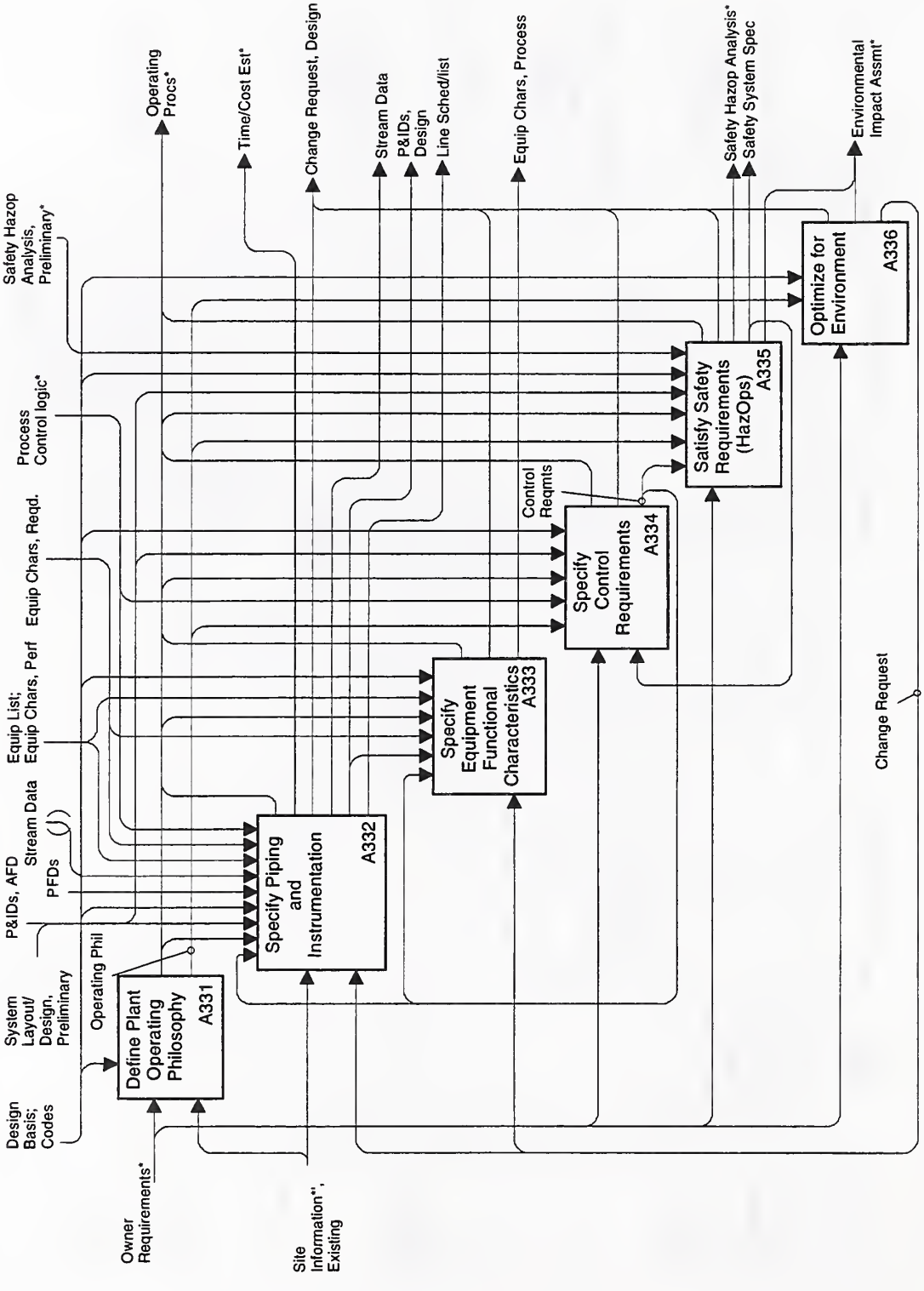
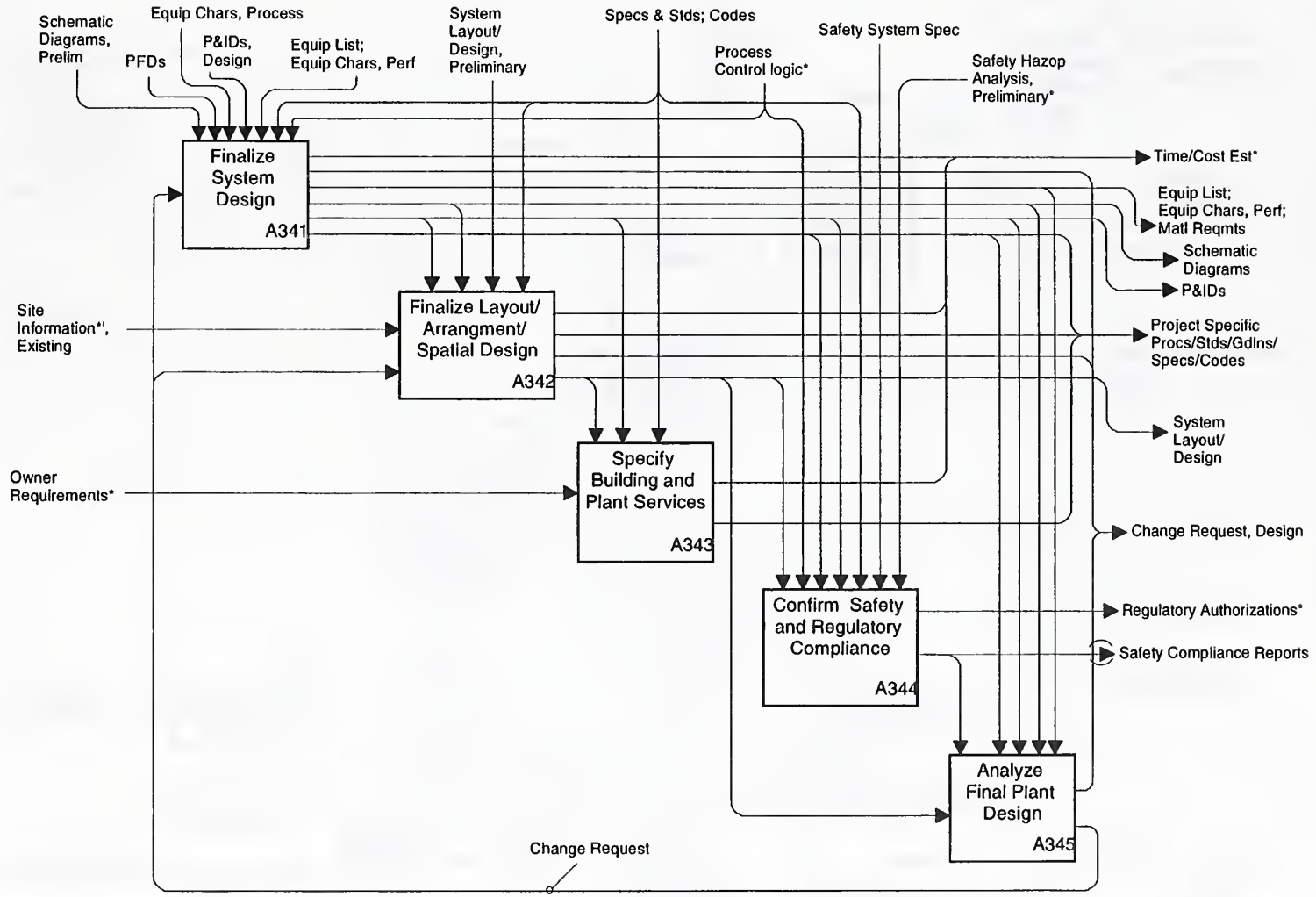
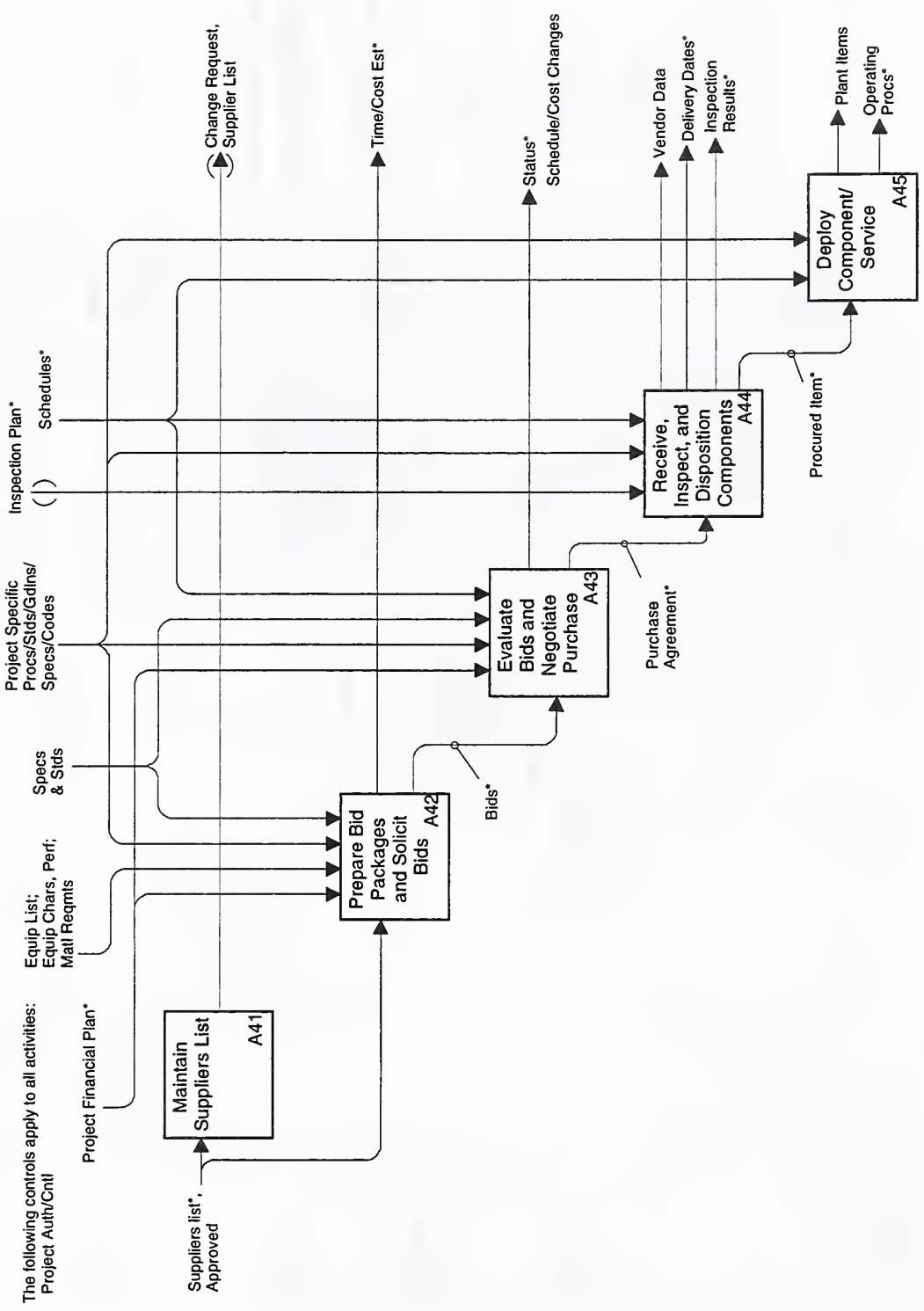


Figure F.9 - A33: Produce Final Process Design in IDEF0



A34 - Produce Final Plant Design

Figure F.10 - A34: Produce Final Plant Design in IDEF0



A4 - Procure Component

Figure F.11 - A4: Procure Component in IDEF0

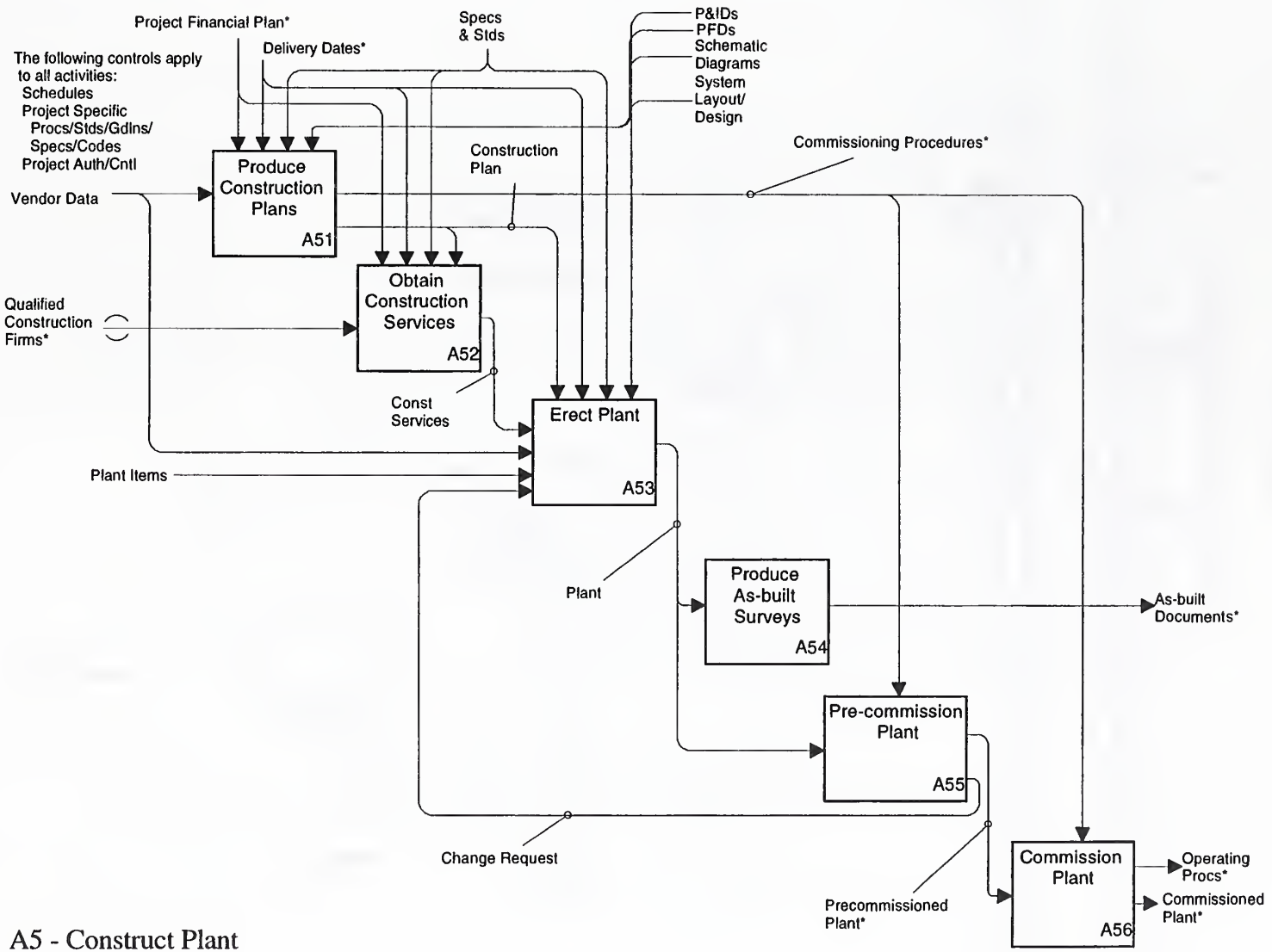
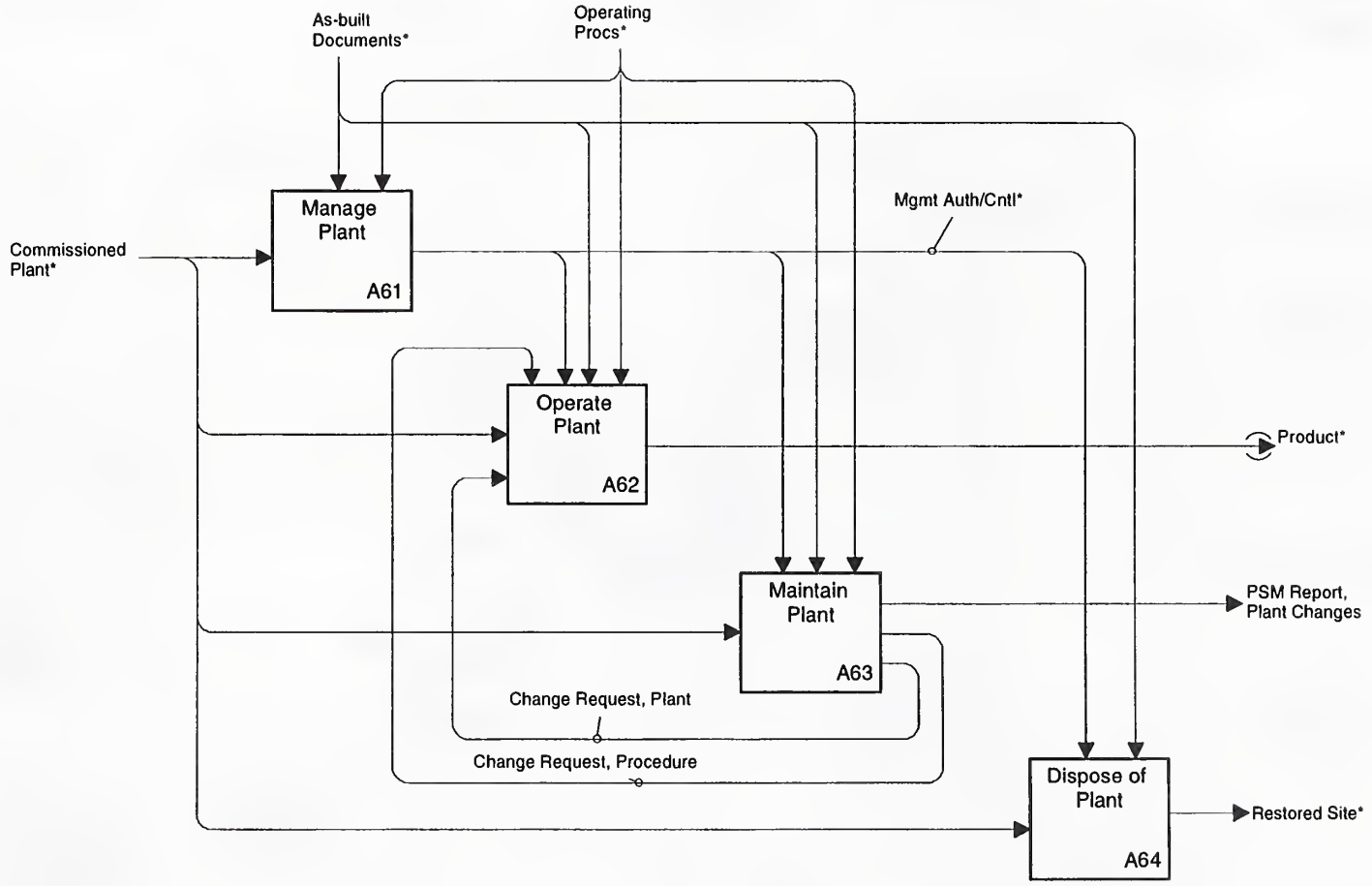


Figure F.12 - A5: Construct Plant in IDEF0

ISO/CD 10303-227:1995(E)



A6 - Manage, Operate, Maintain, and Dispose of Plant

Figure F.13 - A6: Manage, Operate, Maintain, and Dispose of Plant in IDEF0

Annex G
(informative)

Application reference model

This annex provides the application reference model for this part of ISO 10303. The application reference model is a graphical representation of the structure and constraints of the application objects specified in clause 4. The application reference model is independent from any implementation method.

The graphical form of the application reference model is presented in IDEF1X. Extensions to the IDEF1X notation are used within the diagrams. The graphical extensions are based on EXPRESS-G off page connectors.

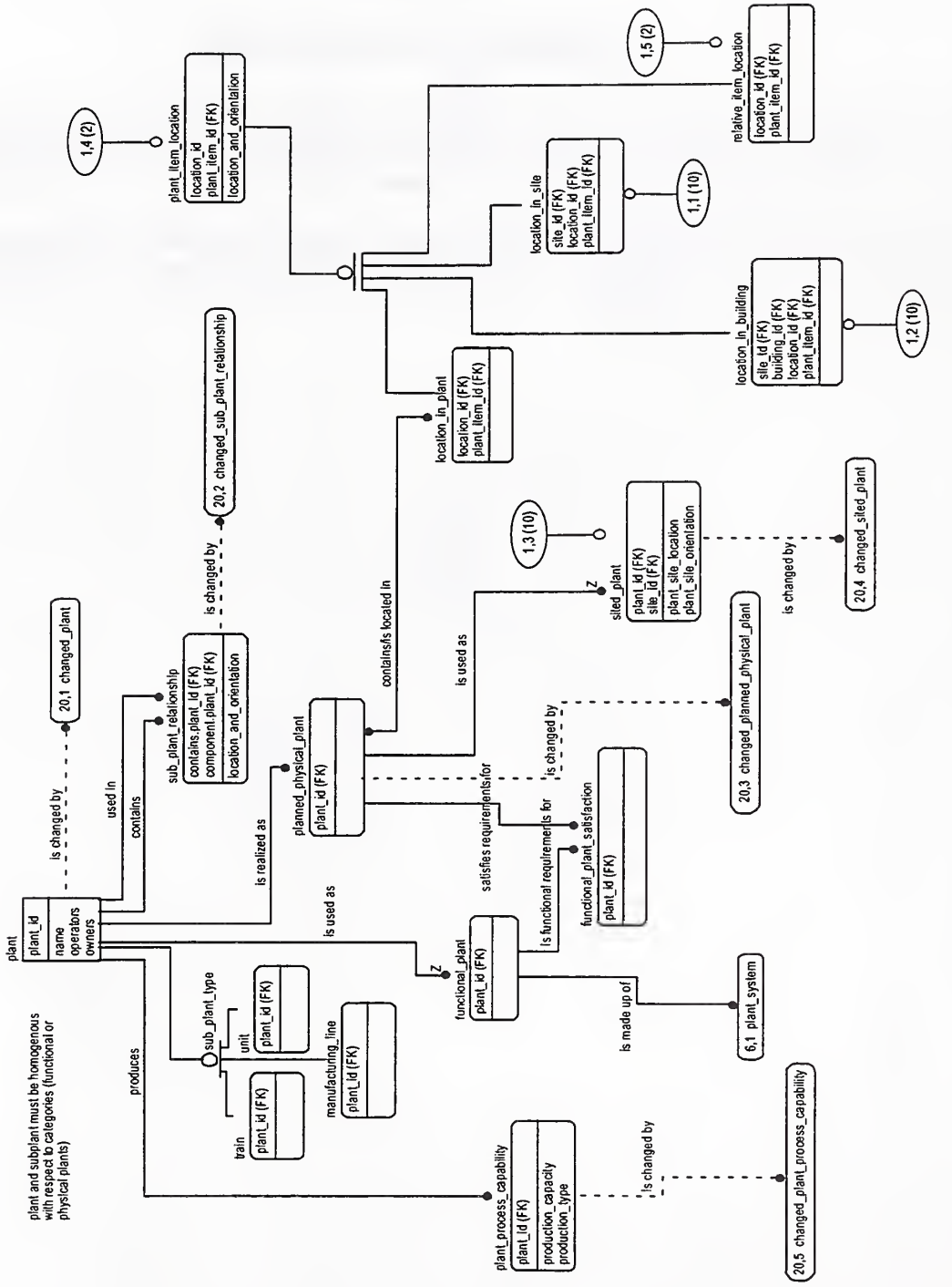


Figure G.1 - ARM diagram 1 of 25 in IDEF1X

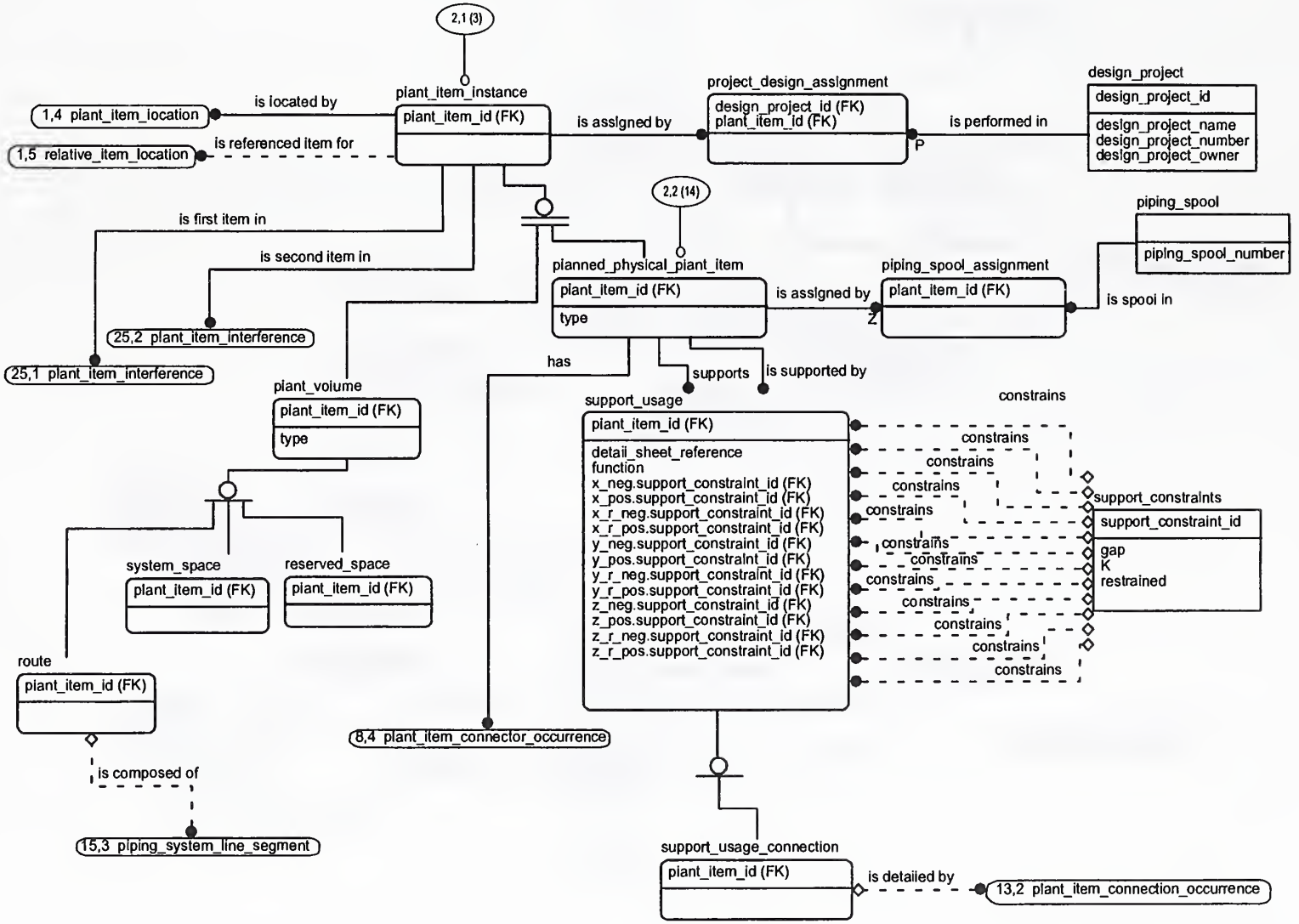


Figure G.2 - ARM diagram 2 of 25 in IDEF1X

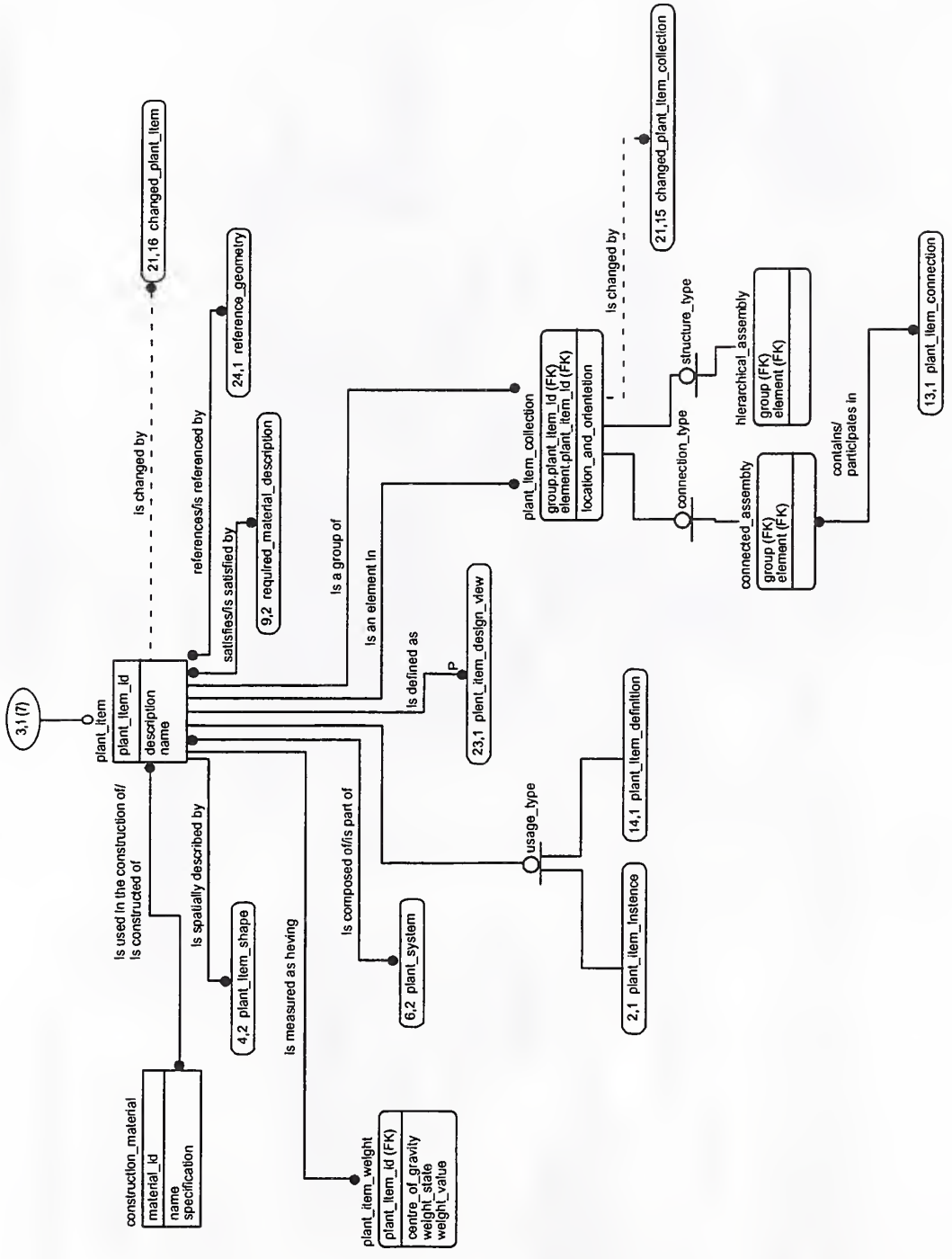


Figure G.3 - ARM diagram 3 of 25 in IDEF1X

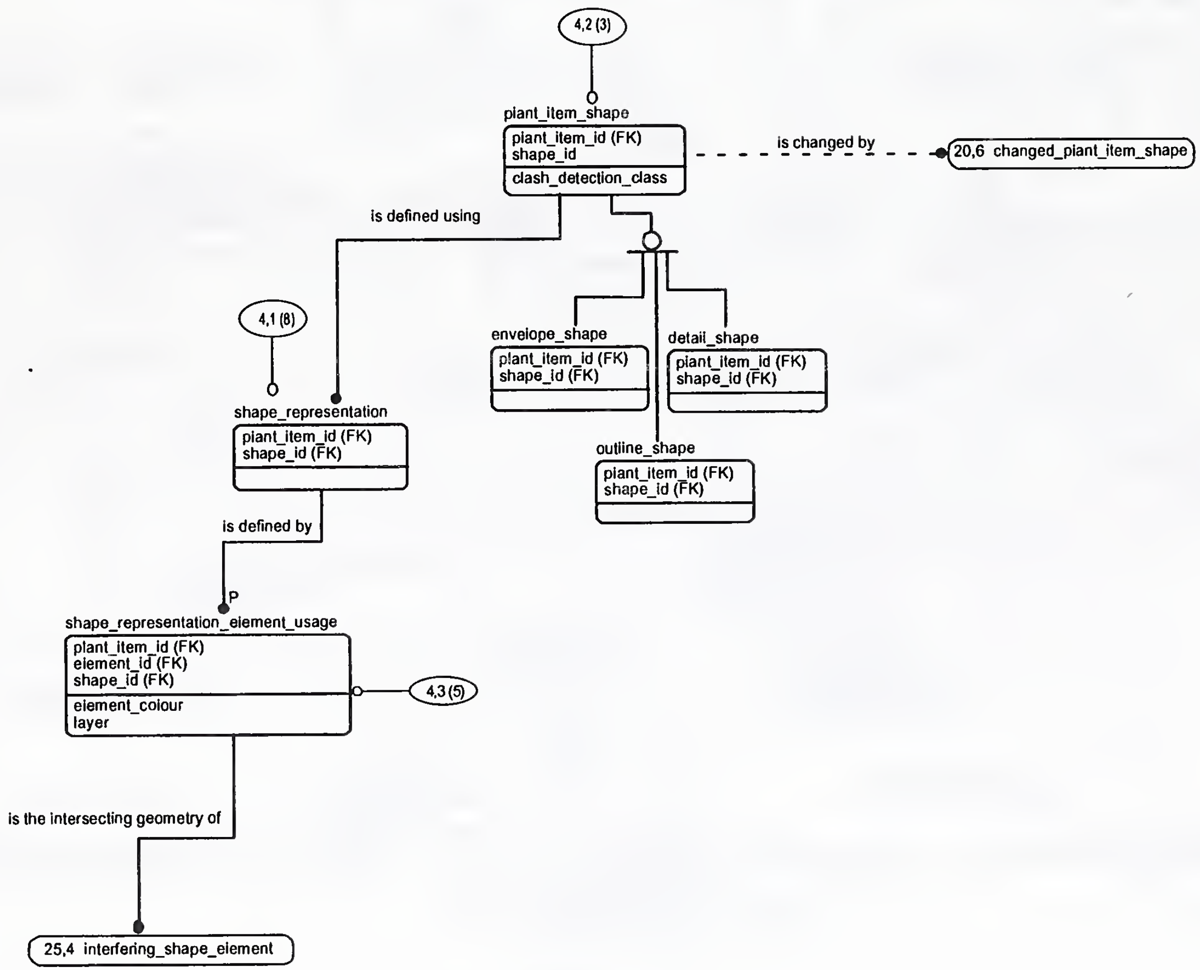


Figure G.4 - ARM diagram 4 of 25 in IDEF1X

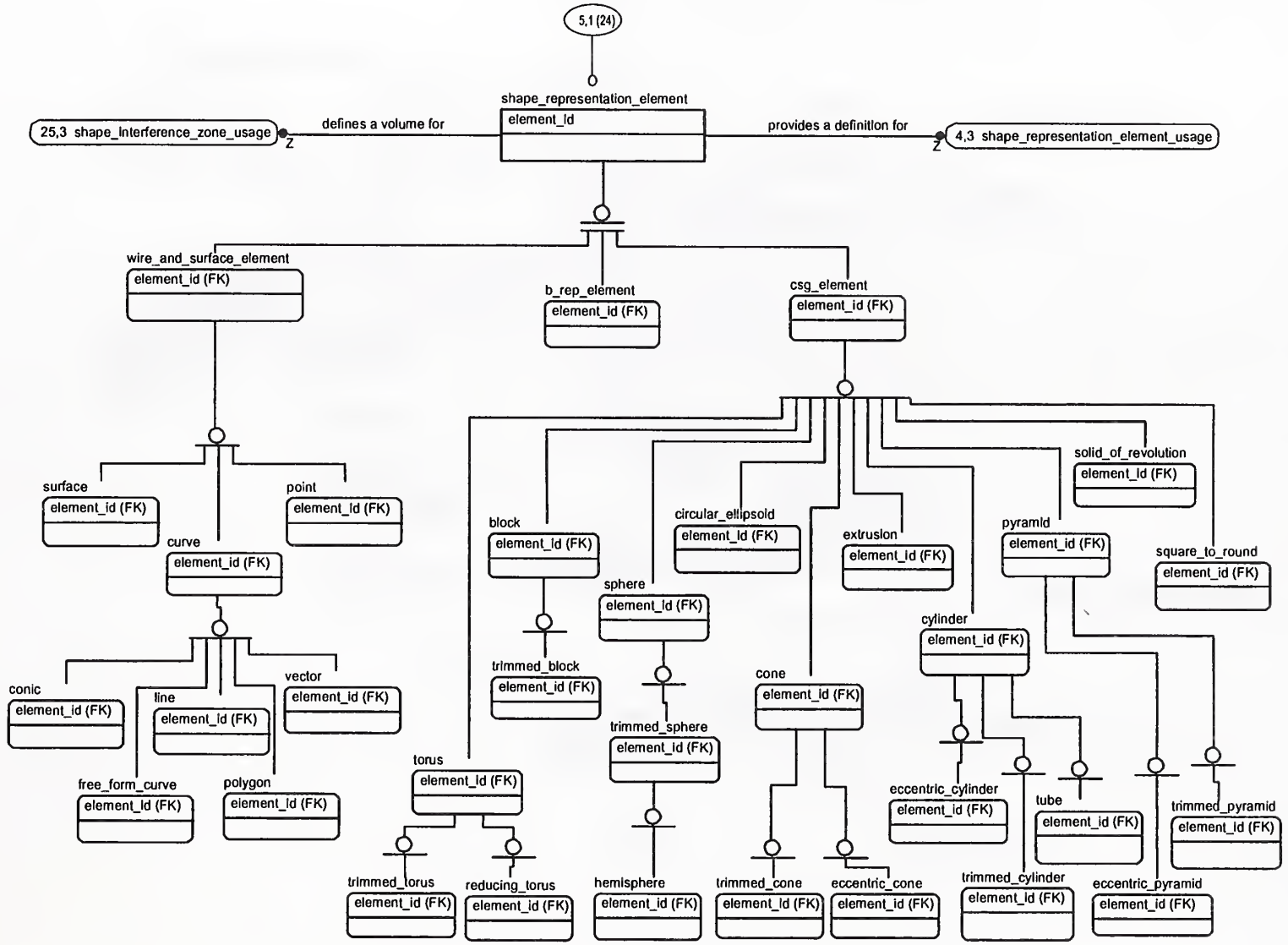


Figure G.5 - ARM diagram 5 of 25 in IDEF1X

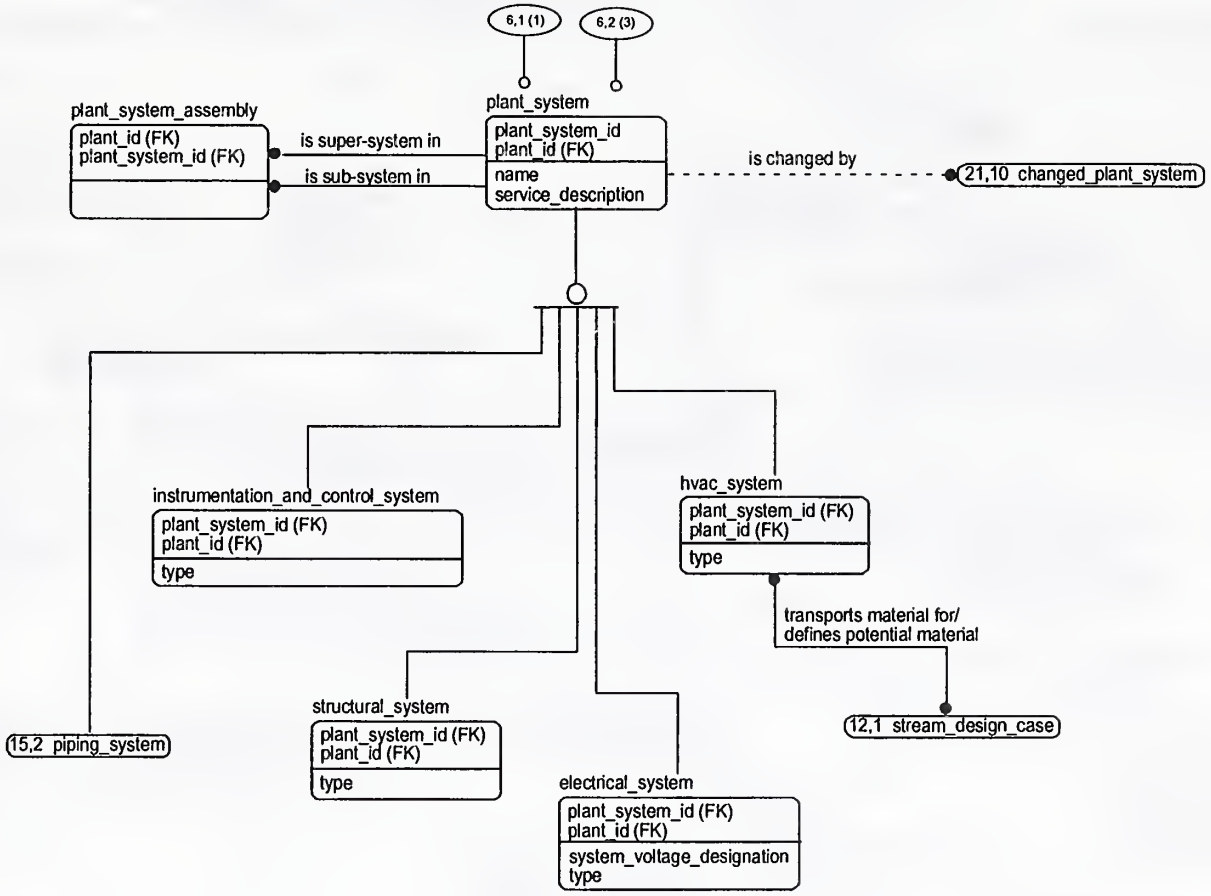


Figure G.6 - ARM diagram 6 of 25 in IDEF1X

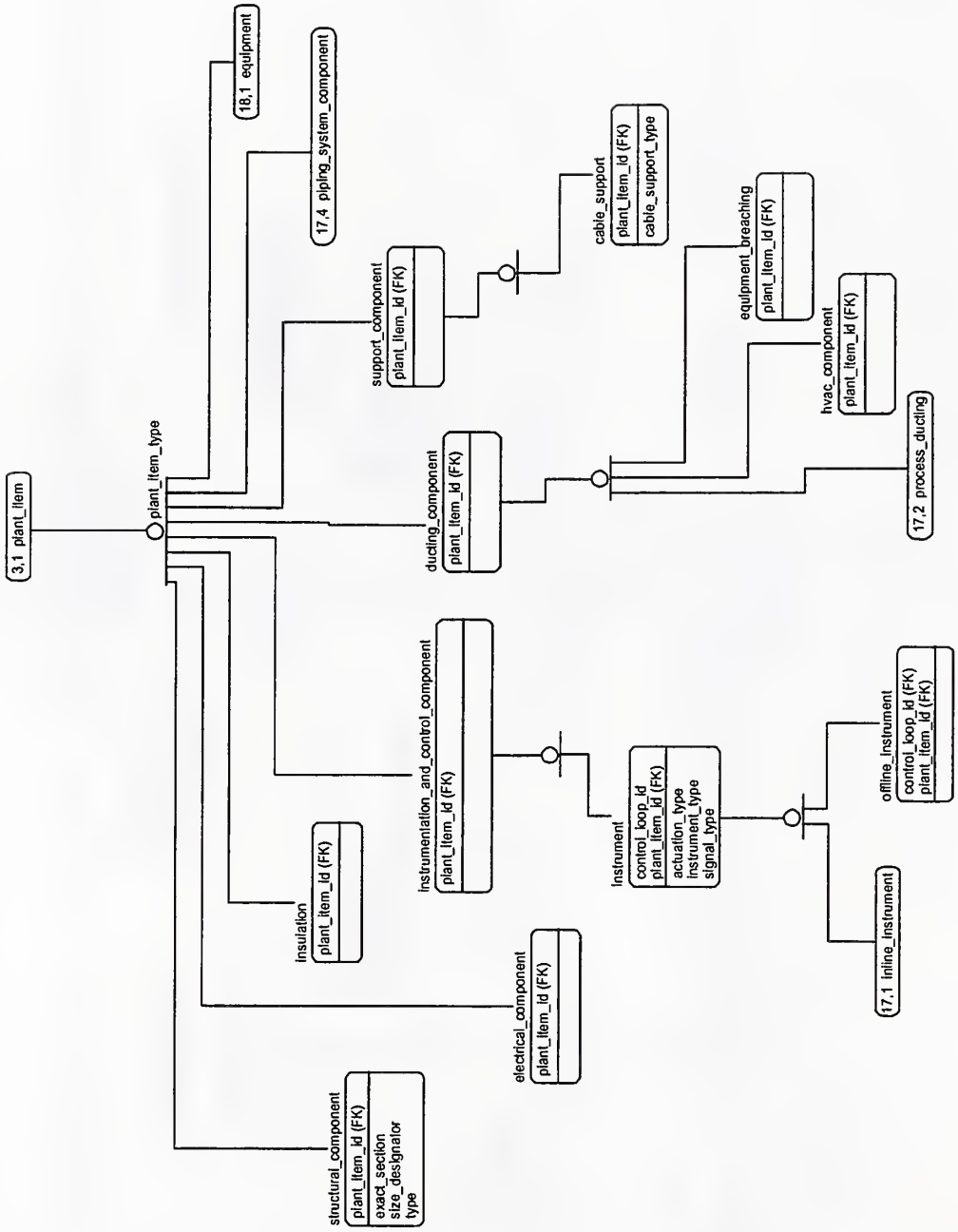


Figure G.7 - ARM diagram 7 of 25 in IDEF1X

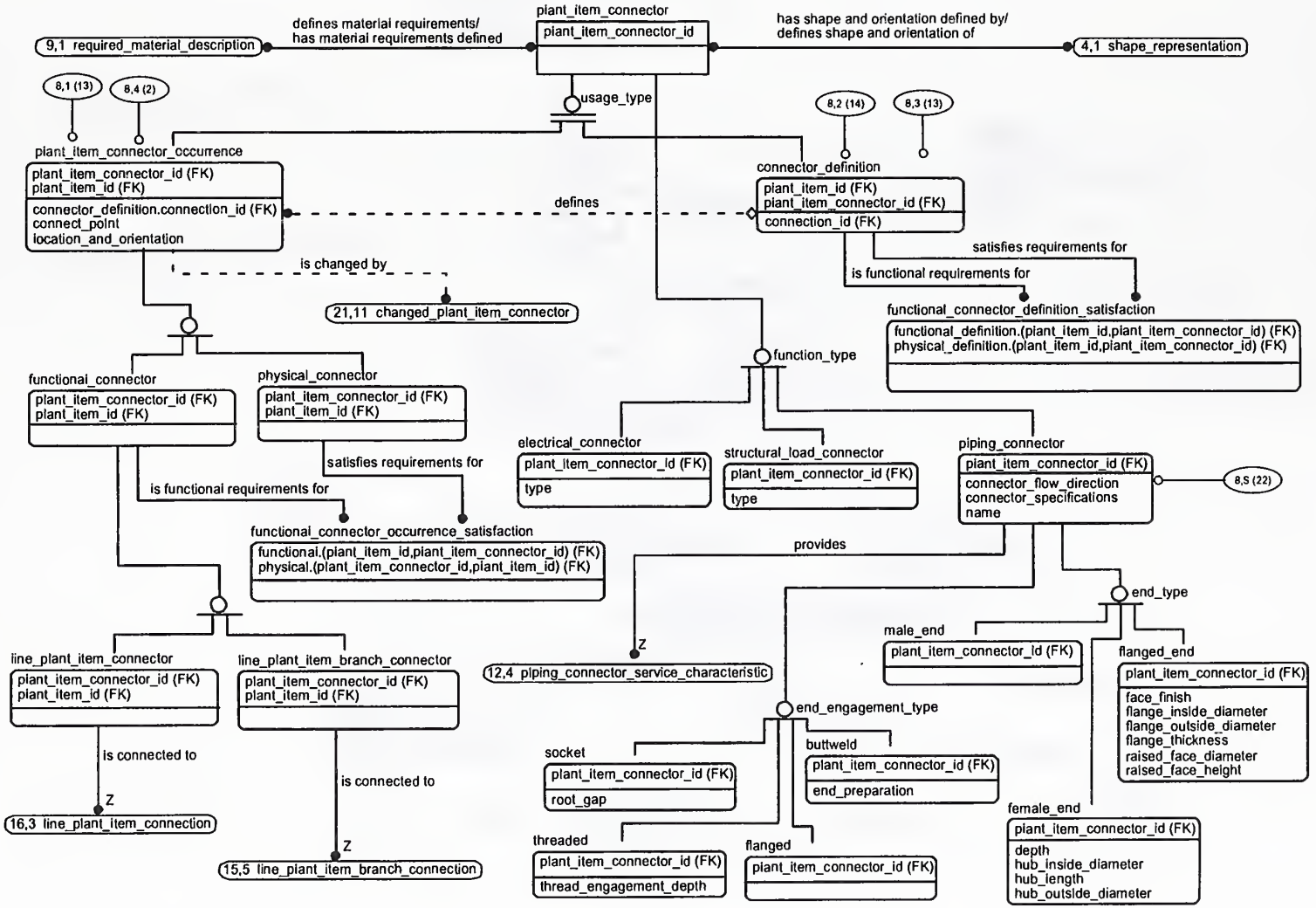


Figure G.8 - ARM diagram 8 of 25 in IDEF1X

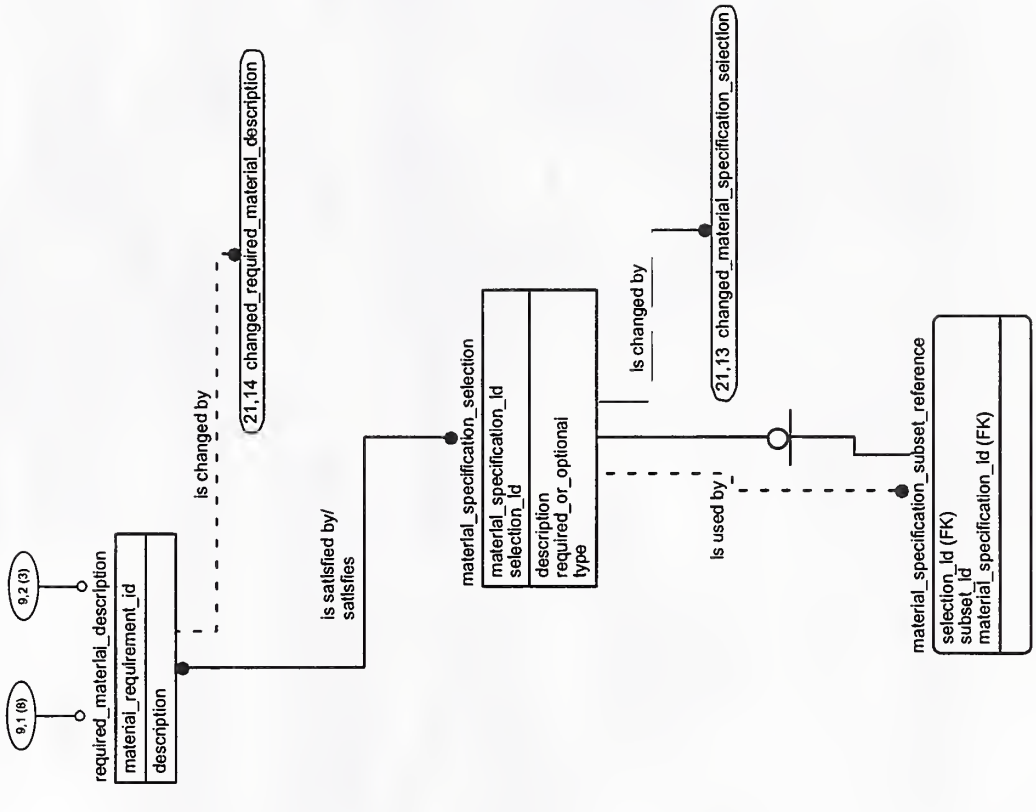


Figure G.9 - ARM diagram 9 of 25 in IDEF1X

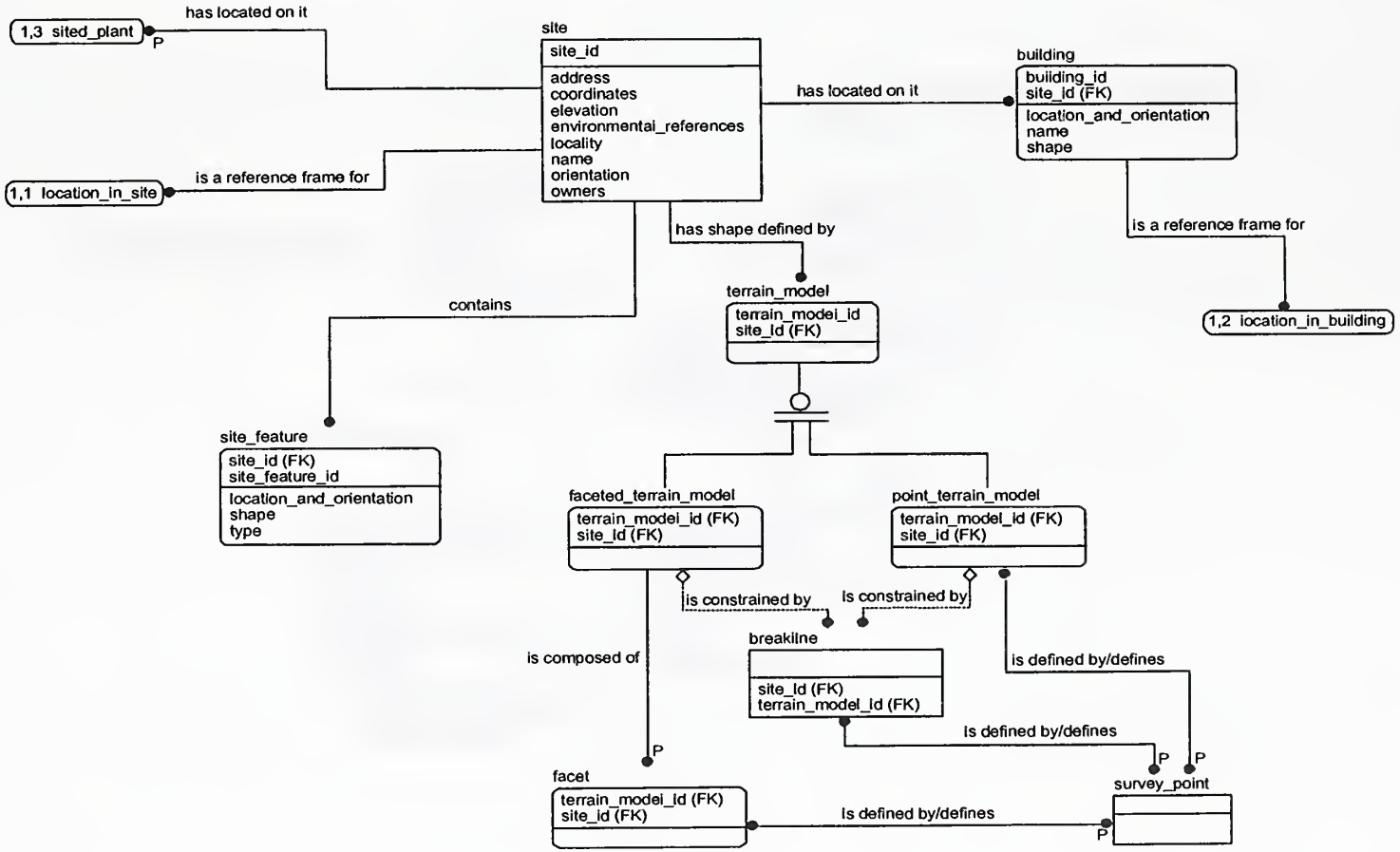


Figure G.10 - ARM diagram 10 of 25 in IDEF1X

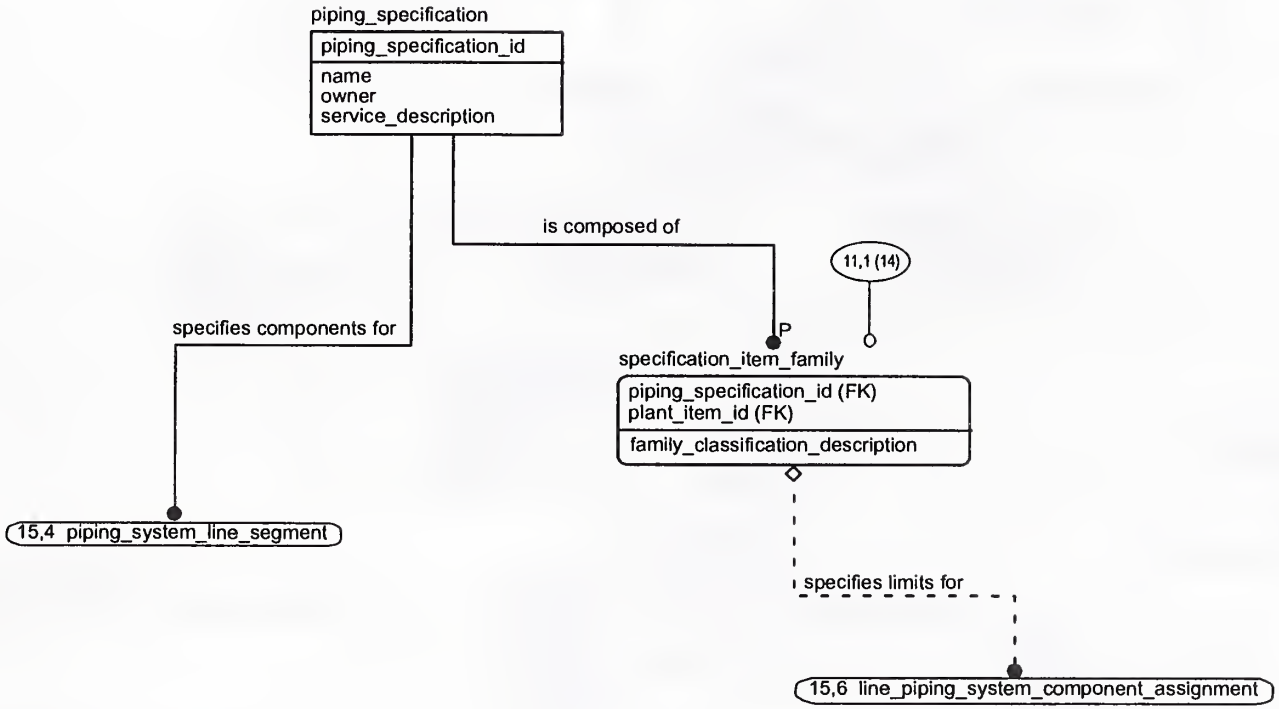


Figure G.11 - ARM diagram 11 of 25 in IDEF1X

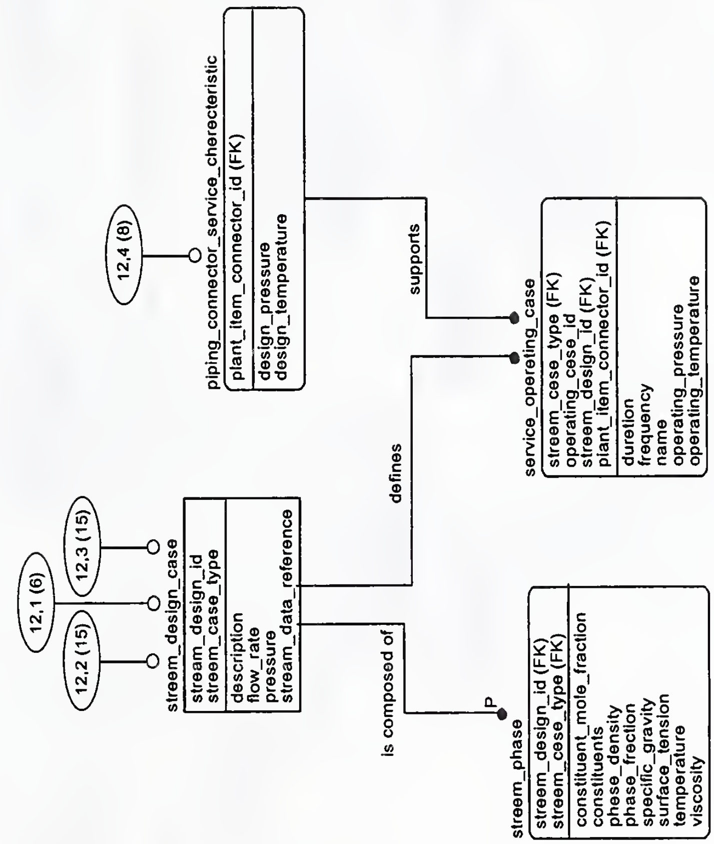


Figure G.12 - ARM diagram 12 of 25 in IDEF1X

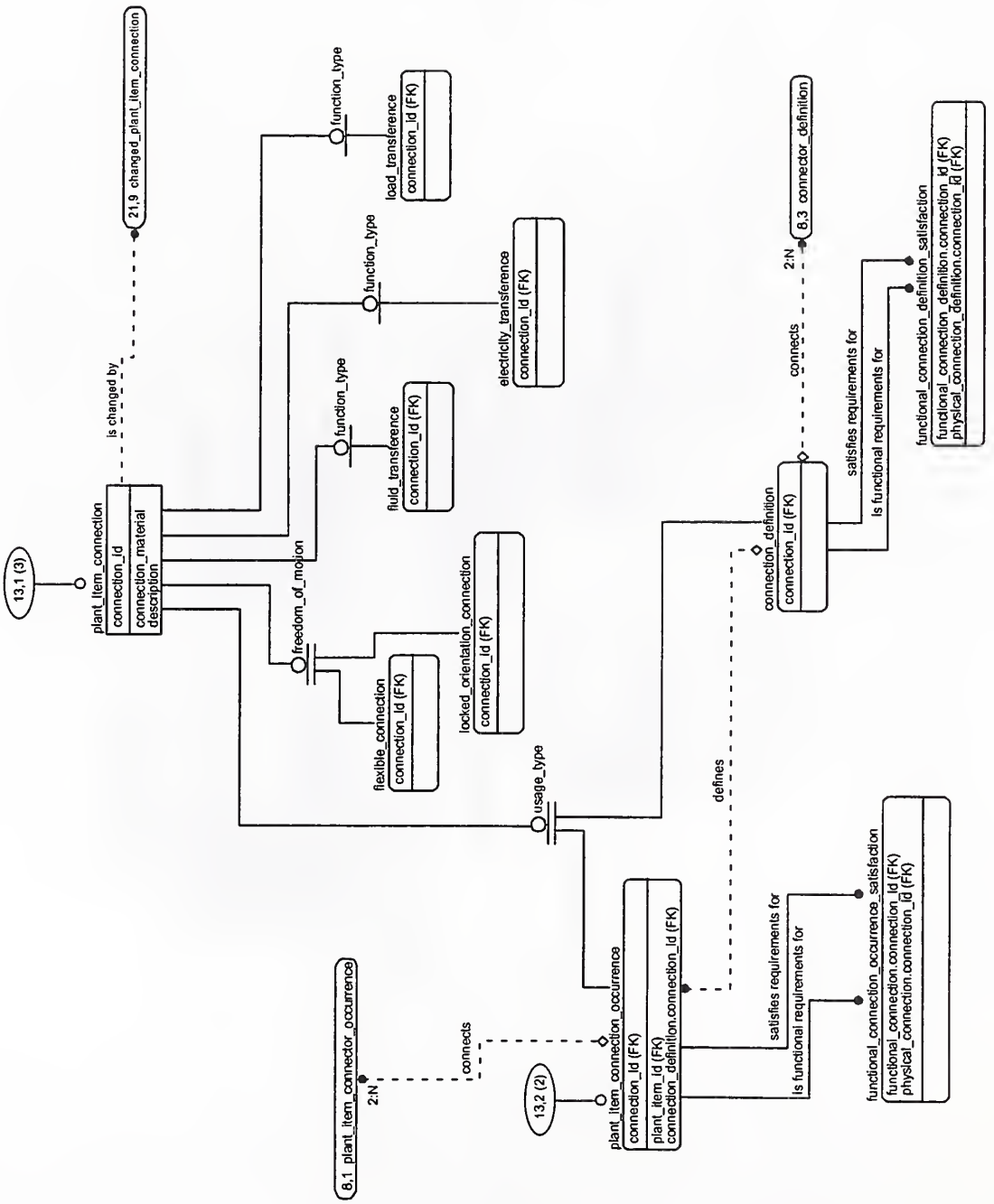


Figure G.13 - ARM diagram 13 of 25 in IDEF1X

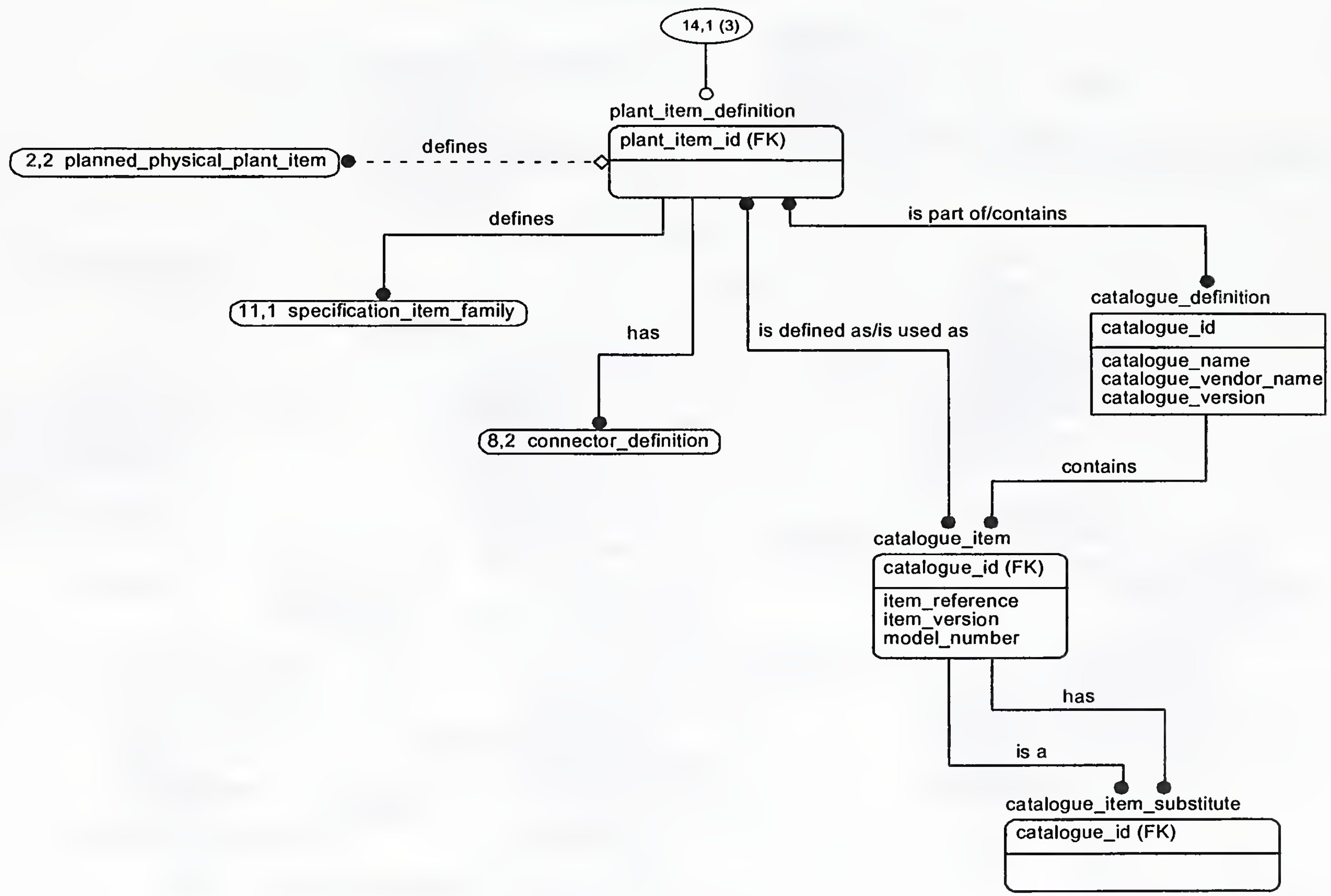


Figure G.14 - ARM diagram 14 of 25 in IDEF1X

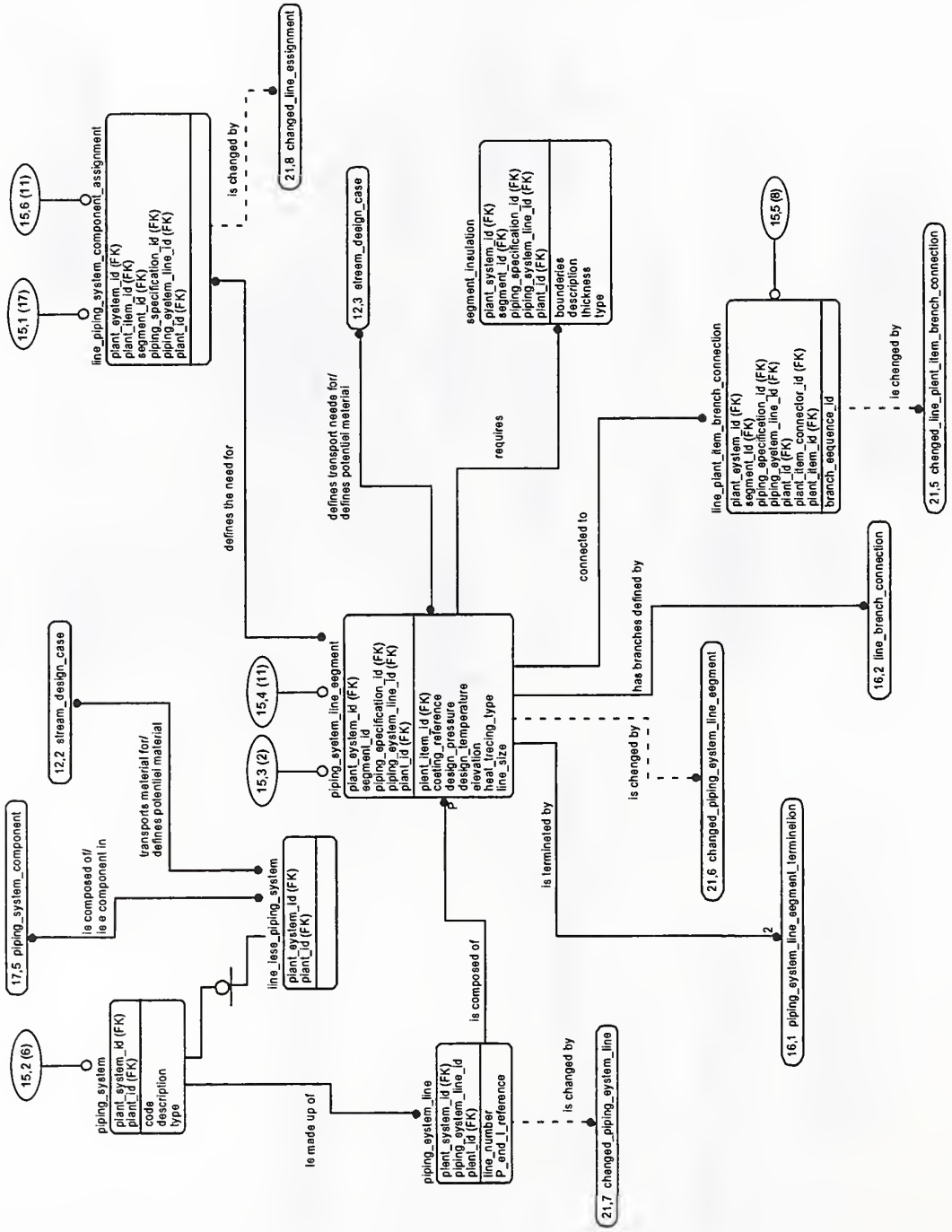


Figure G.15 - ARM diagram 15 of 25 in IDEF1X

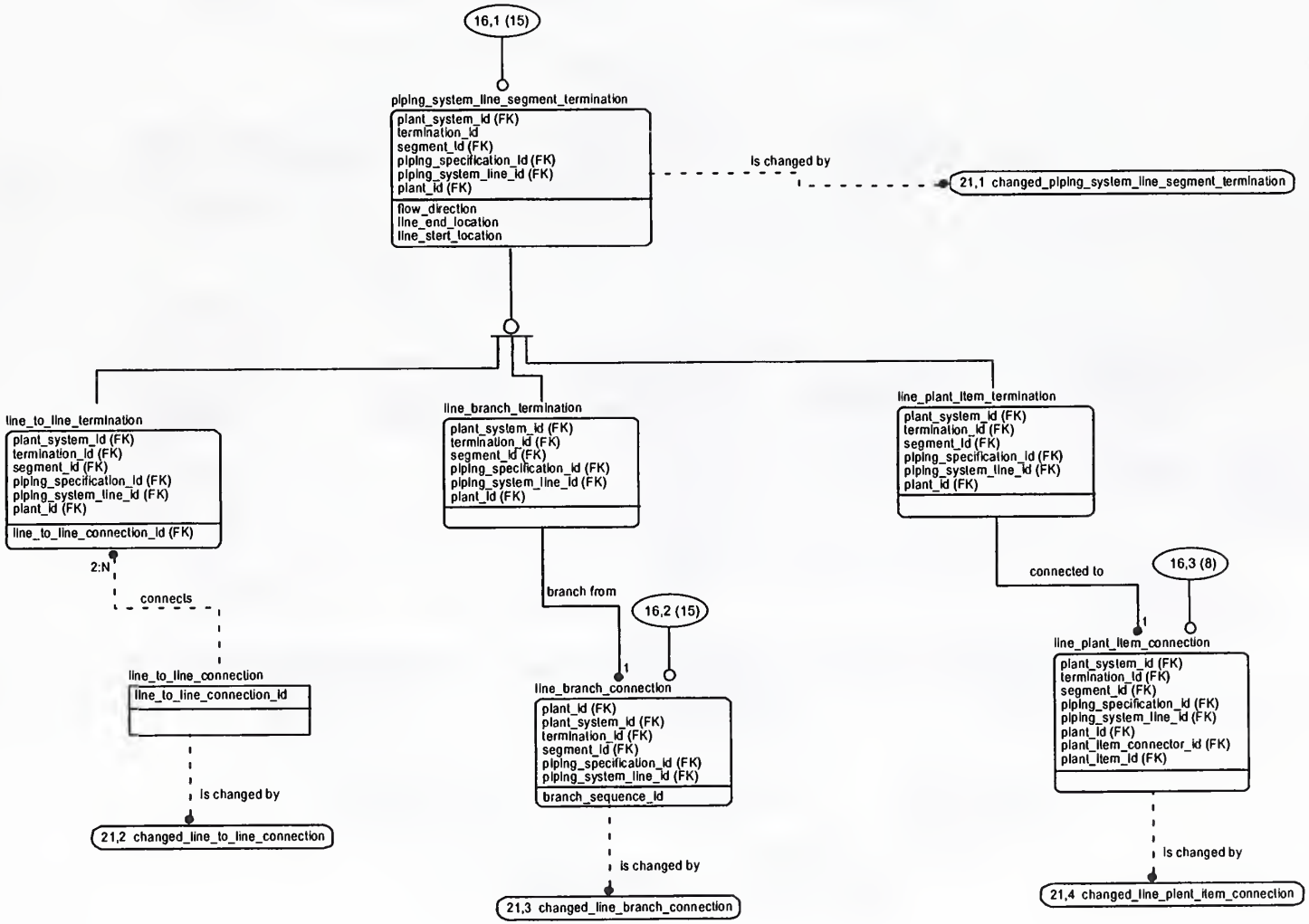


Figure G.16 - ARM diagram 16 of 25 in IDEF1X

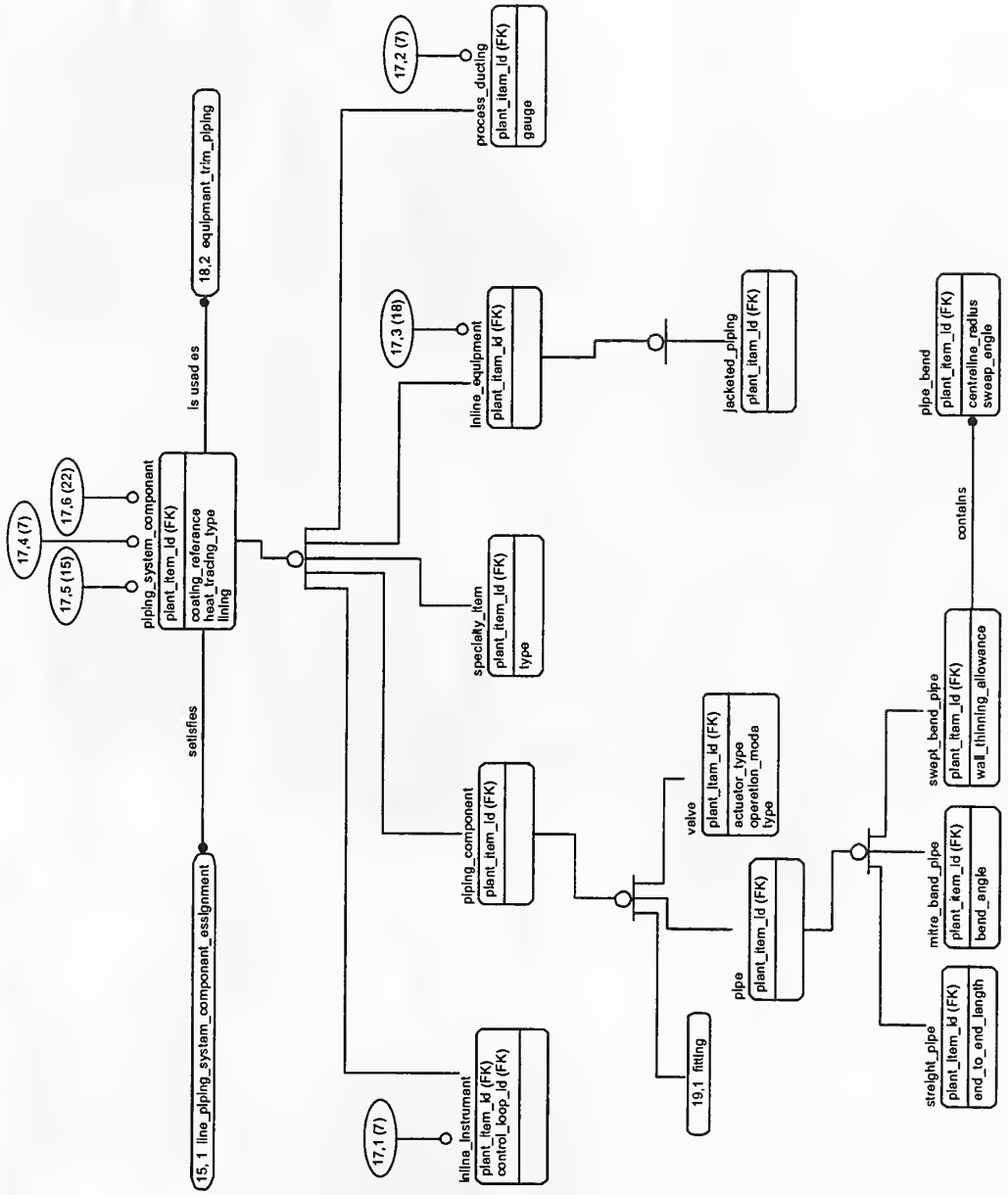


Figure G.17 - ARM diagram 17 of 25 in IDEF1X

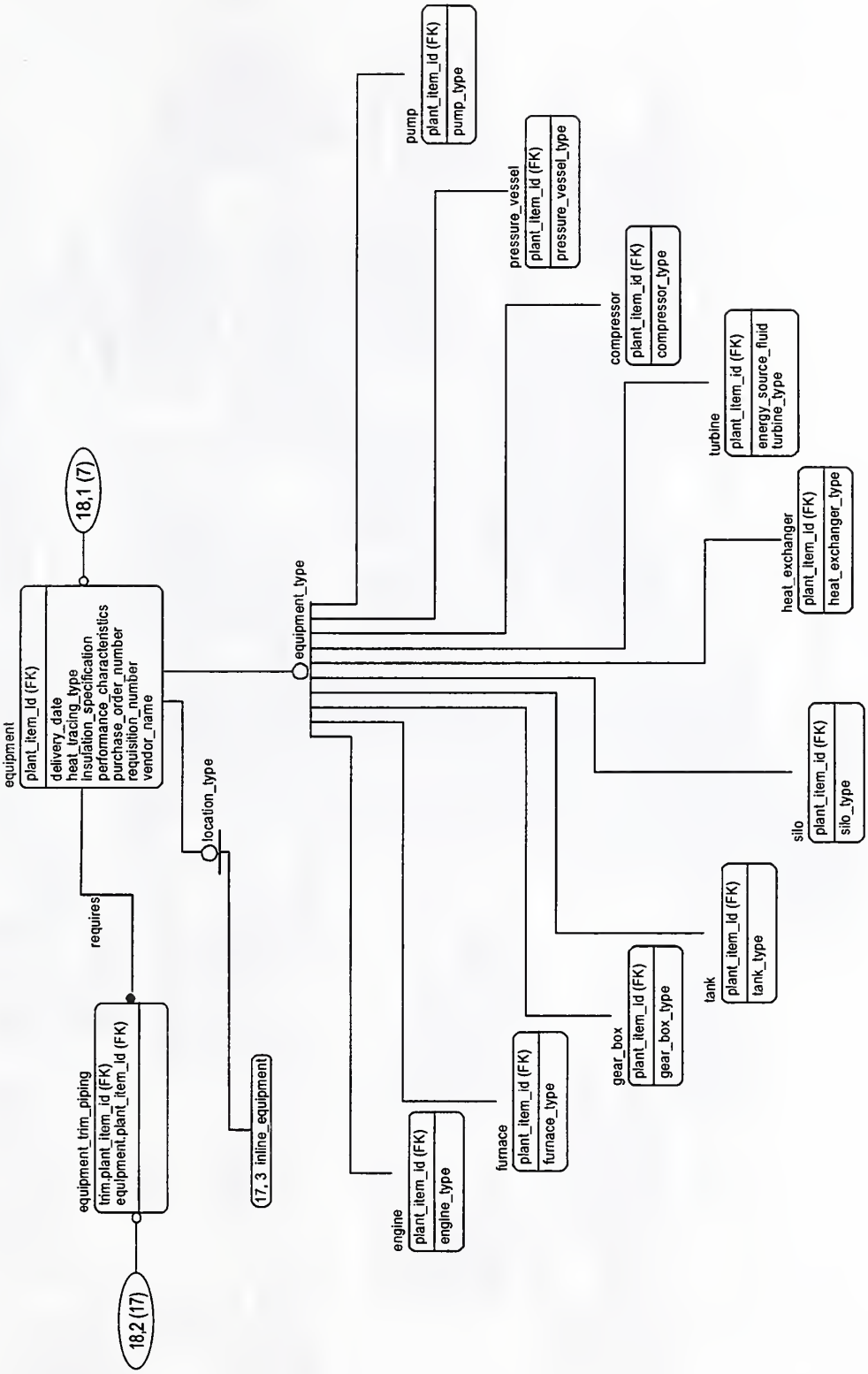


Figure G.18 - ARM diagram 18 of 25 in IDEF1X

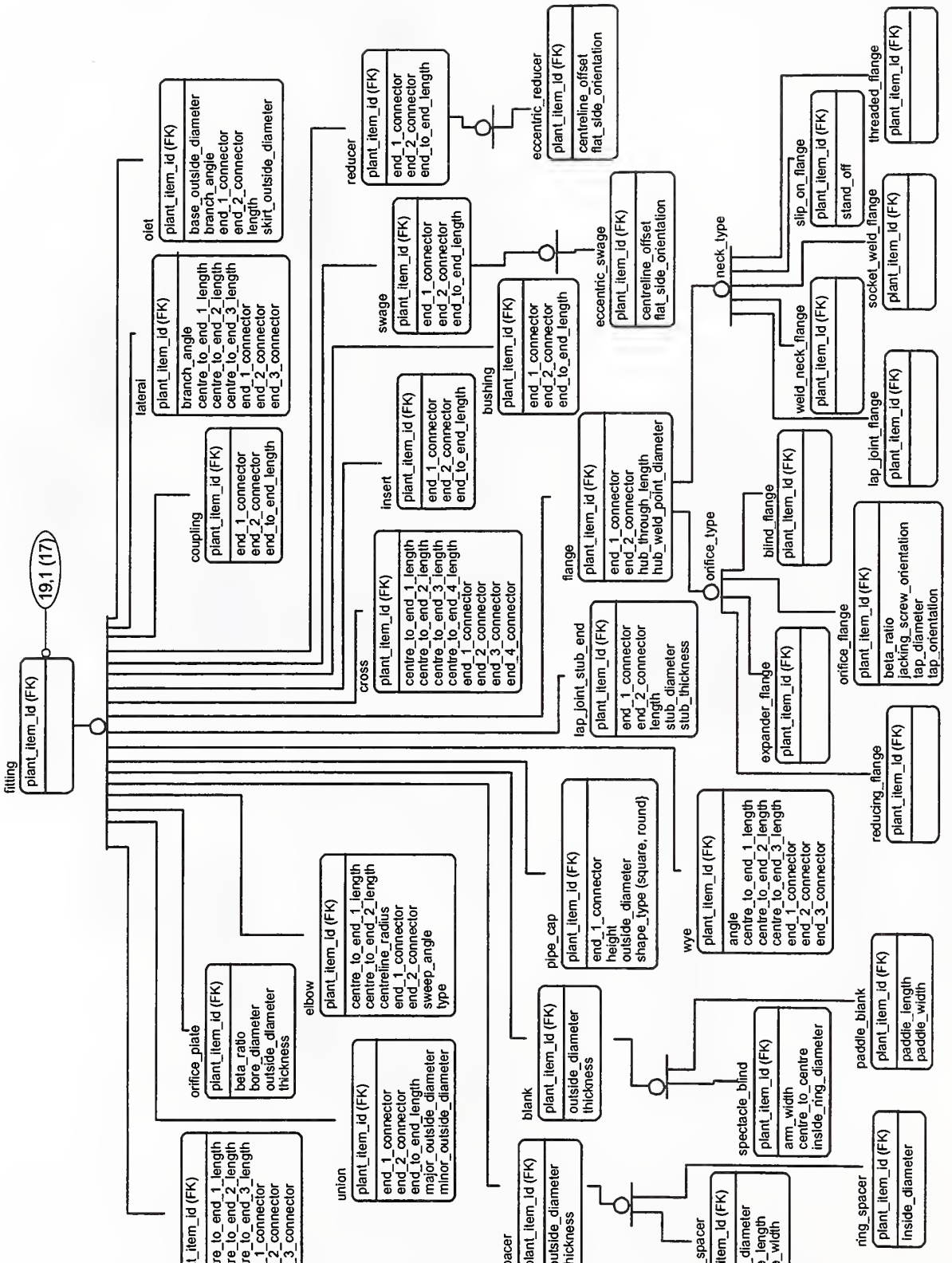


Figure G.19 - ARM diagram 19 of 25 in IDEF1X

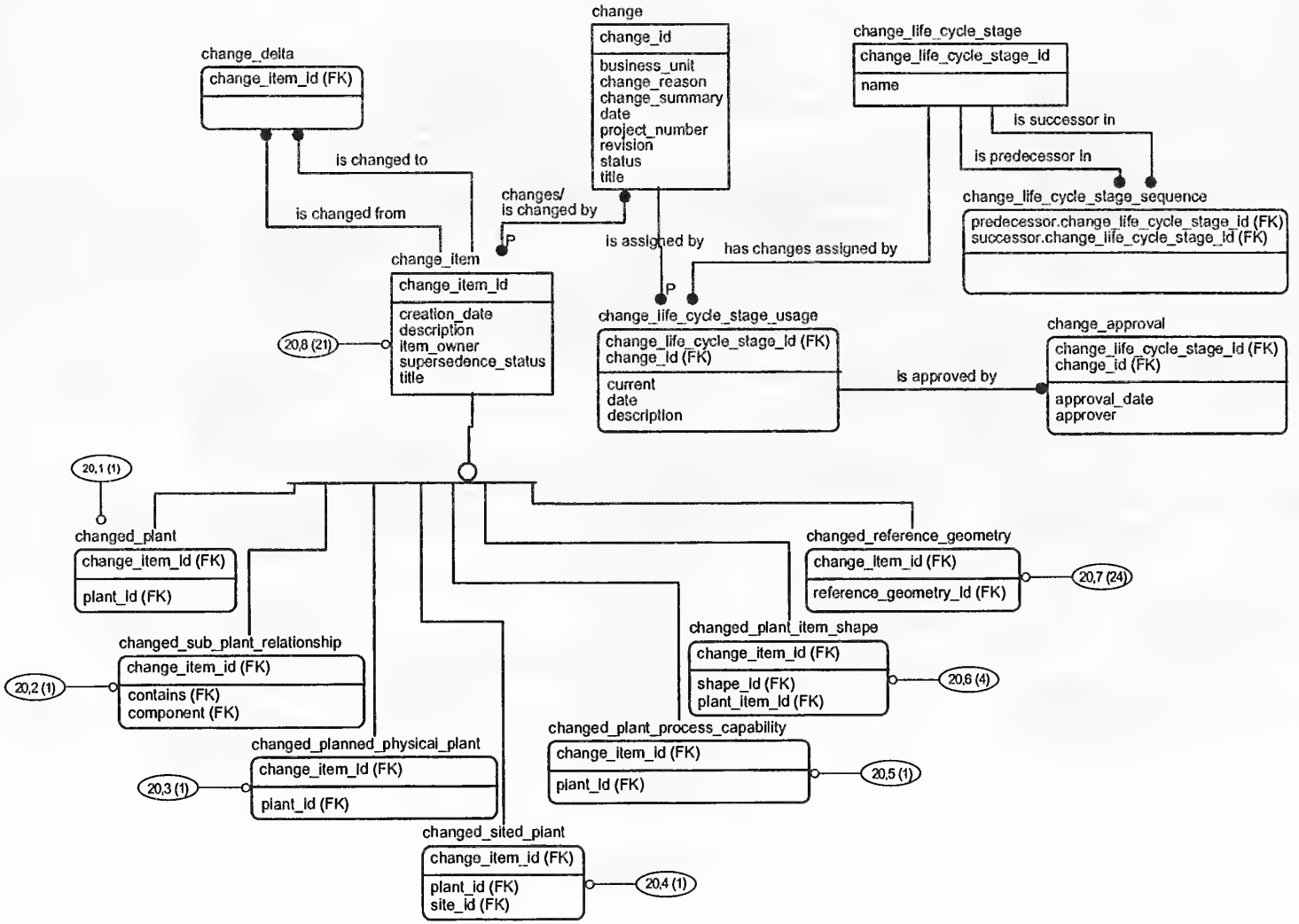


Figure G.20 - ARM diagram 20 of 25 in IDEF1X

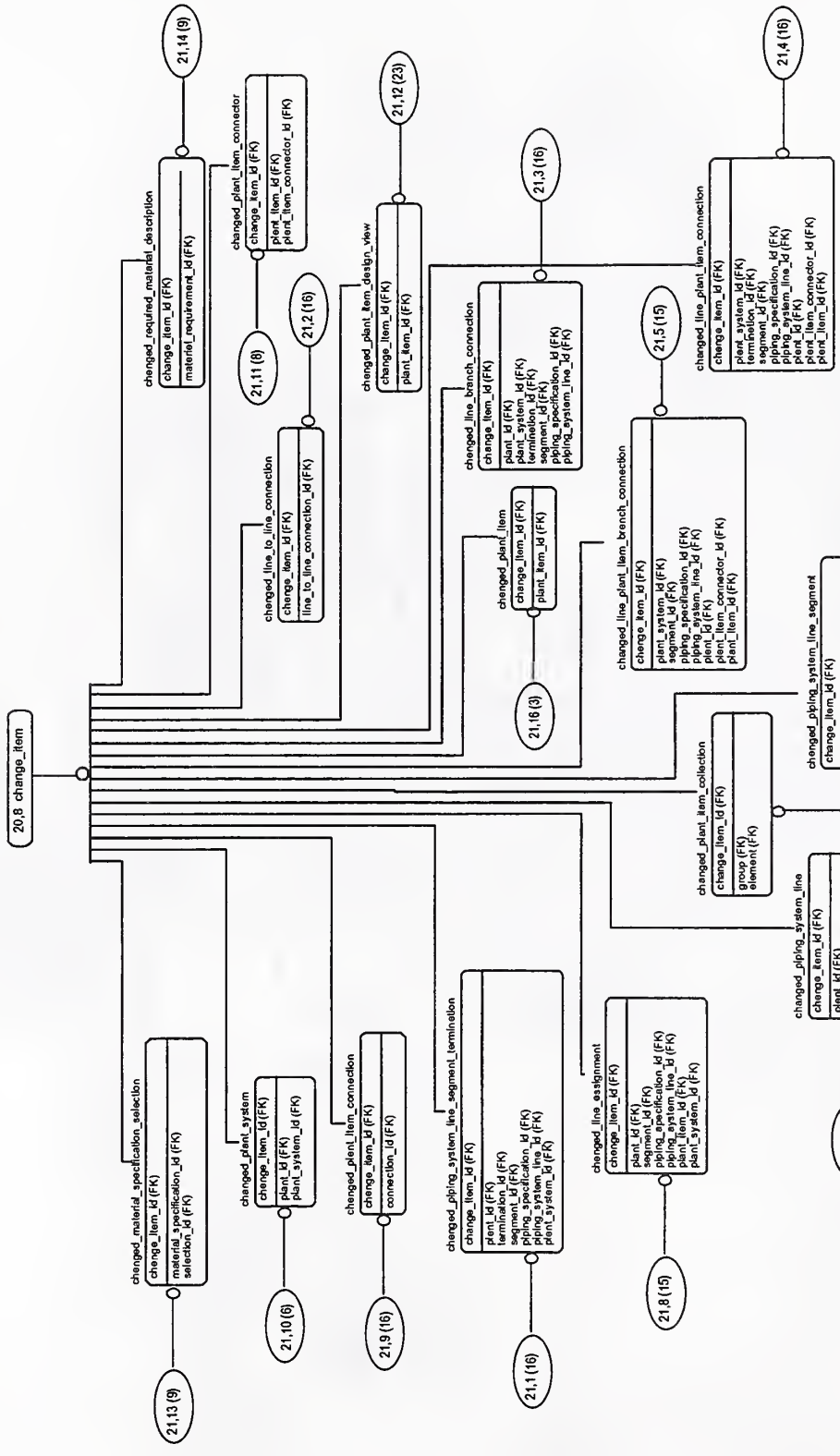


Figure G.21 - ARM diagram 21 of 25 in IDEF1X

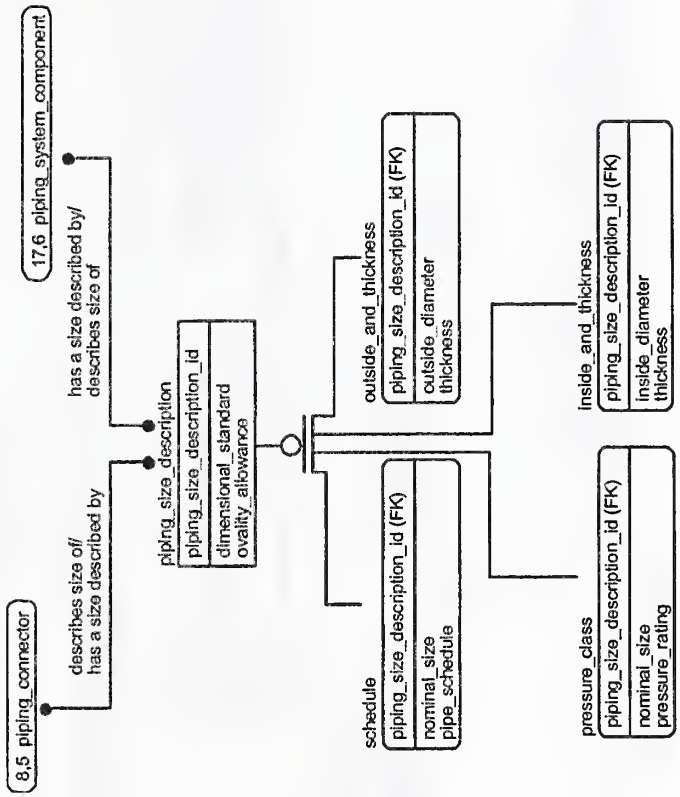


Figure G.22 - ARM diagram 22 of 25 in IDEF1X

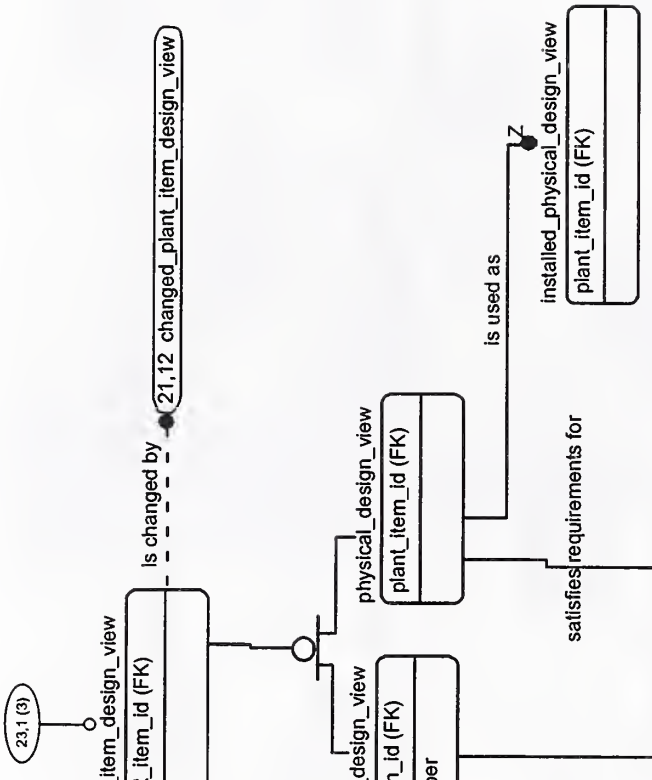


Figure G.23 - ARM diagram 23 of 25 in IDEF1X

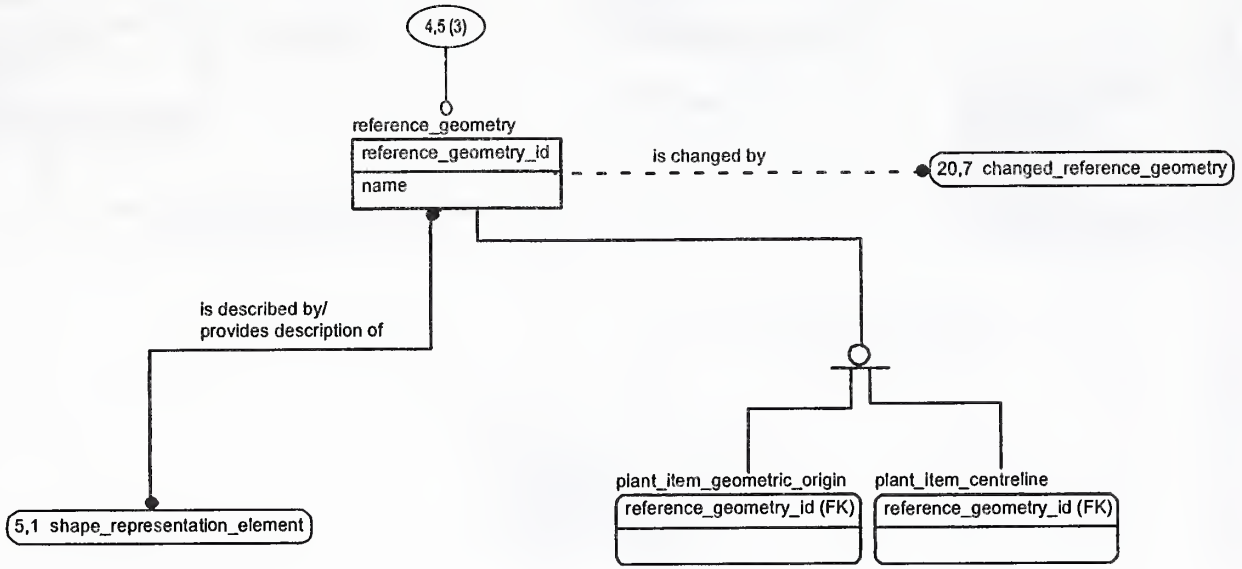


Figure G.24 - ARM diagram 24 of 25 in IDEF1X

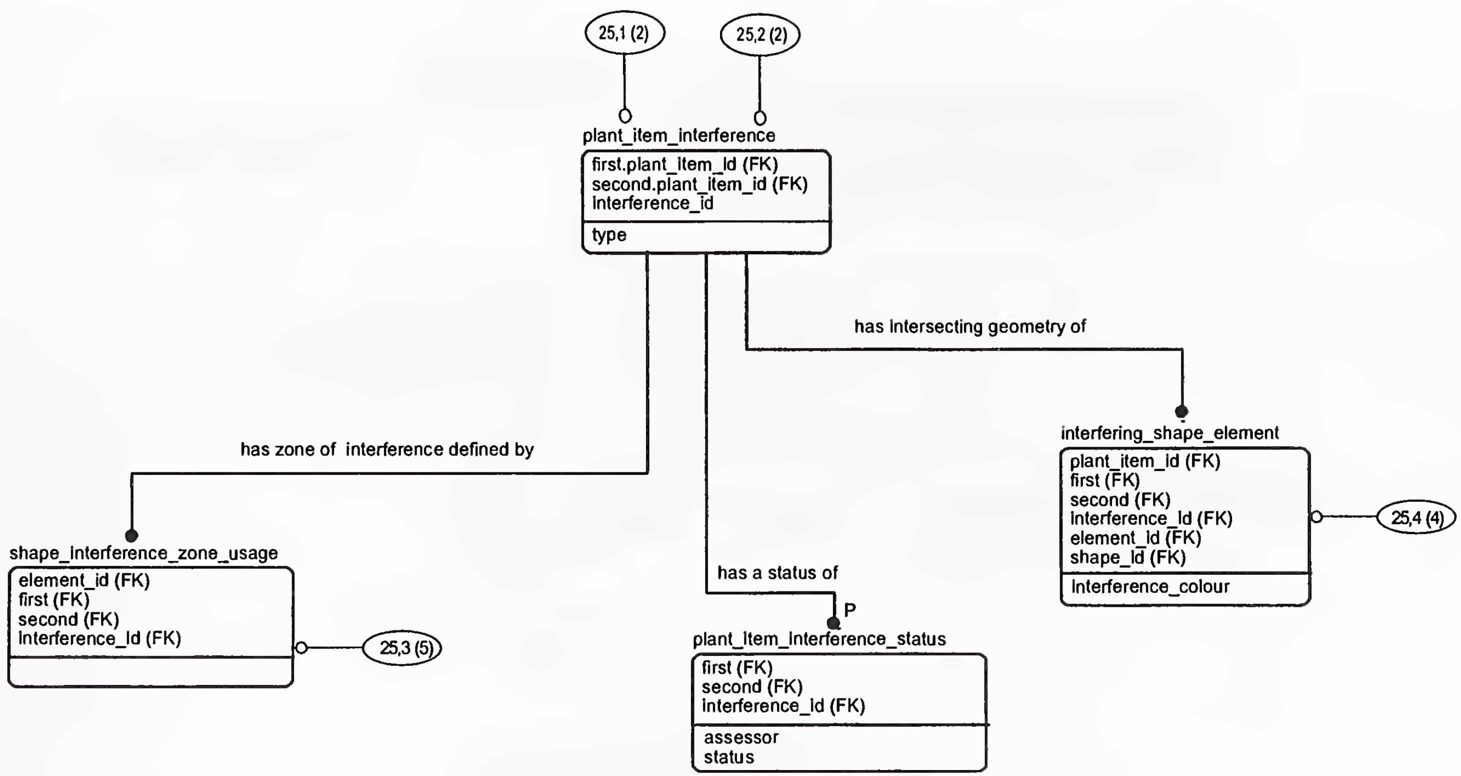


Figure G.25 - ARM diagram 25 of 25 in IDEF1X

Annex H
(informative)

AIM EXPRESS-G

Figures H.1 through H.32 correspond to the AIM EXPRESS expanded listing given in annex A. The figures use the EXPRESS-G graphical notation for the EXPRESS language. EXPRESS-G is defined in annex A of ISO 10303-11.

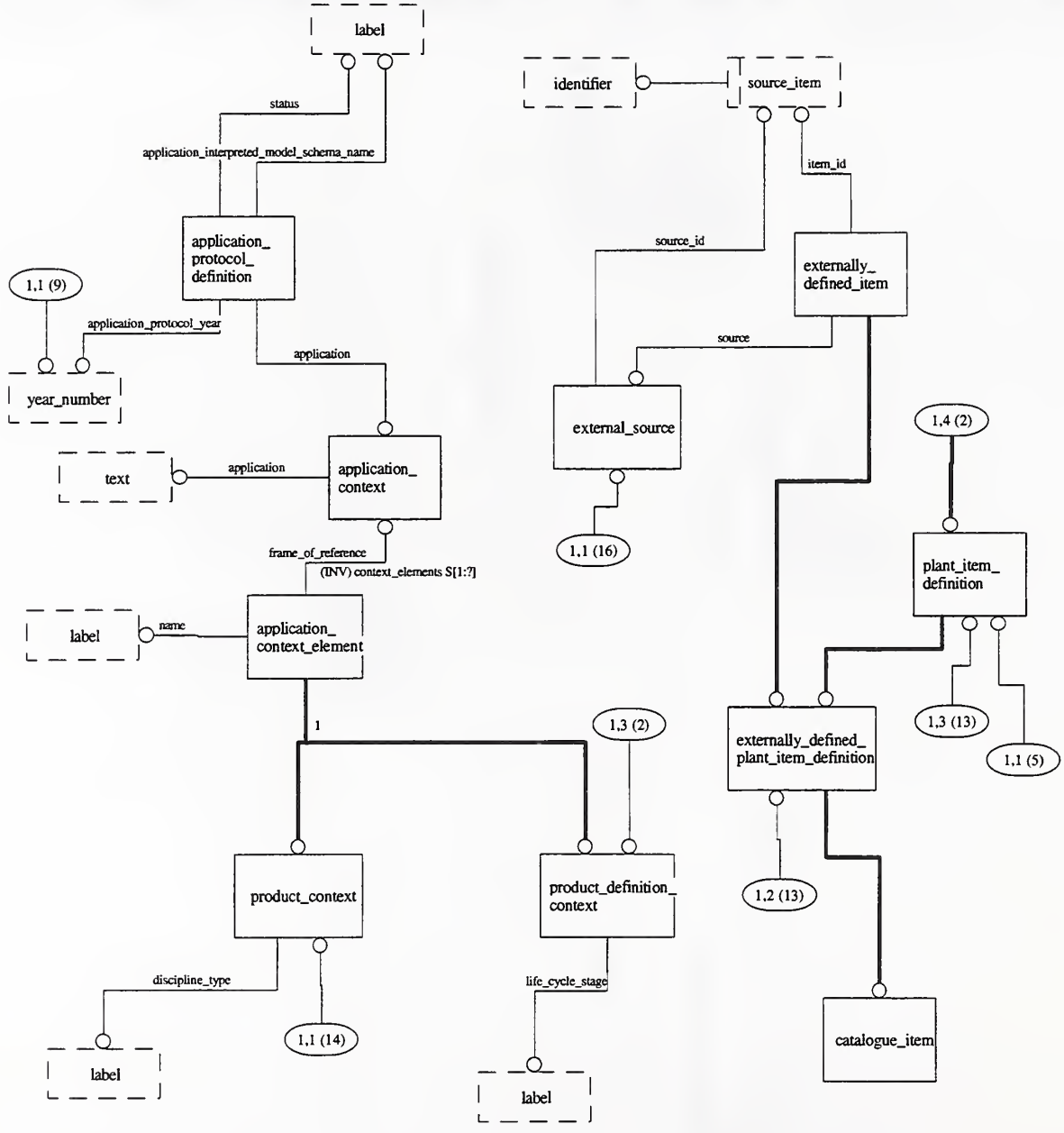


Figure H.1 - AIM EXPRESS-G diagram 1 of 32

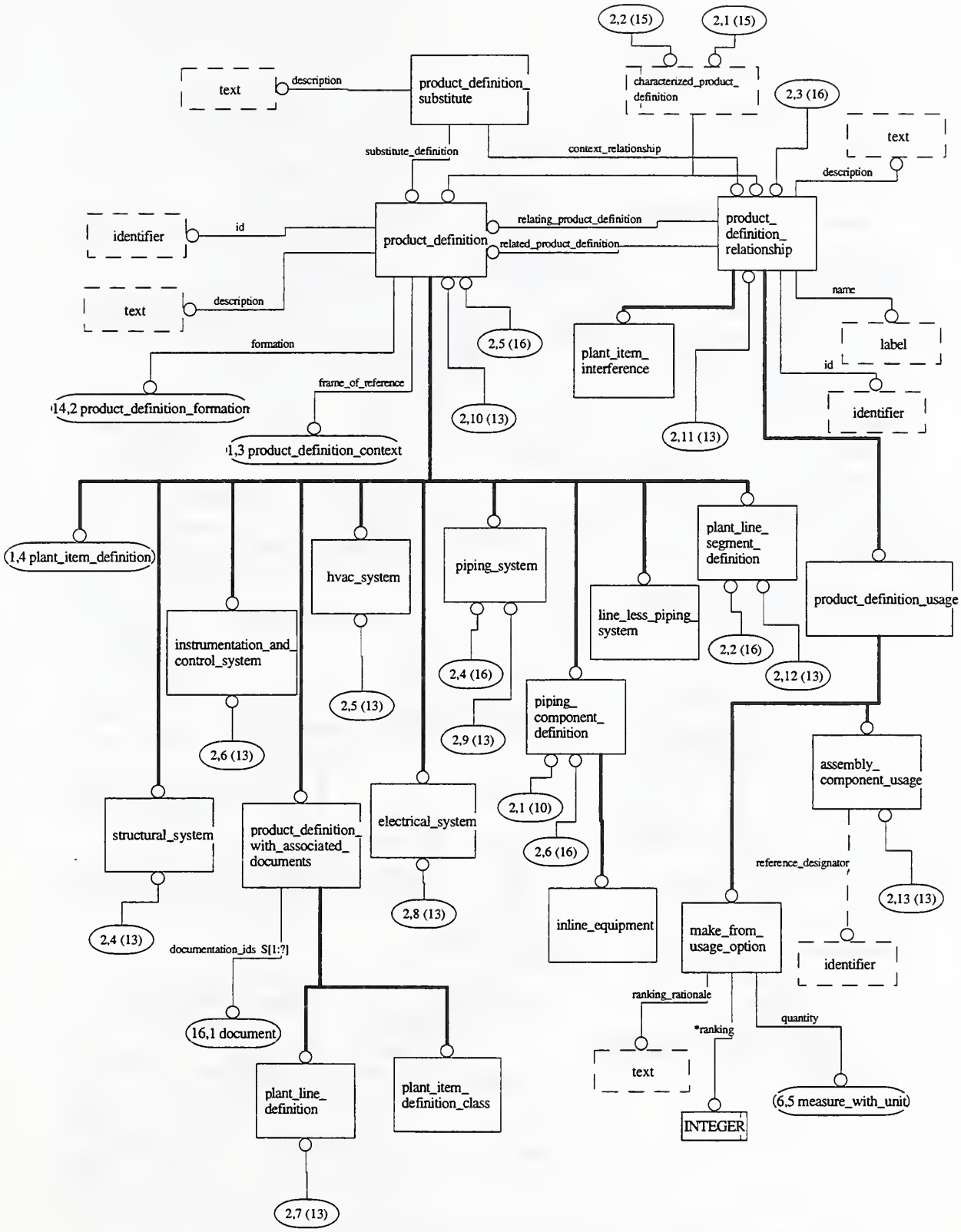


Figure H.2 - AIM EXPRESS-G diagram 2 of 32

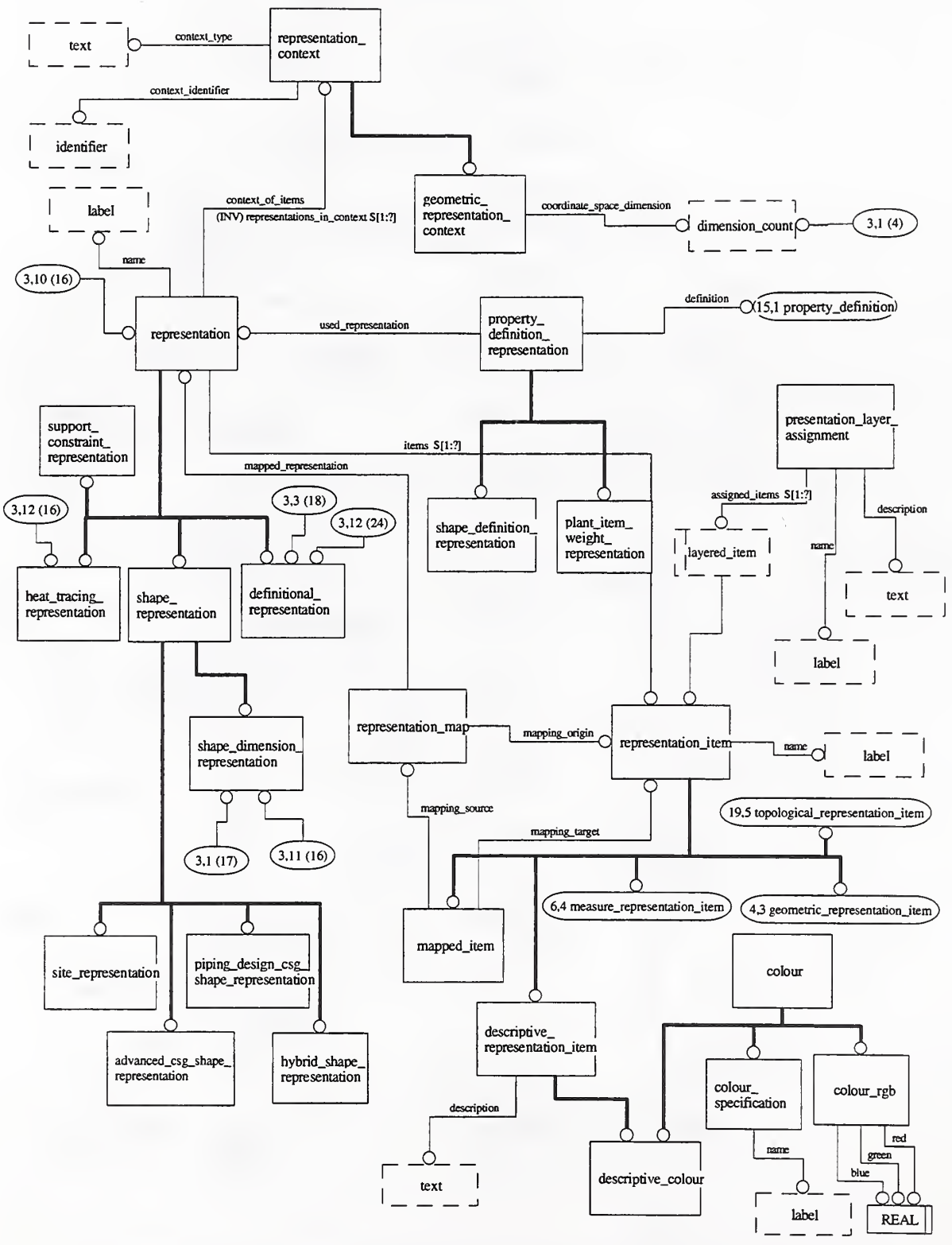


Figure H.3 - AIM EXPRESS-G diagram 3 of 32

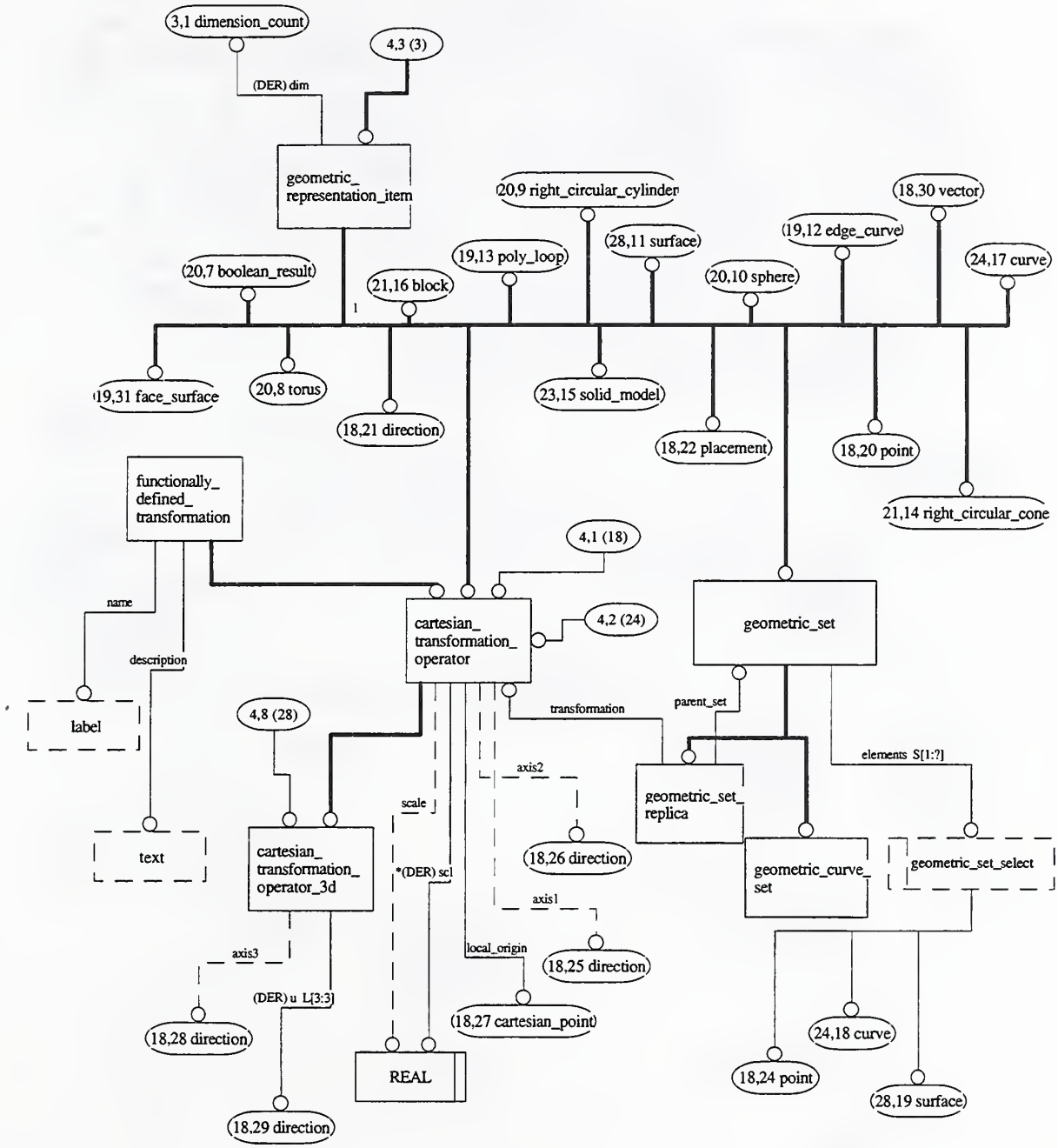


Figure H.4 - AIM EXPRESS-G diagram 4 of 32

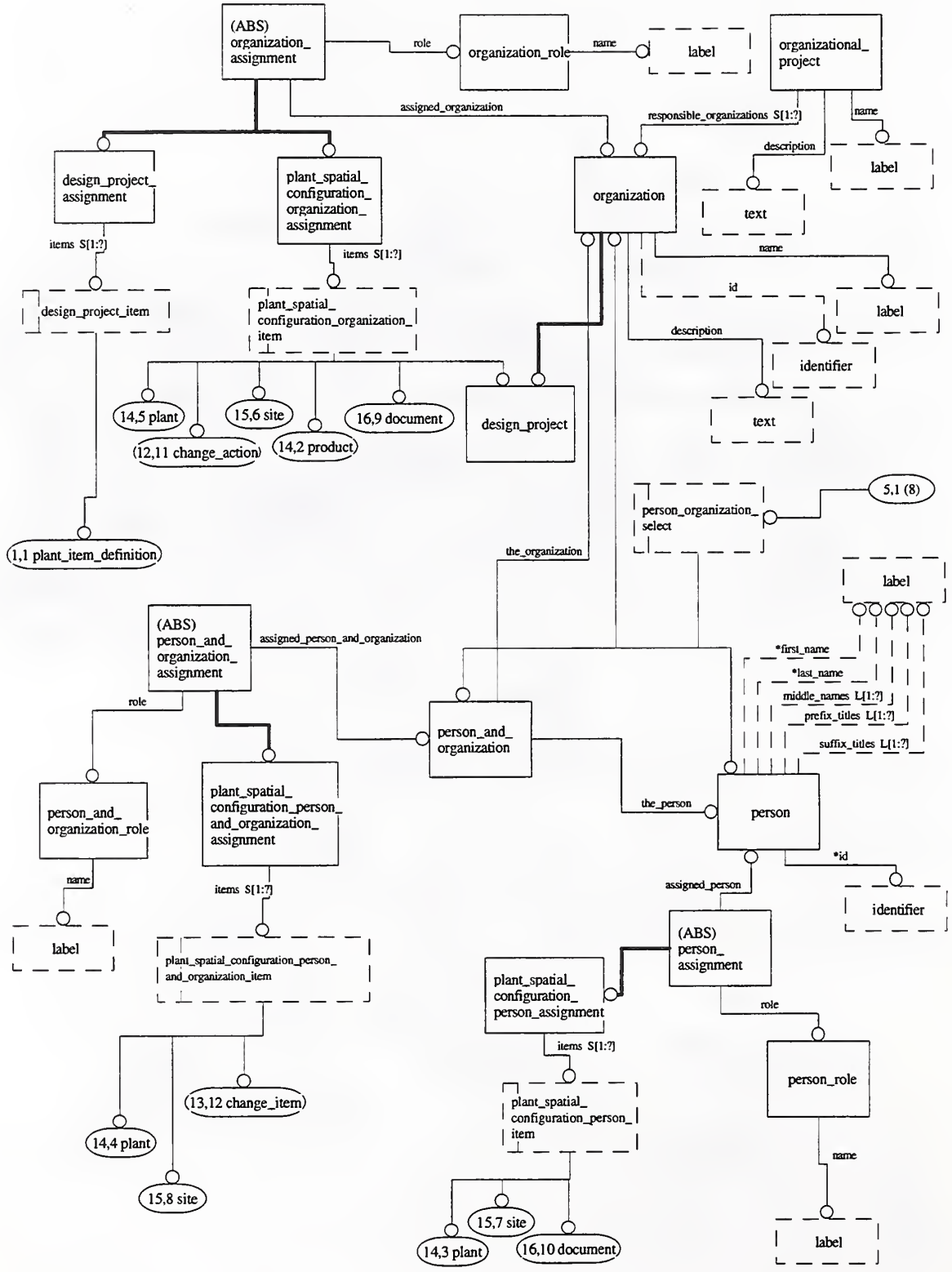


Figure H.5 - AIM EXPRESS-G diagram 5 of 32

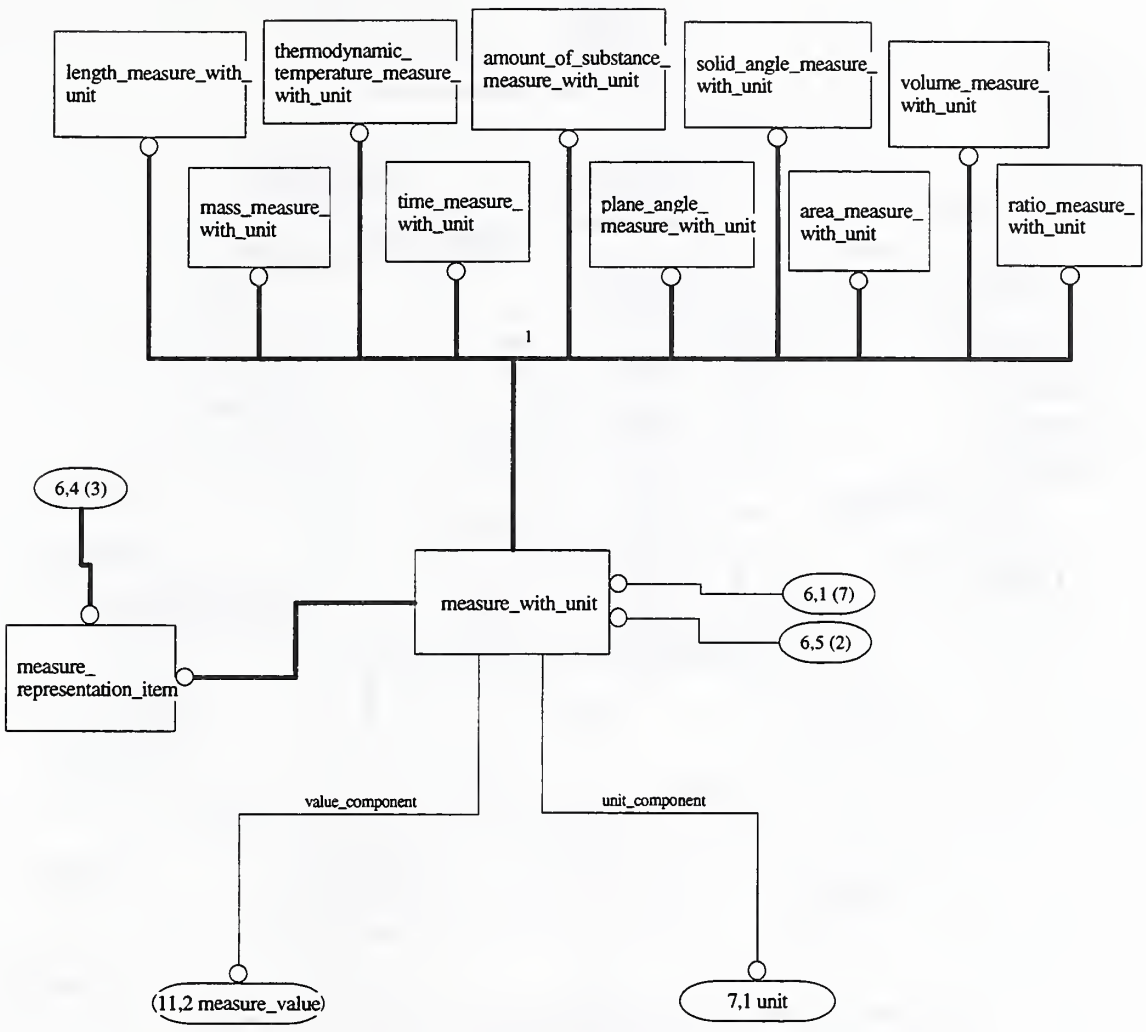


Figure H.6 - AIM EXPRESS-G diagram 6 of 32

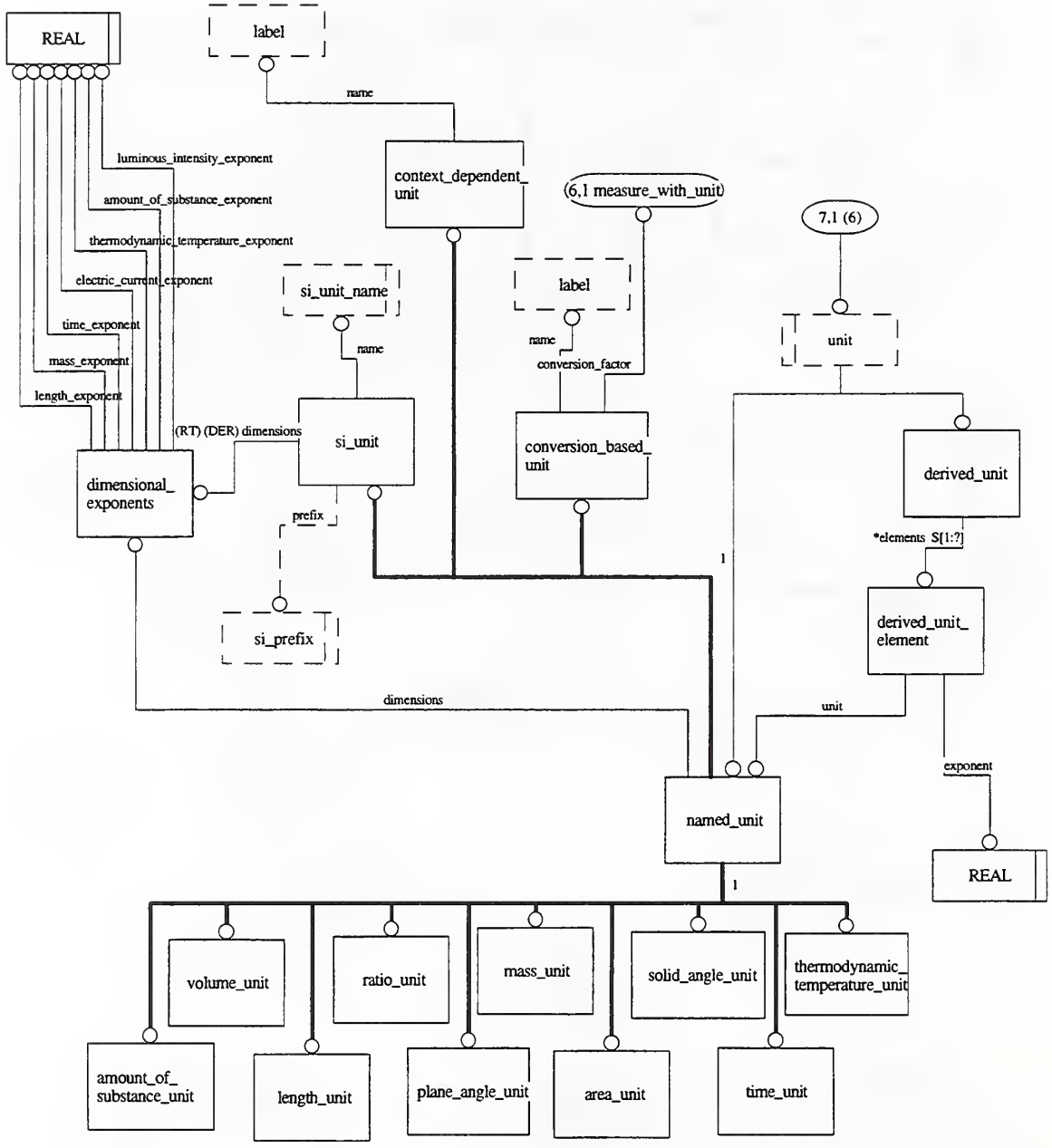


Figure H.7 - AIM EXPRESS-G diagram 7 of 32

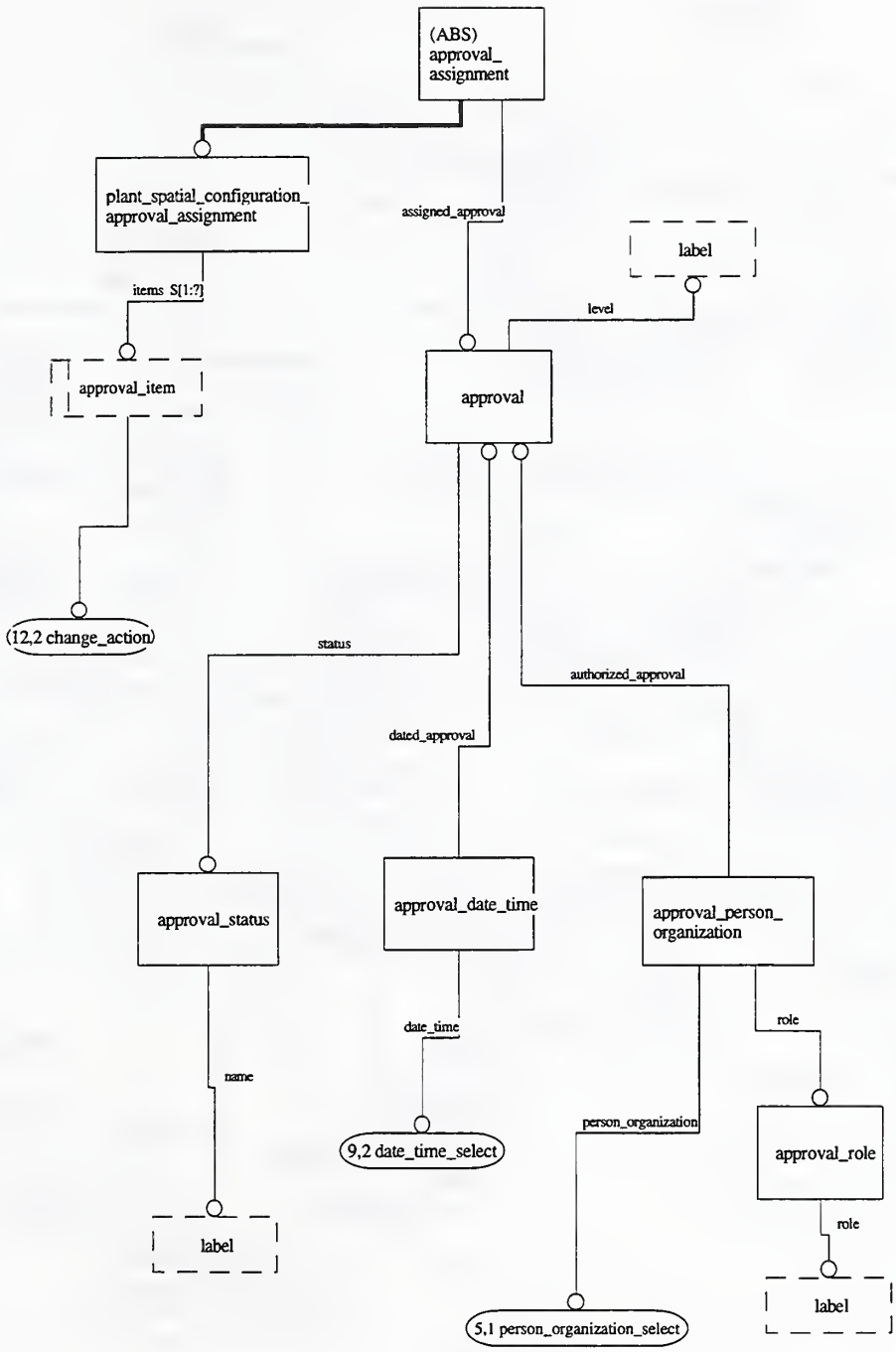


Figure H.8 - AIM EXPRESS-G diagram 8 of 32

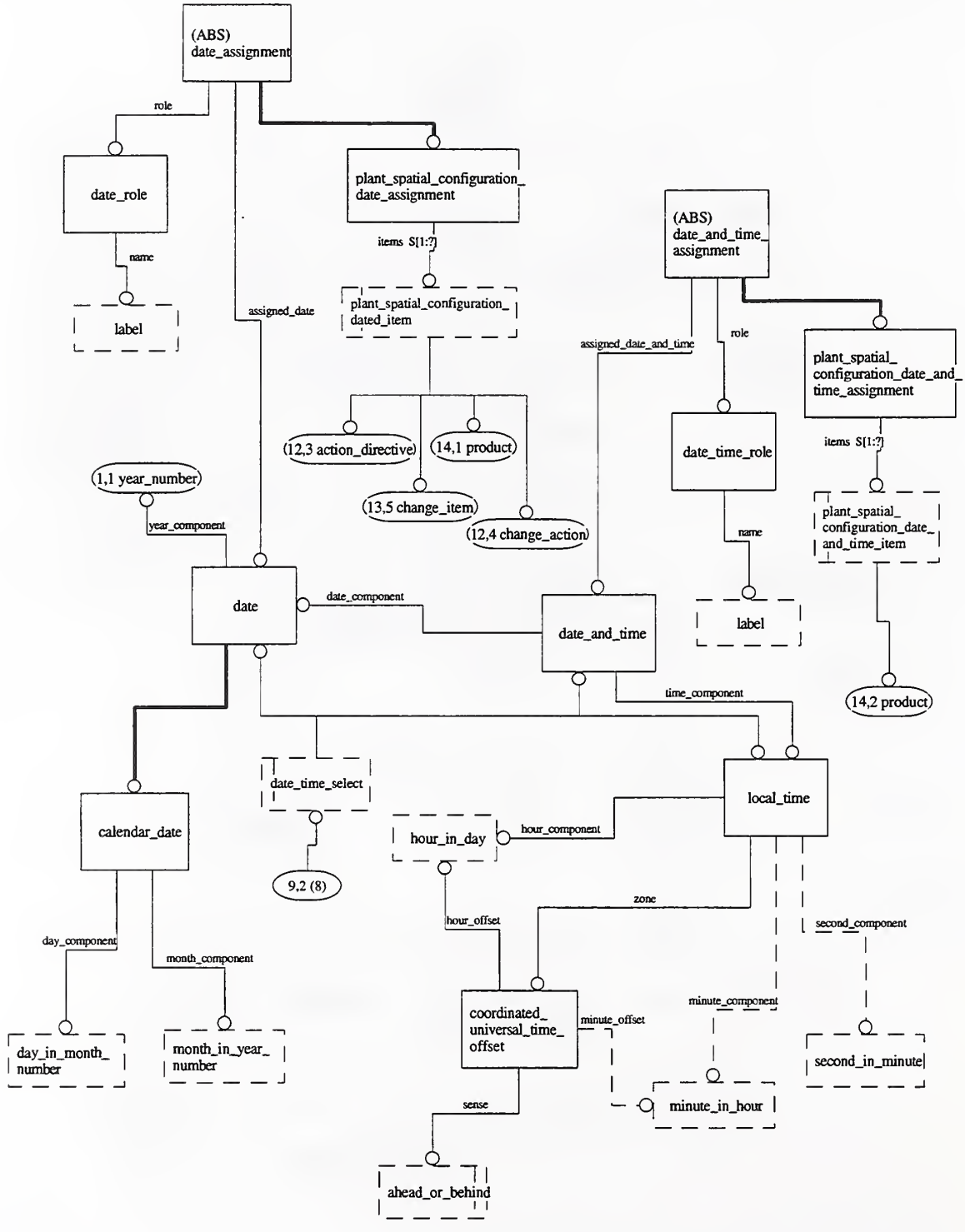


Figure H.9 - AIM EXPRESS-G diagram 9 of 32

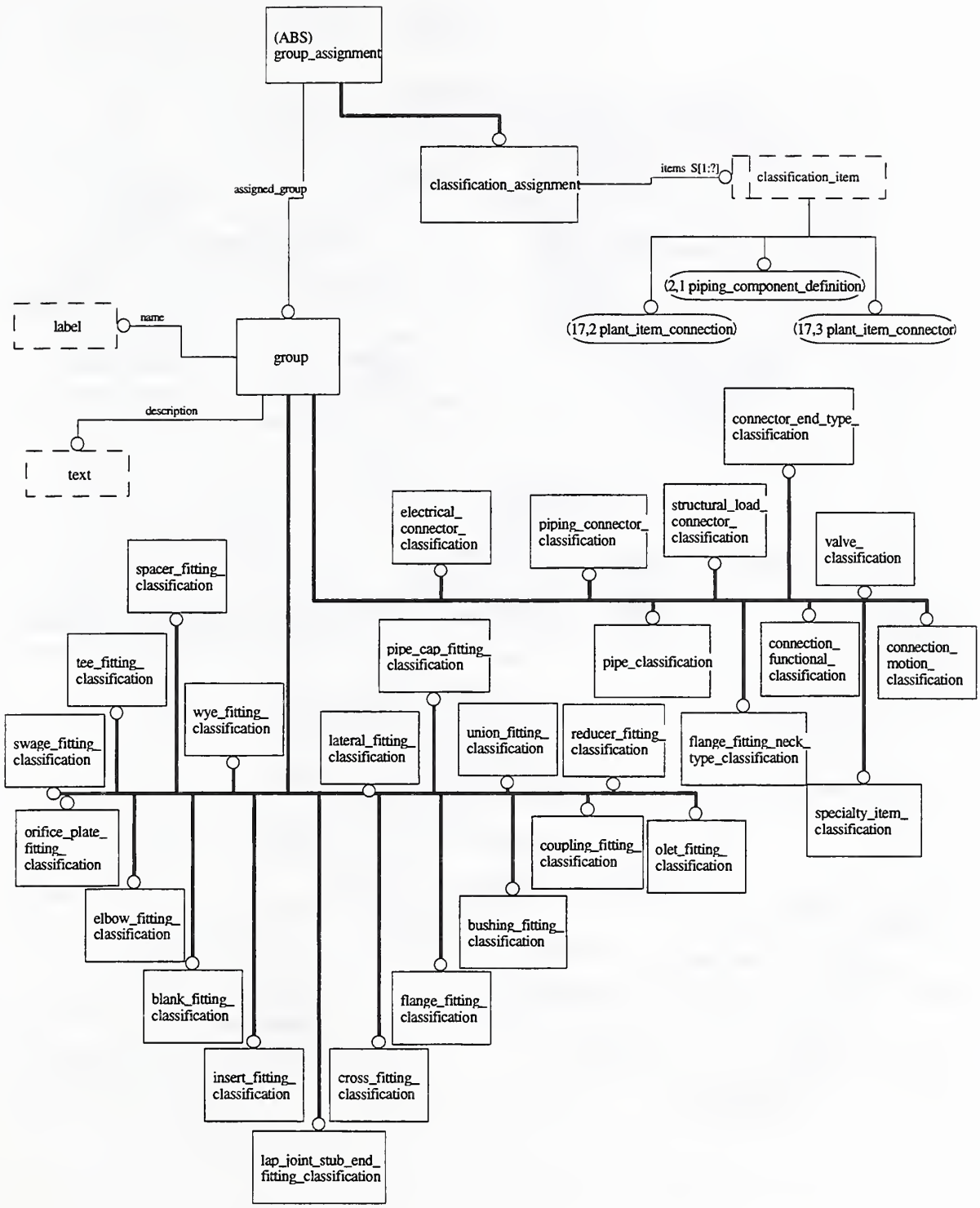


Figure H.10 - AIM EXPRESS-G diagram 10 of 32

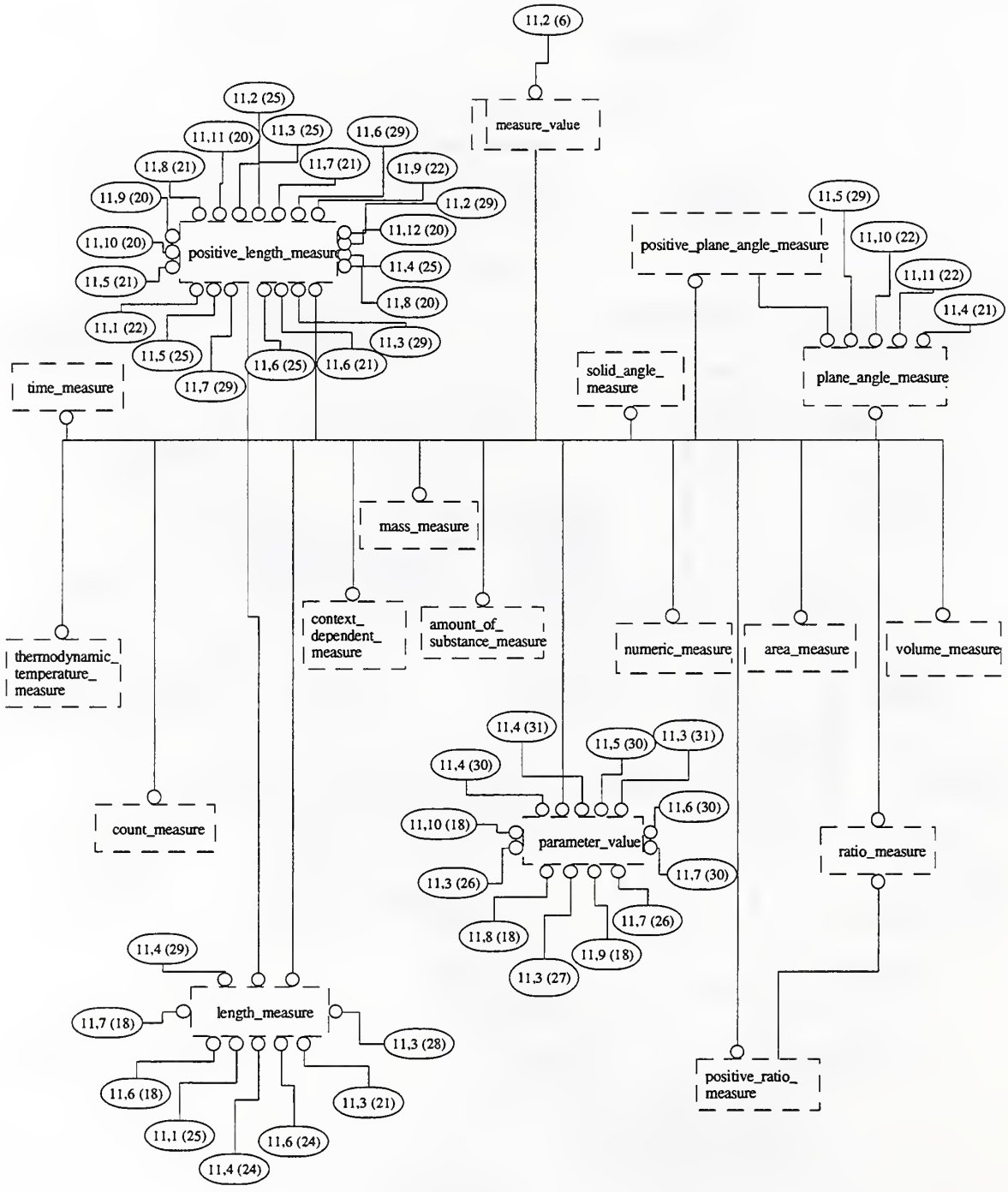


Figure H.11 - AIM EXPRESS-G diagram 11 of 32

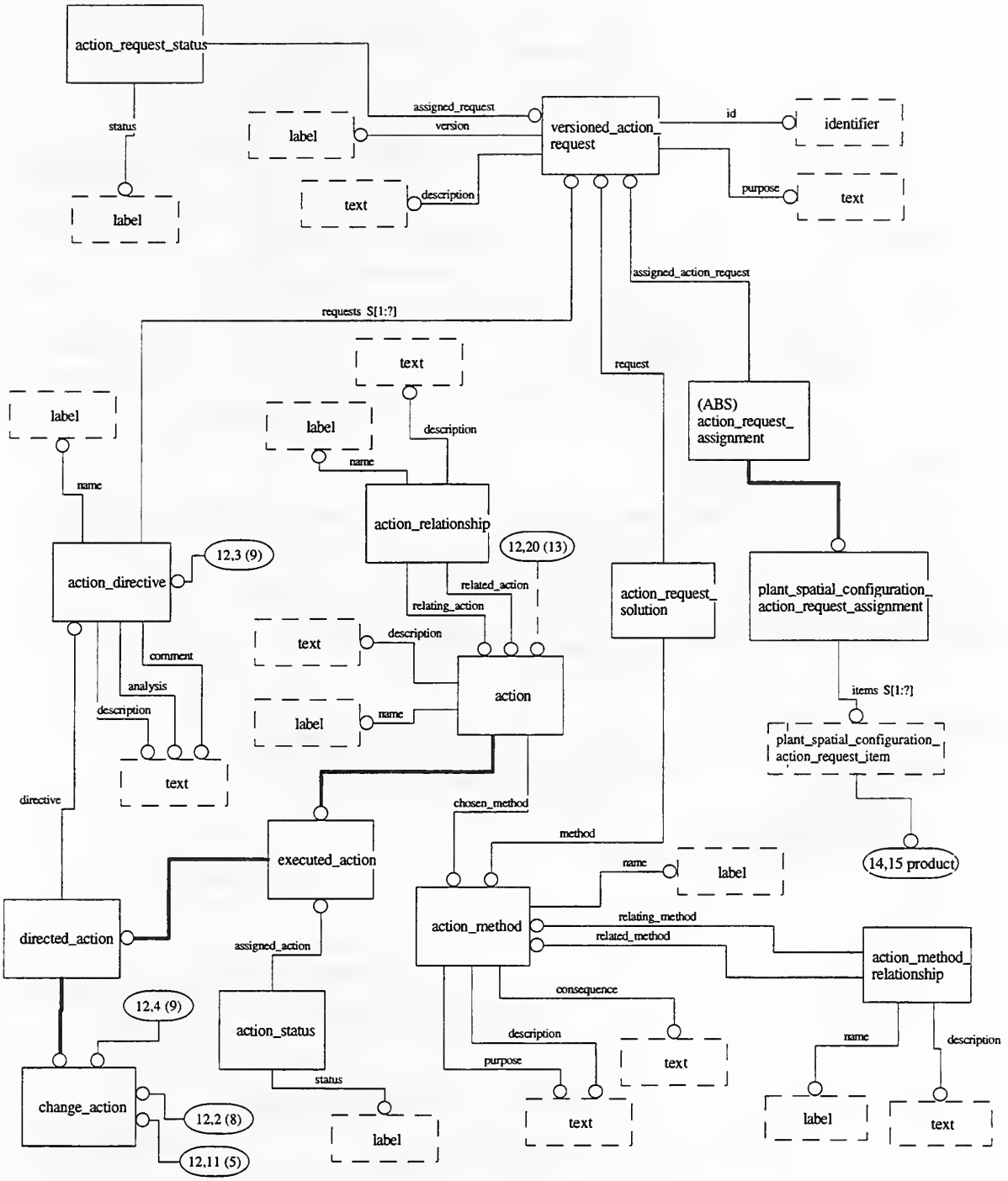


Figure H.12 - AIM EXPRESS-G diagram 12 of 32

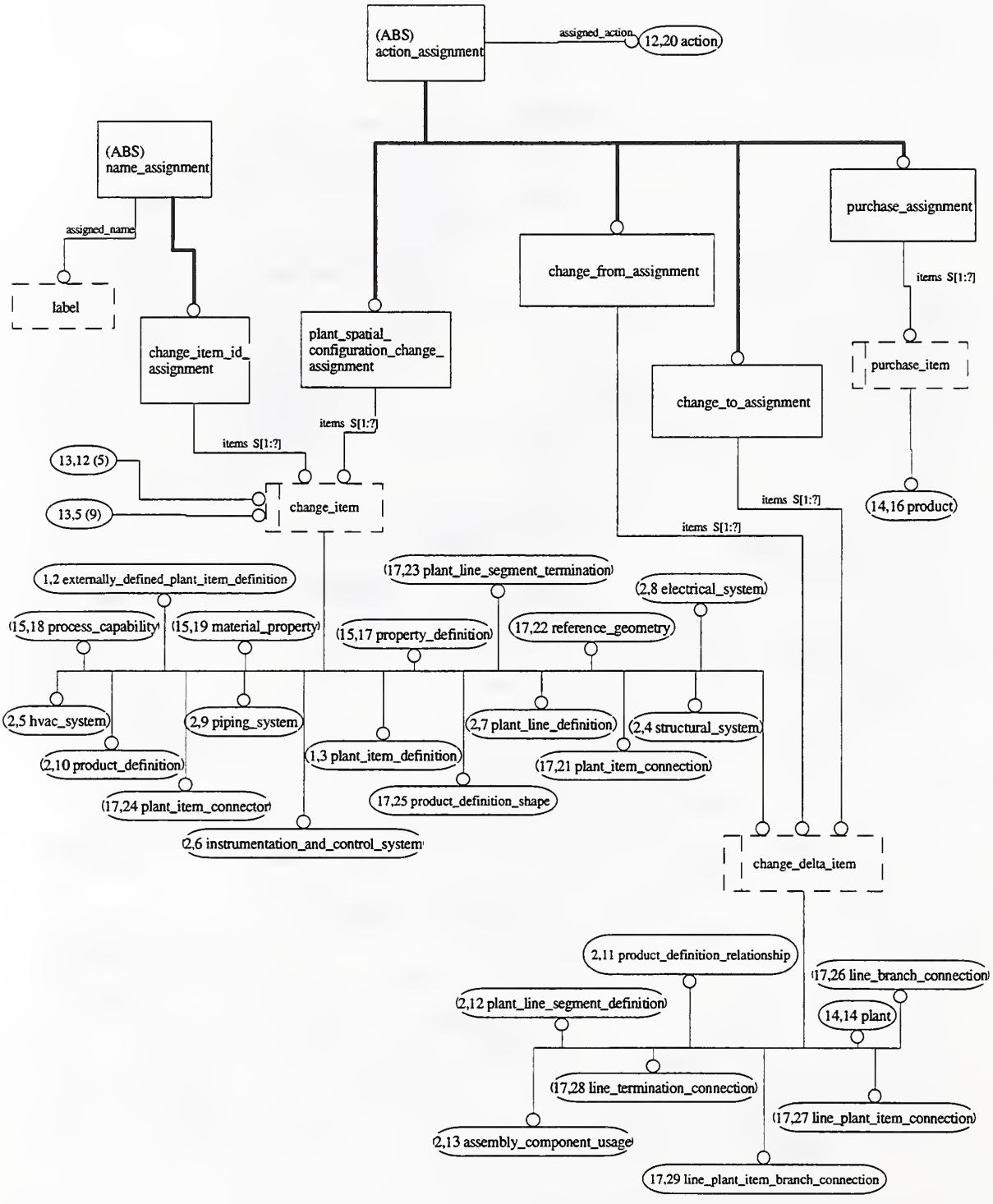


Figure H.13 - AIM EXPRESS-G diagram 13 of 32

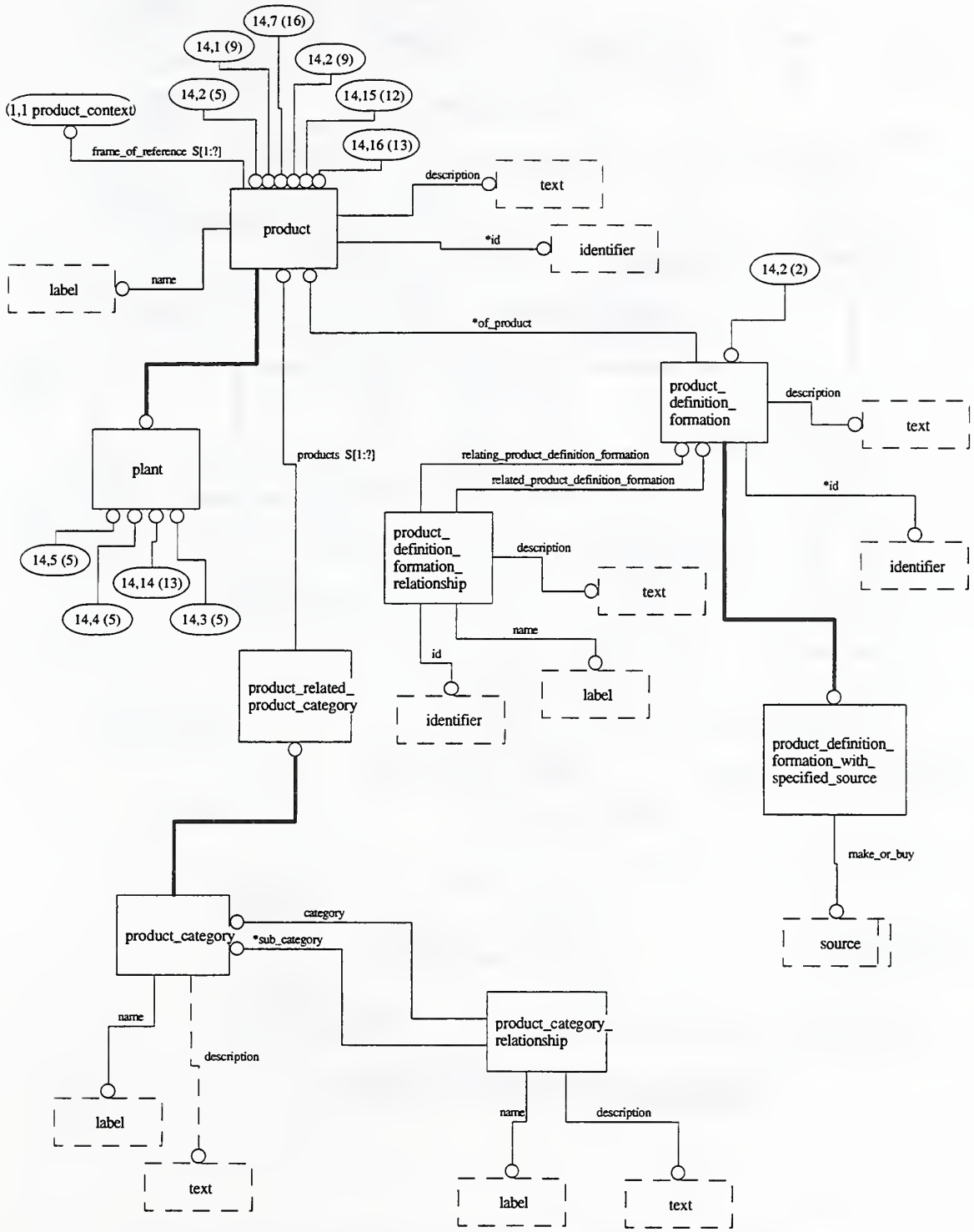


Figure H.14 - AIM EXPRESS-G diagram 14 of 32

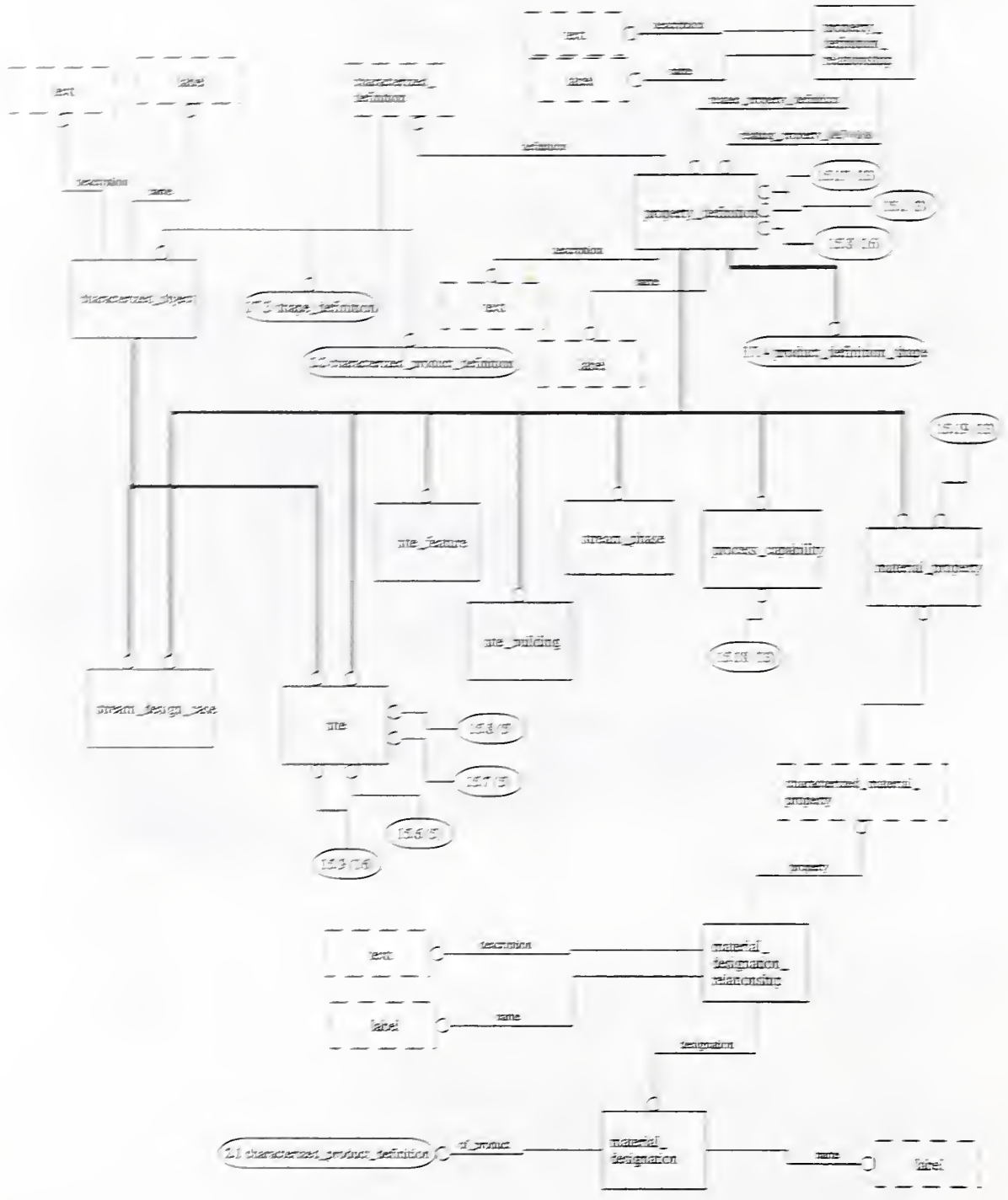


Figure H.15 - AIM EXPRESS-G diagram 15 of 32

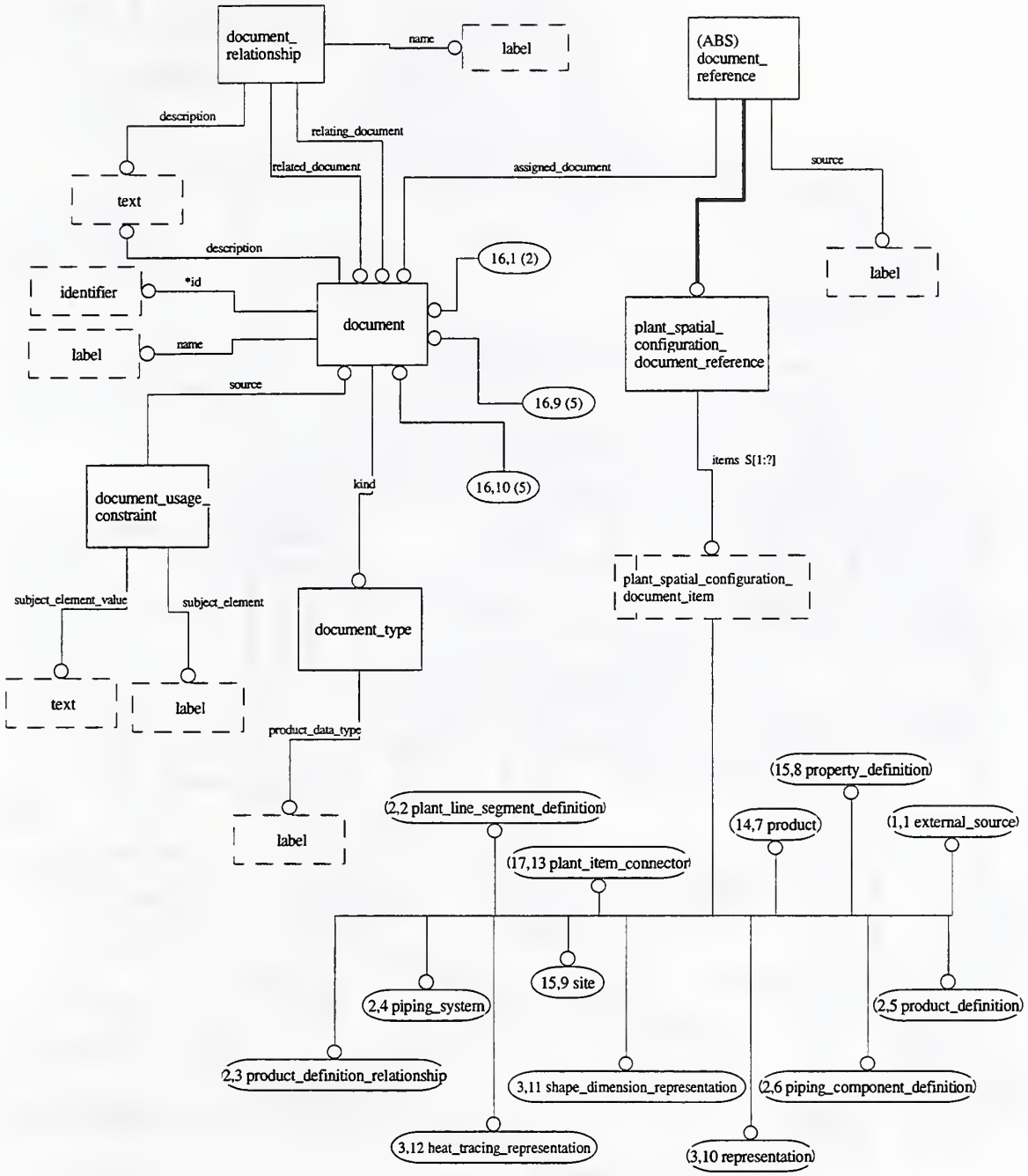


Figure H.16 - AIM EXPRESS-G diagram 16 of 32

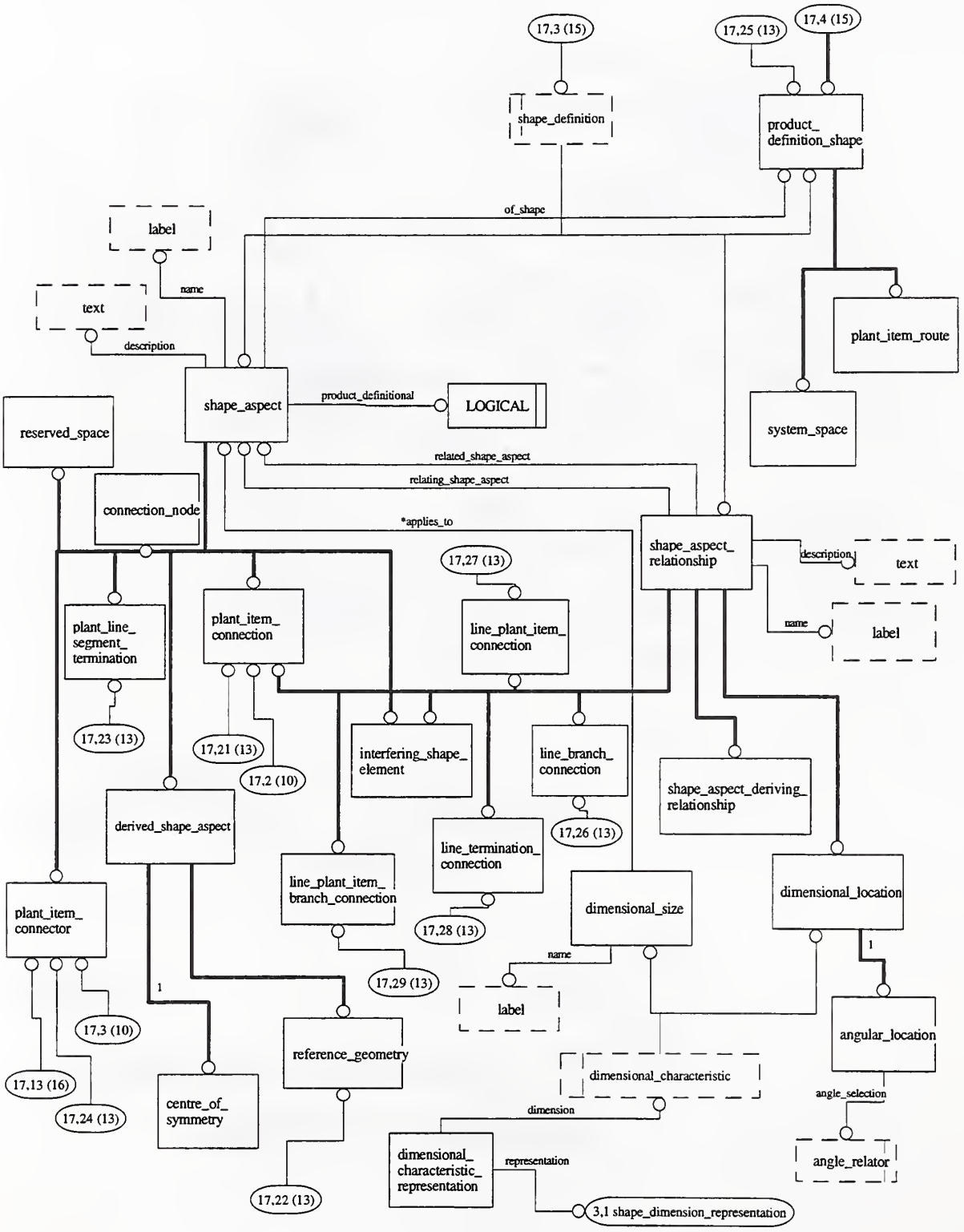


Figure H.17 - AIM EXPRESS-G diagram 17 of 32

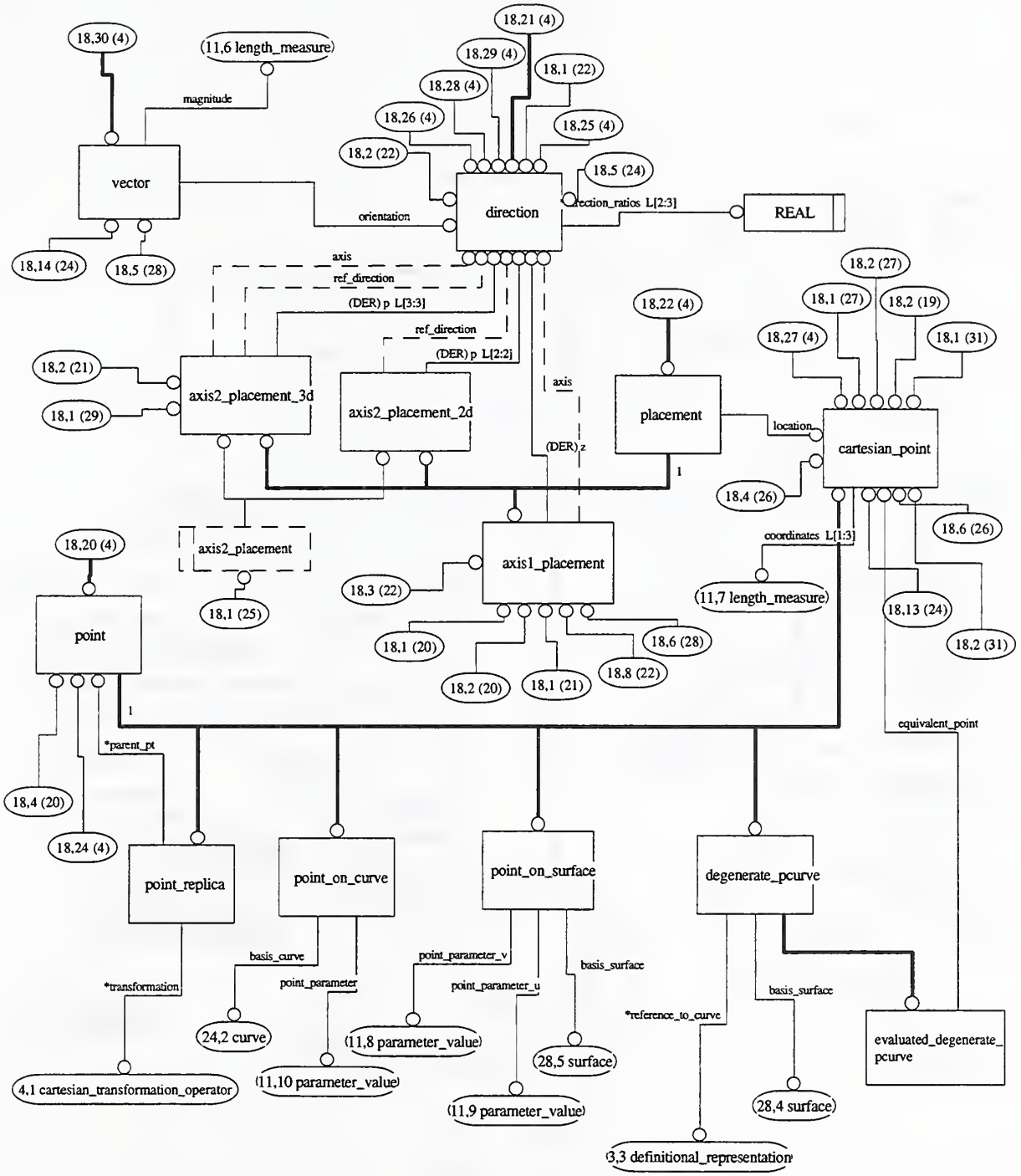


Figure H.18 - AIM EXPRESS-G diagram 18 of 32

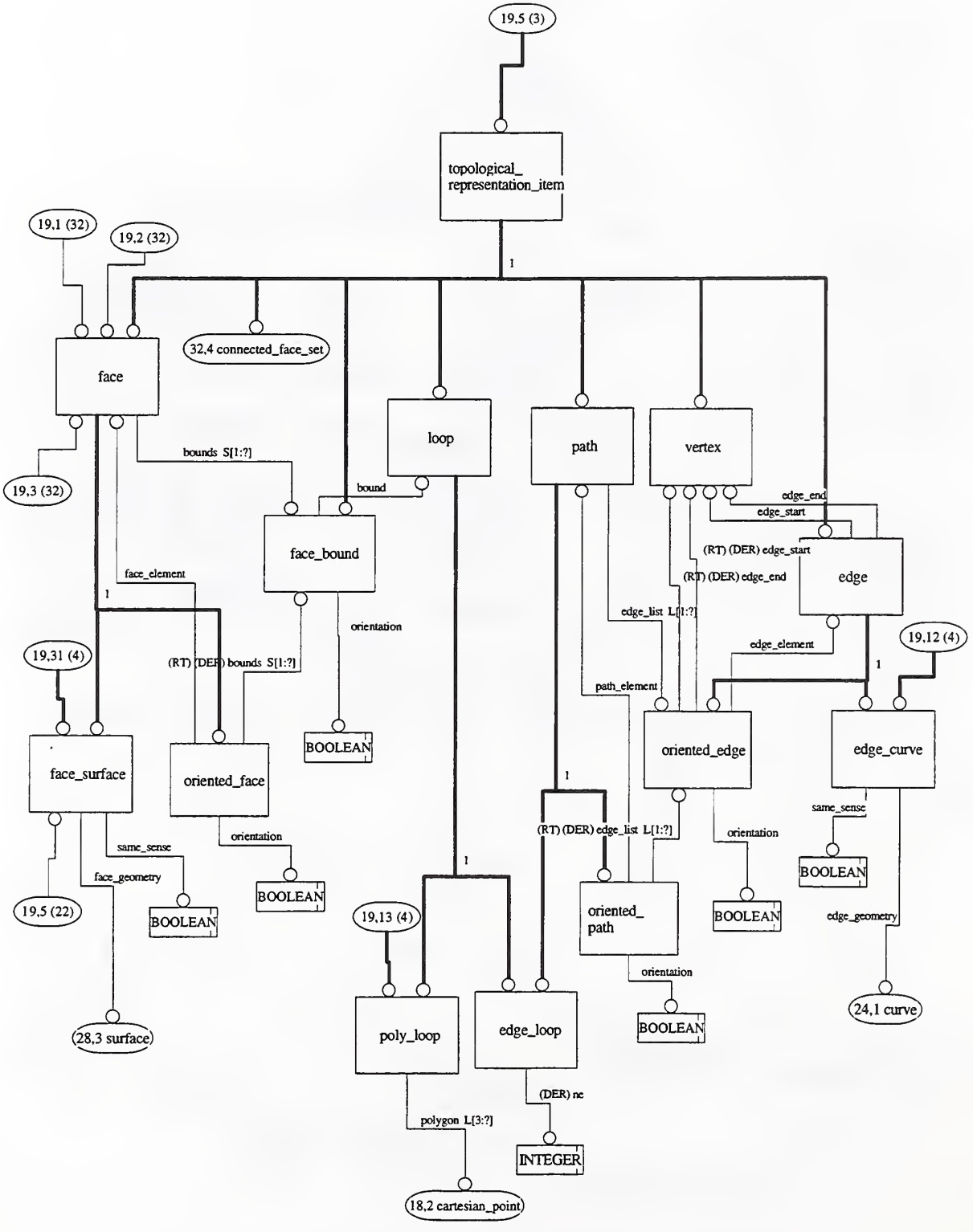


Figure H.19 - AIM EXPRESS-G diagram 19 of 32

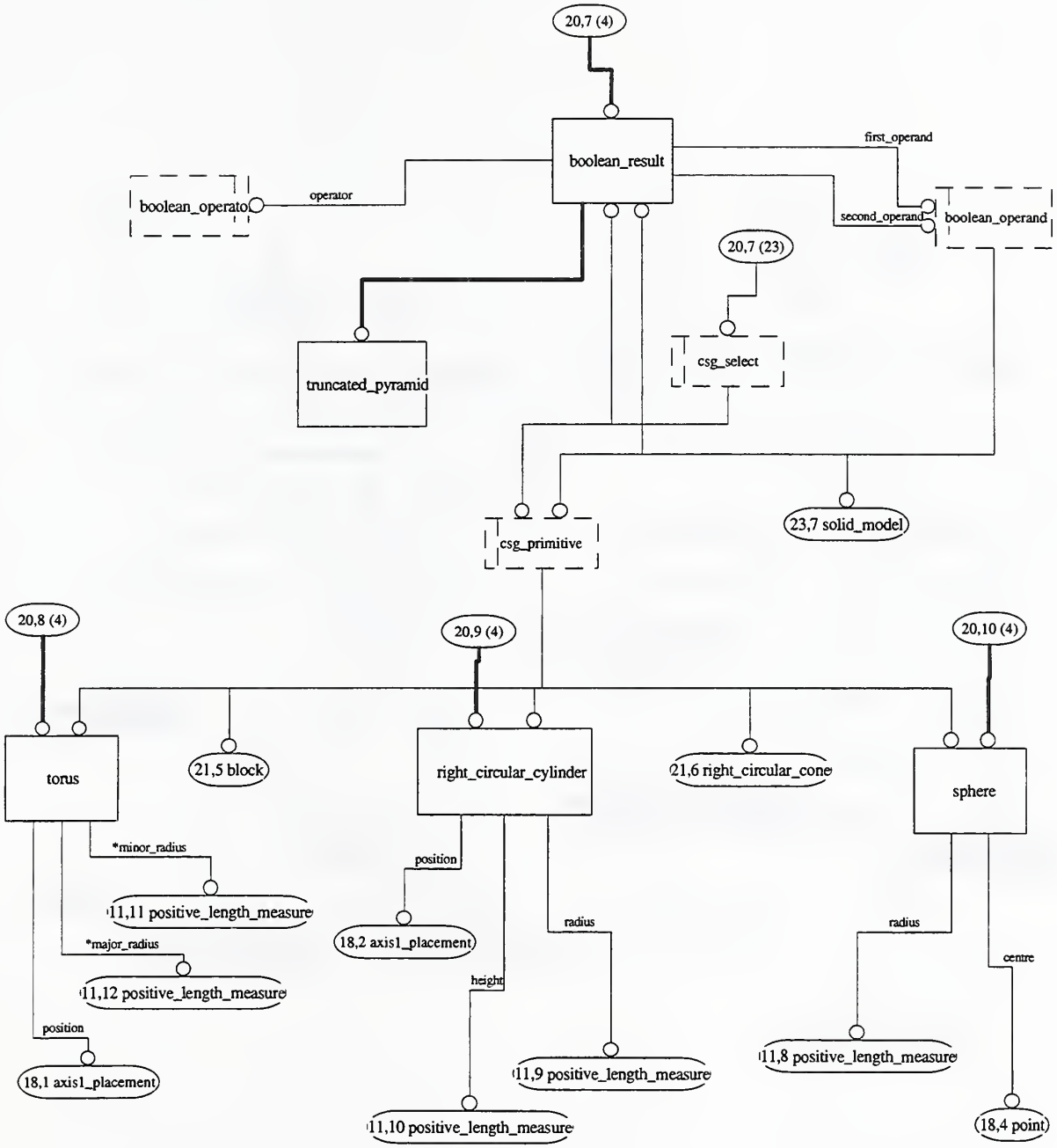


Figure H.20 - AIM EXPRESS-G diagram 20 of 32

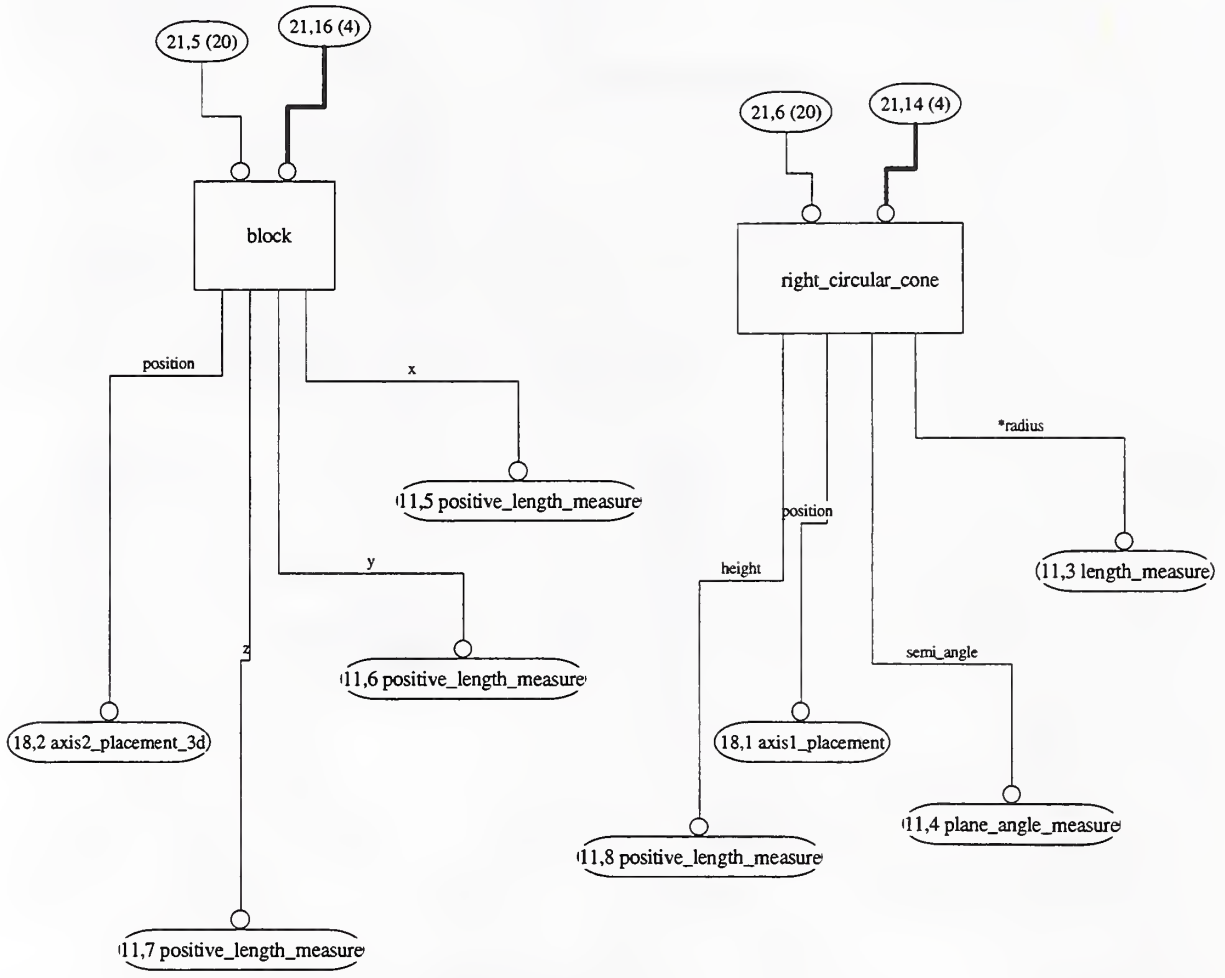


Figure H.21 - AIM EXPRESS-G diagram 21 of 32

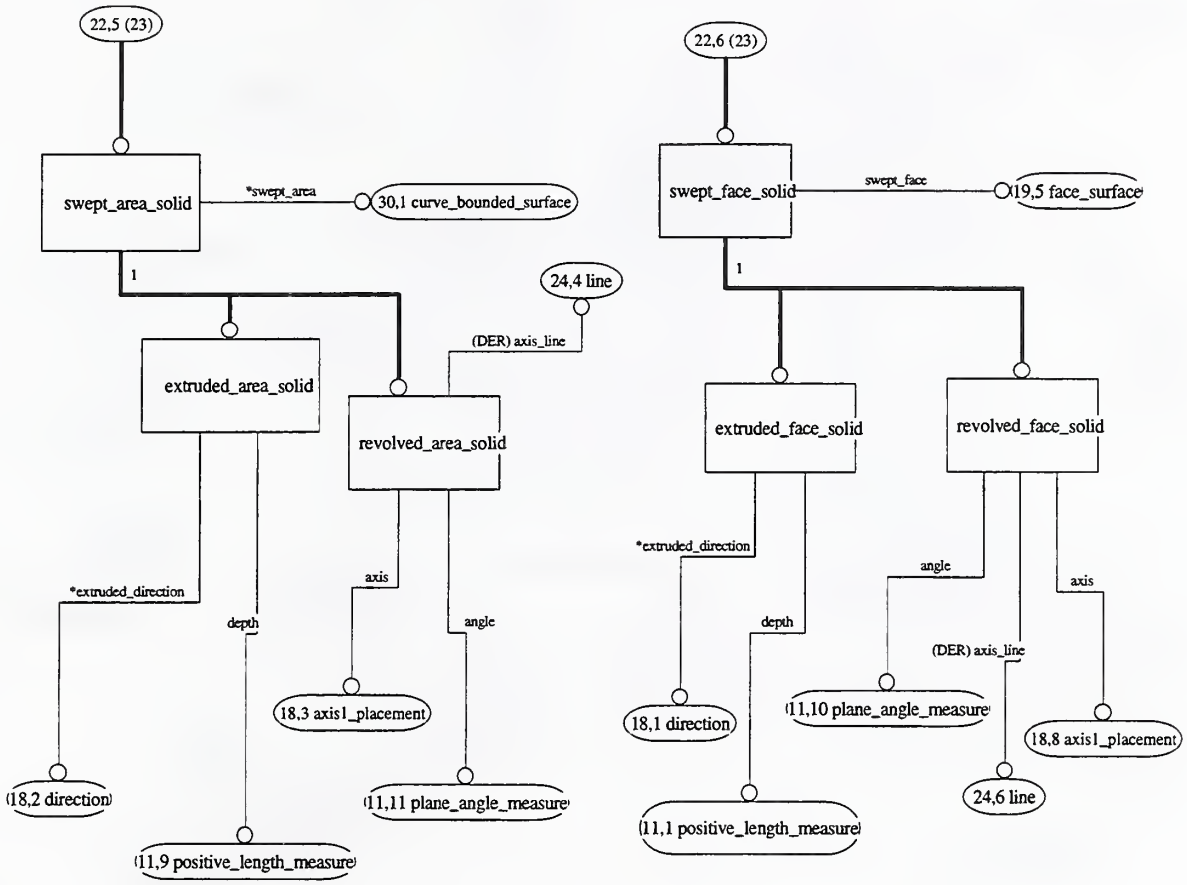


Figure H.22 - AIM EXPRESS-G diagram 22 of 32

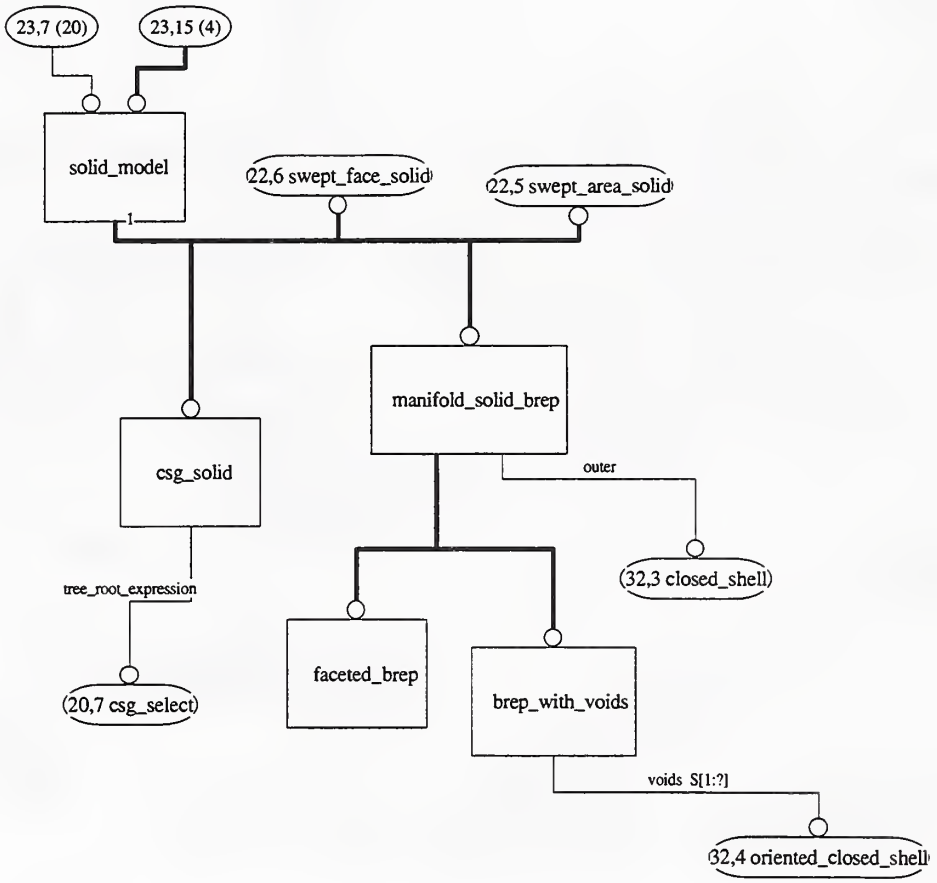


Figure H.23 - AIM EXPRESS-G diagram 23 of 32

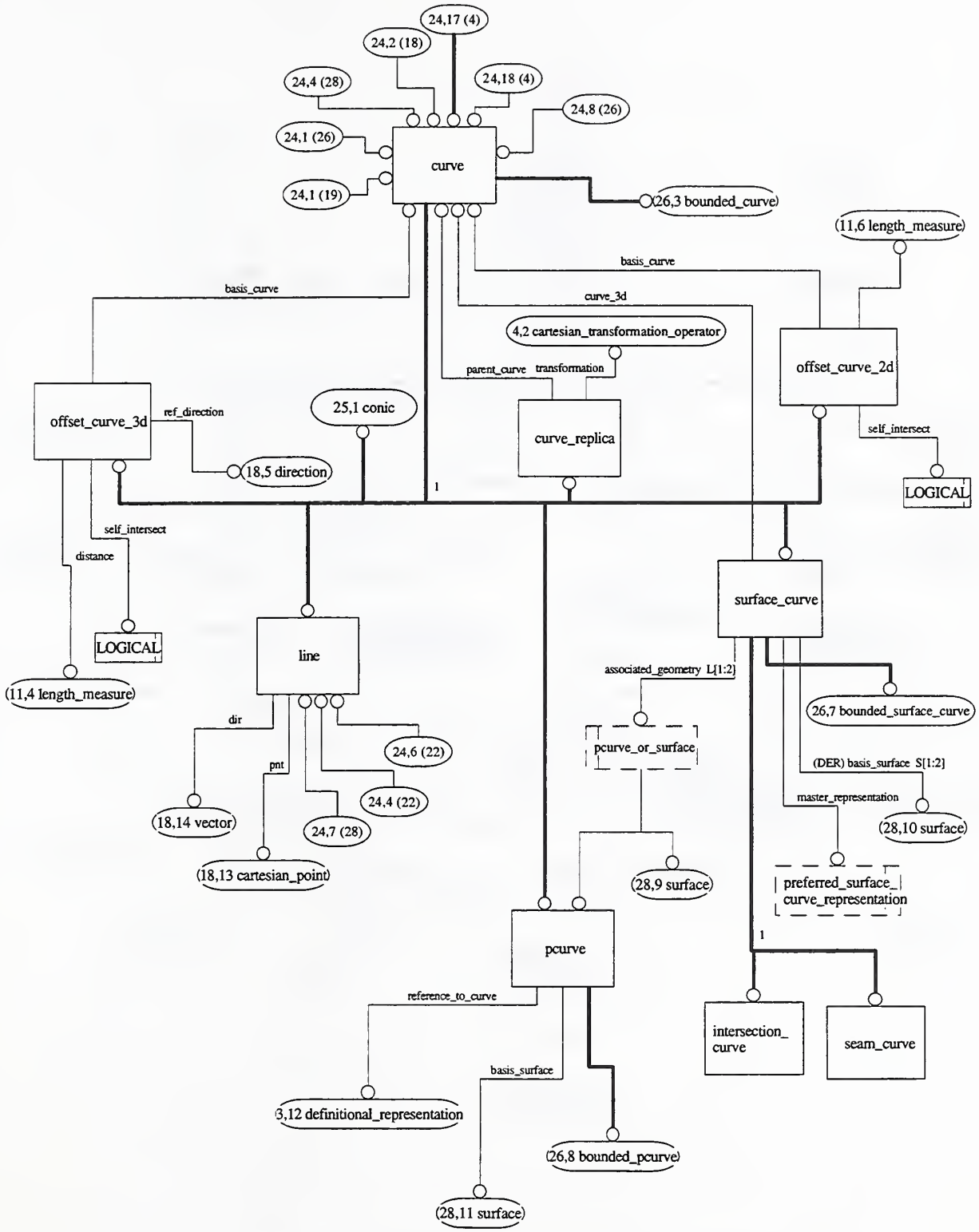


Figure H.24 - AIM EXPRESS-G diagram 24 of 32

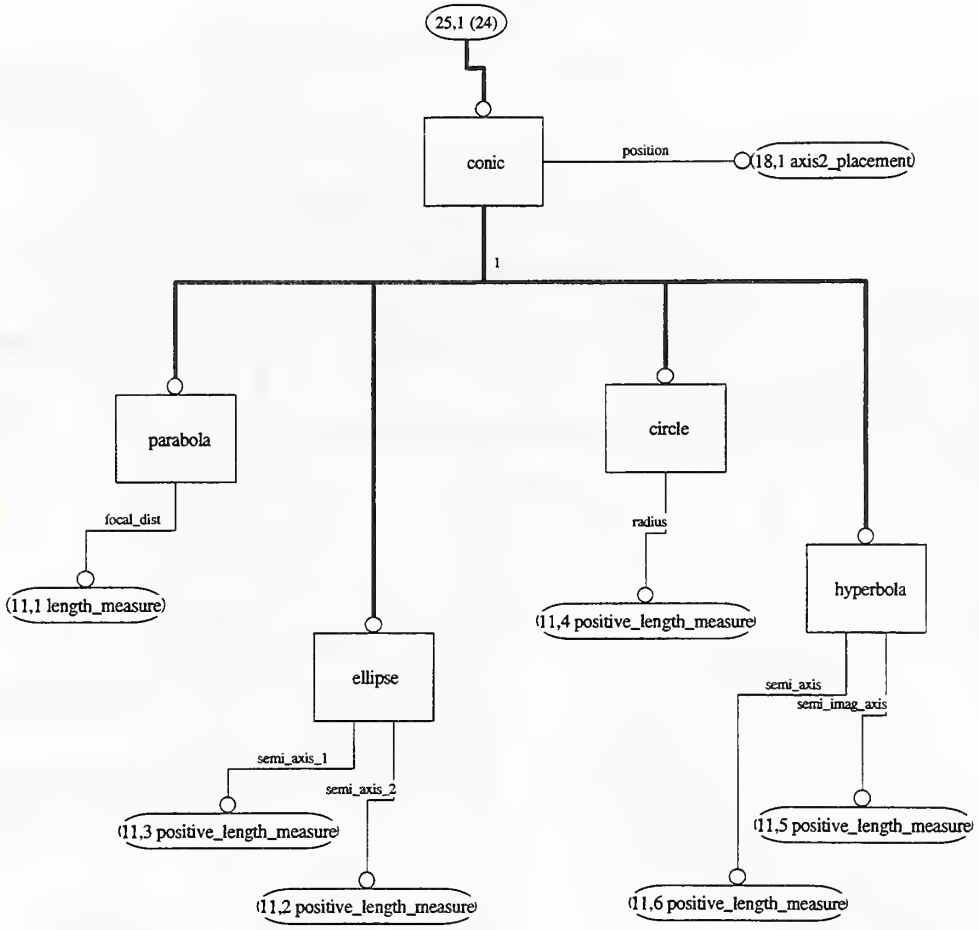


Figure H.25 - AIM EXPRESS-G diagram 25 of 32

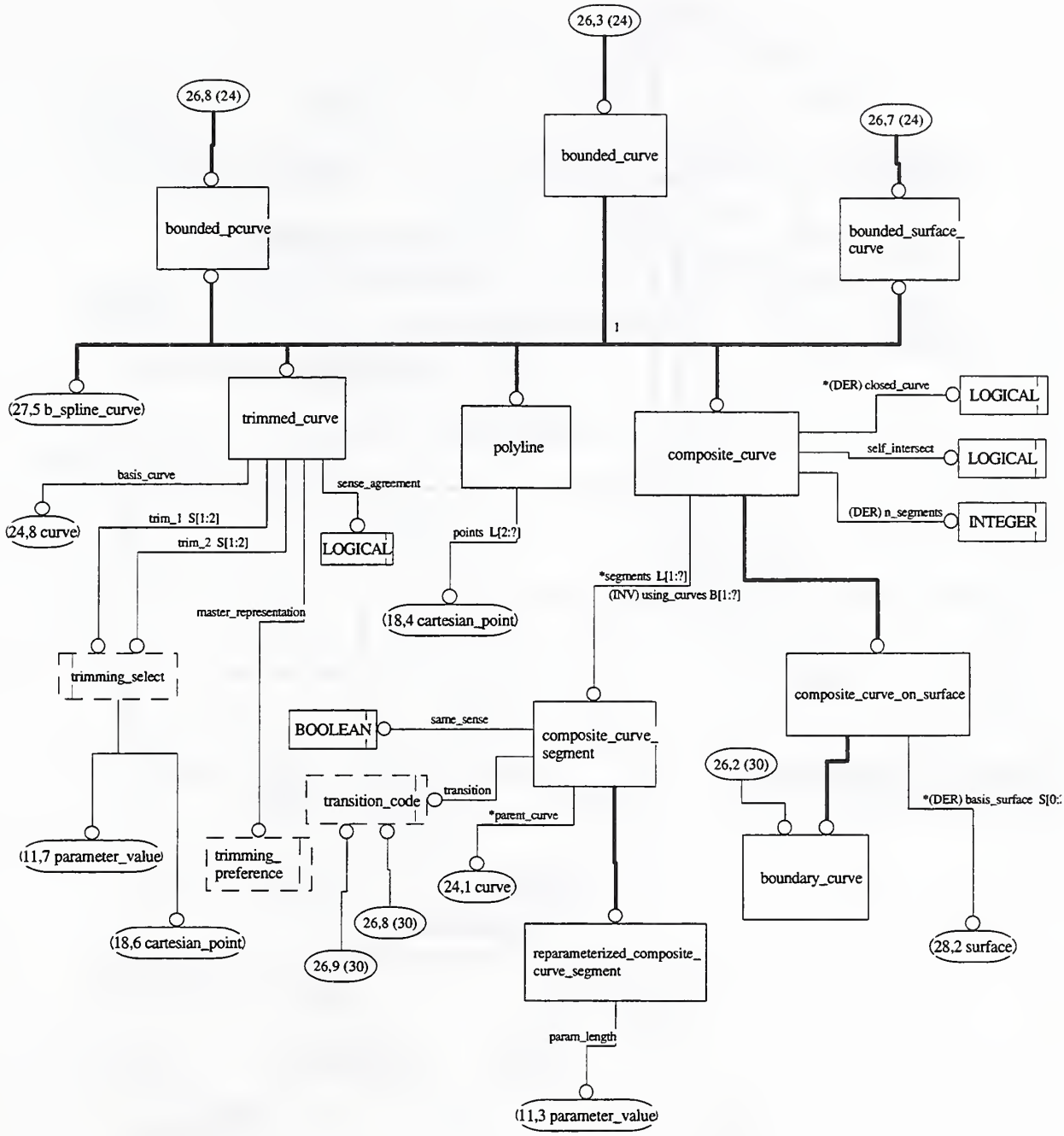


Figure H.26 - AIM EXPRESS-G diagram 26 of 32

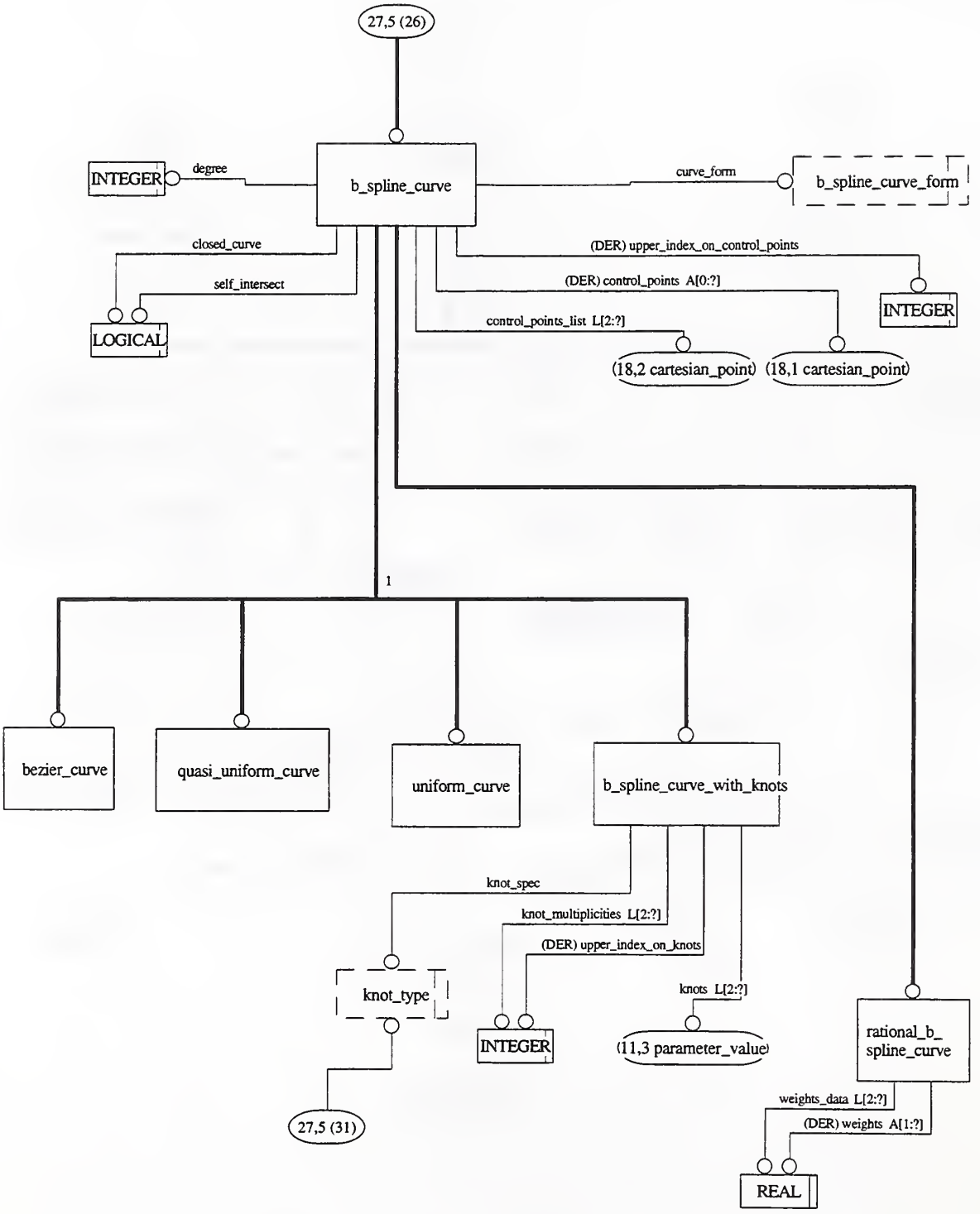


Figure H.27 - AIM EXPRESS-G diagram 27 of 32

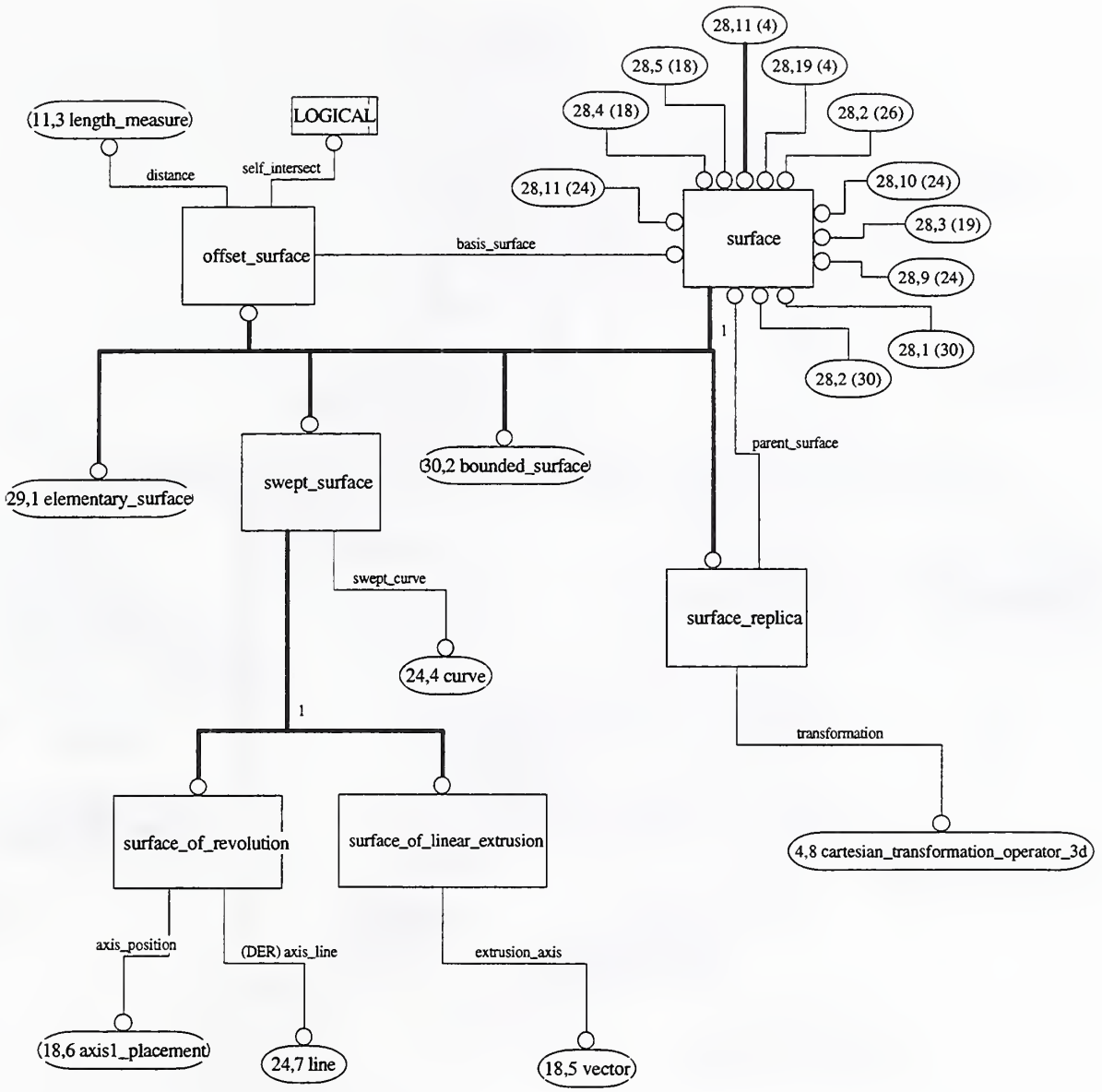


Figure H.28 - AIM EXPRESS-G diagram 28 of 32

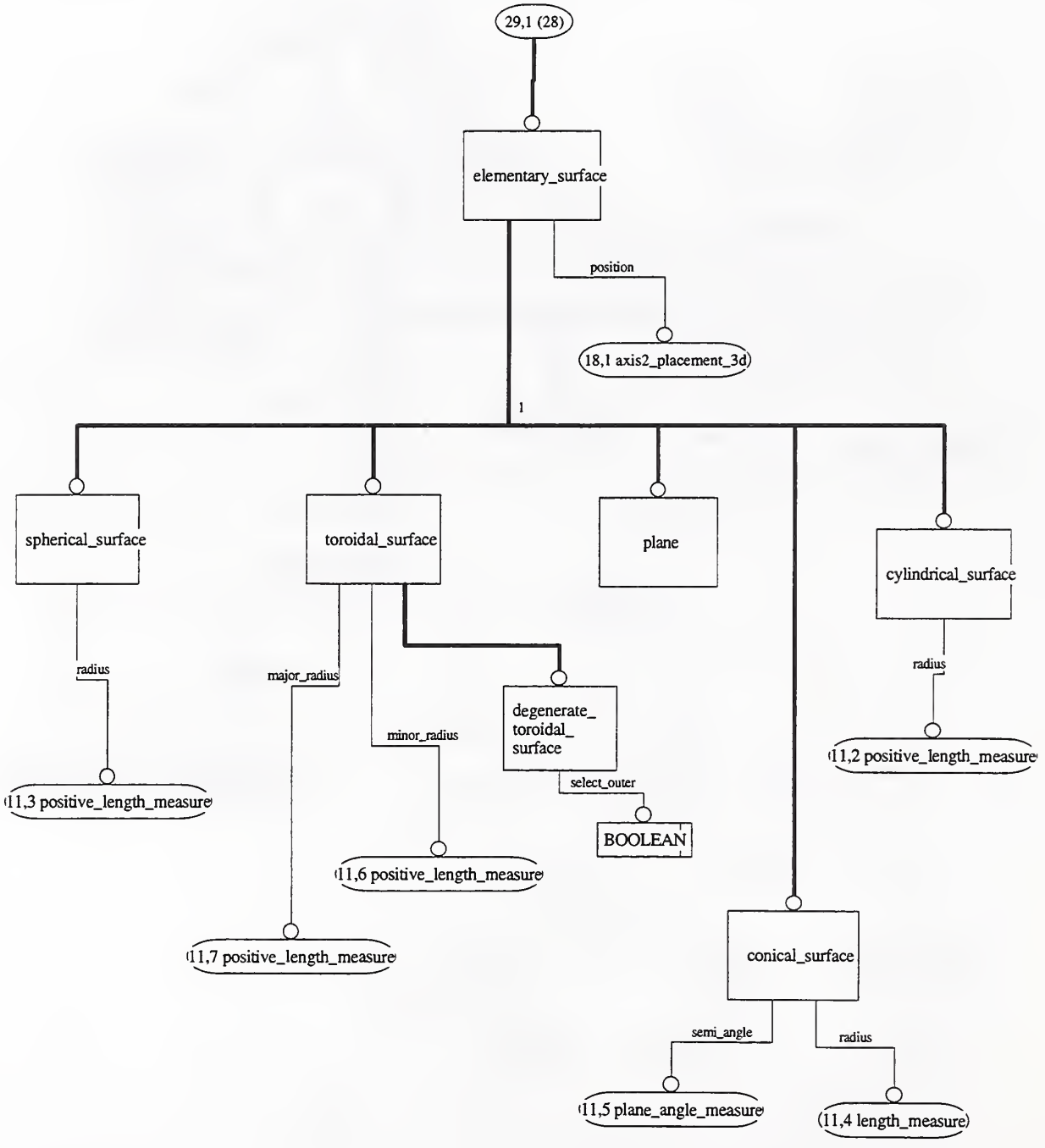


Figure H.29 - AIM EXPRESS-G diagram 29 of 32

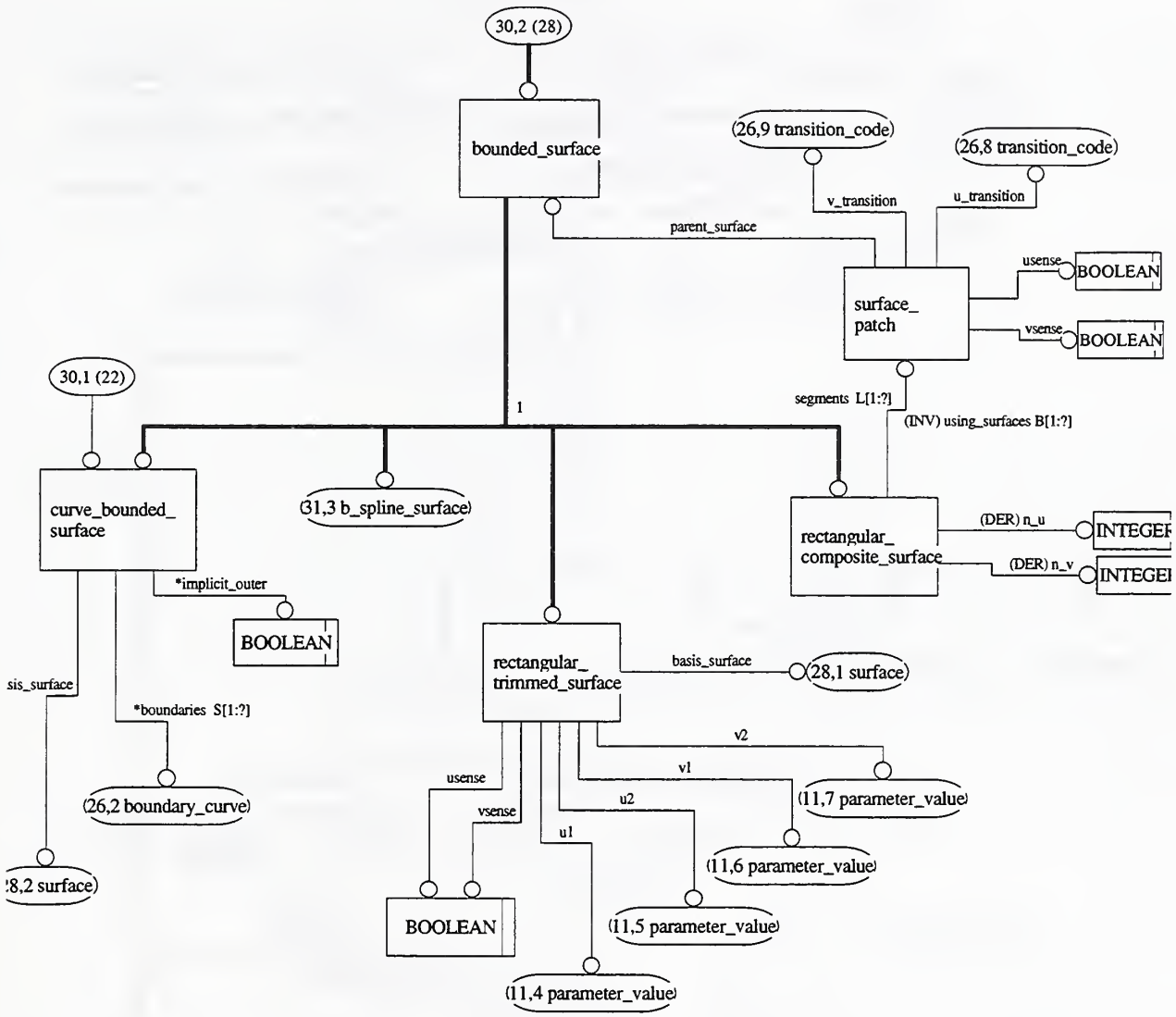


Figure H.30 - AIM EXPRESS-G diagram 30 of 32

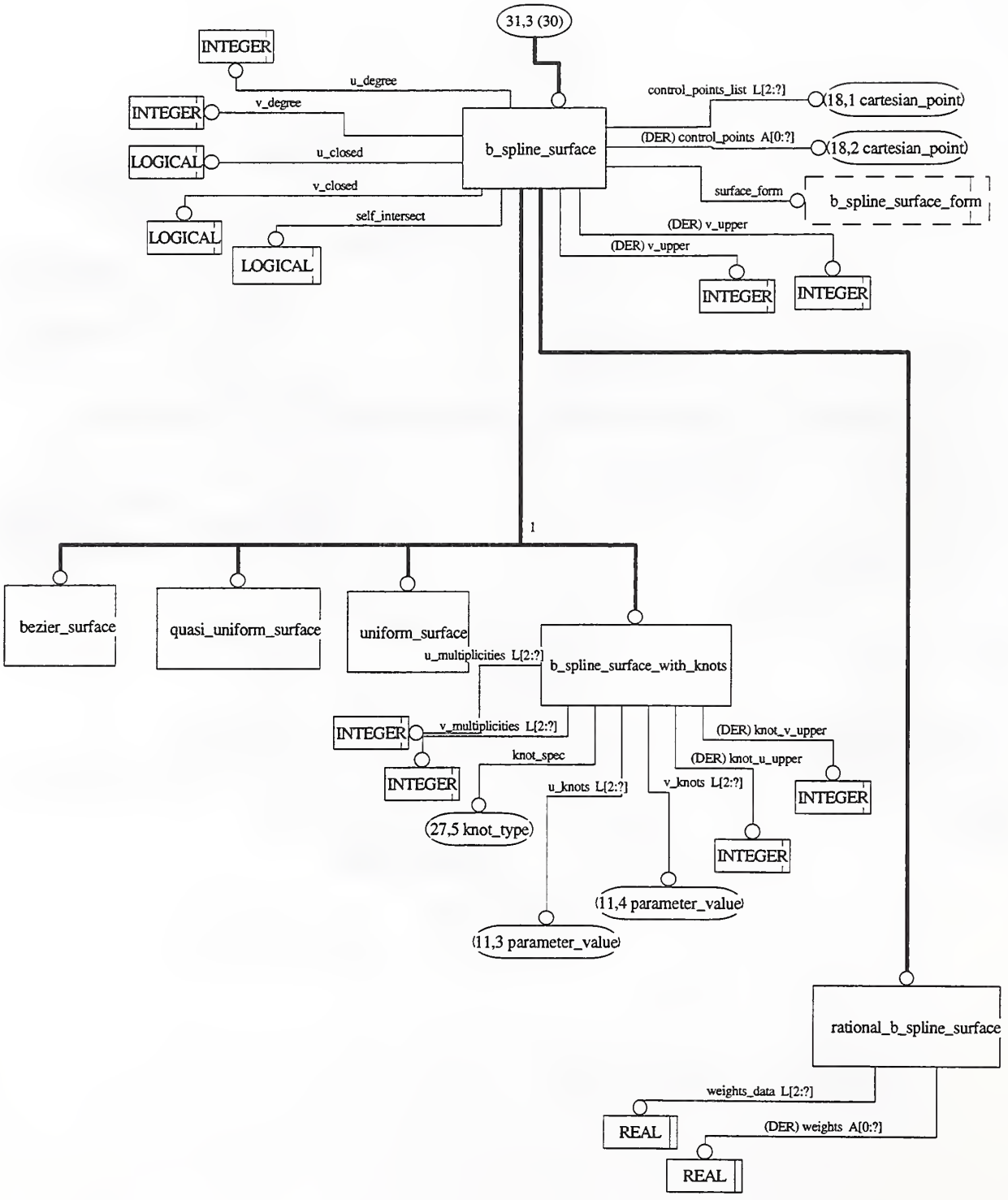


Figure H.31 - AIM EXPRESS-G diagram 31 of 32

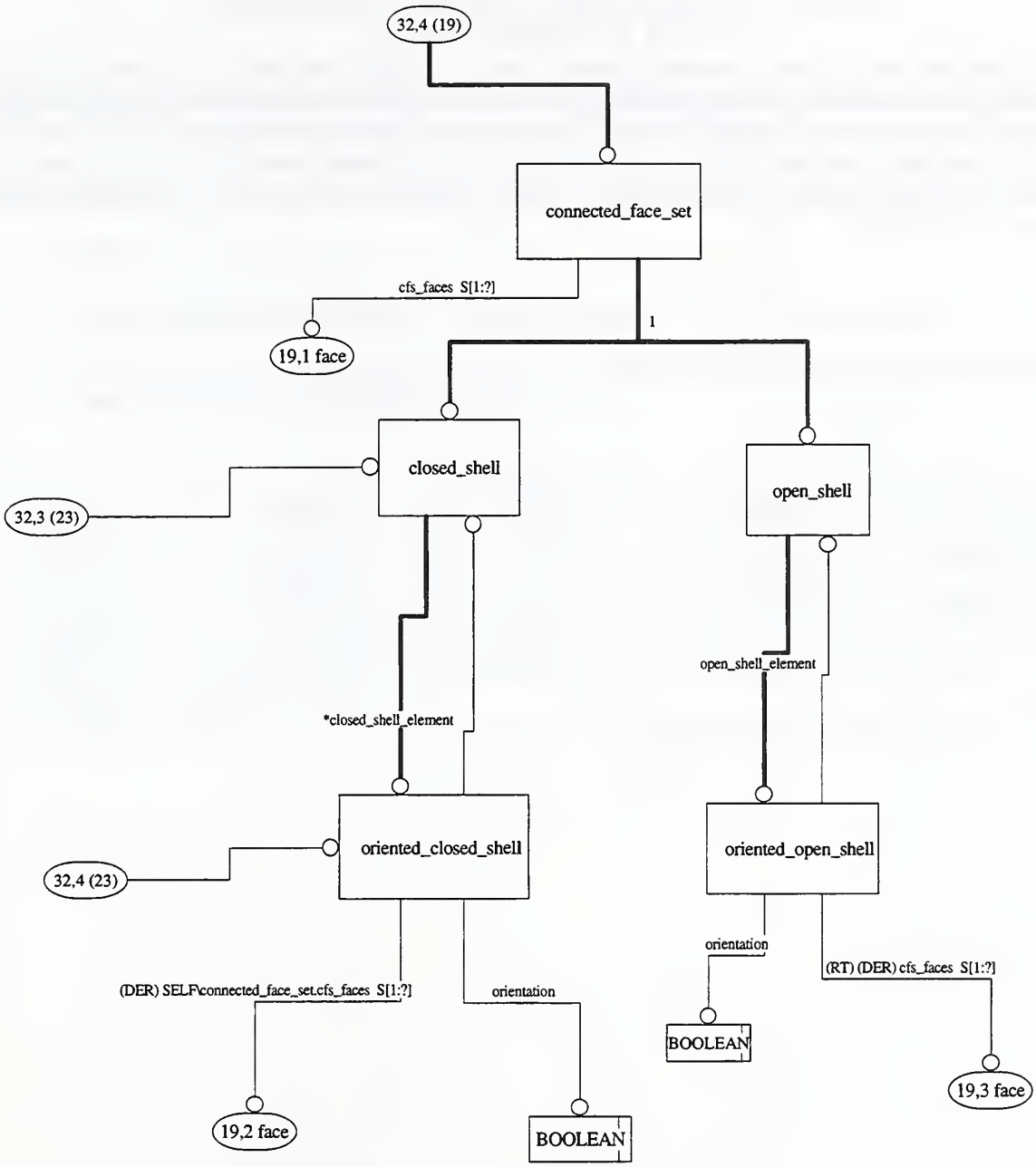


Figure H.32 - AIM EXPRESS-G diagram 32 of 32

Annex J
(informative)

AIM EXPRESS listing

This annex provides a listing of the EXPRESS specified in the AIM of this part of ISO 10303. No text or annotation is included. This annex is provided only in computer-interpretable form.

NOTE - The information provided on this diskette is informative; the normative text is that contained in the body of this part of ISO 10303.

Annex K
(informative)

Application reference model wallpaper version

This annex provides a "wallpaper" version of the application reference model for the exchange of plant spatial configuration information. This application reference model presents a graphical representation of the structure and constraints of the application objects specified in clause 4 of this part of ISO 10303. This version of the application reference model is structured for pasting together to facilitate viewing of the complete model. The application reference model is independent of any implementation method.

NOTES

1 - The application reference model is represented using the IDEF1X modelling language.

2 - The application reference model is presented in a tiled diagram format. The complete diagram may be assembled according to the following template:

Figure K.1	Figure K.2	Figure K.3	Figure K.4	Figure K.5
Figure K.6	Figure K.7	Figure K.8	Figure K.9	Figure K.10
Figure K.11	Figure K.12	Figure K.13	Figure K.14	Figure K.15
Figure K.16	Figure K.17	Figure K.18		
Figure K.19				

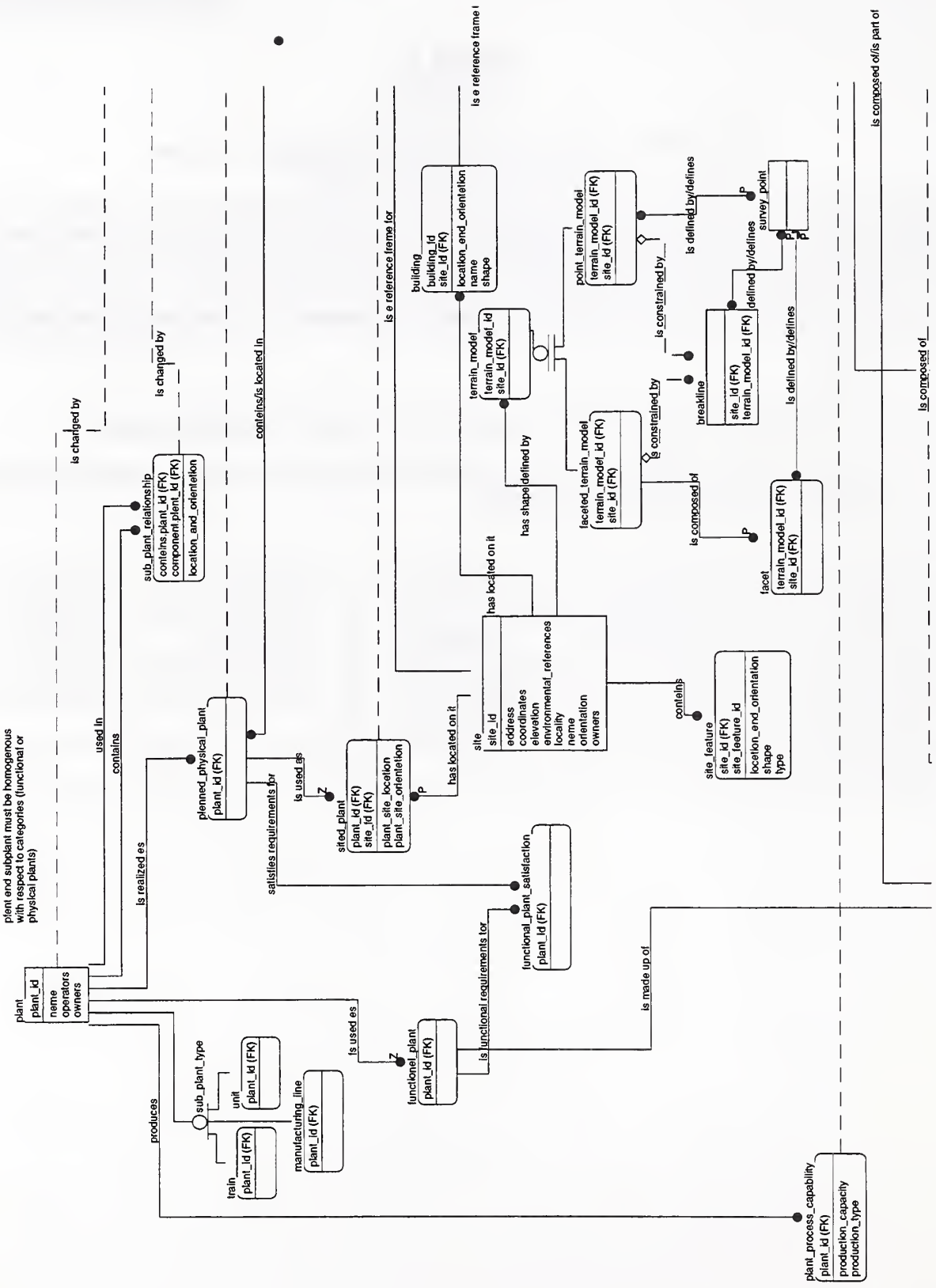


Figure K.1 - ARM diagram 1 of 19 in IDEF1X

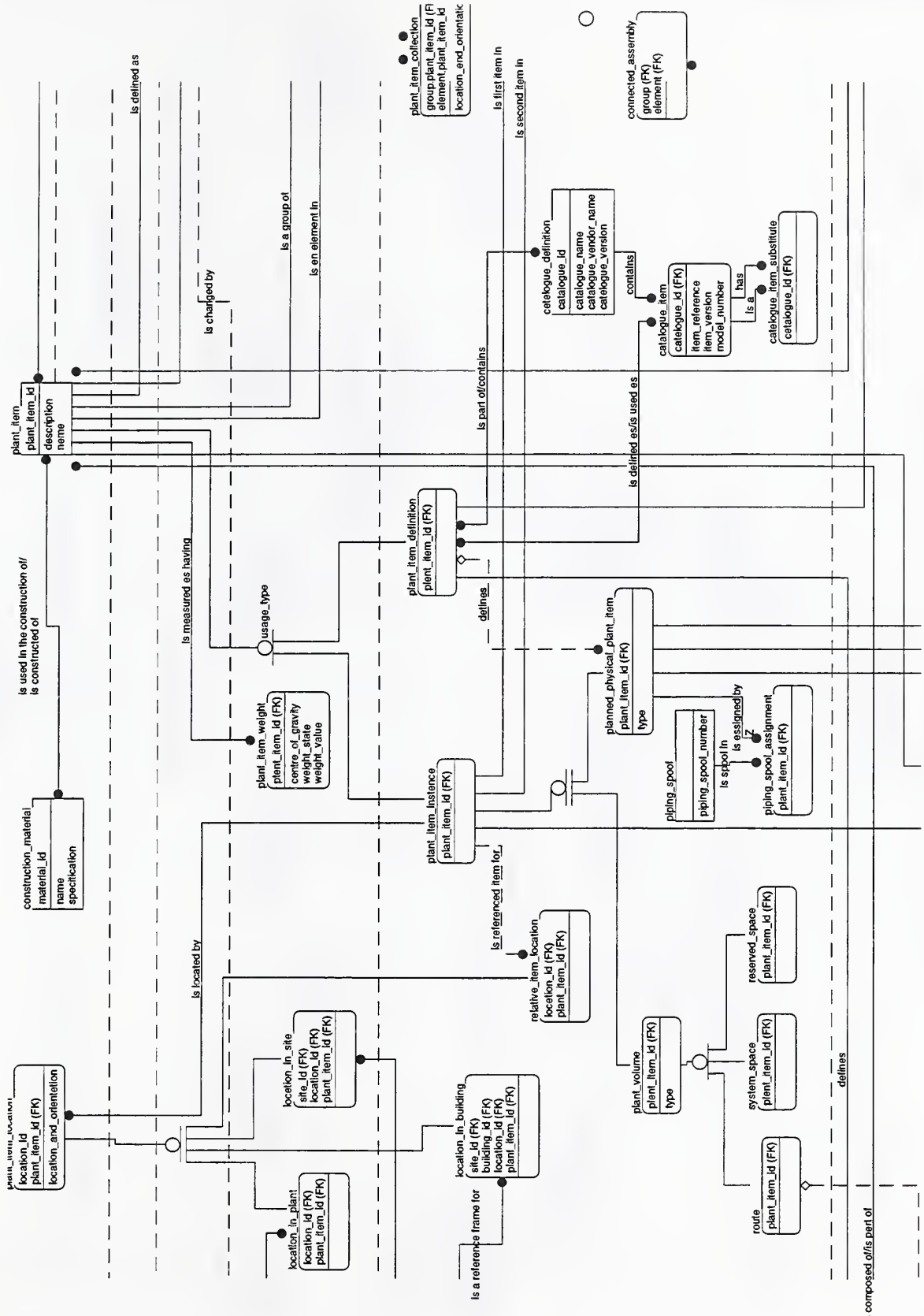


Figure K.2 - ARM diagram 2 of 19 in IDEF1X

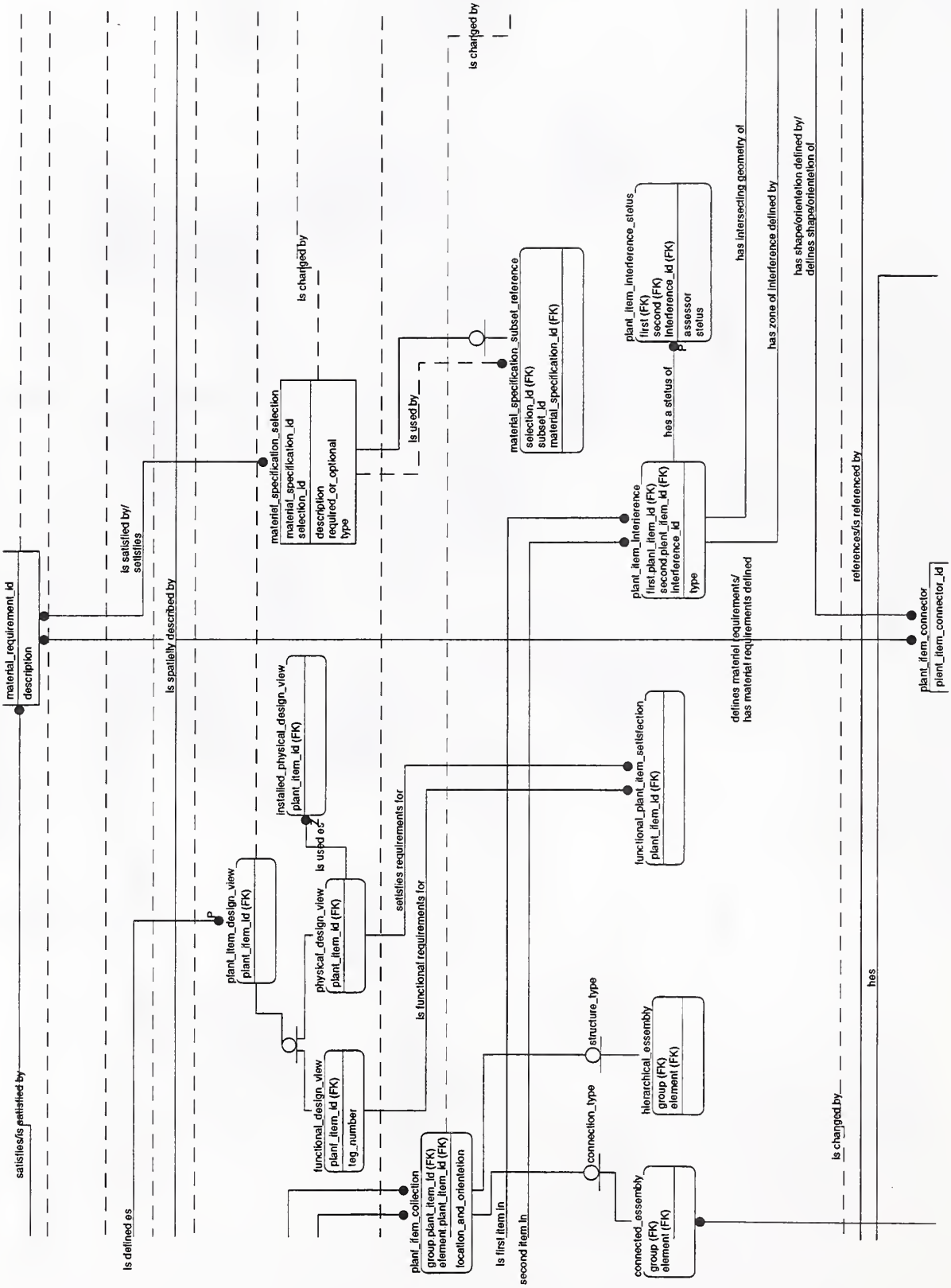


Figure K.3 - ARM diagram 3 of 19 in IDEF1X

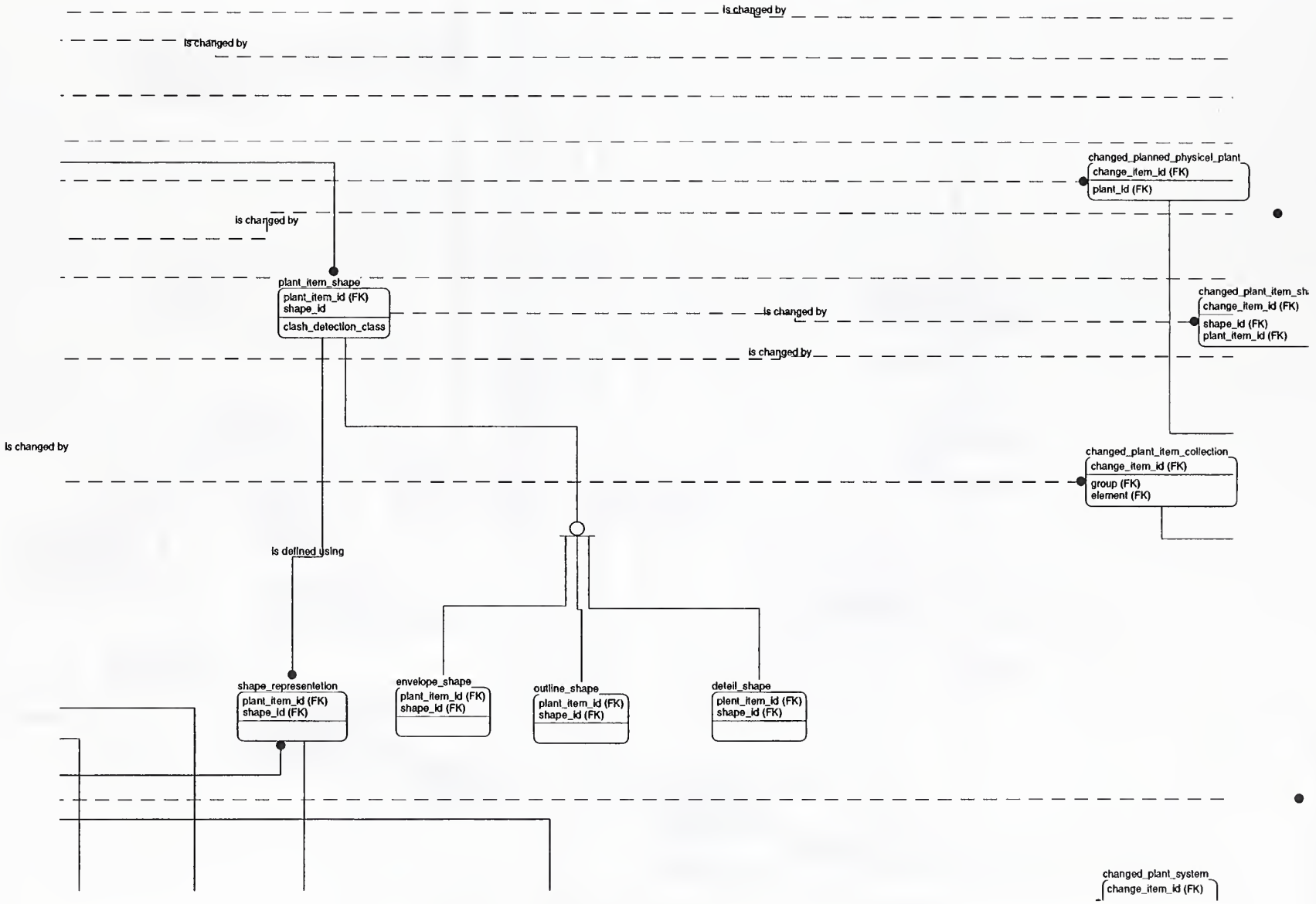


Figure K.4 - ARM diagram 4 of 19 in IDEF1X

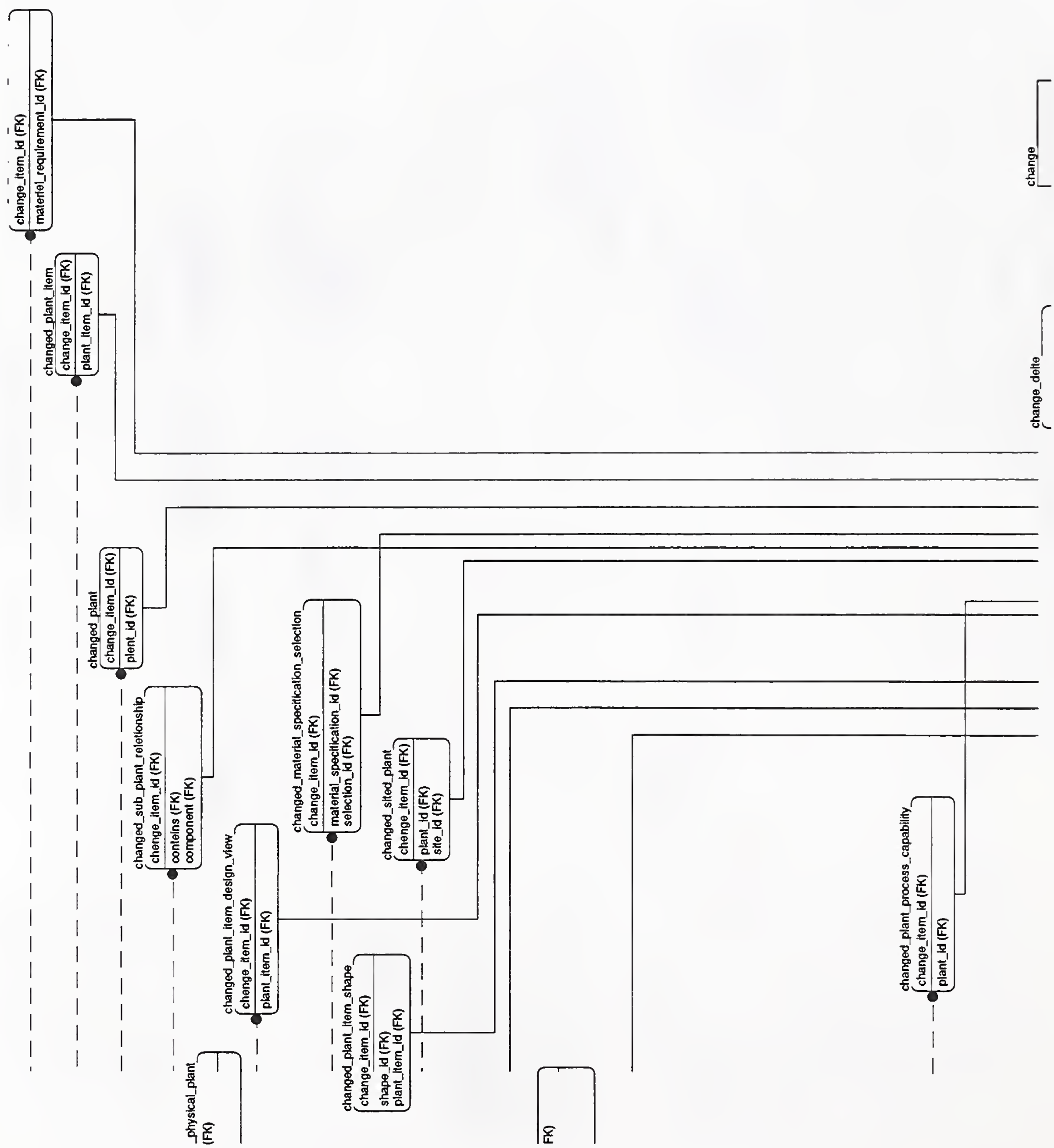


Figure K.5 - ARM diagram 5 of 19 in IDEF1X

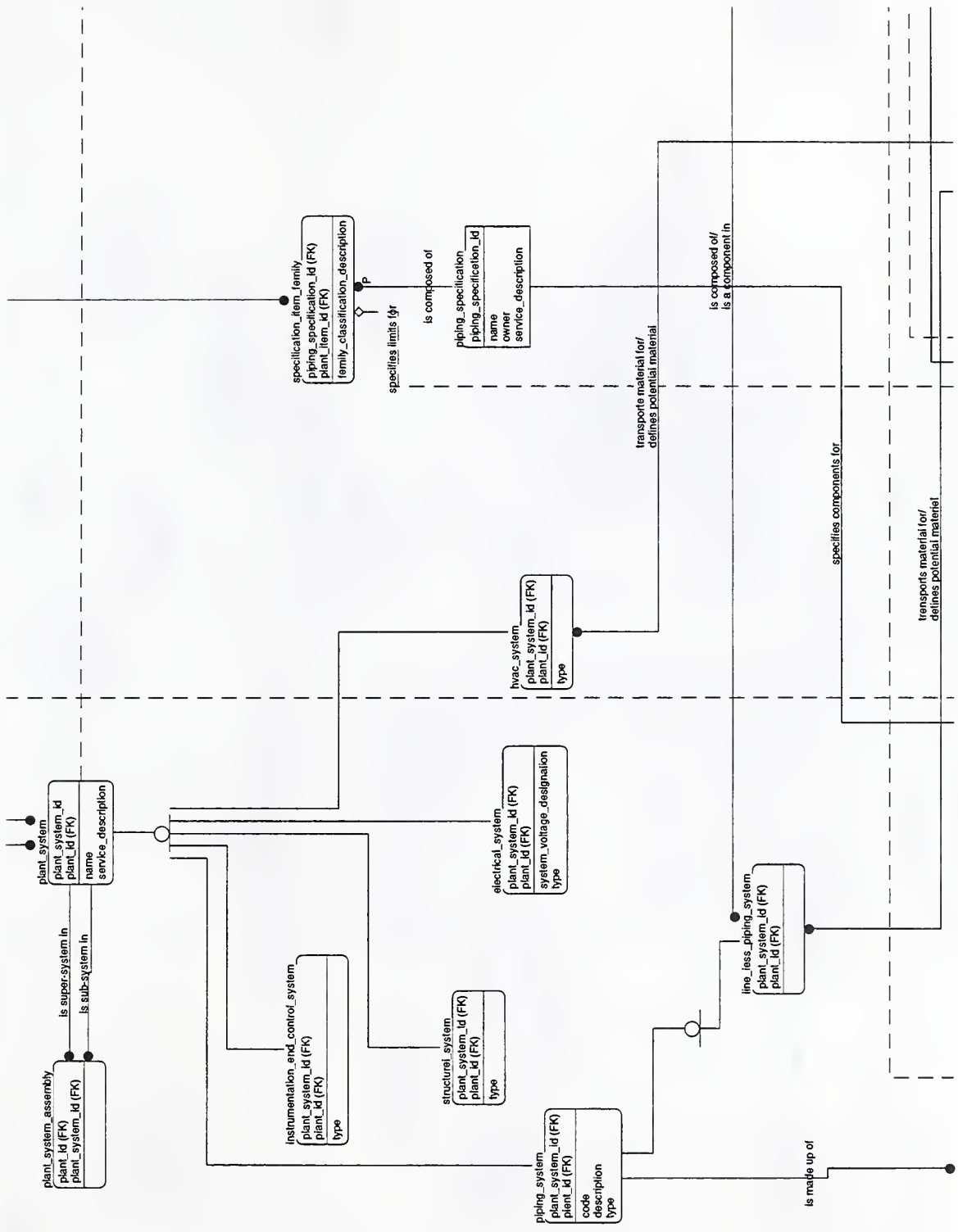


Figure K.6 - ARM diagram 6 of 19 in IDEF1X

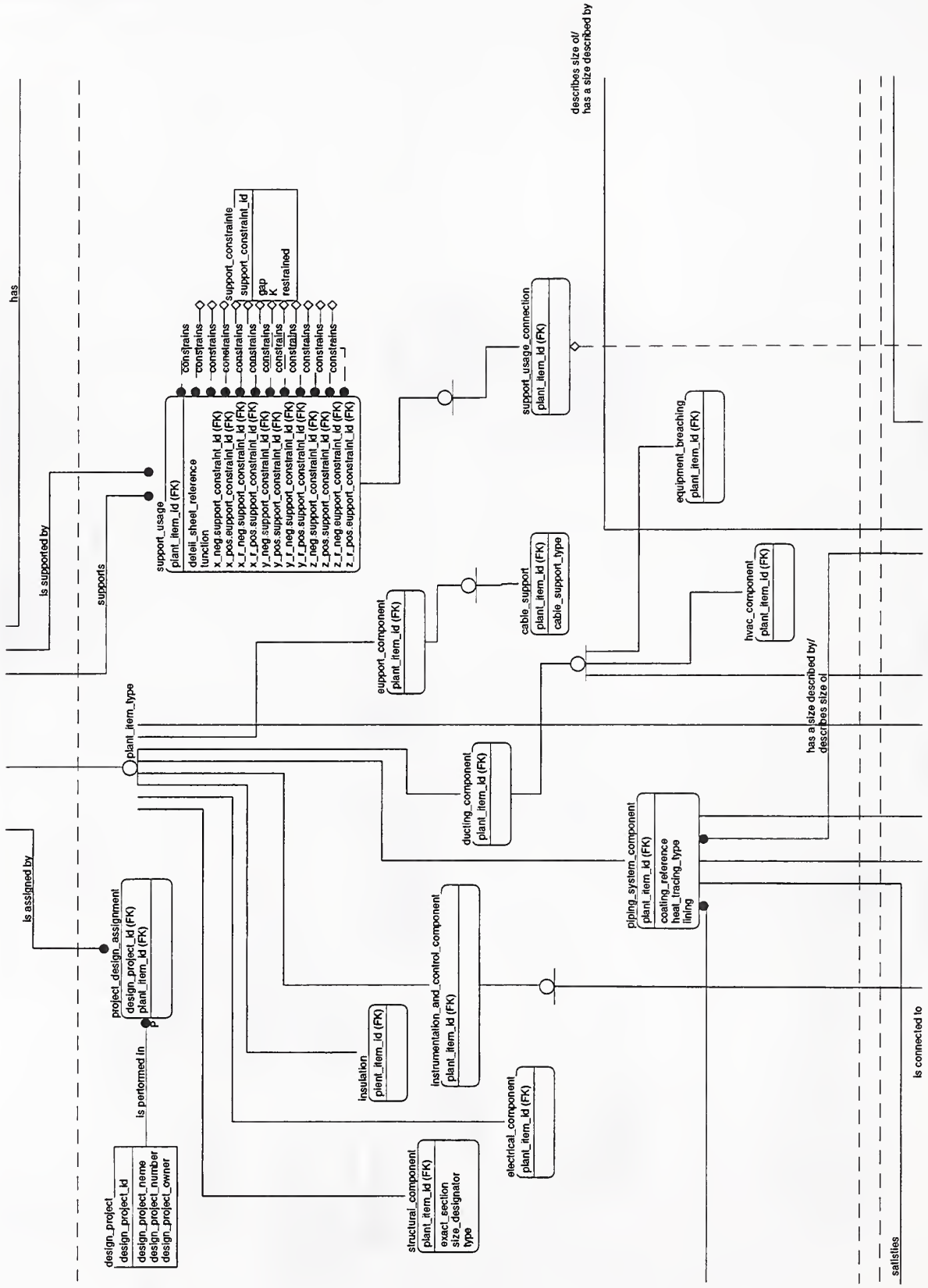


Figure K.7 - ARM diagram 7 of 19 in IDEF1X

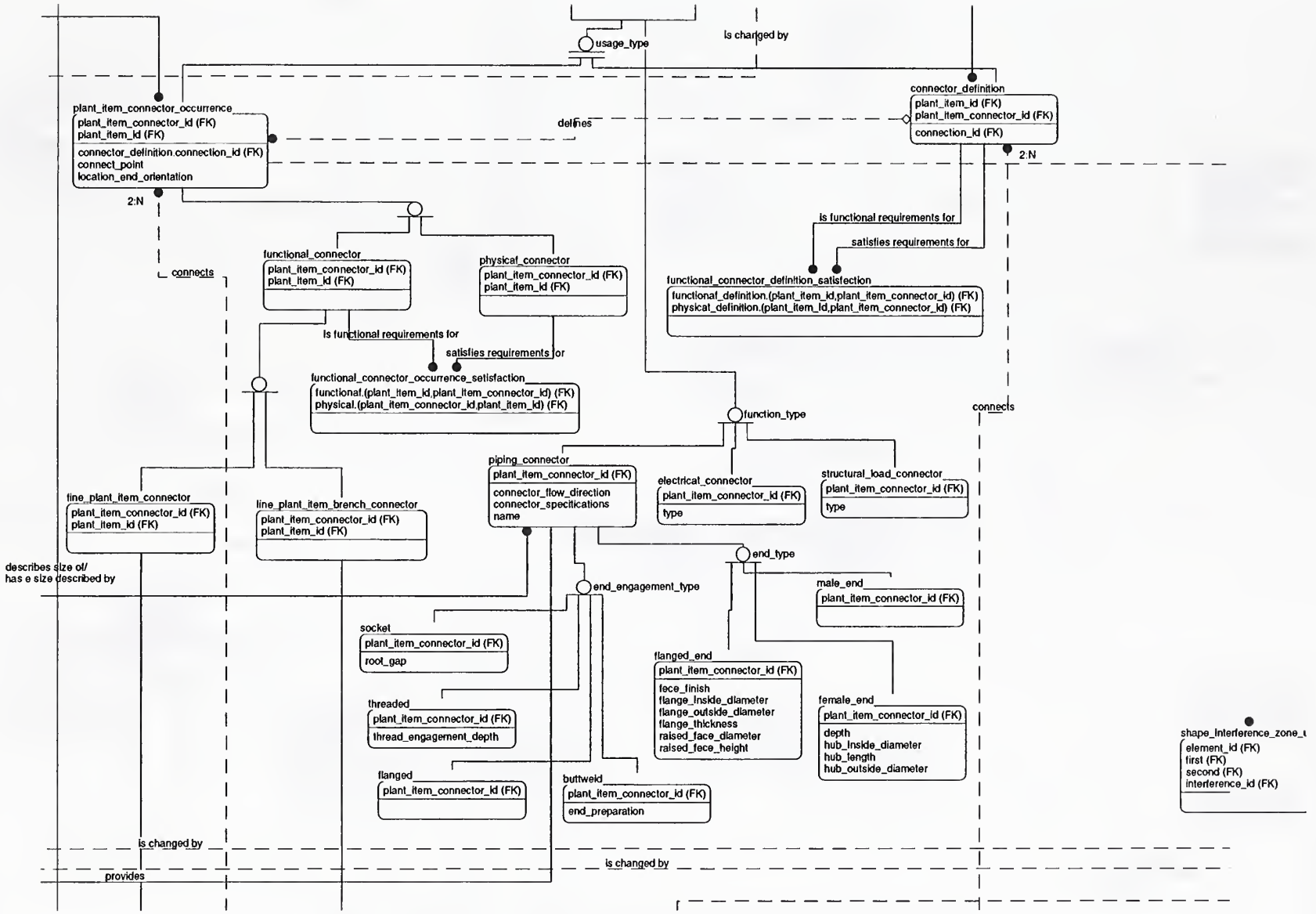


Figure K.8 - ARM diagram 8 of 19 in IDEF1X

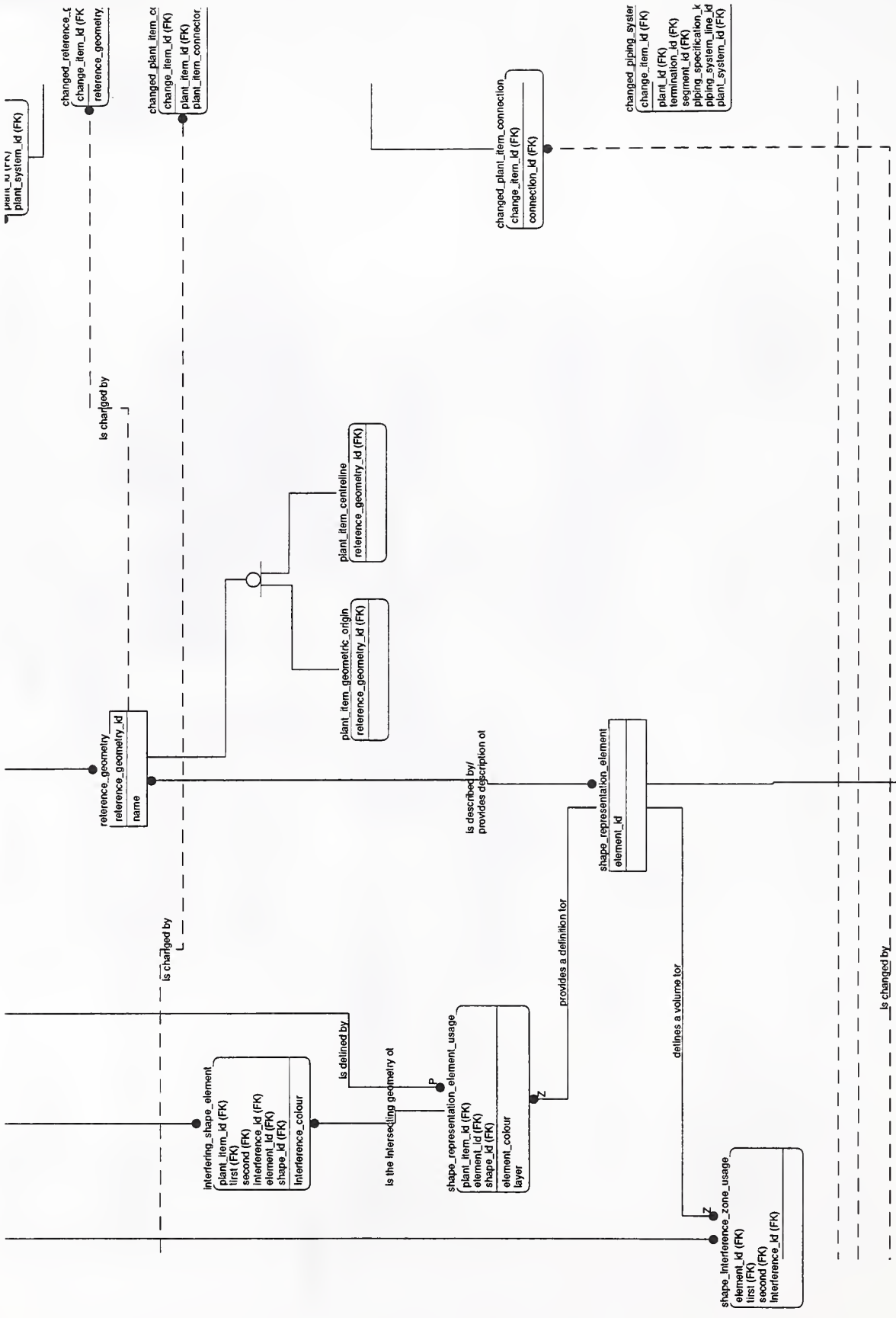


Figure K.9 - ARM diagram 9 of 19 in IDEF1X

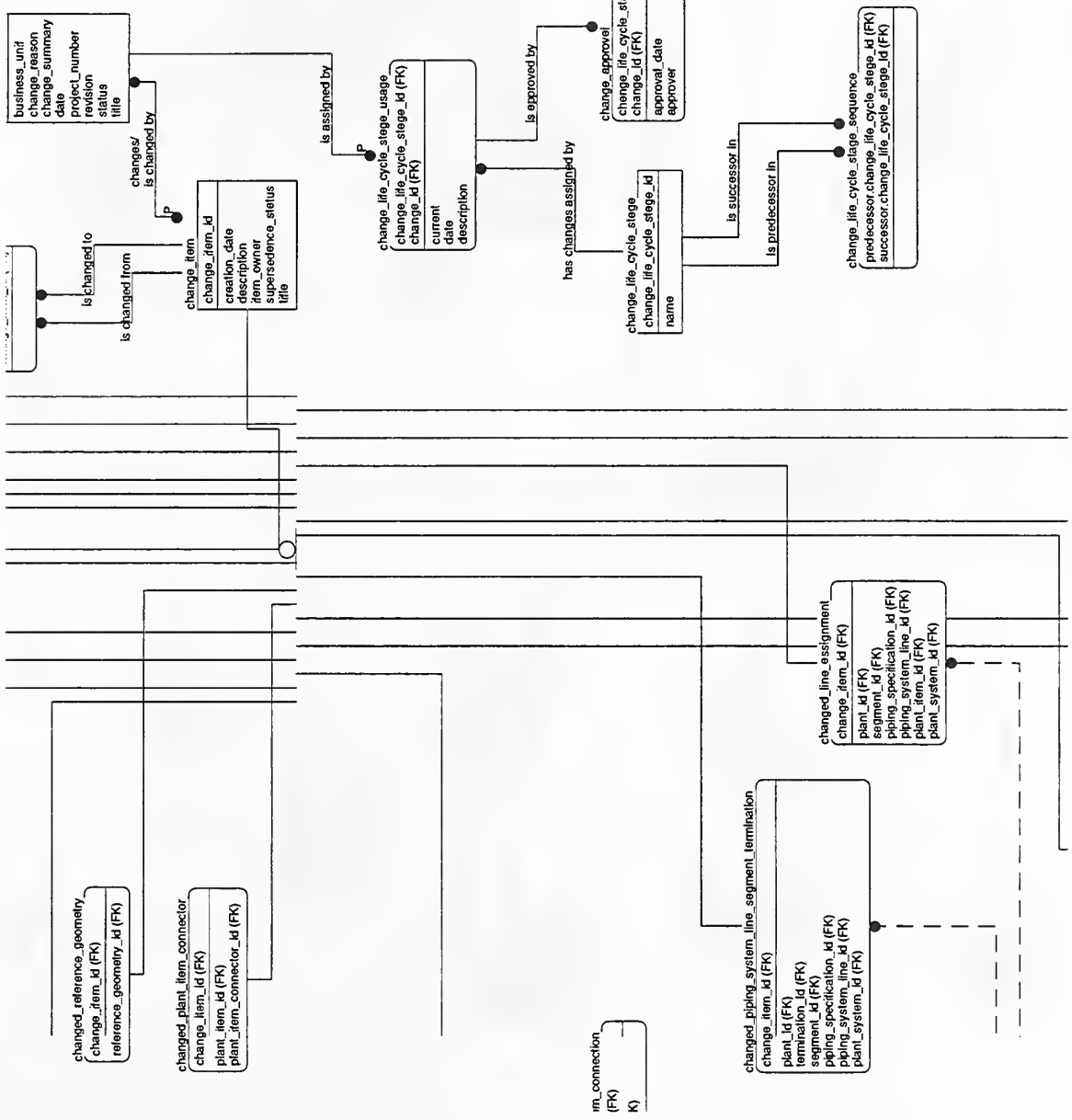


Figure K.10 - ARM diagram 10 of 19 in IDEF1X

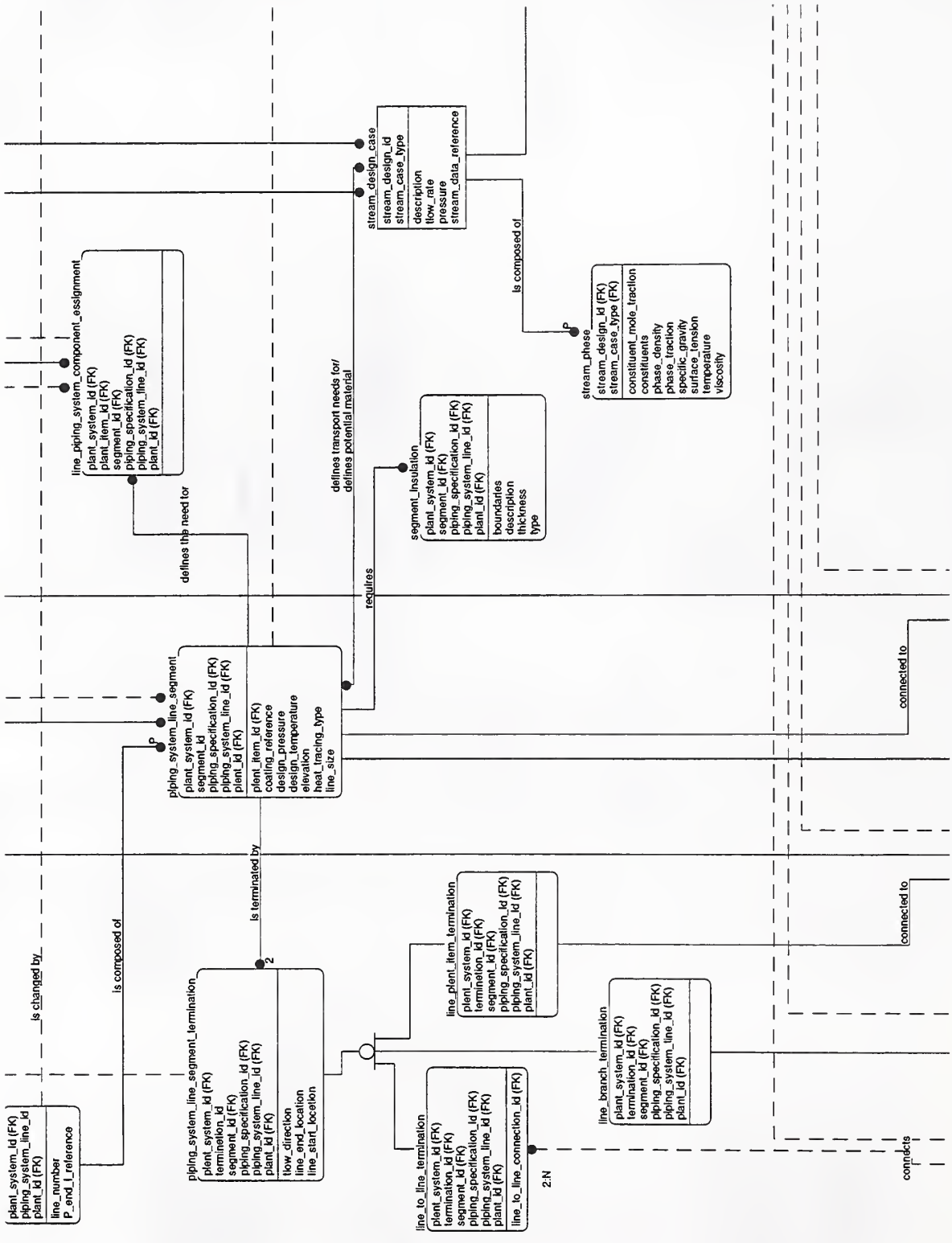


Figure K.11 - ARM diagram 11 of 19 in IDEF1X

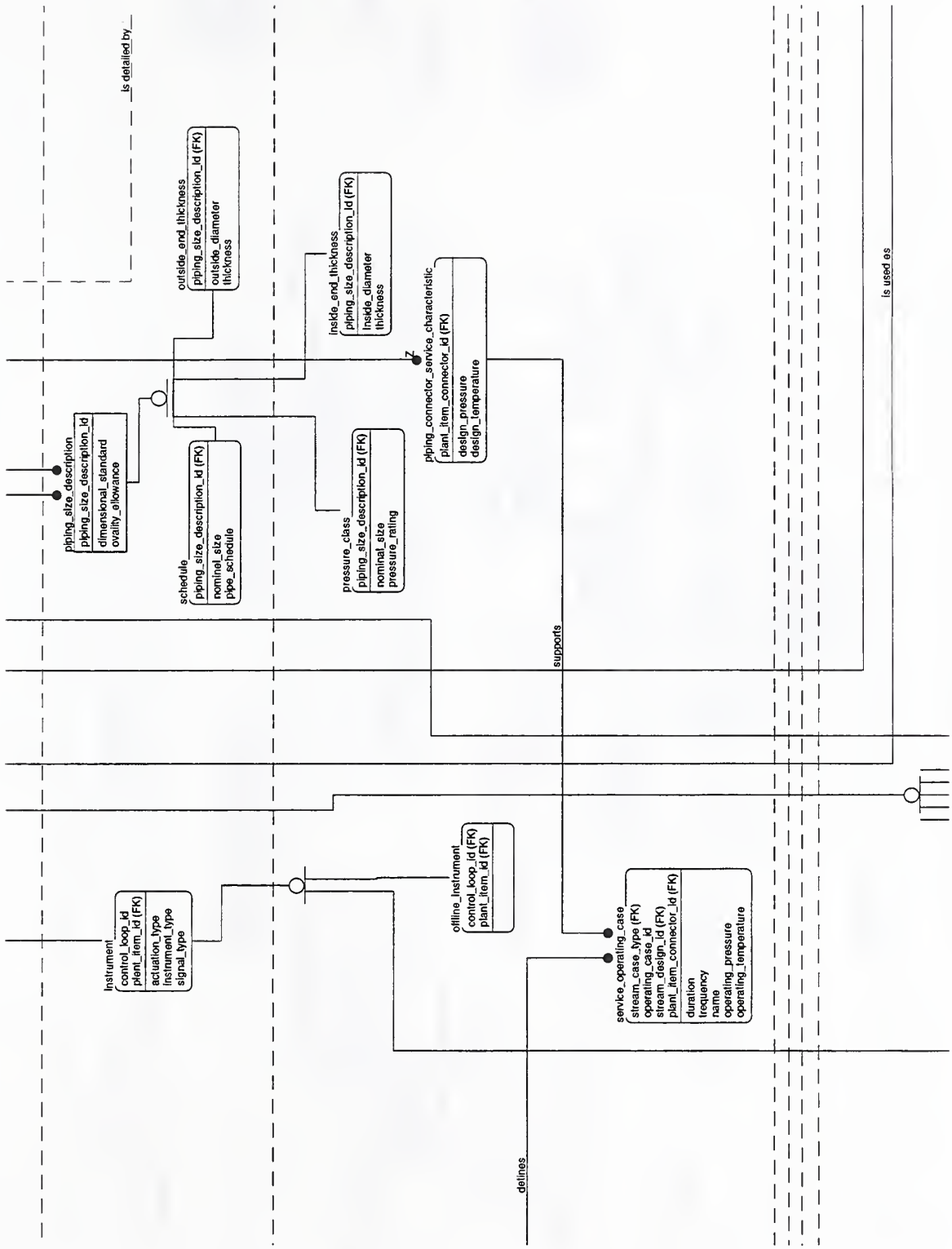


Figure K.12 - ARM diagram 12 of 19 in IDEF1X

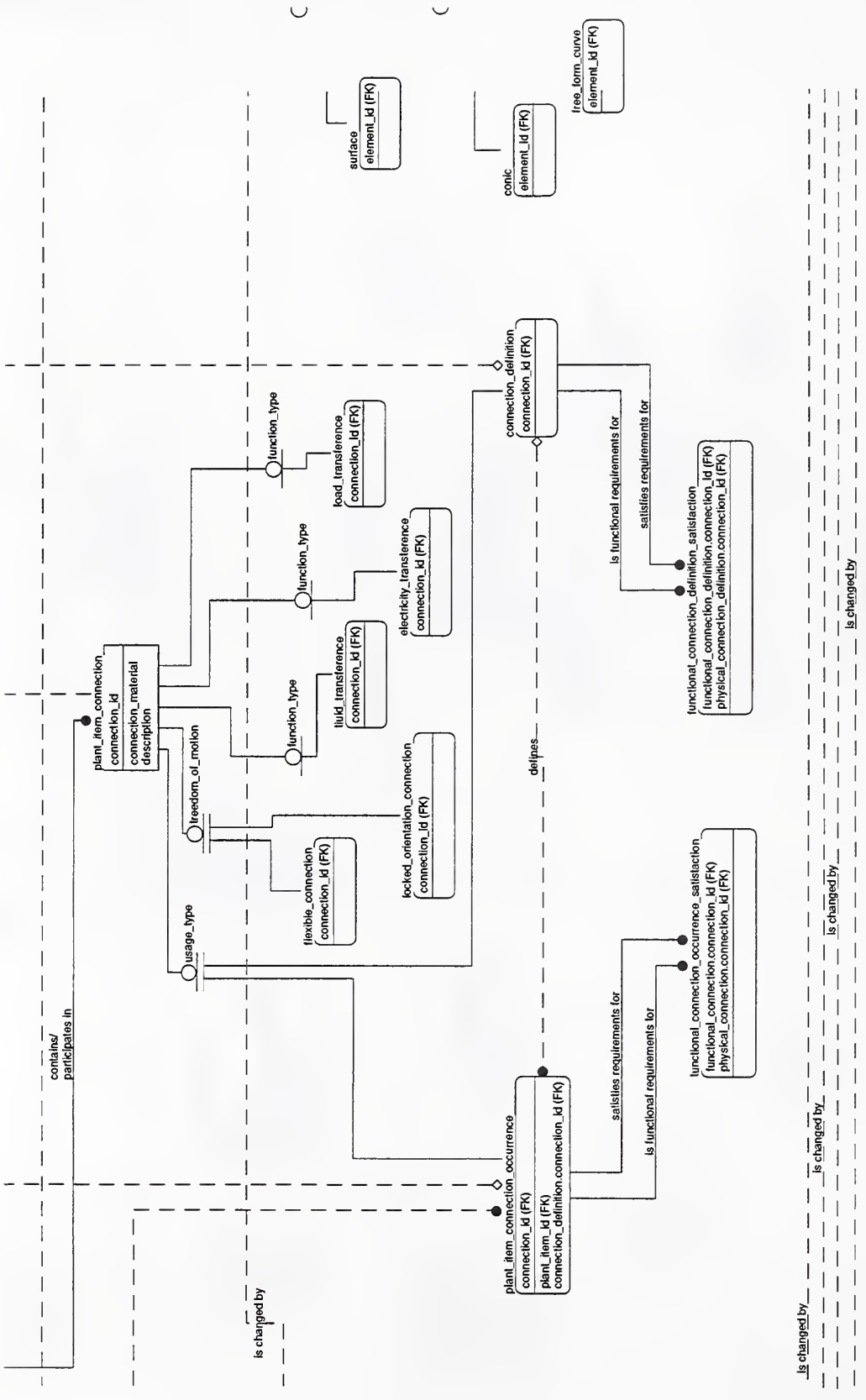


Figure K.13 - ARM diagram 13 of 19 in IDEF1X

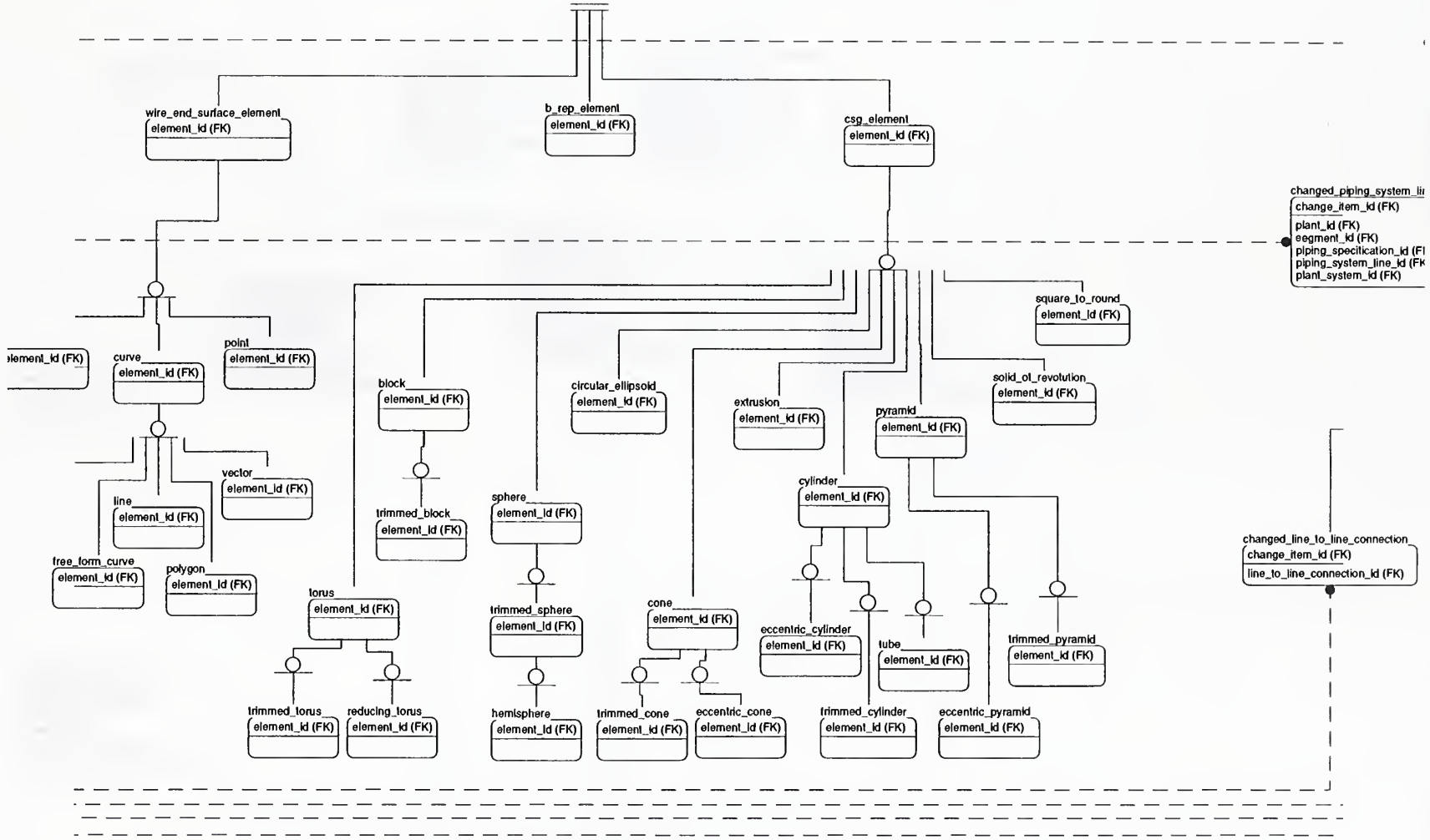


Figure K.14 - ARM diagram 14 of 19 in IDEF1X

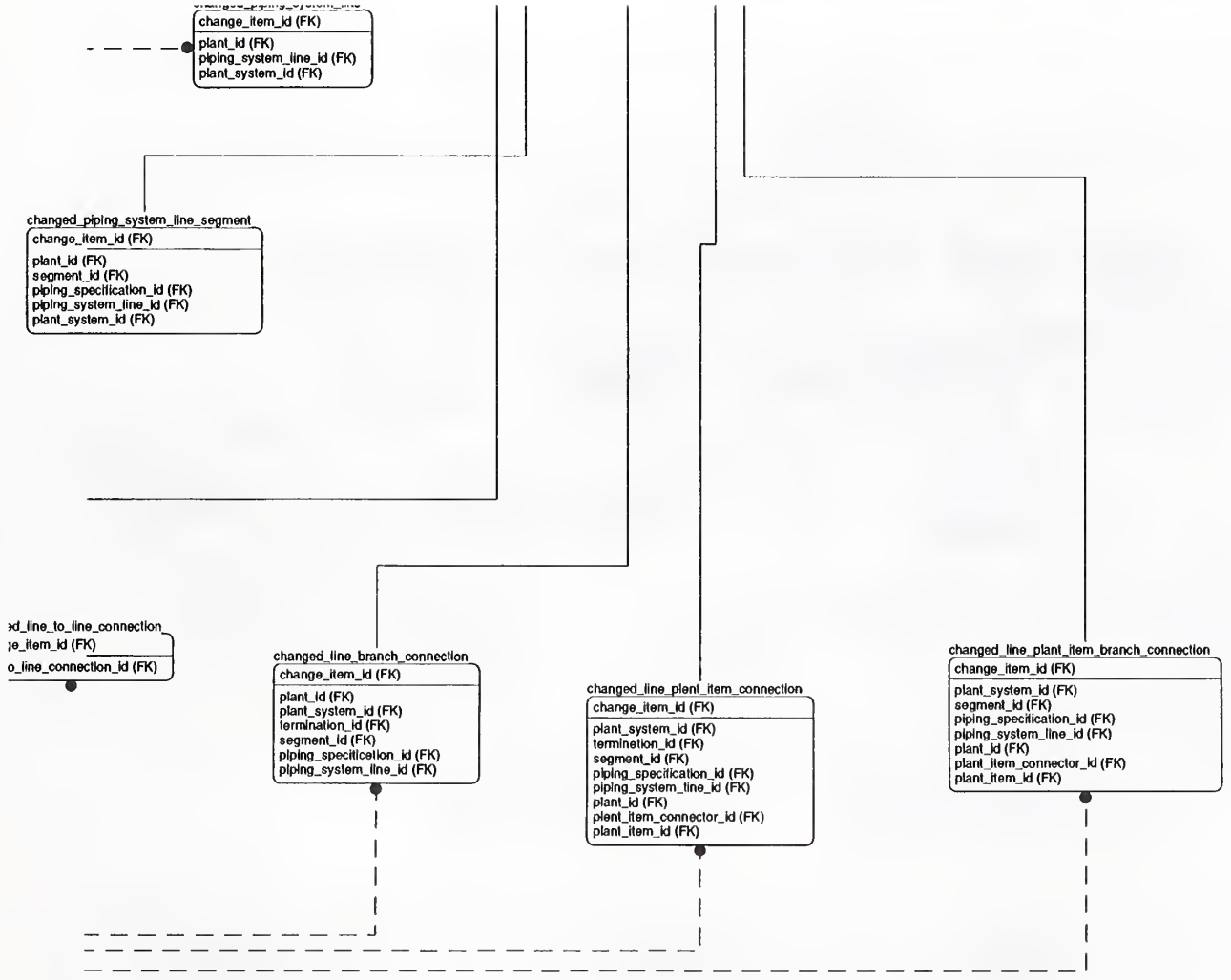


Figure K.15 - ARM diagram 15 of 19 in IDEF1X

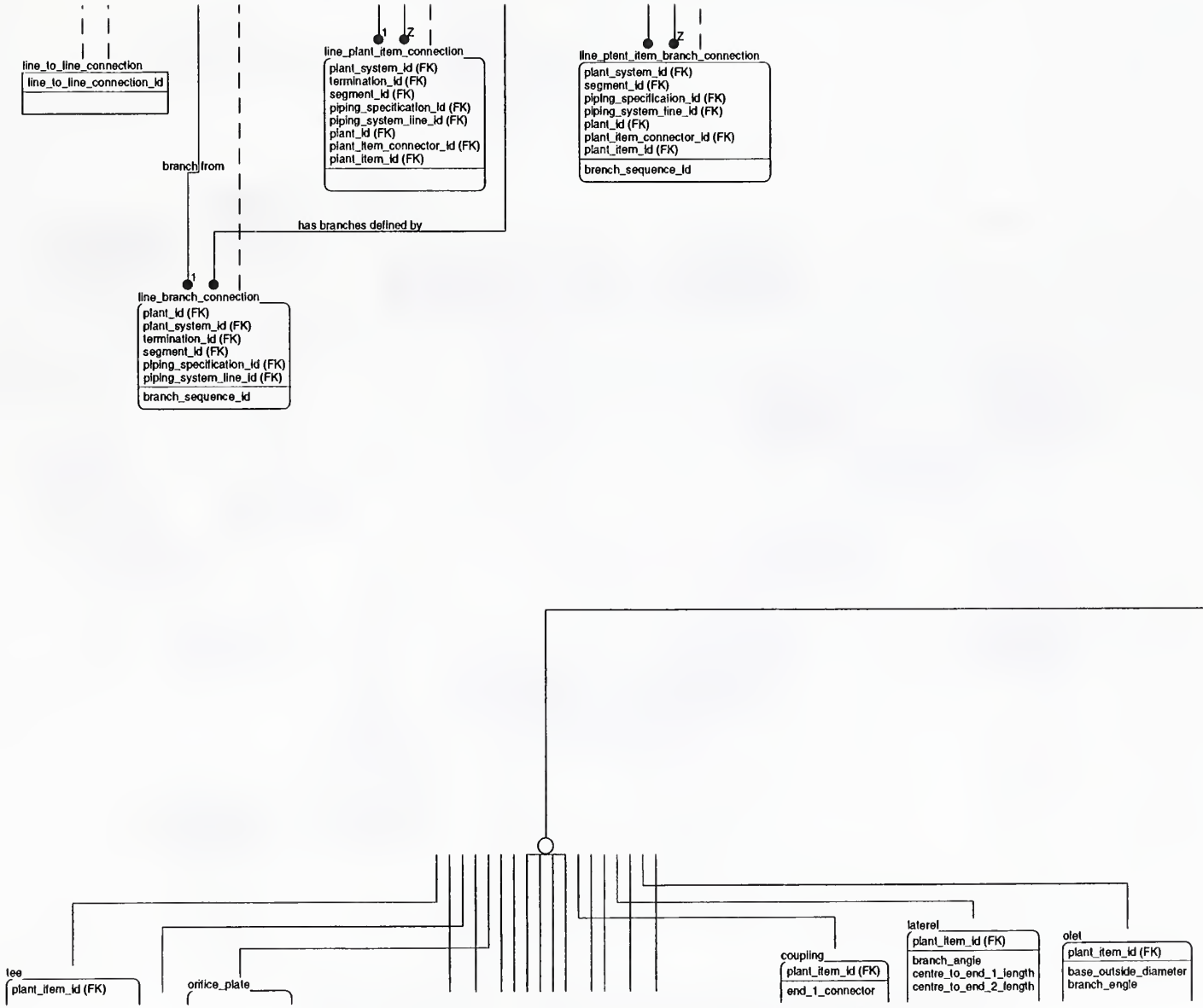


Figure K.16 - ARM diagram 16 of 19 in IDEF1X

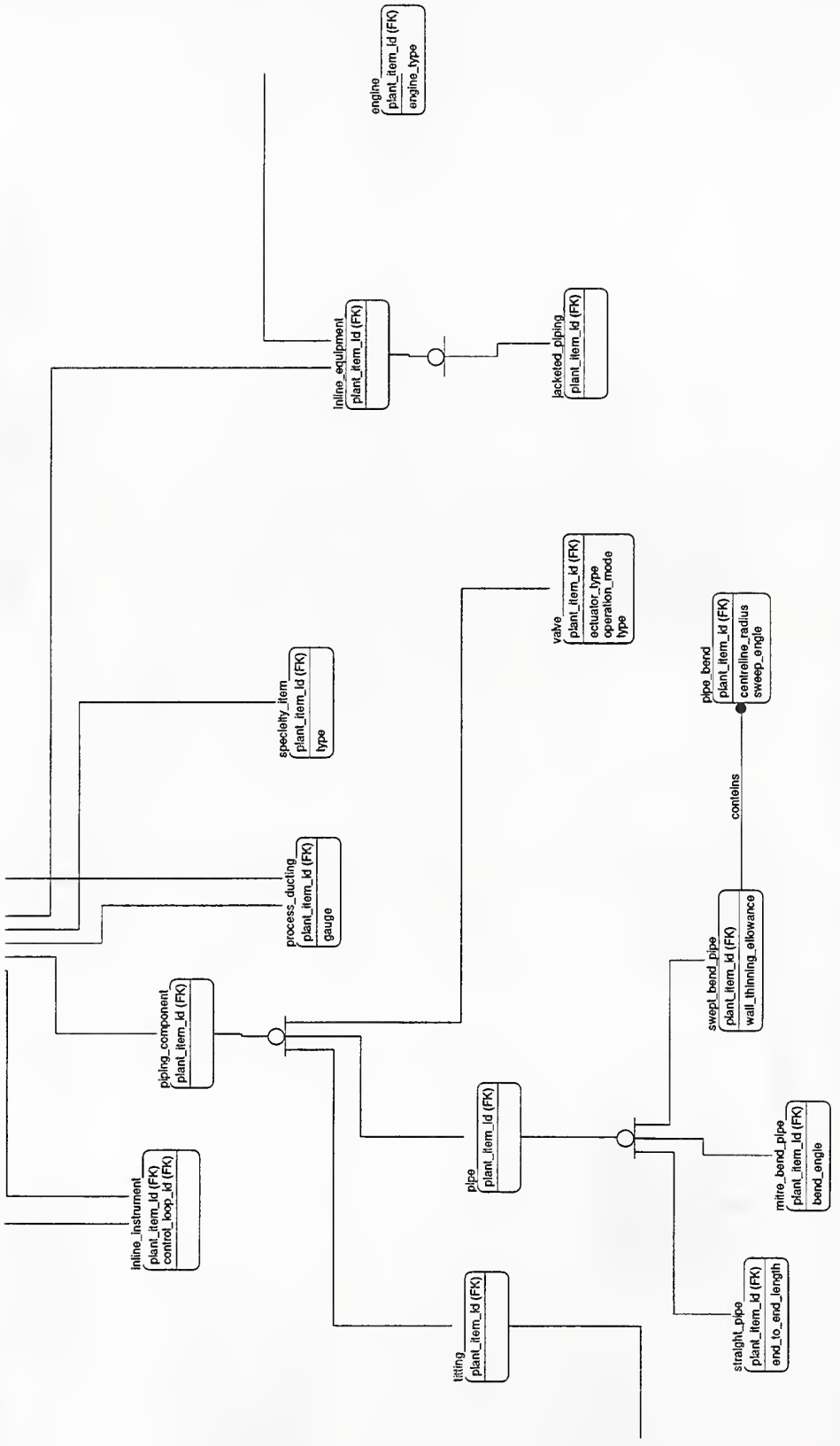


Figure K.17 - ARM diagram 17 of 19 in IDEF1X

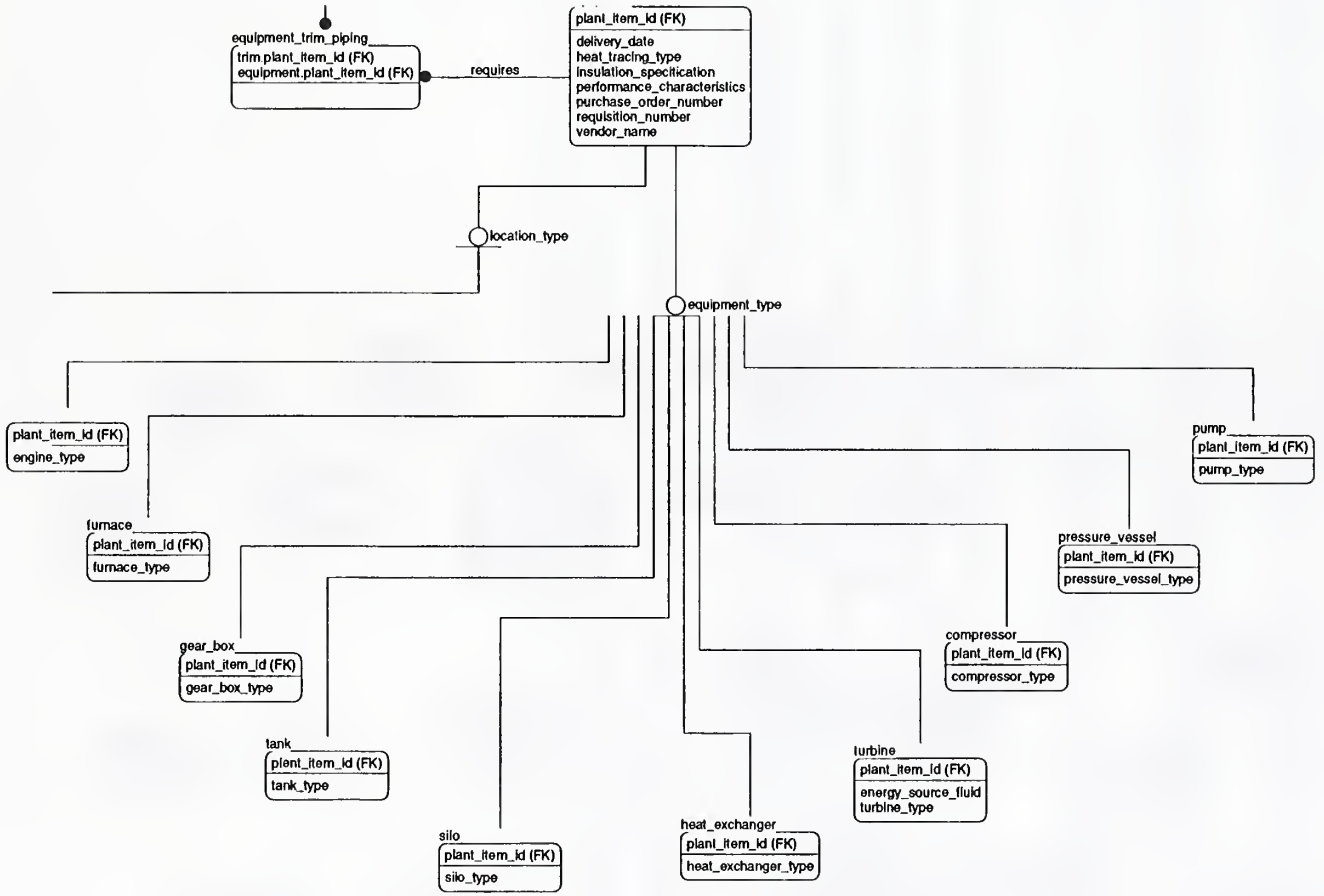


Figure K.18 - ARM diagram 18 of 19 in IDEF1X

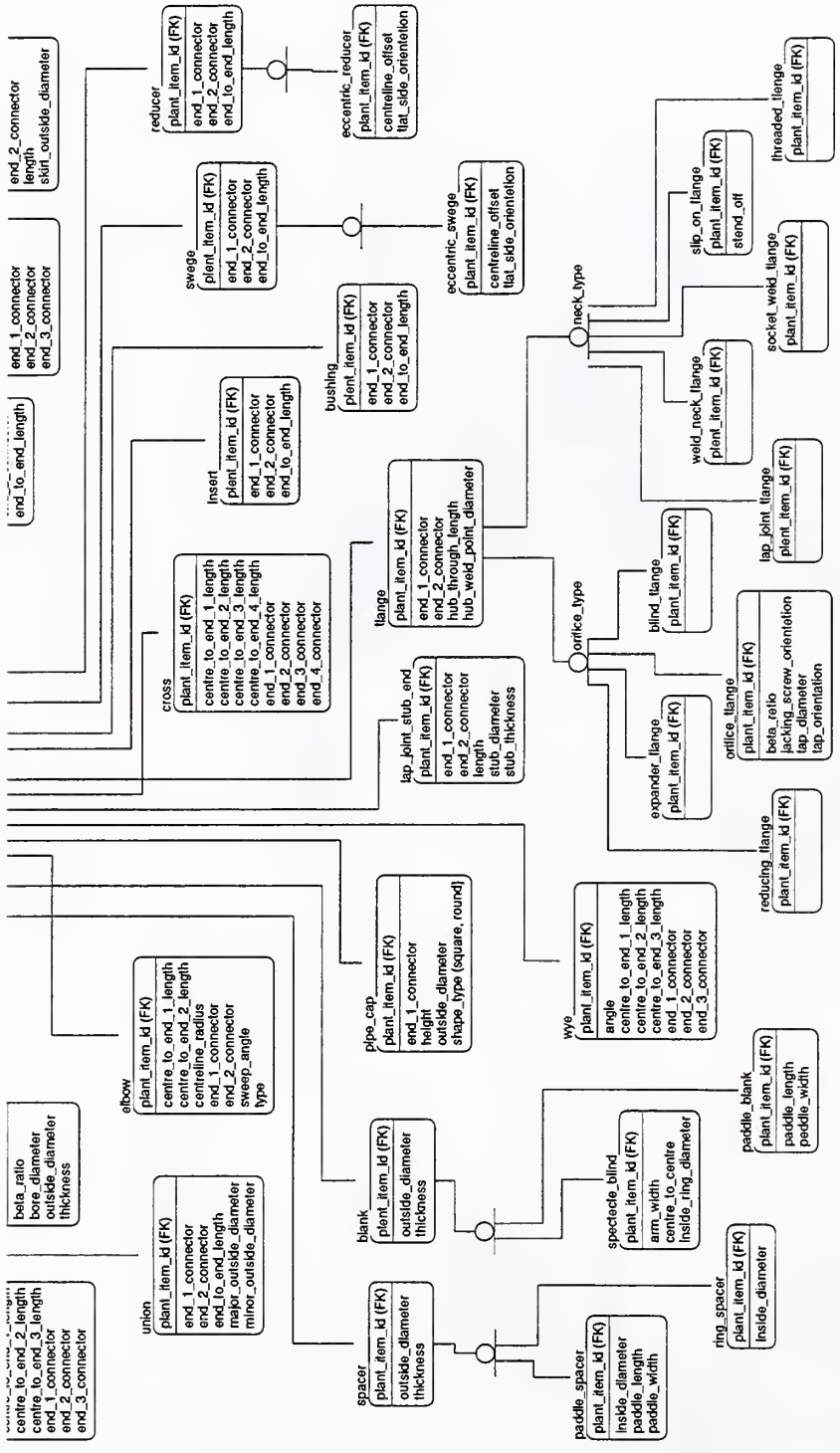


Figure K.19 - ARM diagram 19 of 19 in IDEF1X

Annex L
(informative)

Bibliography

[1]IDEF0 *Federal Information Processing Standards Publication 183, Integration Definition for Function Modeling (IDEF0)*, FIPS PUB 183, National Institute of Standards and Technology, December 1993.

[2]IDEF1X *Federal Information Processing Standards Publication 184, Integration Definition for Information Modeling (IDEF1X)*, FIPS PUB 184, National Institute of Standards and Technology, December 1993.

[3] *Functional Data and their Schematic Representation for Process Plant*, ISO 10303-221, CDC Version, 28 November 1994.

[4] *Cast Iron Pipe Flanges and Flanged Fittings*, ANSI B16.1, American National Standards Institute, 1989.

[5] *Pipe Flanges and Flanged Fittings*, ANSI B16.5, American National Standards Institute, 1988.

[6] *Welded and Seamless Wrought Steel Pipe*, ANSI B36.10, American National Standards Institute, 1985.

[7] *Stainless Steel Pipe*, ANSI B36.19, American National Standards Institute, 1985.

[8] *The Piping Guide for the Design and Drafting of Industrial Piping Systems*, Sherwood, David R., Whistance, Dennis J., Syentek Books Company, Inc., 1991.

[9] *Piping Handbook*, Nayyar, Mohinder L., McGraw-Hill, Inc., Sixth Edition, 1992.

[10] *Guidelines on Terminology for Valves and Fittings*, MSS SP-96, Manufacturers Standardization Society of the Valve and Fittings Industry, Inc., 1991.

Index