NISTIR 88-4004

Product Data Exchange Specification
First Working Draft

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology
(Formerly National Bureau of Standards)
National Engineering Laboratory
Center for Manufacturing Engineering
Factory Automation Systems Division
Gaithersburg, MD 20899

December 1988

National Bureau of Standards became the National Institute of Standards and Technology on August 23, 1988, when the Omnibus Trade and Competitiveness Act was signed. NIST retains all NBS functions. Its new programs will encourage improved use of technology by U.S. industry.

Chair, IGES/PDES Organization
Bradford Smith

Coordinator, IGES/PDES Organization
Gaylen Rinaudot

U.S. DEPARTMENT OF COMMERCE
C. William Verity, Secretary

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
Ernest Ambler, Director

4 FORM FEATURES INFORMATION MODEL

4.1 Purpose & Scope

4.1.1 Purpose

This document presents a conceptual model of form feature information. It is named "Form Features Information Model" (FFIM).

The FFIM is intended as a reference model for PDES (and STEP) Version 1. It is intended to be "plug compatible" with other PDES topical models. That is, the FFIM and any other PDES topical model(s) should be combinable into a single model that is correct in syntax, semantics, and substance. In particular, the FFIM combined with other shape-oriented PDES models should form an integrated information model for the toleranced nominal shape of industrial objects.

The FFIM is intended to be independent of product class and application. That is, it is a portion of an overall model of information which describes toleranced shape without assuming the type of product having that shape or the use to which the shape information is put. It may be that the FFIM is not equally congenial to all applications and product classes. But any cant is due to the state of knowledge, the PDES environment, the experience of contributors, coincidence, etc., rather than intent.

The FFIM has a dual objective: (i) to explicate a flexible approach to the role of form feature data in shape representation and (ii) to exhibit a variety of particular feature types of interest to industry. The first of these has hopefully been accomplished with considerable generality. As for the second, the best that can be hoped is that a good start has been made in "standardizing" features of widespread industrial interest. (As a practical matter, it is probably wise to limit the number of feature types in the model until the FFIM's approach has proven durable.)

4.1.2 Scope

The FFIM regards a form feature as a portion of a shape which (i) conforms to some stereotypical pattern and (ii) is considered a unit for some purpose. The model does not attempt to limit the shape subsets that might be treated as features, the nomenclature that is applied to these, or the purposes for which they are created.

The FFIM is concerned with shape. It is understood that a variety of nonshape information will be associated with form features – functionality, processes, surface finish, etc. However, the FFIM's role is to provide shape representation information to which such data might be "attached", not to provide for the nonshape data.

The FFIM is limited to form features of a specific shape. Optional features and alternate features are excluded on the grounds that these imply modeling of different shapes.

The model addresses nominal shape. Tolerance information is covered in a different PDES topical model. It is that model's job to provide for tolerancing of nominal shape, including form features. (However, there has been a conscious effort to make the model compatible with the PDES model of shape tolerance information.)

The model is limited to form features of individual, rigid objects. Flexibility, assembly interfaces, mechanism joints, and the like are not addressed.

In the FFIM, a form feature is limited to being a portion, rather than the entirety, of a shape. It is not intended that "standard" parts such as screws or bars be regarded as form features. ("Standard" parts are a likely area for future expansion.)

The FFIM is concerned with "macro shape". This excludes surface finish, the wave pattern produced by some processes, thin coatings, etc.

The majority of the FFIM is devoted to information for implicit (constructive, descriptive) representations of form features. In contrast, information for explicit representations and compound representations are treated quite generally. Refinement of these areas is a potential future extension.

4.1.3 Viewpoint

A form feature is viewed as having dimensionality 2. That is, the feature is a portion of the "skin" of a shape. This viewpoint is taken because all features considered seem amenable to that view and because the view seemed most compatible with other PDES models. (Note, though, that a major portion of the FFIM is devoted to volumetric features; i.e., to specification of features using volume additions/subtractions.

Feature data is considered optional in shape modeling. The fact that, for example, a cylindrical passage in a shape would generally be considered a "hole" does not imply any obligation to have a "hole" entity in a model of the shape. Any other attitude would be incompatible with most current CAD/CAM practice.

The form features seen in a shape are viewpoint dependent and may vary depending on application, company, and personal inclination. Different models of a given shape may contain different form feature information.

Form feature data is applied in the context of an abstract geometric model (e.g., surfaced wireframe, BREP, CSG) of the shape which includes the feature. This model may be a fully detailed representation of the shape or it may be a simplified representation. This view is consistent with the scope limitation that form features are portions of shape rather than complete shapes.

Form feature representations serve one or both of two functions in the context of a geometric model:

1. Group abstract mathematical elements of the geometric model and classify the grouping. For example, form feature data might group a cylindrical, a planar, and a conical face of a BREP model and declare the group to constitute a (chamfered) circular boss. This is termed an "explicit" representation of the feature.

2. Add information to a model of shape. For instance, form feature data might say "The shape has a fillet of radius R blending surfaces A and B". There may or may not be a surface or face in the geometric model corresponding to the fillet surface. If so, the feature data have added interpretation and parameters of interest to the model. If not, the feature data provide information needed to "realize" the feature in the model if desired; i.e., to calculate the fillet's surface, its intersections with the blended surfaces, etc. Representing features by information needed to "realize" them in a geometric model is called "implicit" (or "constructive" or "descriptive") representation.

The FFIM takes the view that there may be both an explicit and an implicit representation of a form feature associated with the same geometric model. This attitude opens the door to contradictory models, but is felt to be a practical necessity.

The implicit representation data of the FFIM are aimed at 3D modeling contexts. This is done despite the fact that there are useful geometric models which have 2D contexts; e.g., bodies of revolution and extrusions. It is done because PDES geometric modeling work appears to take the same attitude. (It is probably desirable to offer special geometric contexts and implicit representations tailored to them. The matter should be addressed in future work.)

In entities for implicit feature representation, the FFIM strives for a minimal set of data. For example, a regular n-sided polygon requires a value for inscribed radius but does not provide for a side length value. This is in accordance with the authors' understanding that PDES models should avoid redundancy.

A selected minimal set of parameters may suffice for nominal shape representation, but industry cannot be limited to that set in dimensioning and tolerancing practice. To use the example of the previous paragraph, there may be good reason to design an n-sided polygon by toleranced side length dimension. This being so, PDES must provide a means to (i) make it clear that the polygon's side length is the dimension of interest and (ii) give the tolerance on that dimension.

4.1.4 Abbreviations and Acronyms

BREP	Boundary Representation geometric model
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
CSG	Constructive Solid Geometry model
FFIM	Form Features Information Model
PDES	Product Data Exchange Standard
STEP	Standard for Exchange of Product Data

4.2 Assumptions and Basic Concepts

4.2.1 Assumptions

It is assumed that form feature information is exchanged in association with an abstract mathematical geometric model which, to some degree of detail, represents the shape of interest. This geometric model may be a BREP, surfaced wireframe, etc. It may be fully detailed, it may merely describe a simple shape to be detailed by implicit form features, or it may lie between these extremes.

It is assumed that the geometric model is given in a 3D context. (PDES supports two types of geometric models that are essentially planar, solids of revolution and extrusions. However, the planes are in 3-space.)

It is assumed that the creator and the receiver of a PDES model containing form feature information have coordinated on the use of form feature data in the model. Some areas where such coordination may be needed:

1. Which portions of the form features subschema are supported by the creating and target systems and their PDES translators.

2. The informational value, if any, contained in feature nomenclature.

3. The algorithms for calculating values of derivable attributes.

4. The modeling "style" to be employed. (This covers a large variety of choices, since the FFIM typically offers several reasonable tactics for modeling a particular situation.)

With respect to dimensions/tolerances, it is assumed that the FFIM need deal specifically only with those intrinsic to implicit form features. The following are assumed to be modeled elsewhere in a manner sufficient to satisfy the needs of form features; i.e., can merely refer to entire form features rather than to aspects of features.

1. Geometric tolerances.

2. Dimensions/tolerances applicable to geometric model entities which may be employed in explicit feature representations.

3. Dimensions/tolerances relating implicit form features to geometric model entities or other form features.

4.2.2 General Concepts

Shape is a primitive term associated with the size and proportions of a real or conceived object. A shape element is an identifiable aspect of a shape. A dimensionality-2 shape element is a shape element associated with the "skin" of the shape.

A geometric model is an abstract mathematical representation of a shape. BREPs, CSG models, and wireframe models are examples. There is an obvious correspondence between shapes and geometric models—a geometric model is employed to represent a shape. In form feature representation, though, it is important to recognize that the geometric model may differ from the shape of the product with which it corresponds. In particular, the geometric model may be considerably simpler than the product's nominal shape, the difference being made up by implicit features.

A form feature corresponds to a portion of the skin of a shape; i.e., is a dimensionality-2 shape element. As a matter of utility, a form feature should conform to some pattern or stereotype with a name; e.g., bevel gear, hole, bend.

Form features of a shape have representations that exist in association with geometric models of that shape. (The models may be simplifications or approximations of the represented shape.) There are two types of form feature representations:

1. An implicit (or constructive or descriptive) representation employs parameters and other data to describe the feature. For example, a through hole might be implicitly represented by diameter, centerline, entry point, and exit point. An implicit feature representation must suffice to compute or evaluate the feature. That is, there must exist a logic which will edit the geometric model in such a way that the form feature is incorporated. In a BREP geometric model, for example, this editing would consist of adding/modifying/deleting faces, edges, vertices, etc. (Note that it is not required that this editing logic be implemented for implicit form features to be used. Implicitly represented holes can be drilled without being evaluated.)

2. Enumerative representations are lists of dimensionality-2 shape elements that compose the form feature. The constituents are shape elements which have dimensionality 2, including perhaps other form features. The concept of enumerative representation includes two familiar notions:

   (a) Explicit or evaluated feature representations, which are lists of shape elements represented by geometric model entities, and

   (b) Compound feature representations, which are lists of constituent features.

As mentioned earlier, the FFIM allows both an implicit and an enumerative representation of a form feature to be associated with the same geometric model.

Enumerative representations group existing model entities and apply a meaningful name to the group. In a sense, they interpret existing data rather than add new information.

Implicit representations may or may not add wholly new information to a representation of shape.

1. If the geometric model has no entities corresponding to the feature, the implicit representation adds new information to yield a more detailed understanding of shape.

2. If the geometric model contains entities corresponding to the feature, an implicit representation gives a description of the feature that, for some purpose, is a convenient supplement or alternative to abstract geometric model data. (In this case, of course, the implicit representation could theoretically be derived from geometric model entities. In practice, this is very difficult.)

4.2.3 Implicit Feature Representation

Implicit representations offer a means of modeling "popular" shape configurations in a convenient way. It is important to realize that this is a limited capability which does not offer the generality of abstract geometric modeling methods. If the entities available for implicit representation cannot handle a "feature", explicit representation must be used. This will happen most often in two circumstances:

1. There is no way to model the desired feature's shape with the entities available.

2. There is a need to refer to portions of the feature, rather than its entirety. Implicit representations are "atomic" or "unitary" in the sense that they can be referred t wholly or not at all. For example, there can be no way to associate a surface finish with the sides of a boss, but not its top. In this case, an explicit representation must be used. (But an implicit representation may also be present.)

The concept of pre-existing shape is crucial to implicit feature representations. An implicit representation serves as a description of how to change a geometric model without the feature in such a way that it models shape inclusive of the feature. (Embedded in this concept is the fact that an implicit feature representation may apply to a pre-existing shape which includes other implicitly represented features. Thus it may be necessary to partially order implicit feature representations. This is addressed below.)

Implicit feature representations can be classified according to the effect of the feature upon pre-existing shape or, more or less equivalently, upon data associations and interpretations:

1. Passages are subtractions of material from pre-existing shape. The subtracted volume intersects the pre-existing shape's boundary at both ends, increasing the shape's genus by 1.

2. Depressions are subtractions of material whose subtracted volume intersects the pre-existing shape's boundary at one end. The genus of the shape is unchanged.

3. Protrusions are additions of material whose added volume intersects the pre-existing shape's boundary at one end. The genus of the shape is unchanged.

4. Transitions are smoothings or gradualizations of the intersections of elements of pre-existing shape.

5. Area features are treated as being applied to dimensionality-2 elements of pre-existing shape.

6. Deformations involve bending, stretching, etc. of pre-existing shape.

The classifications above are not, in theory, mutually exclusive. To date, however, treating them as mutually exclusive has not led to problems.

4.2.4 Volume-Associated Features

Of the six classes above, the first three have definite volumetric associations. That is, their implicit representations are aimed at specifying a volume added to or subtracted from pre-existing shape. Sweeps and rulings are the primary vehicles for defining the added/subtracted volume.

A feature sweep defines a volume by a profile and a sweep path. The FFIM provides for typical end conditions so that the sweeps can be de facto solid sweeps. For example, an axisymmetric feature sweep can have a spherical end condition, which can be used to model sweeps of ball-nosed drill bits.

A feature ruling defines a volume using a ruled surface definition.

Implicit feature bounds indicate where the added/subtracted volume intersects pre-existing shape. For example, a through hole modeled as a swept implicit passage might have two such bounds identifying the hole's entry face(s) and exit face(s). Note that this information may not be necessary, merely convenient. In the example given, the entry and exit faces could in theory be deduced from the pre-existing shape and the sweep-defined volume; in practice, this deduction might be quite difficult.

The FFIM gives considerable attention to edge blends (chamfers, fillets, etc.) associated with volumetric features. These blend surfaces occur in three ways:

1. Intersections between portions of feature-defining volumes; e.g., between the walls of a sweep-represented pad, between the ruled wall and one if its end surfaces in a ruling-defined pocket.

2. Intersections between the added/subtracted volumes and pre-existing shape; e.g., between pocket walls and the surfa ce on which the pocket is installed.

3. Intersections between wall surfaces and "ends" of the feature; e.g., between walls and bottom of a pocket.

4.2.5 Ordering Implicit Features

An implicit feature representation may depend on other implicit feature representations. For example, a hole may be installed in a boss, with both being implicitly represented. While the boss representation is understandable without knowledge of the hole, the converse is not true.

The FFIM regards implicit feature representations as partially ordered by a relation that might be called "is understood in the context of" or "is dependent on". There are two ways in which dependence information is shown:

1. One implicit representation refers to another. This may happen in several ways. In the example above, the boss would be a bound of the hole. Or an implicit hole might be identified as the area on which an implicit thread is installed.

2. A non-specific statement is made that one implicit feature should preceed another. This can be used when the model does not provide a specific precedence-establishing relation. (Or when the mere statement of dependence is preferred to a more verbose modeling of the dependence.)

4.2.6 Locating Implicit Form Feature Representations

An implicit representation must specify both the intrinsic shape of a feature and its location. Depending on the nature of a feature, the FFIM uses one of three ways to locate features:

1. A local coordinate system positions an axis system, the feature's location with respect to that axis system being known.

2. A geometric entity locates the feature: e.g., a bend is located by bendline.

3. The feature is located by reference to pre-existing shape; e.g.; a fillet is located by identifying the surfaces it blends.

4.2.7 Replication of Implicit Features

Form features are often identical except for location. The FFIM provides two means to "reuse" an implicit representation in such situations — replication and feature patterns.

Replication is the modeling of a feature by declaring it to be identical to another, except for location, and specifying its location.

Feature patterns are a strong form of replication. An original representation is declared to be repeated in a geometric pattern. The FFIM provides for circular and matrix patterns.

In replication and patterns, all information associated with the "original" feature is assumed to apply to the "copies".

4.2.8 Dimensional Information

Most of the information giving nominal form feature shape is dimensional. Dimensional parameters in the FFIM falls into three classes, depending on whether "referenceability" is required and on whether the parameter is independent or derivable. (Referenceability is, sloppily, the ability to "point" or be "pointed" at. The most obvious example of a referenceability requirement is the need to associate a tolerance with the dimension.)

1. Independent parameters which must be referenceable are modeled via relations to entities having an attribute giving parameter value.

2. Independent parameters not needing referencing are modeled as attributes.

3. Dependent parameters which must be referenceable are modeled via relations to entities in which parameter value is not an attribute.

The entities required by the first and third cases are native to the Shape Variation Tolerances Information Model.

4.3 Planning Model

4.3.1 Entity Pool

Following are definitions of the entities in the FFIM Planning Model. Two concepts are important to understanding these definitions and the overall Planning Model: shape and pre-existing shape. These are discussed in 4.2.2, Basic Concepts.

AREA SHAPE ELEMENT: A DIMENSIONALITY-2 SHAPE ELEMENT that is arcwise connected and has a uniform underlying mathematical surface.

AXIS PLACEMENT: Defines a local coordinate system used to locate an implicit form feature.

DERIVABLE SIZE PARAMETER: A dimension which is dependent upon other dimensions or information and thus has no nominal value given in implicit form feature definition.

DIMENSIONALITY-2 SHAPE ELEMENT: A distinguishable aspect of the "skin" of a shape.

FEATURE RULING: The definition of a feature volume using a ruled surface description.

FEATURE SWEEP: The definition of a feature volume using a two dimensional profile and a sweep path.

FEATURE VOLUME: A volume added to or subtracted from pre-existing shape. Used to specify implicit definitions of form features modeled as increments/decrements of material.

FORM FEATURE: A portion of shape that conforms to some stereotypical pattern and is considered a unit for some purpose.

GEOMETRIC MODEL: An abstract mathematical representation of a shape. Boundary representations, Constructive Solid Geometry (CSG) models, and wireframe models are examples. It is important to note that a GEOMETRIC MODEL used to represent a shape may be considerably simpler than the shape, with the difference being made up by implicit feature representations. As an extreme example, the GEOMETRIC MODEL of a turned part might merely represent a bar (OD and two ends), with implicit representations of steps, IDs, holes, chamfers, fillets, etc. providing all detail.

GEOMETRY ENTITY: A point, curve, surface, vector, etc. Note that a GEOMETRIC ENTITY is not necessarily employed in a GEOMETRIC MODEL. It may, for example, be used in an implicit feature representation that augments a GEOMETRIC MODEL.

IMPLICIT BLEND: An implicit representation of a 'gradualization' of the intersection of two DIMENSIONALITY-2 SHAPE ELEMENTs. Fillets and chamfers are blends.

IMPLICIT FEATURE BOUND: Indicates where an added/subtracted volume defining an IMPLICIT FORM FEATURE intersects pre-existing shape; e.g., the entry or exit surface of a hole. The bound 'closes' the added/subtracted volume.

IMPLICIT FEATURE PRECEDENCE: An indication of existence precedence between two implicit feature representations.

IMPLICIT FORM FEATURE: A descriptive representation of a FORM FEATURE. These employ parameters, geometric entities, and/or references to pre-existing shape to specify the feature. For example, a hole might be described by centerline, diameter, entry face, and exit face. The specification is independent of whether a GEOMETRIC MODEL of the shape contains elements corresponding to the feature. It must be sufficient to 'realize' the feature in a GEOMETRIC MODEL; i.e., to modify the model in such a way that the feature is present.

INDEPENDENT SIZE PARAMETER: A dimension whose value is used in an implicit representation of the nominal shape of a form feature.

SHAPE ELEMENT: A distinguishable aspect of a shape.

SIZE PARAMETER: A dimensional parameter used in an implicit representation of a form feature or derivable from an implicit representation of the feature.

TOLERANCE RANGE: An indication of the range of acceptable values of a dimensional attribute of a feature. This is done by giving a plus/minus value applicable to the nominal dimension.

ZONE: A collection of two or more DIMENSIONALITY-2 SHAPE ELEMENTs.

4.3.2 Entity-Relationship Diagram

The following IDEF1X diagram shows the FFIM Planning Model. This model is an overview of the FFIM Reference Model.

In the diagram, shaded boxes are entities from other topical models – the Integration Core Model, Geometry Model, and Shape Variation Tolerances Model.

Note that non-specific relations are not resolved in the Planning Model.

4.3.3 Business Rules

The following business rules state the relations between entities of the Planning Model. Each of these relations is shown in figure 36.

A DIMENSIONALITY-2 SHAPE ELEMENT is zero or one FORM FEATUREs.
A FORM FEATURE is exactly one DIMENSIONALITY-2 SHAPE ELEMENT.

A DIMENSIONALITY-2 SHAPE ELEMENT may be either an AREA SHAPE ELEMENT or a ZONE SHAPE ELEMENT.
An AREA SHAPE ELEMENT is a type of DIMENSIONALITY-2 SHAPE ELEMENT.
A ZONE SHAPE ELEMENT is a type of DIMENSIONALITY-2 SHAPE ELEMENT.

A TOLERANCE RANGE tolerances zero, one, or many SIZE PARAMETERs.
A SIZE PARAMETER is toleranced by zero or one TOLERANCE RANGEs.

A SIZE PARAMETER must be either an INDEPENDENT SIZE PARAMETER or a DERIVABLE SIZE PARAMETER.
An INDEPENDENT SIZE PARAMETER is a type of SIZE PARAMETER.
A DERIVABLE SIZE PARAMETER is a type of SIZE PARAMETER.

An IMPLICIT FORM FEATURE has as dimension zero, one, or many SIZE PARAMETERs.
A SIZE PARAMETER is dimension of zero, one, or many IMPLICIT FORM FEATUREs.

A FORM FEATURE is represented by zero, one, or many IMPLICIT FORM Features.
An IMPLICIT FORM FEATURE represents exactly one FORM FEATURE.

An IMPLICIT FORM FEATURE is located by zero or one AXIS PLACEMENTs.
An AXIS PLACEMENT locates zero, one, or many IMPLICIT FORM FEATUREs.

A GEOMETRIC MODEL is the context of zero, one, or many IMPLICIT FORM FEATUREs.
An IMPLICIT FORM EATURE augments one or many GEOMETRIC MODELs.

A GEOMETRY ENTITY is referenced by zero, one, or many IMPLICIT FORM FEATUREs.
An IMPLICIT FORM FEATURE references zero, one, or many GEOMETRY ENTITYs.

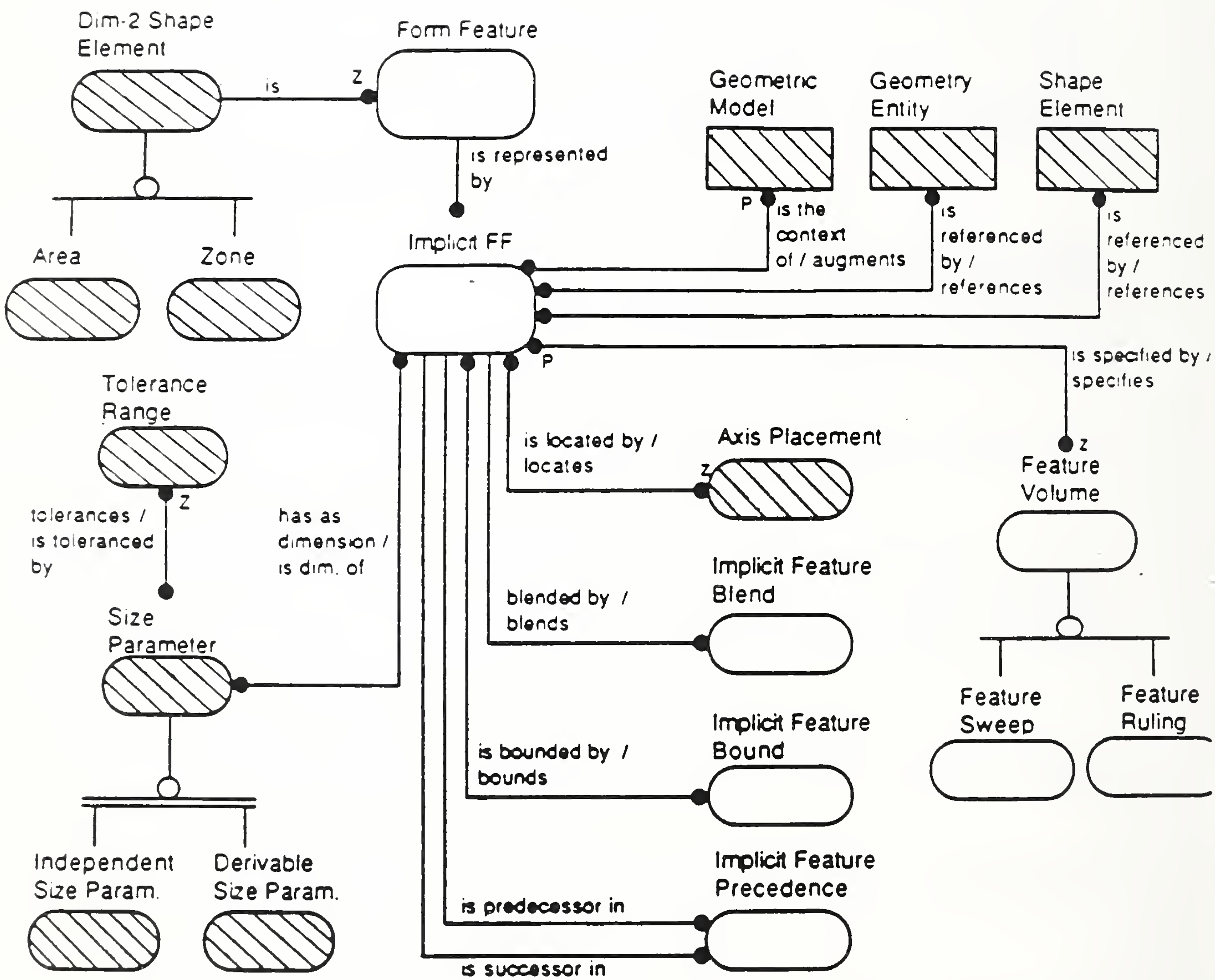


Figure D-36: FFIM Planning Model

A SHAPE ELEMENT is referenced by zero, one, or many IMPLICIT FORM FEATUREs
An IMPLICIT FORM FEATURE references zero, one, or many SHAPE ELEMENTs.

An IMPLICIT FORM FEATURE is specified by zero or one FEATURE VOLUMEs.
A FEATURE VOLUME specifies one or many IMPLICIT FORM FEATUREs.

A FEATURE VOLUME may be either a FEATURE SWEEP or a FEATURE RULING.
A FEATURE SWEEP is a type of FEATURE VOLUME.
A FEATURE RULING is a type of FEATURE VOLUME.

An IMPLICIT FORM FEATURE is bounded by zero, one, or many IMPLICIT FEATURE BOUNDs.

An IMPLICIT FEATURE BOUND bounds zero, one, or many IMPLICIT FORM FEATUREs.

An IMPLICIT FORM FEATURE is blended by zero, one, or many IMPLICIT FEATURE BLENDs.

An IMPLICIT FEATURE BLEND blends exactly one IMPLICIT FORM FEATURE.

An IMPLICIT FORM FEATURE is predecessor in zero, one, or many IMPLICIT FEATURE PRECEDENCEs.
An IMPLICIT FEATURE PRECEDENCE has as predecessor exactly one IMPLICIT FORM FEATURE.
An IMPLICIT FORM FEATURE is successor in zero, one, or many IMPLICIT FEATURE PRECEDENCEs.
An IMPLICIT FEATURE PRECEDENCE has as successor exactly one IMPLICIT FORM FEATURE.

4.4 Reference Model

This chapter gives a diagrammatic IDEF1X conceptual model of form feature information. Diagrammatic IDEF1X was the working methodology of the Form Features Committee. Models in other languages were produced by translation from the IDEF1X.

The reference model is documented in three parts:

1. Entity Pool. The entities of the FFIM are enumerated.

2. Diagrams. IDEF1X diagrams show entities and entity relationships.

3. Entity Descriptions. Each "native" FFIM entity is specified in detail.

4.4.1 Entity Pool

This section lists all entities "native" to the FFIM. In addition, entities from other topical models are included if these entities appear in the FFIM's diagrams.

There are two entity lists. The first is in order of entity number; the second, entity name.

4.4.2 Numeric Entity Listing

Table 1 lists FFIM entities in numeric order. Native entities are given first without the "FF-" prefix of FFIM entities. Entities from other topical models follow, ordered by alphabetical prefix (INT, TOL, etc.) and subordered by number within their own topical model.

NUMBER	NAME	DIAGRAMS
001	FORM FEATURE	37,38,39
002	IMPLICIT FORM FEATURE	37,40,38,39
003	IMPLICIT PASSAGE	40,41,42
004	IMPLICIT PROTRUSION	40,43
005	IMPLICIT DEPRESSION	40,43
006	IMPLICIT TRANSITION	40,44
024	IMPLICIT KNURL	45
025	IMPLICIT THREAD	45
029	IMPLICIT FEATURE BOUND	40,41 (twice),43
030	FEATURE BOUND ELEMENT	40
036	IMPLICIT EDGE FLAT	44
037	IMPLICIT EDGE ROUND	41,44,46
049	IMPLICIT FORM FEATURE PATTERN	39
053	IMPLICIT CIRCULAR FORM FEATURE PATTERN	39
054	IMPLICIT ARRAY FORM FEATURE PATTERN	39
055	PARAMETRIC EQUAL SPACING ARRAY PATTERN	39
056	PARALLEL EQUAL SPACING ARRAY PATTERN	39
064	IMPLICIT DEFORMATION	40,47
067	REPLICATE FORM FEATURE	38
073	CIRCULAR PATTERN OMISSION	39
074	CIRCULAR PATTERN OFFSET MEMBER	39
081	IMPLICIT EMBOSS	47,48
082	IMPLICIT TWIST	47
083	IMPLICIT PARTIAL CUTOUT	47,49
084	IMPLICIT BEND	47,42
086	IMPLICIT TUBE DEFORMATION	47,50
087	IMPLICIT V-BEAD	48
088	IMPLICIT ROUND BEAD	48
089	IMPLICIT CORNER RIB	48
090	IMPLICIT LOUVER	49
091	IMPLICIT CIRCULAR KNOCKOUT	49
092	IMPLICIT GENERAL BEND	42
093	IMPLICIT TUBE BEND	50
094	IMPLICIT CUTOUT FLANGE	42
095	IMPLICIT STRAIGHT BEND	42
096	BEND POINT	42
098	IMPLICIT TUBE FLARE	50
099	IMPLICIT TUBE NECK	50
100	IMPLICIT TUBE FLATTENING	50
101	IMPLICIT TUBE ROLL	50

Table 1: Entity Pool – Numeric Order

NUMBER	NAME	DIAGRAMS
111	IMPLICIT AREA FEATURE	40,45
120	FEATURE SWEEP	51
121	ALONG FEATURE SWEEP	51,52
122	AXISYMMETRIC FEATURE SWEEP	51,46
123	IN/OUT FEATURE SWEEP	51,52
124	FEATURE SWEEP PATH	52,53
125	LINEAR FEATURE SWEEP PATH	53
126	CIRCULAR FEATURE SWEEP PATH	53
127	PARTIAL CIRCULAR FEATURE SWEEP PATH	53
128	COMPLETE CIRCULAR FEATURE SWEEP PATH	53
129	IN/OUT FEATURE SWEEP WALL/END BLEND	52
130	AXISYMMETRIC FEATURE SWEEP END	46
131	AXISYMMETRIC FEATURE SWEEP WALL/END BLEND	46
132	AXISYMMETRIC FEATURE SWEEP FLAT END	46
133	AXISYMMETRIC FEATURE SWEEP SPHERICAL END	46
134	AXISYMMETRIC FEATURE SWEEP CONICAL END	46
135	AXISYMMETRIC FEATURE SWEEP CONICAL END TIP BLEND	46
136	FEATURE SWEEP PROFILE	52,54
137	CLOSED FEATURE SWEEP PROFILE	54
139	RECTANGULAR FEATURE SWEEP PROFILE	54
140	N-GON FEATURE SWEEP PROFILE	54
141	STANDARD CLOSED FEATURE SWEEP PROFILE BLEND	54
142	OPEN FEATURE SWEEP PROFILE	52,54,55
143	CIRCULAR ARC FEATURE SWEEP PROFILE	55
144	ROUNDED-U FEATURE SWEEP PROFILE	55
145	VEE FEATURE SWEEP PROFILE	55
146	SQUARE-U FEATURE SWEEP PROFILE	55
147	TEE FEATURE SWEEP PROFILE	55,56
148	ELL FEATURE SWEEP PROFILE	55,56
149	CONSTANT DIAMETER AXISYMMETRIC FEATURE SWEEP	46
150	TAPERED AXISYMMETRIC FEATURE SWEEP	46
151	OTHER AXISYMMETRIC FEATURE SWEEP	46
152	PASSAGE BLEND	41
153	PROTRUSION BLEND	41
154	DEPRESSION BLEND	43
158	ALONG FEATURE SWEEP END	52
159	ALONG FEATURE SWEEP FLAT END	52

Table 1: Entity Pool – Numeric Order (Continued)

NUMBER	NAME	DIAGRAMS
160	ALONG FEATURE SWEEP RADIUSED END	52
161	ALONG FEATURE SWEEP FLAT END BLEND	52
162	GENERAL PROFILE	46,57
163	OPEN GENERAL PROFILE	55,57
164	CLOSED GENERAL PROFILE	54,57
165	PROFILE PAIR	57
166	PROFILE PAIR BLEND	57
167	STANDARD CLOSED FEATURE SWEEP PROFILE	54
168	OTHER CLOSED FEATURE SWEEP PROFILE	54
169	OTHER OPEN FEATURE SWEEP PROFILE	55
170	SQUARE-U BLEND1	55
171	SQUARE-U BLEND2	55
172	VEE BLEND	55
173	TEE STEM/CROSSBAR BLEND1	56
174	TEE STEM/CROSSBAR BLEND2	56
175	TEE CROSSBAR BLEND1	56
176	TEE CROSSBAR BLEND2	56
177	TEE CROSSBAR BLEND3	56
178	TEE CROSSBAR BLEND4	56
179	ELL STEM/ENDBAR BLEND	56
180	ELL ENDBAR BLEND1	56
181	ELL ENDBAR BLEND2	56
182	ELL ENDBAR BLEND3	56
184	LINE PLUS RADIUS FEATURE SWEEP PROFILE	55
185	HALF OBROUND FEATURE SWEEP PROFILE	55
186	PASSAGE INTERMEDIATE BOUND	41
187	PASSAGE INTERMEDIATE BOUND BLEND	41
188	DEPRESSION INTERMEDIATE BOUND	43
189	DEPRESSION INTERMEDIATE BOUND BLEND	43
190	CONSTANT PROFILE IN/OUT SWEEP	52
192	TAPERED PROFILE IN/OUT SWEEP	52
193	CONTOURED PROFILE IN/OUT SWEEP	52
302	FORM FEATURE REFLECTION	38
305	PARALLEL ARRAY PATTERN OFFSET MEMBER	39
306	ARRAY PATTERN OMISSION	39
307	PARAMETRIC ARRAY PATTERN OFFSET MEMBER	39
308	IMPLICIT EDGE BLEND	41,43,44,51,52,46, 54,55,56,57
309	IMPLICIT CORNER BLEND ASSOCIATION	44

Table 1: Entity Pool – Numeric Order (Continued)

NUMBER	NAME	DIAGRAMS
310	IMPLICIT CORNER FLAT	44
311	IMPLICIT OUTSIDE CORNER ROUND	44
315	GEOMETRIC MODEL/IMPLICIT FORM FEATURE	37
354	IMPLICIT STRAIGHT KNURL	45
355	IMPLICIT DIAGONAL KNURL	45
356	IMPLICIT DIAMOND KNURL	45
359	EDGE-BLENDED INTERSECTION	44
400	IMPLICIT SPHERICAL EMBOSS	48
401	IMPLICIT TAB	49
402	PASSAGE BOUND	41
403	DEPRESSION BOUND	43
404	PROTRUSION BOUND	41
406	IMPLICIT MARKING	45
411	FEATURE RULING	51
415	IMPLICIT COUPLING	45
416	FEATURE RULING WALL/END BLEND	51
417	FEATURE VOLUME	41 (twice),43.51
418	FEATURE RULING LCS	51
419	IMPLICIT FEATURE PRECEDENCE	37
420	COUPLING TIMER	45
421	FULL THREADING SPECIFICATION	45
422	SPIRAL FEATURE SWEEP PATH	53
423	SURFACE CONFORMING FEATURE SWEEP PATH	53
424	OTHER FEATURE SWEEP PATH	53
425	PITCH APEX	45
426	THREAD APEX	45
427	SPIRAL TAPER	53
428	THREAD TIMER	45
429	BEND DIMENSION	47,49,50,48,42
430	BEND MOVEMENT INDICATOR	42
431	GENERAL BEND MOVEMENT INDICATOR	42
432	STRAIGHT BEND MOVEMENT INDICATOR	42
433	TUBE BEND MOVEMENT INDICATOR	50
GEO-2	POINT	50.48.42,45
GEO-3	VECTOR	50
GEO-4	AXIS PLACEMENT	47,49,50,48,38,51
GEO-6	CURVE	42,53,57
GEO-7	SURFACE	39
GEO-17	VECTOR WITH MAGNITUDE	39
GEO-20	LINE	42,39

Table 1: Entity Pool – Numeric Order (Continued)

NUMBER	NAME	DIAGRAMS
GEO-22	BOUNDED CURVE	48,52
GEO-23	CURVE ON SURFACE	51,53
INT-7	DIMENSIONALITY-2 SHAPE ELEMENT	37,40,44,45
INT-8	AREA SHAPE ELEMENT	37
INT-11	ZONE SHAPE ELEMENT	37
INT-15	EDGE SHAPE ELEMENT	44
INT-20	CORNER SHAPE ELEMENT	44
INT-23	GEOMETRIC MODEL	37
INT-103	ZONE SHAPE ELEMENT COMPONENT	37
TOL-6041	INDEPENDENT SIZE PARAMETER	47,58
TOL-6042	DERIVABLE SIZE PARAMETER	58
TOL-6101	INDEPENDENT ANGLE SIZE PARAMETER	58
TOL-6102	DERIVABLE ANGLE SIZE PARAMETER	58

Table 1: Entity Pool - Numeric Order (Continued)

4.4.3 Alphabetic Entity Listing

Table 2 lists FFIM entities in alphabetic order of entity name. Native and non-native entities are intermixed.

NUMBER	NAME	DIAGRAMS
121	ALONG FEATURE SWEEP	51,52
158	ALONG FEATURE SWEEP END	52
159	ALONG FEATURE SWEEP FLAT END	52
161	ALONG FEATURE SWEEP FLAT END BLEND	52
160	ALONG FEATURE SWEEP RADIUSED END	52
INT-8	AREA SHAPE ELEMENT	37
306	ARRAY PATTERN OMISSION	39
GEO-4	AXIS PLACEMENT	47,49,50,48,38,51
122	AXISYMMETRIC FEATURE SWEEP	51,46
134	AXISYMMETRIC FEATURE SWEEP CONICAL END	46
135	AXISYMMETRIC FEATURE SWEEP CONICAL END TIP BLEND	46
130	AXISYMMETRIC FEATURE SWEEP END	46
132	AXISYMMETRIC FEATURE SWEEP FLAT END	46
133	AXISYMMETRIC FEATURE SWEEP SPHERICAL END	46
131	AXISYMMETRIC FEATURE SWEEP WALL/END BLEND	46
429	BEND DIMENSION	47,49,50,48,42
430	BEND MOVEMENT INDICATOR	42
096	BEND POINT	42
GEO-22	BOUNDED CURVE	48,52
143	CIRCULAR ARC FEATURE SWEEP PROFILE	55
126	CIRCULAR FEATURE SWEEP PATH	53
074	CIRCULAR PATTERN OFFSET MEMBER	39
073	CIRCULAR PATTERN OMISSION	39
137	CLOSED FEATURE SWEEP PROFILE	54
164	CLOSED GENERAL PROFILE	54,57
128	COMPLETE CIRCULAR FEATURE SWEEP PATH	53
149	CONSTANT DIAMETER AXISYMMETRIC FEATURE SWEEP	46
[illegible]	CONSTANT PROFILE IN/OUT SWEEP	52
193	CONTOURED PROFILE IN/OUT SWEEP	52
INT-20	CORNER SHAPE ELEMENT	44
420	COUPLING TIMER	45
GEO-6	CURVE	42,53,57
GEO-23	CURVE ON SURFACE	51,53
154	DEPRESSION BLEND	43
188	DEPRESSION INTERMEDIATE BOUND	43
189	DEPRESSION INTERMEDIATE BOUND BLEND	43
TOL-6102	DERIVABLE ANGLE SIZE PARAMETER	58
TOL-6042	DERIVABLE SIZE PARAMETER	58

Table 2: Entity Pool – Alphabetic Order

NUMBER	NAME	DIAGRAMS
INT-7	DIMENSIONALITY-2 SHAPE ELEMENT	37,40,44,45
INT-15	EDGE SHAPE ELEMENT	44
359	EDGE-BLENDED INTERSECTION	44
180	ELL ENDBAR BLEND1	56
181	ELL ENDBAR BLEND2	56
182	ELL ENDBAR BLEND3	56
148	ELL FEATURE SWEEP PROFILE	55,56
179	ELL STEM/ENDBAR BLEND	56
403	DEPRESSION BOUND	43
030	FEATURE BOUND ELEMENT	40
411	FEATURE RULING	51
418	FEATURE RULING LCS	51
416	FEATURE RULING WALL/END BLEND	51
417	FEATURE VOLUME	41 (twice),43,51
120	FEATURE SWEEP	51
124	FEATURE SWEEP PATH	52,53
136	FEATURE SWEEP PROFILE	52,54
001	FORM FEATURE	37,38,39
302	FORM FEATURE REFLECTION	38
421	FULL THREADING SPECIFICATION	45
431	GENERAL BEND MOVEMENT INDICATOR	42
162	GENERAL PROFILE	46,57
INT-23	GEOMETRIC MODEL	37
315	GEOMETRIC MODEL/IMPLICIT FORM FEATURE ASSOCIATION	37
185	HALF OBROUND FEATURE SWEEP PROFILE	55
111	IMPLICIT AREA FEATURE	40,45
054	IMPLICIT ARRAY FORM FEATURE PATTERN	39
084	IMPLICIT BEND	47,42
053	IMPLICIT CIRCULAR FORM FEATURE PATTERN	39
091	IMPLICIT CIRCULAR KNOCKOUT	49
309	IMPLICIT CORNER BLEND	44
310	IMPLICIT CORNER FLAT	44
089	IMPLICIT CORNER RIB	48
415	IMPLICIT COUPLING	45
094	IMPLICIT CUTOUT FLANGE	42
064	IMPLICIT DEFORMATION	40,47
005	IMPLICIT DEPRESSION	40,43
308	IMPLICIT EDGE BLEND	41,43,44.51,52,46, 54,55,56,57

Table 2: Entity Pool – Alphabetic Order (Continued)

NUMBER	NAME	DIAGRAMS
036	IMPLICIT EDGE FLAT	44
037	IMPLICIT EDGE ROUND	41,44,46
081	IMPLICIT EMBOSS	47,48
355	IMPLICIT DIAGONAL KNURL	45
356	IMPLICIT DIAMOND KNURL	45
029	IMPLICIT FEATURE BOUND	40,41 (twice),43
419	IMPLICIT FEATURE PRECEDENCE	37
002	IMPLICIT FORM FEATURE	37,40,38,39
049	IMPLICIT FORM FEATURE PATTERN	39
092	IMPLICIT GENERAL BEND	42
024	IMPLICIT KNURL	45
090	IMPLICIT LOUVER	49
406	IMPLICIT MARKING	45
311	IMPLICIT OUTSIDE CORNER ROUND	44
083	IMPLICIT PARTIAL CUTOUT	47,49
003	IMPLICIT PASSAGE	40,41,42
004	IMPLICIT PROTRUSION	40,43
088	IMPLICIT ROUND BEAD	48
400	IMPLICIT SPHERICAL EMBOSS	48
095	IMPLICIT STRAIGHT BEND	42
354	IMPLICIT STRAIGHT KNURL	45
401	IMPLICIT TAB	49
025	IMPLICIT THREAD	45
006	IMPLICIT TRANSITION	40,44
093	IMPLICIT TUBE BEND	50
086	IMPLICIT TUBE DEFORMATION	47,50
098	IMPLICIT TUBE FLARE	50
100	IMPLICIT TUBE FLATTENING	50
099	IMPLICIT TUBE NECK	50
101	IMPLICIT TUBE ROLL	50
082	IMPLICIT TWIST	47
087	IMPLICIT V-BEAD	48
TOL-6101	INDEPENDENT ANGLE SIZE PARAMETER	58
TOL-6041	INDEPENDENT SIZE PARAMETER	47,58
123	IN/OUT FEATURE SWEEP	51,52
129	IN/OUT FEATURE SWEEP WALL/END BLEND	52
GEO-20	LINE	42,39
184	LINE PLUS RADIUS FEATURE SWEEP PROFILE	55
125	LINEAR FEATURE SWEEP PATH	53
140	N-GON FEATURE SWEEP PROFILE	54

Table 2: Entity Pool – Alphabetic Order (Continued)

NUMBER	NAME	DIAGRAMS
142	OPEN FEATURE SWEEP PROFILE	52,54,55
163	OPEN GENERAL PROFILE	55,57
151	OTHER AXISYMMETRIC FEATURE SWEEP	46
168	OTHER CLOSED FEATURE SWEEP PROFILE	54
424	OTHER FEATURE SWEEP PATH	53
169	OTHER OPEN FEATURE SWEEP PROFILE	55
305	PARALLEL ARRAY PATTERN OFFSET MEMBER	39
056	PARALLEL EQUAL SPACING ARRAY PATTERN	39
307	PARAMETRIC ARRAY PATTERN OFFSET MEMBER	39
055	PARAMETRIC EQUAL SPACING ARRAY PATTERN	39
127	PARTIAL CIRCULAR FEATURE SWEEP PATH	53
152	PASSAGE BLEND	41
402	PASSAGE BOUND	41
186	PASSAGE INTERMEDIATE BOUND	41
187	PASSAGE INTERMEDIATE BOUND BLEND	41
425	PITCH APEX	45
GEO-2	POINT	50,48,42,45
165	PROFILE PAIR	57
166	PROFILE PAIR BLEND	57
153	PROTRUSION BLEND	41
404	PROTRUSION BOUND	41
139	RECTANGULAR FEATURE SWEEP PROFILE	54
067	REPLICATE FORM FEATURE	38
144	ROUNDED-U FEATURE SWEEP PROFILE	55
170	SQUARE-U BLEND1	55
171	SQUARE-U BLEND2	55
146	SQUARE-U FEATURE SWEEP PROFILE	55
422	SPIRAL FEATURE SWEEP PATH	53
427	SPIRAL TAPER	53
167	STANDARD CLOSED FEATURE SWEEP PROFILE	54
141	STANDARD CLOSED FEATURE SWEEP PROFILE BLEND	54
432	STRAIGHT BEND MOVEMENT INDICATOR	42
GEO-7	SURFACE	39
423	SURFACE CONFORMING FEATURE SWEEP PATH	53
150	TAPERED AXISYMMETRIC FEATURE SWEEP	46
192	TAPERED PROFILE IN/OUT SWEEP	52
175	TEE CROSSBAR BLEND1	56
176	TEE CROSSBAR BLEND2	56
177	TEE CROSSBAR BLEND3	56

Table 2: Entity Pool – Alphabetic Order (Continued)

NUMBER	NAME	DIAGRAMS
178	TEE CROSSBAR BLEND4	56
147	TEE FEATURE SWEEP PROFILE	55,56
173	TEE STEM/CROSSBAR BLEND1	56
174	TEE STEM/CROSSBAR BLEND2	56
426	THREAD APEX	45
428	THREAD TIMER	45
433	TUBE BEND MOVEMENT INDICATOR	50
GEO-3	VECTOR	50
GEO-17	VECTOR WITH MAGNITUDE	39
172	VEE BLEND	55
145	VEE FEATURE SWEEP PROFILE	55
INT-11	ZONE SHAPE ELEMENT	37
INT-103	ZONE SHAPE ELEMENT COMPONENT	37

Table 2: Entity Pool – Alphabetic Order (Continued)

4.4.4 IDEF1X Diagrams

The following pages contain entity/relationship diagrams for the FFIM. These are in the standard form of IDEF1X "For Exposition Only" diagrams, with two exceptions

1. Attributes are not shown.
2. Most relations to four entities of the Shape Variation Tolerances are not shown. There is a very large number of these low-level relations. Including them would drastically reduce the readibility of the diagrams. The last diagram gives a "typical" diagram of these relations.

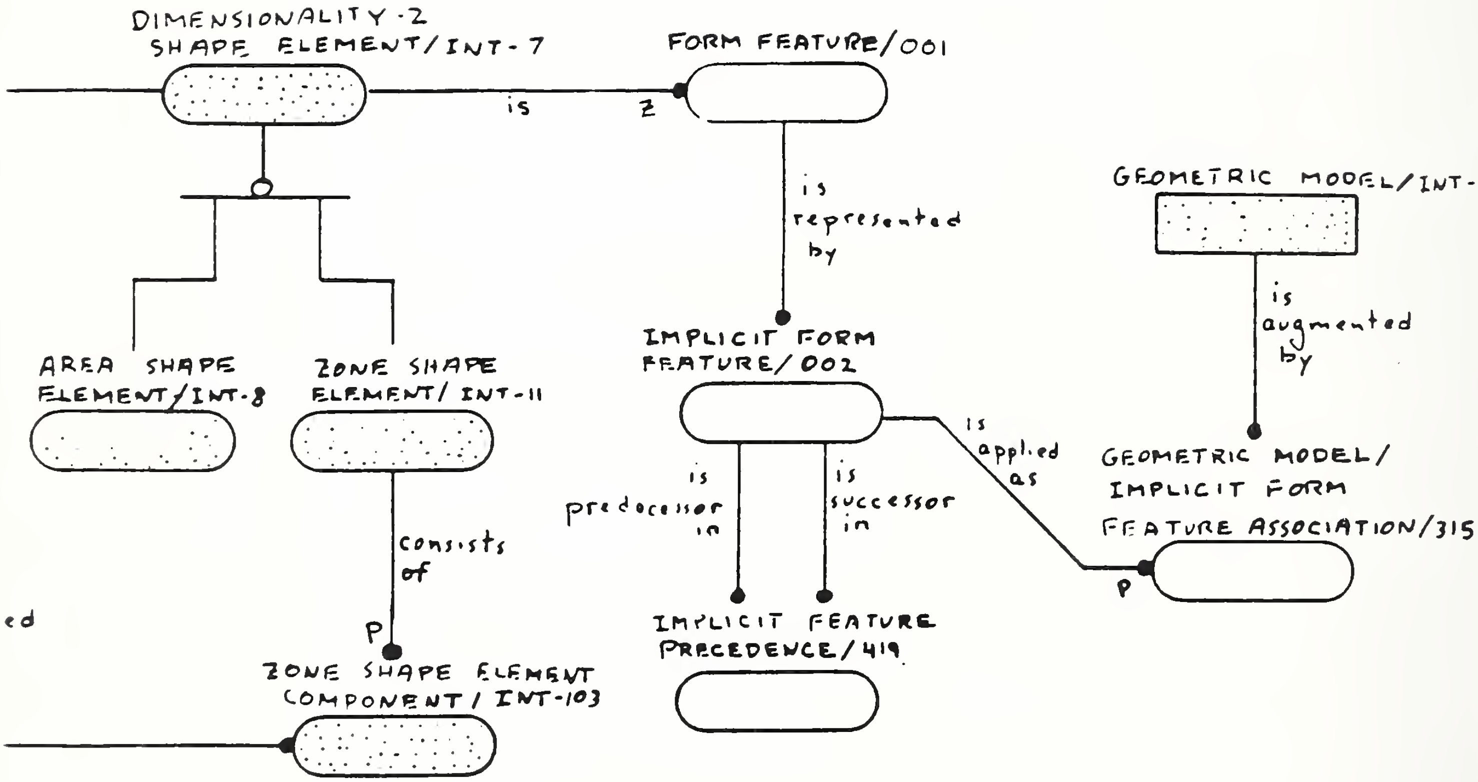


Figure D-37: FFIM Entity/Relationship Diagram

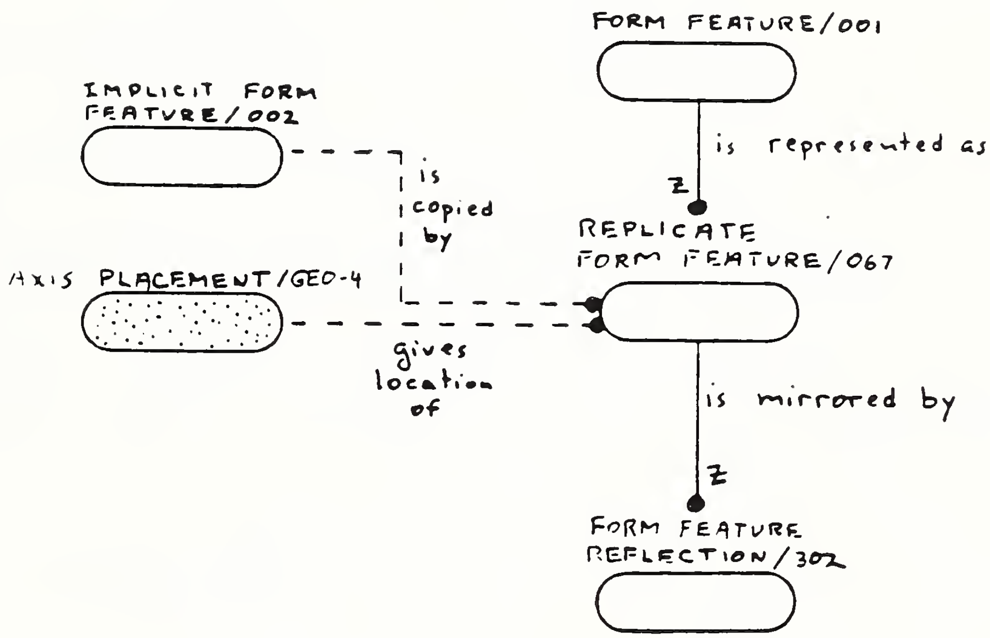


Figure D-38: FFIM Entity/Relationship Diagram

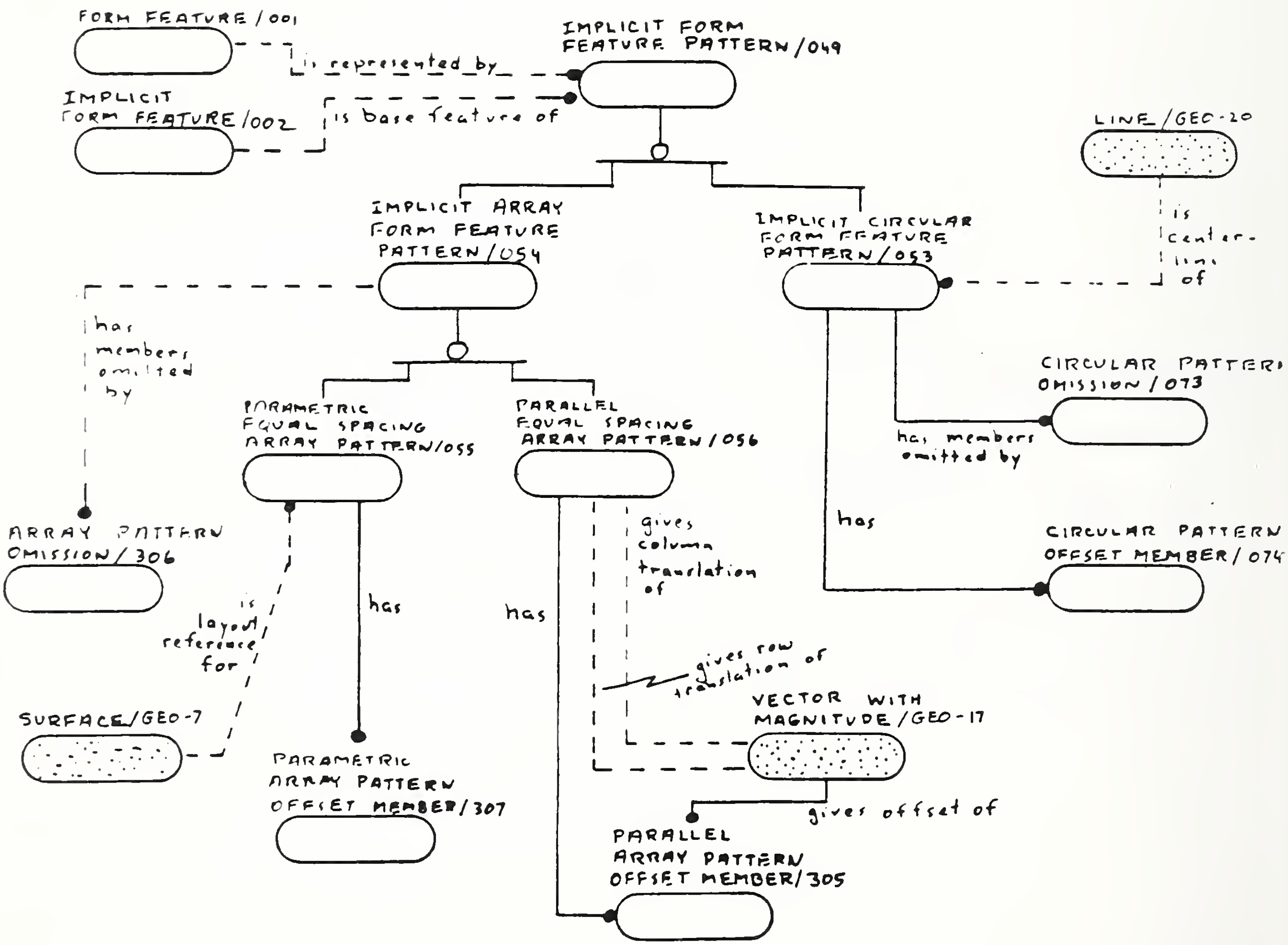


Figure D-39: FFIM Entity/Relationship Diagram

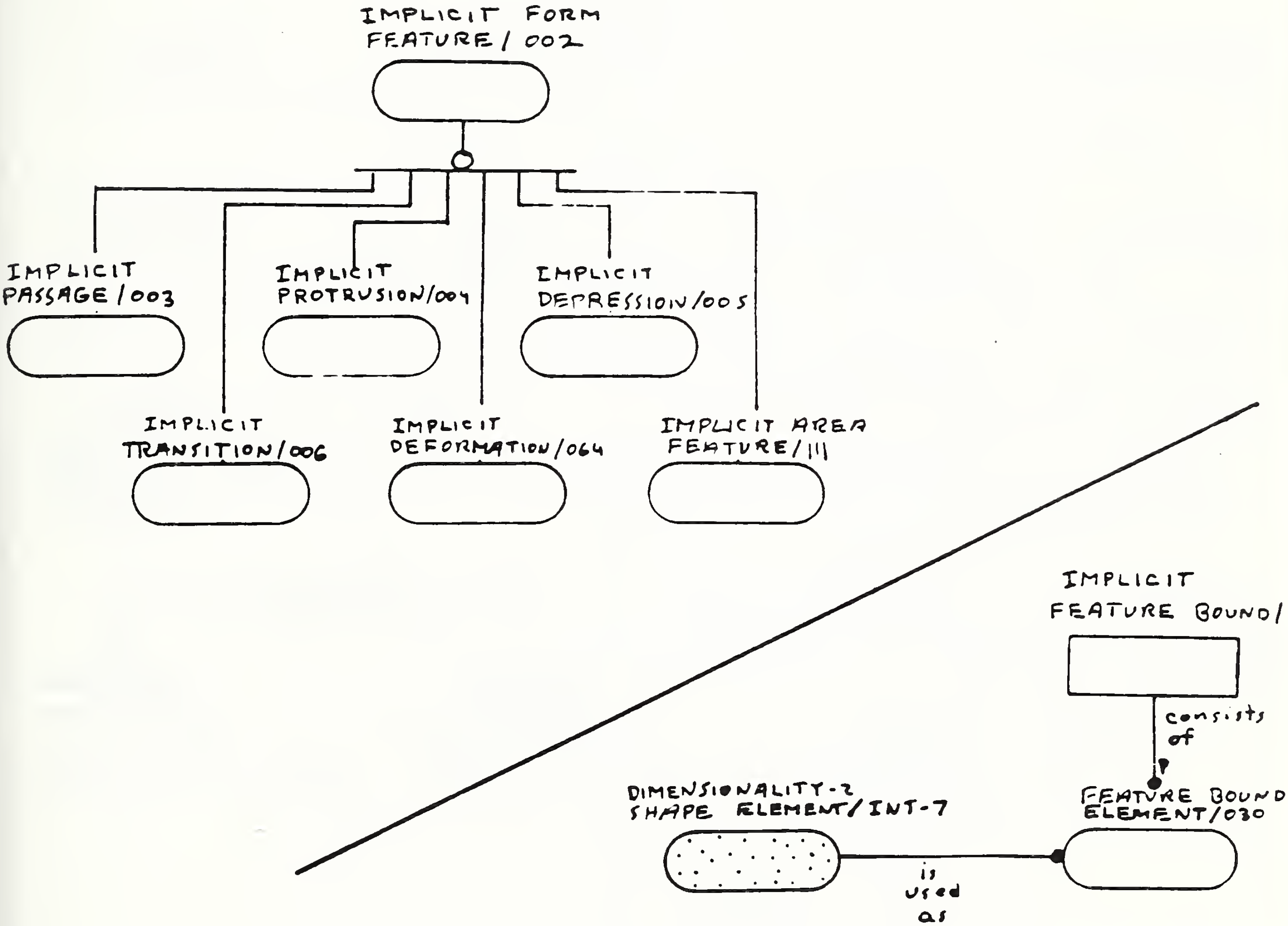


Figure D-40: FFIM Entity/Relationship Diagram

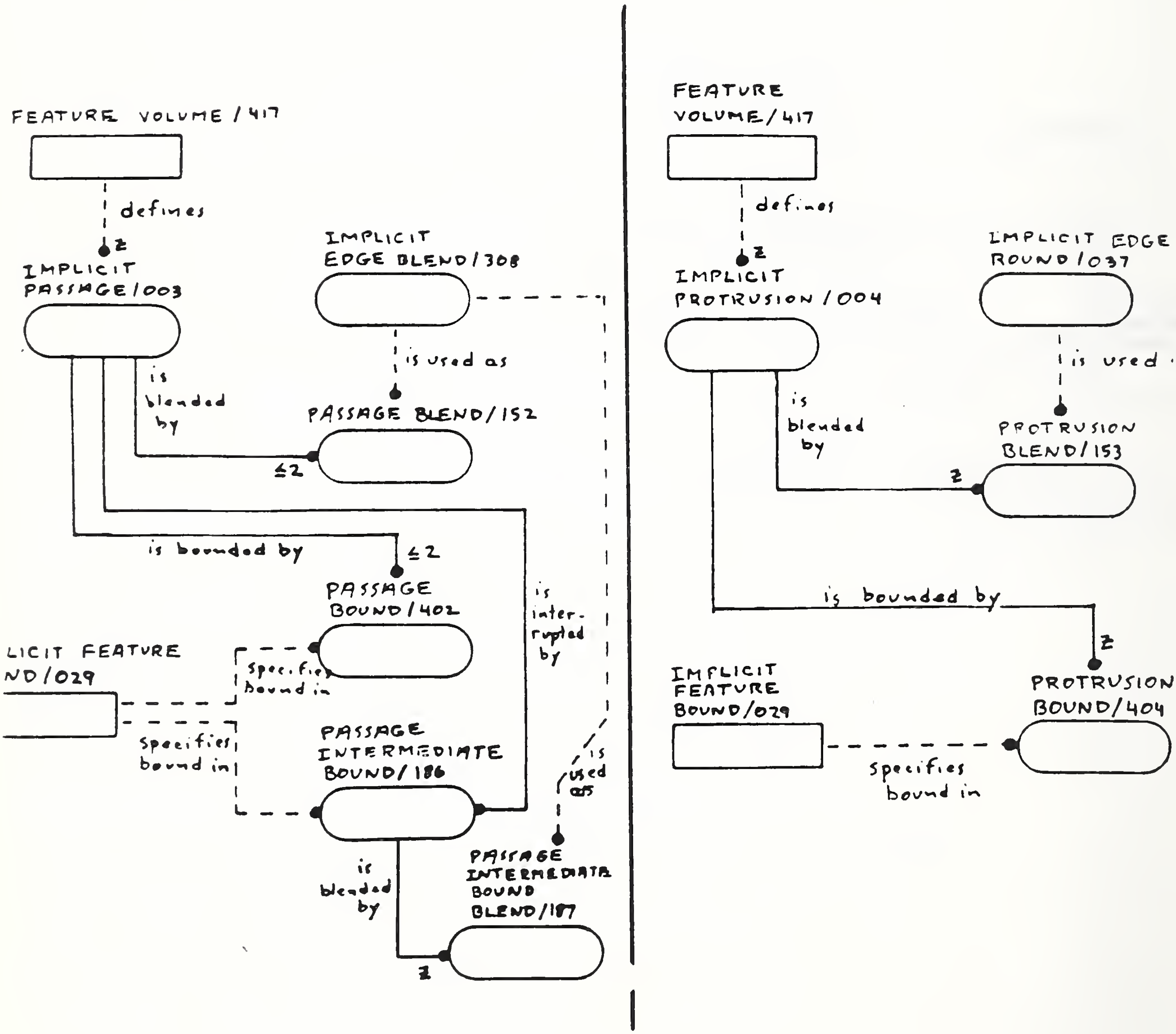


Figure D-41: FFIM Entity/Relationship Diagram

SECTION 4: FORM FEATURES INFORMATION MODEL

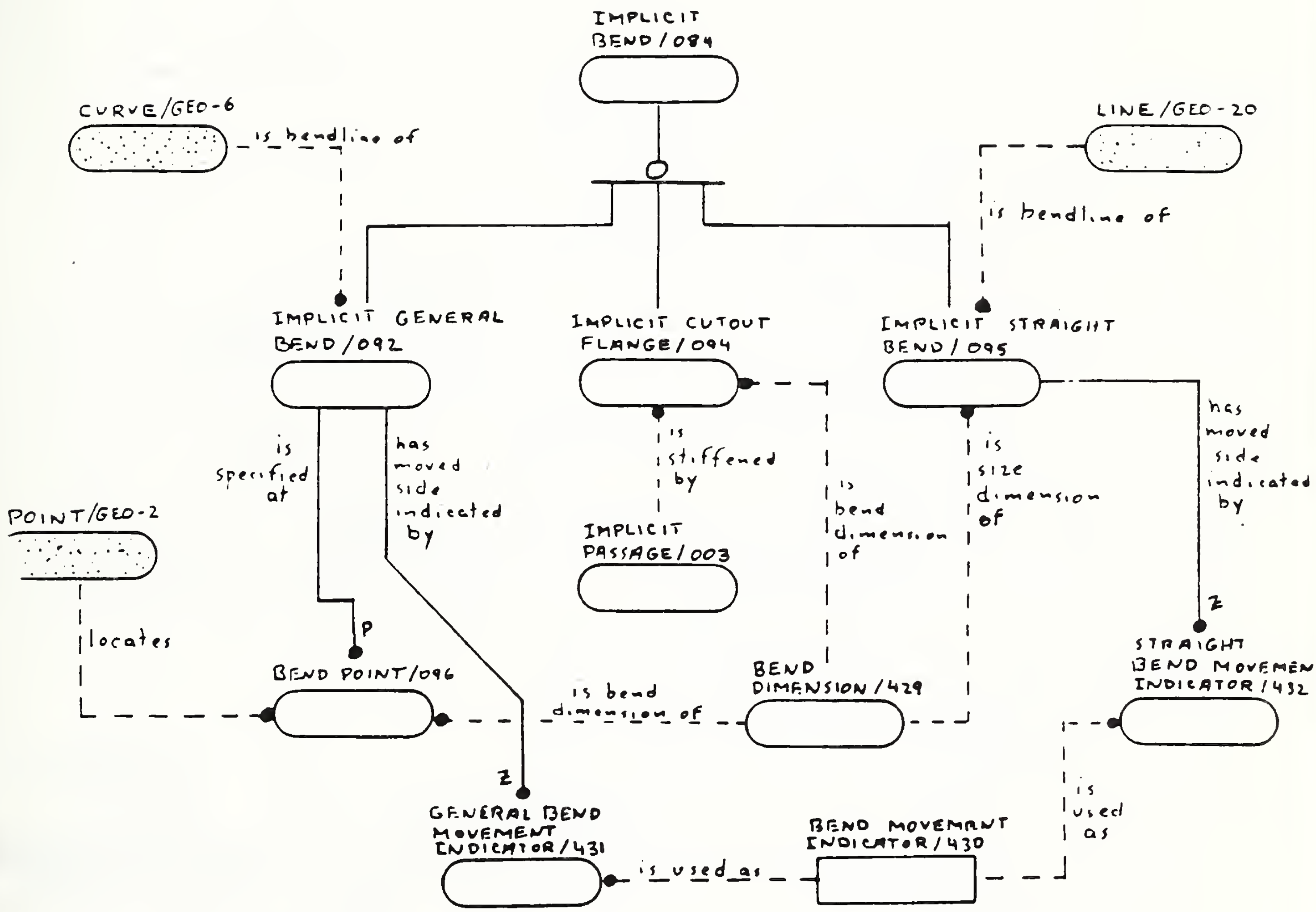


Figure D-42: FFIM Entity/Relationship Diagram

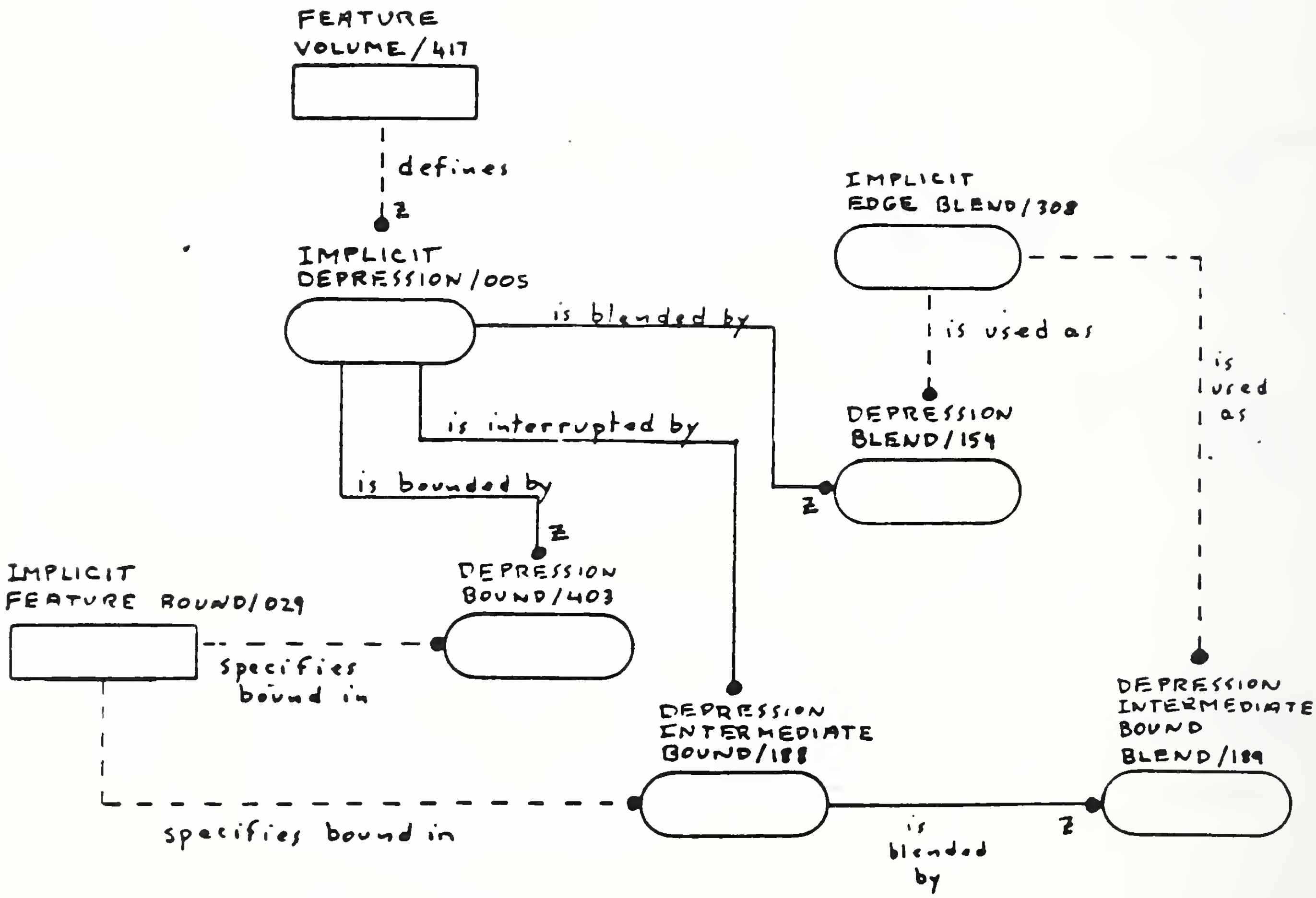


Figure D-43: FFIM Entity/Relationship Diagram

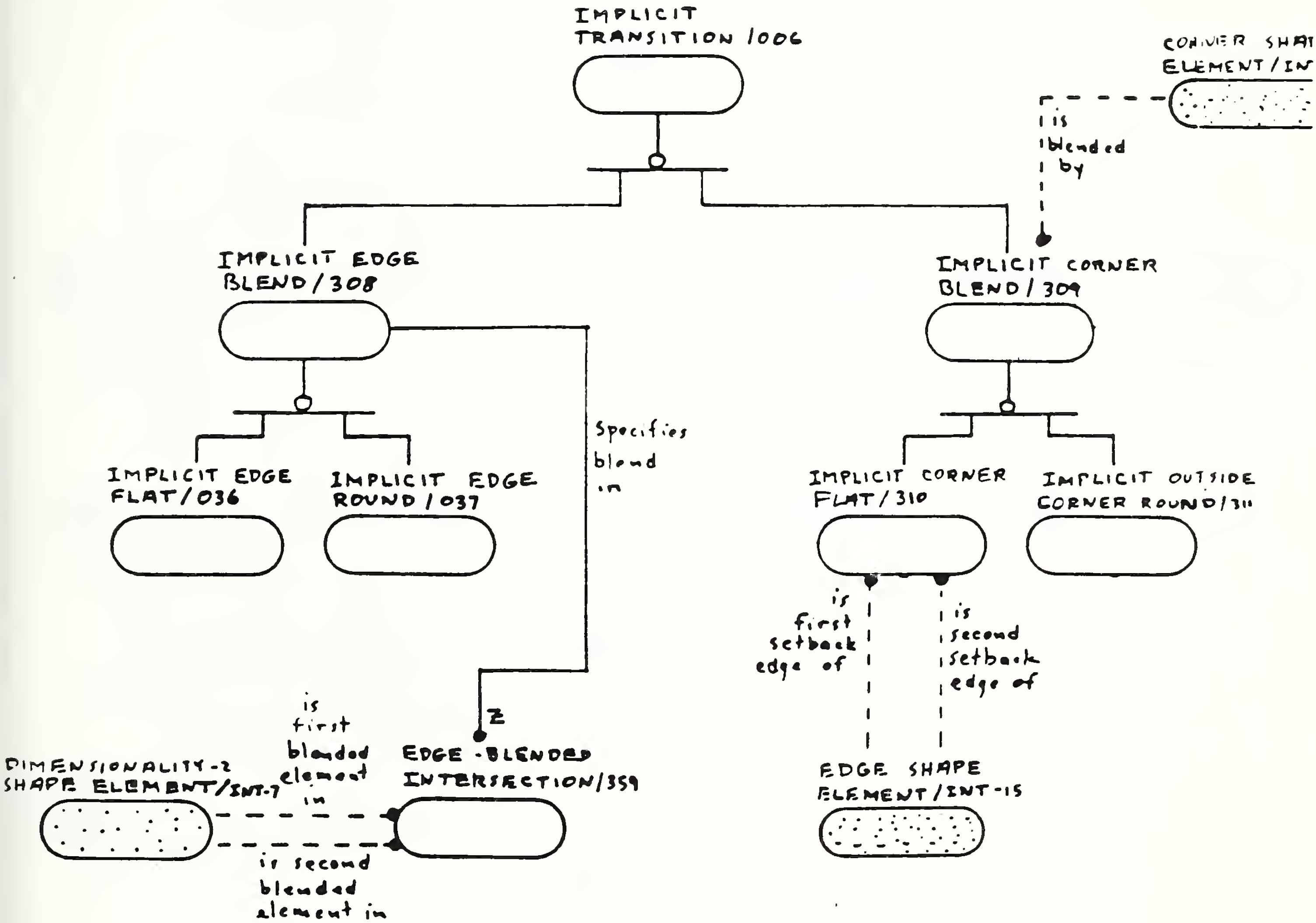


Figure D–44: FFIM Entity/Relationship Diagram

SECTION 4: FORM FEATURES INFORMATION MODEL

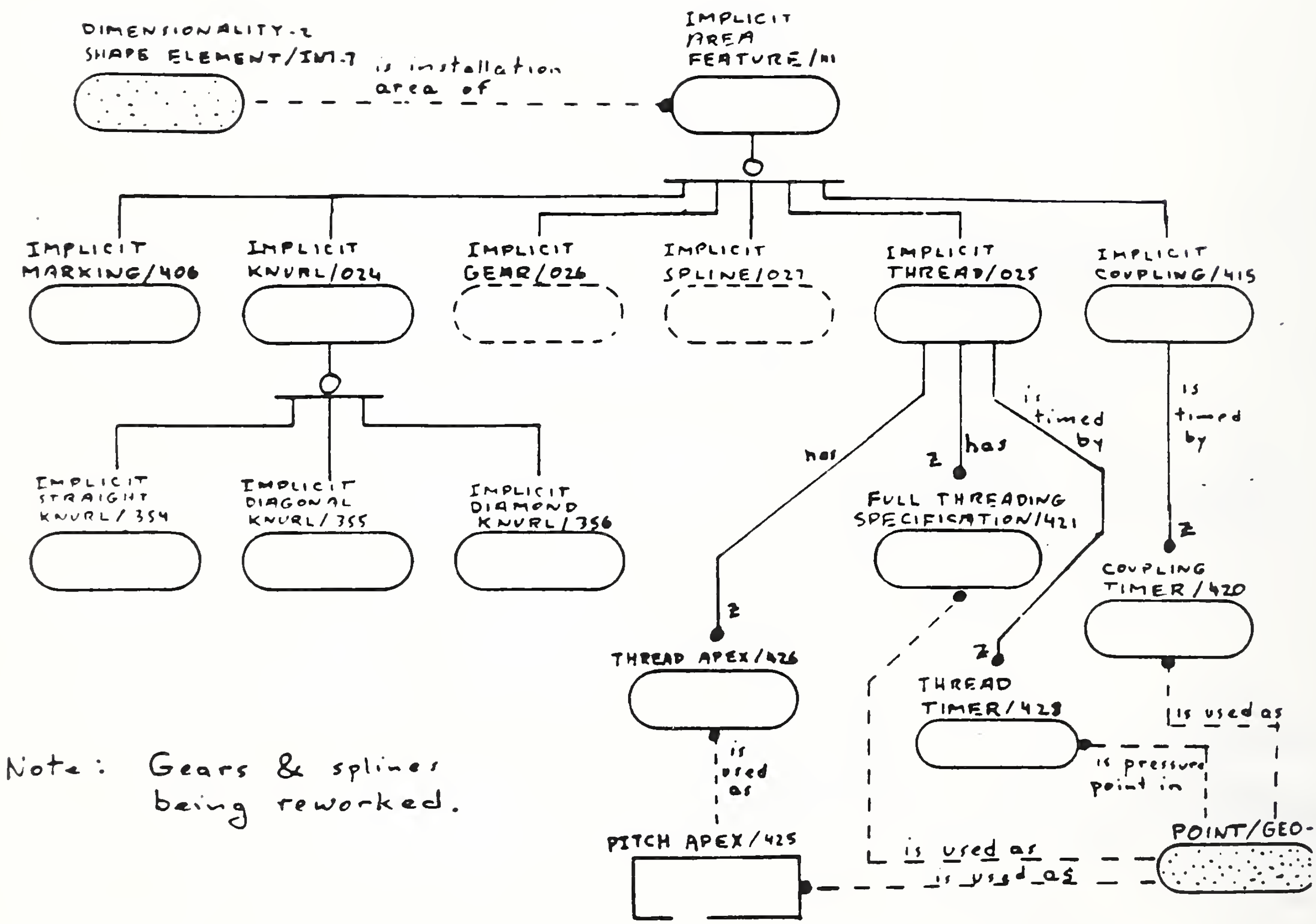


Figure D-45: FFIM Entity/Relationship Diagram

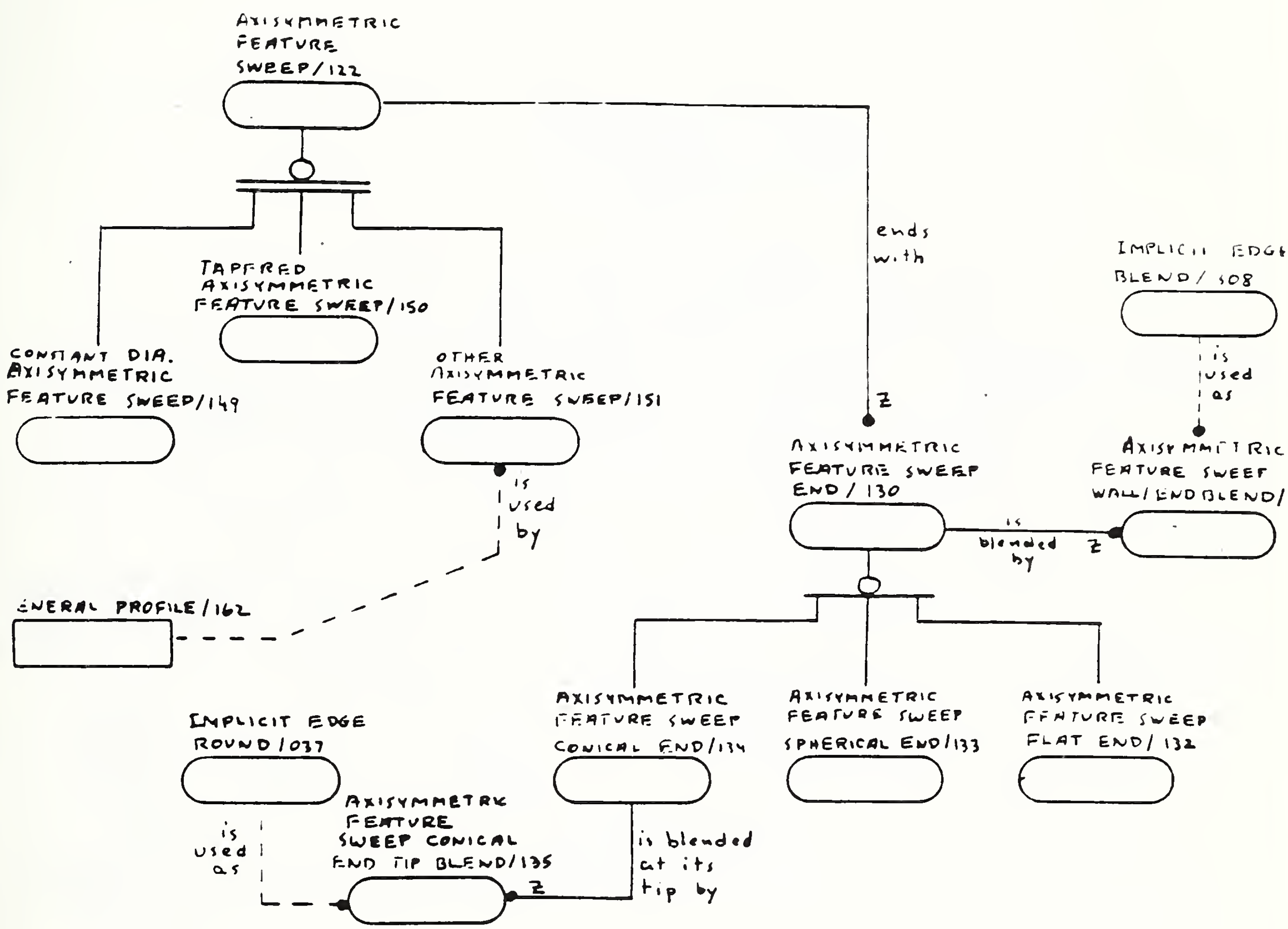


Figure D-46: FFIM Entity/Relationship Diagram

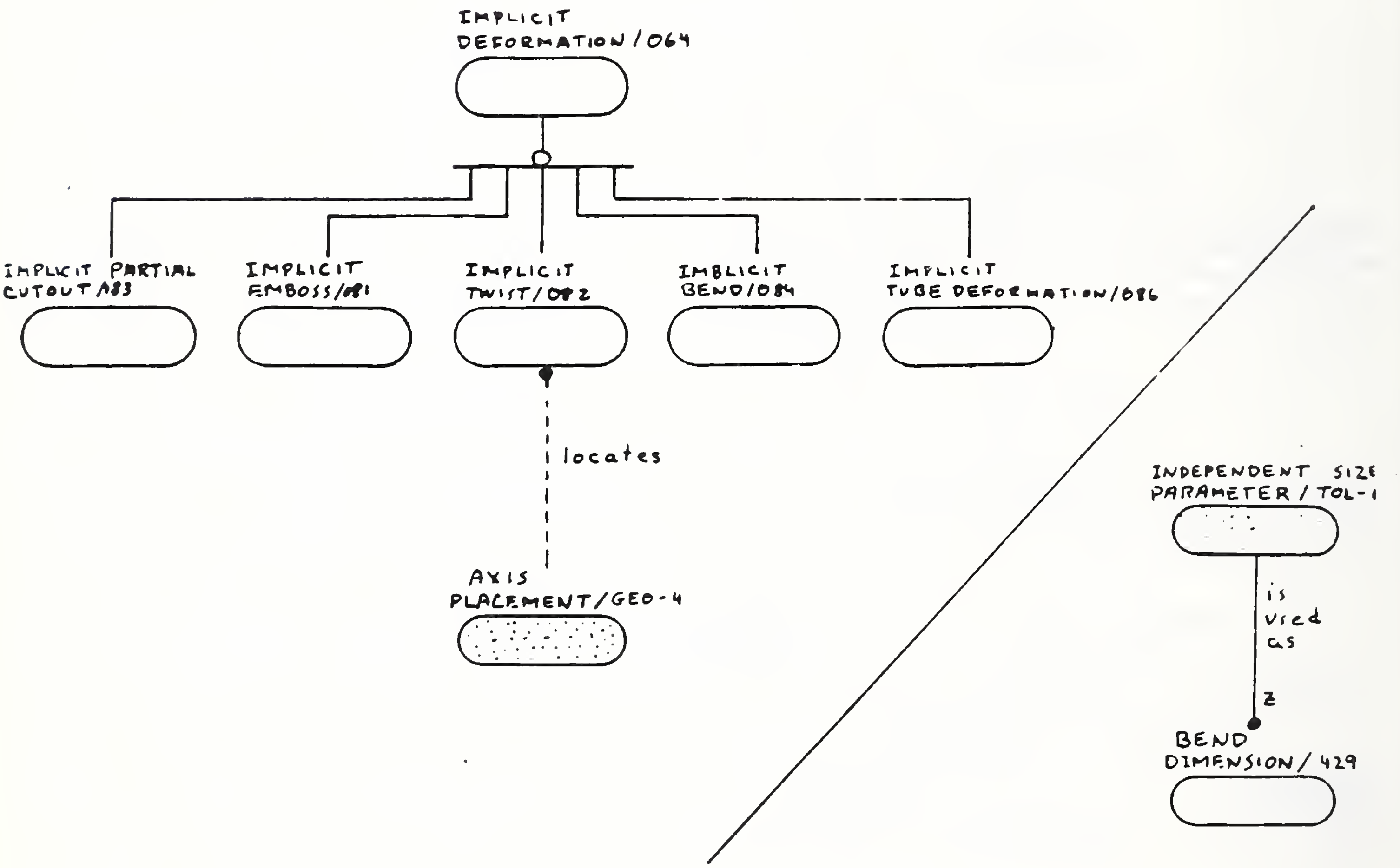


Figure D-47: FFIM Entity/Relationship Diagram

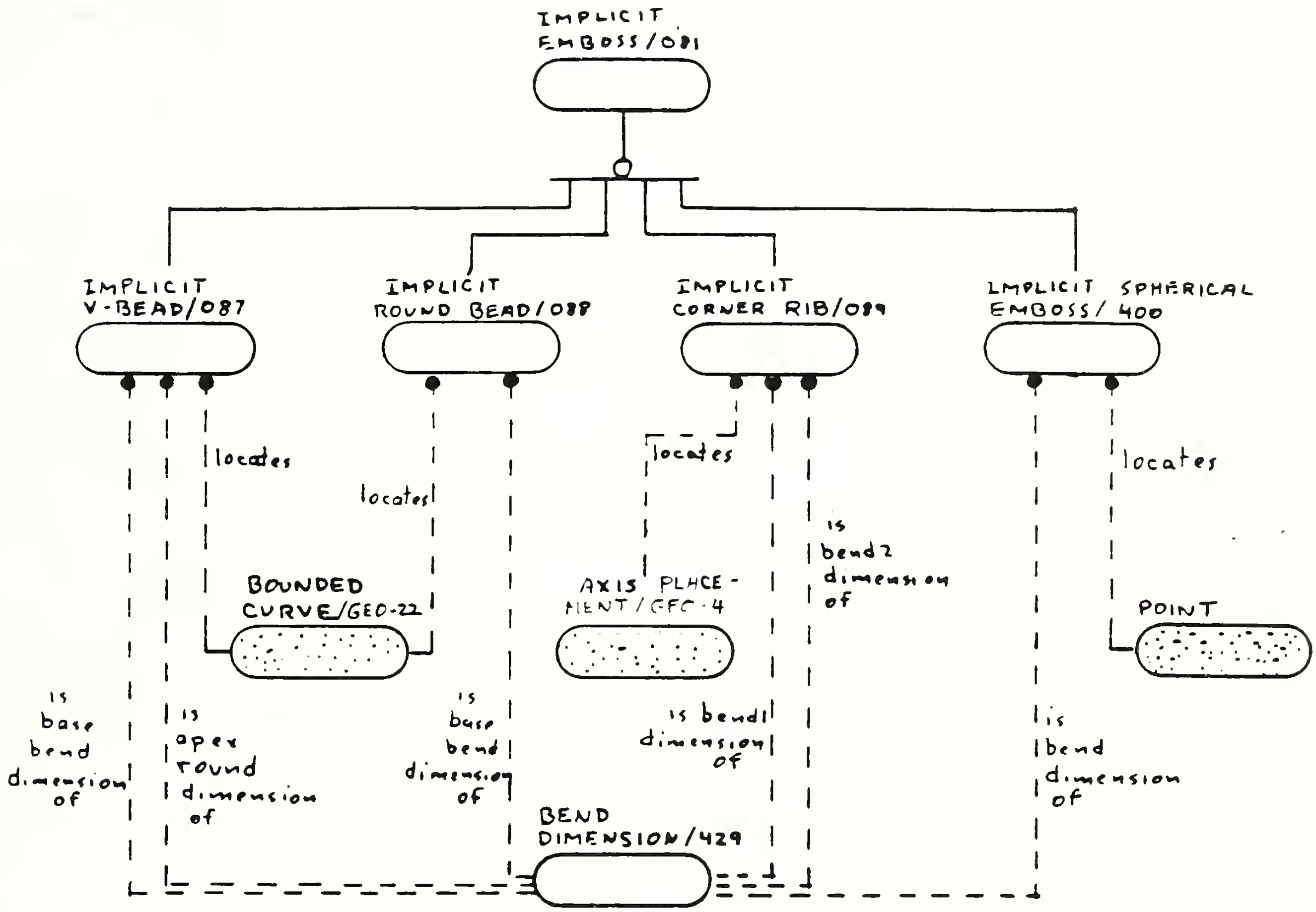


Figure D-48: FFIM Entity/Relationship Diagram

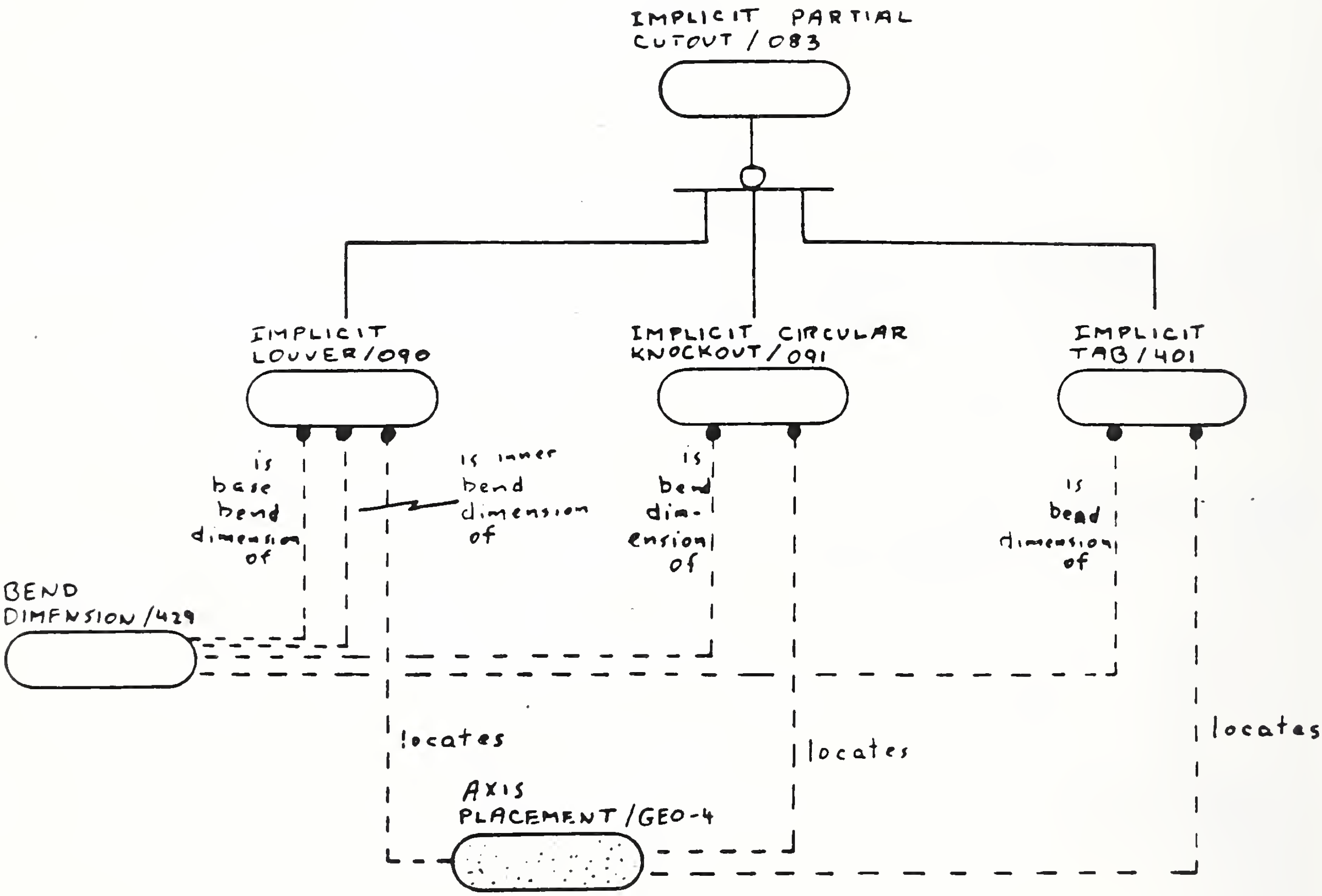


Figure D-49: FFIM Entity/Relationship Diagram

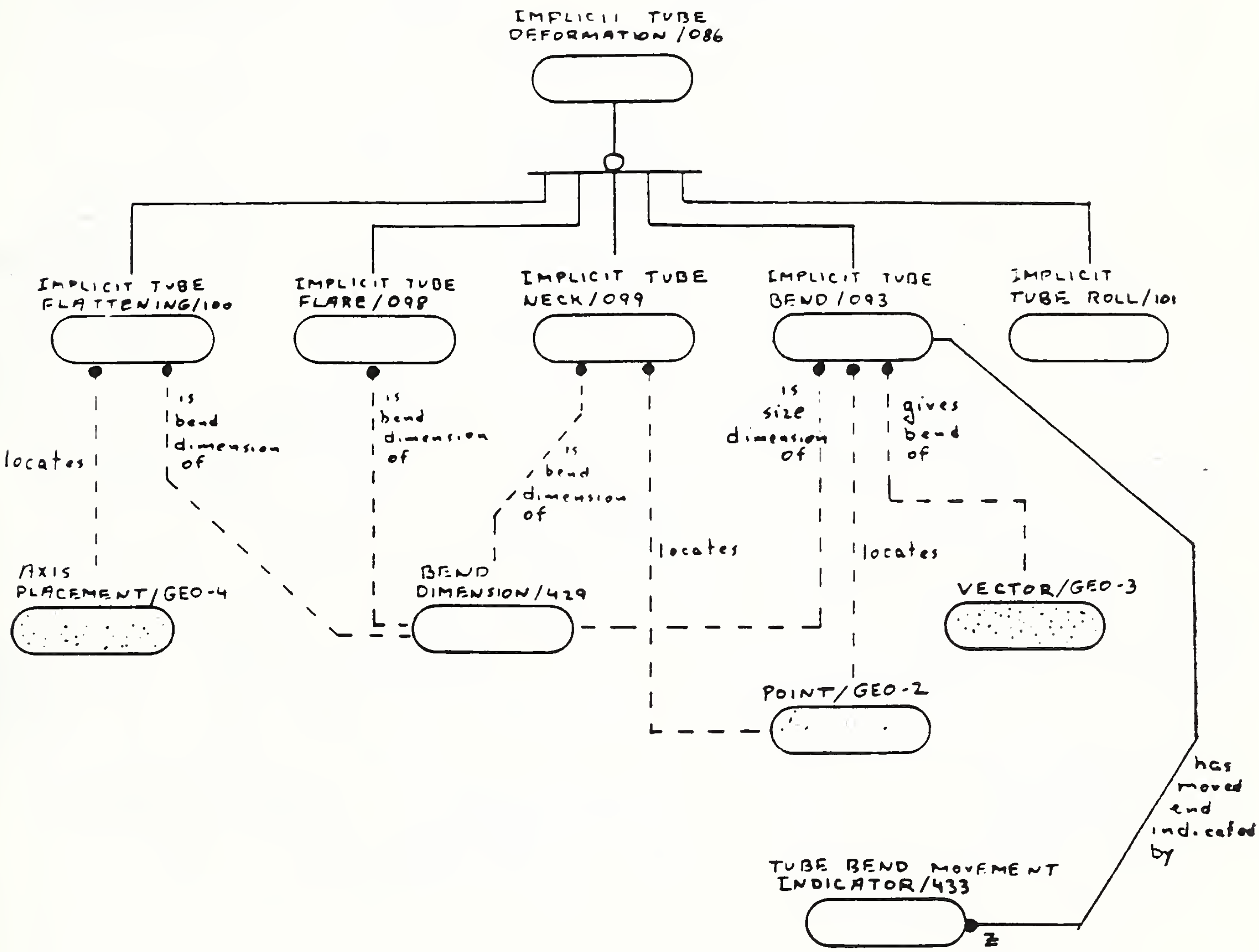


Figure D-50: FFIM Entity/Relationship Diagram

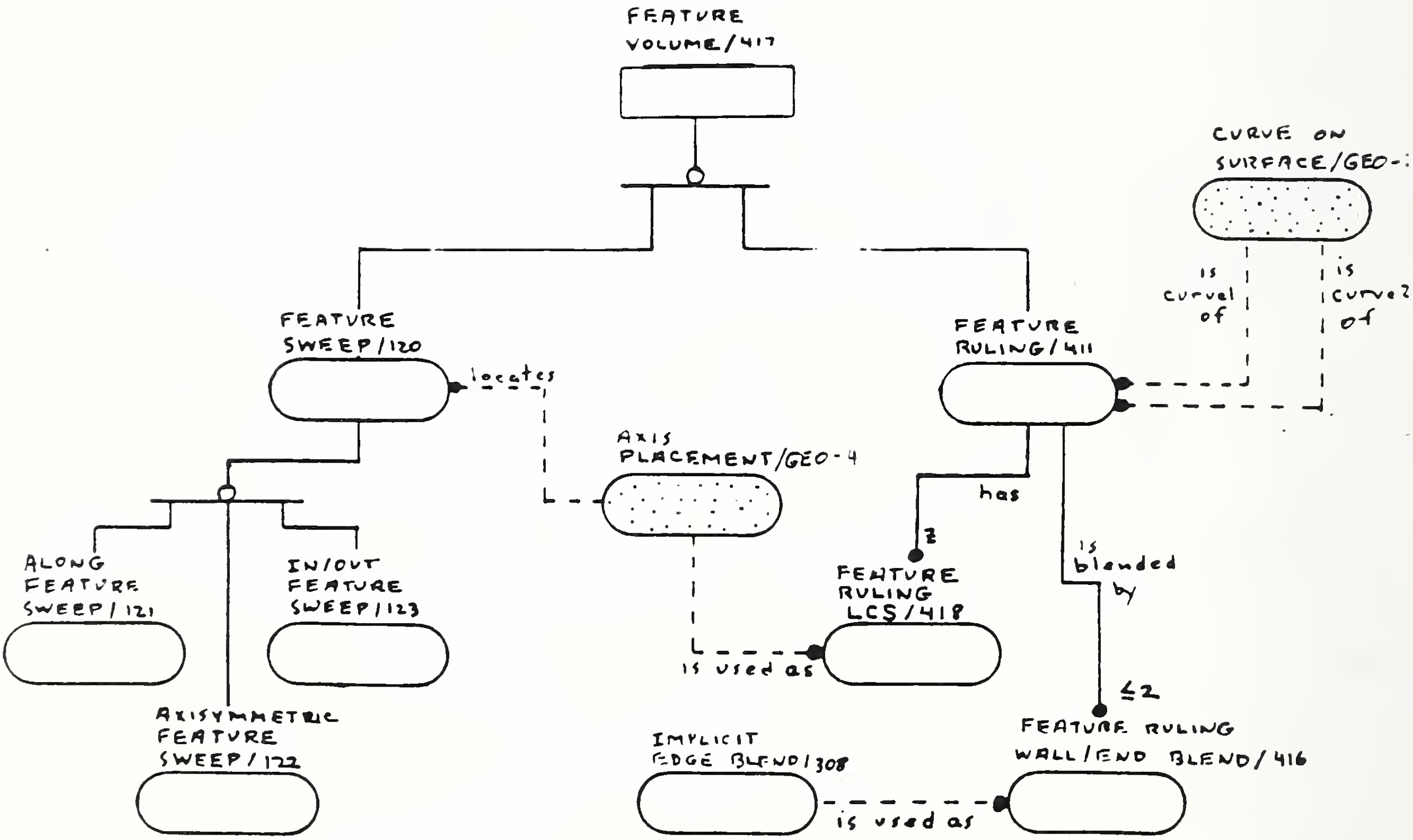


Figure D-51: FFIM Entity/Relationship Diagram

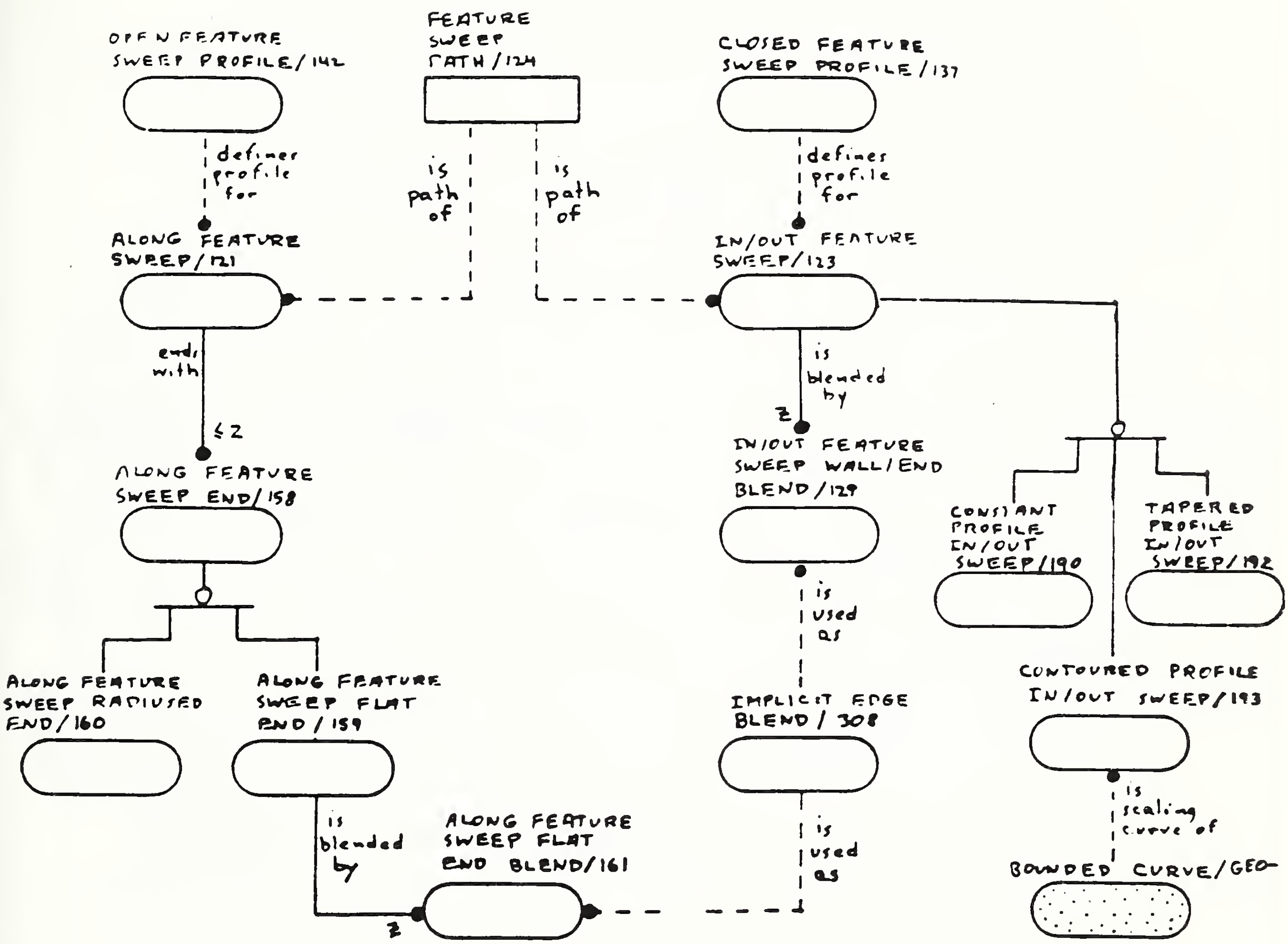


Figure D-52: FFIM Entity/Relationship Diagram

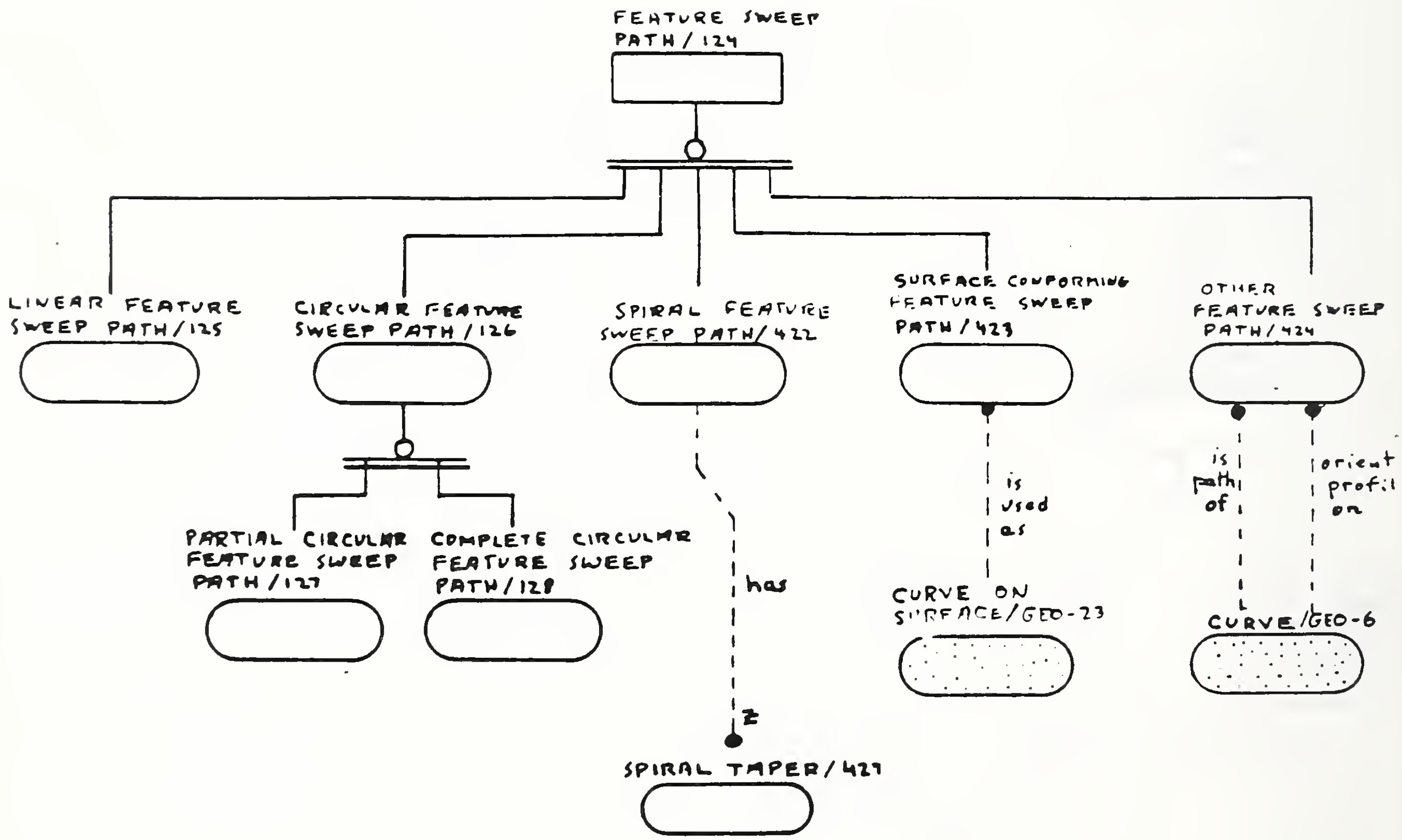


Figure D-53: FFIM Entity/Relationship Diagram

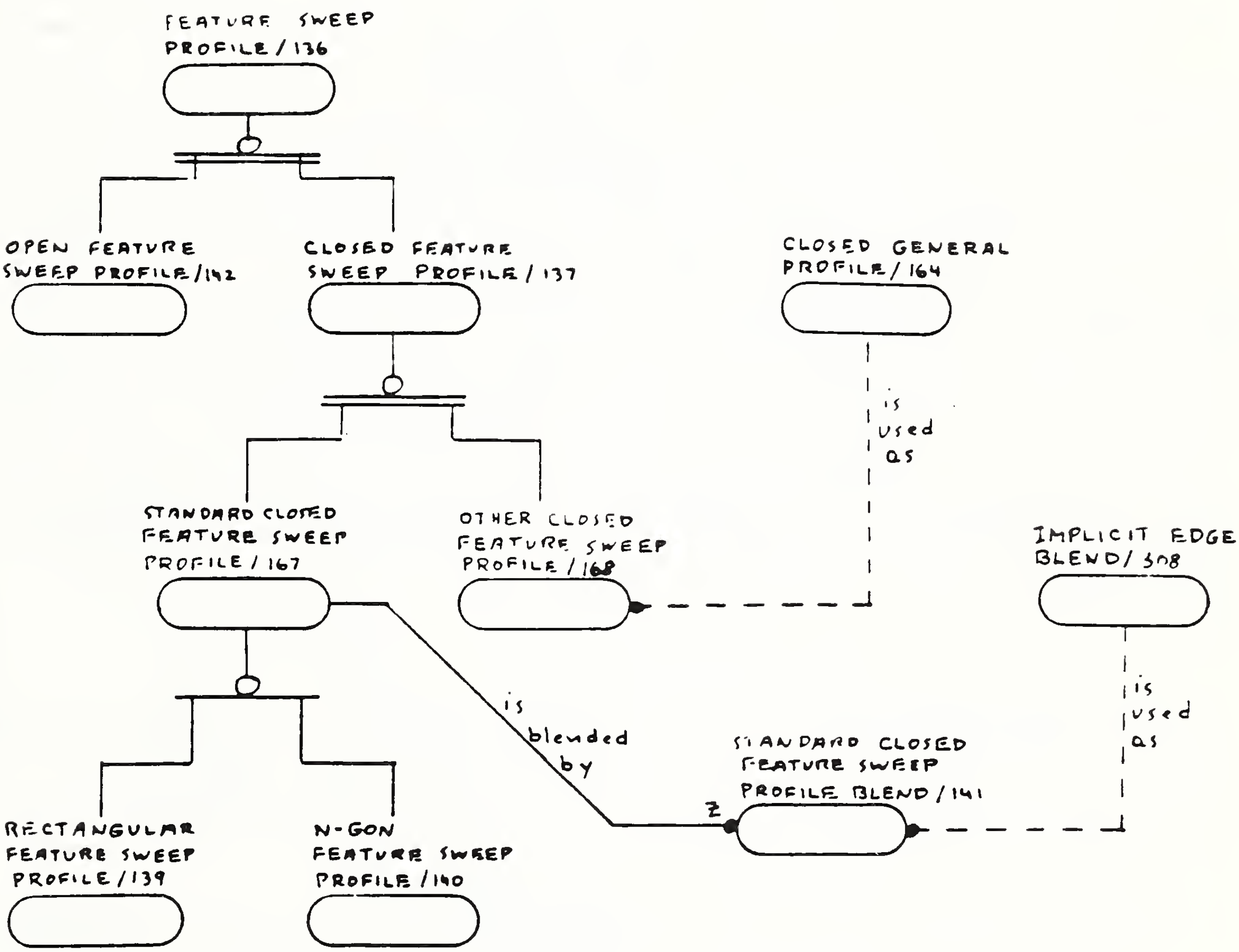


Figure D-54: FFIM Entity/Relationship Diagram

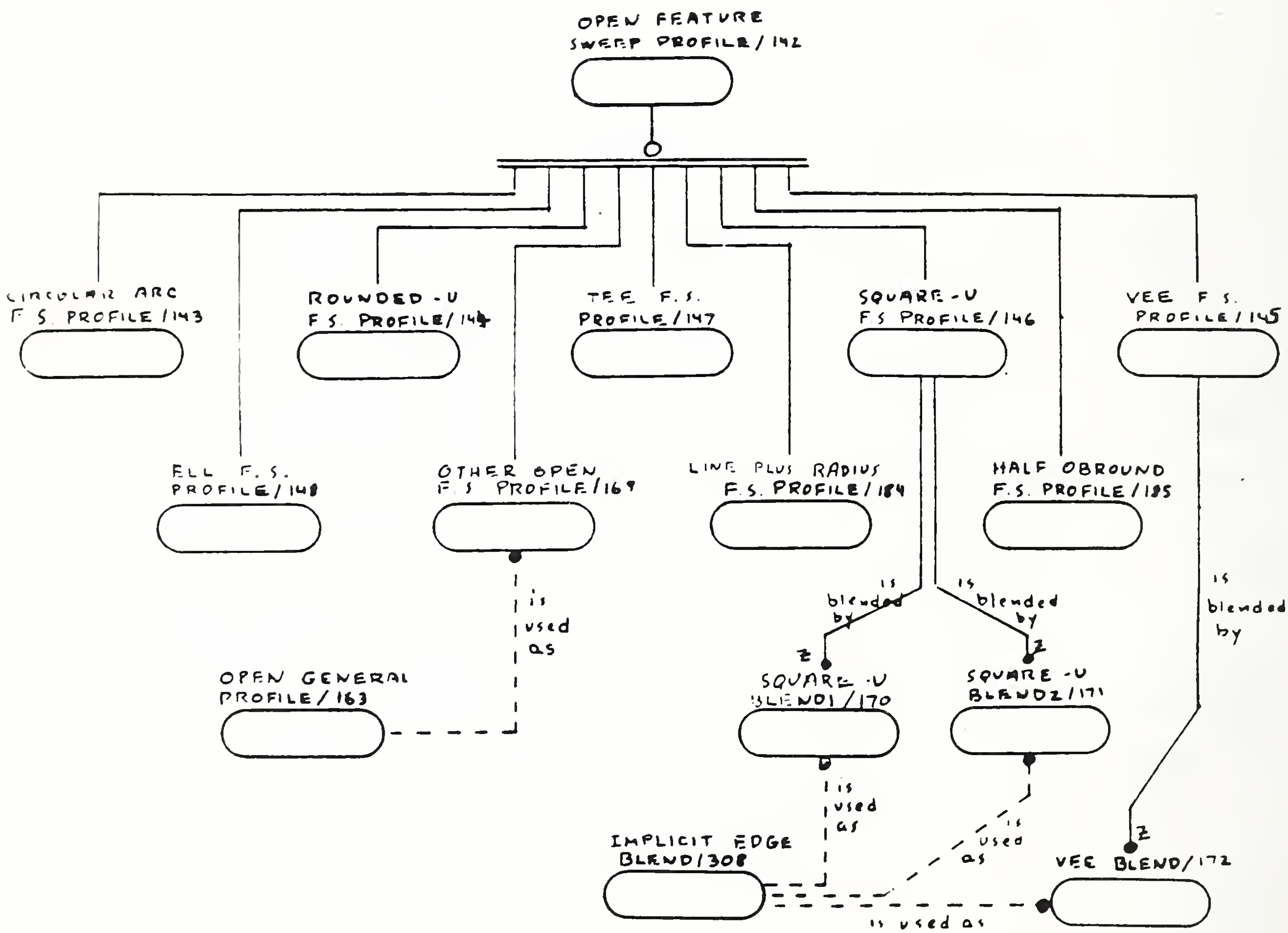


Figure D-55: FFIM Entity/Relationship Diagram

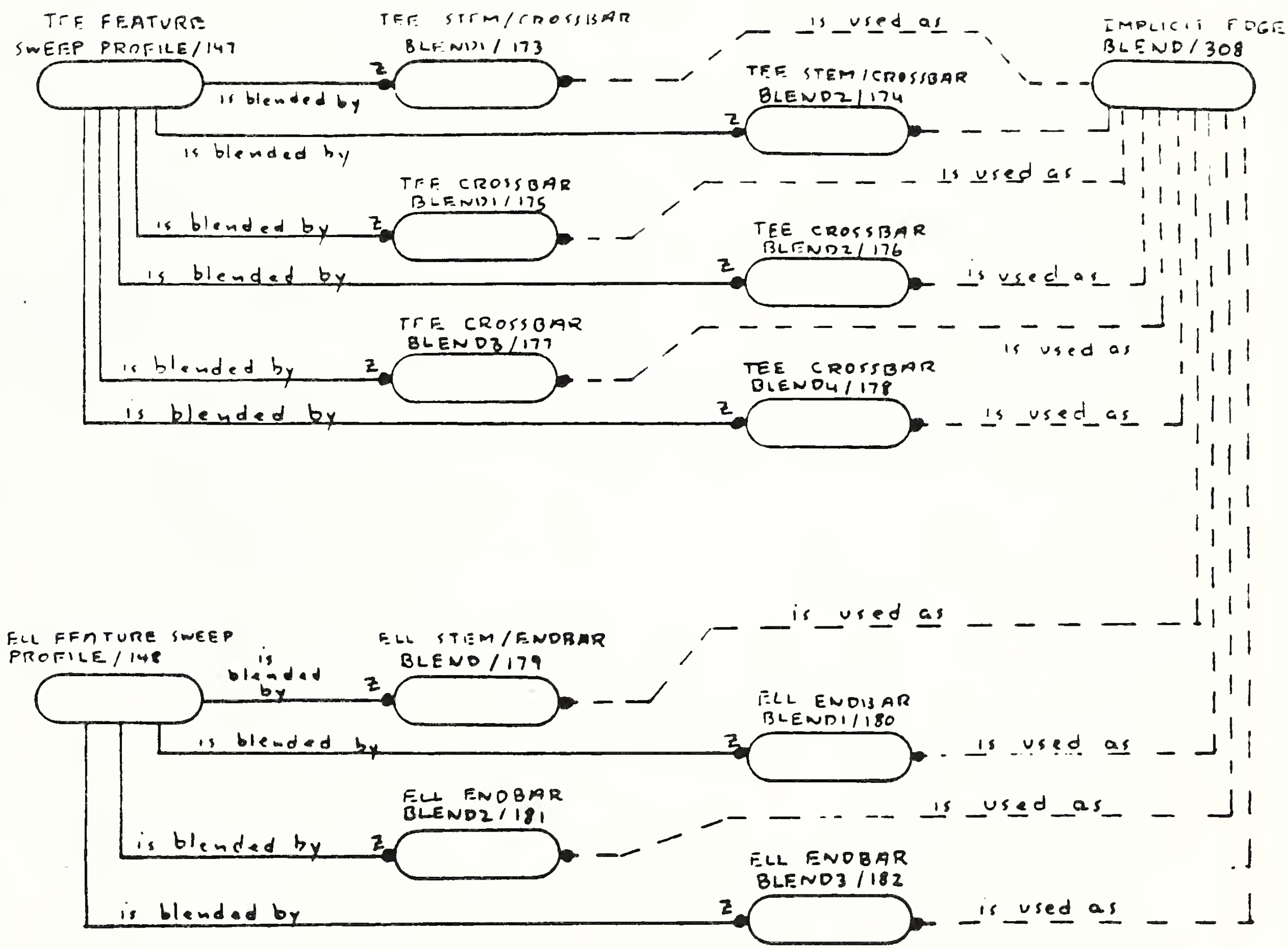


Figure D-56: FFIM Entity/Relationship Diagram

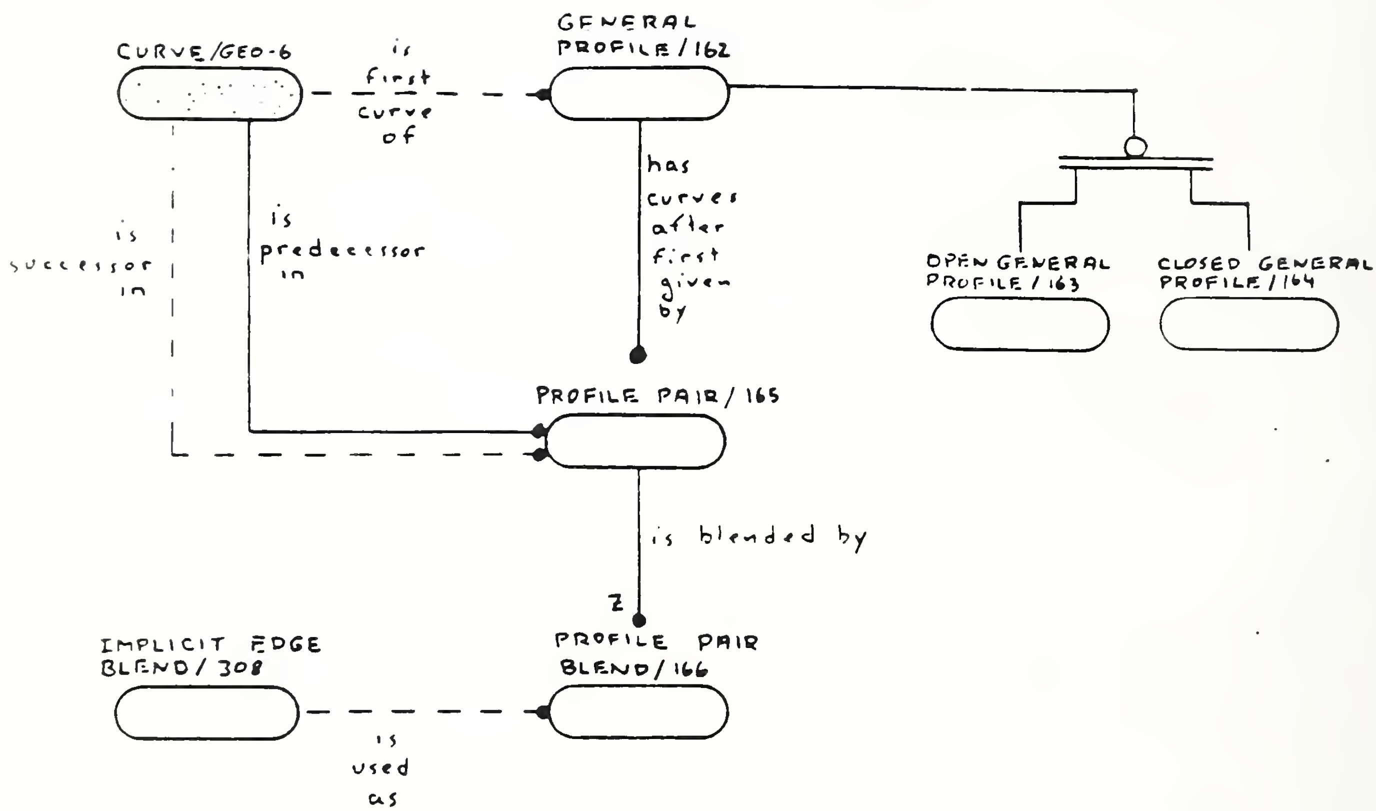


Figure D-57: FFIM Entity/Relationship Diagram

ASSUMED ENVIRONMENT FOR TOLERANCING FEATURE PARAMETERS

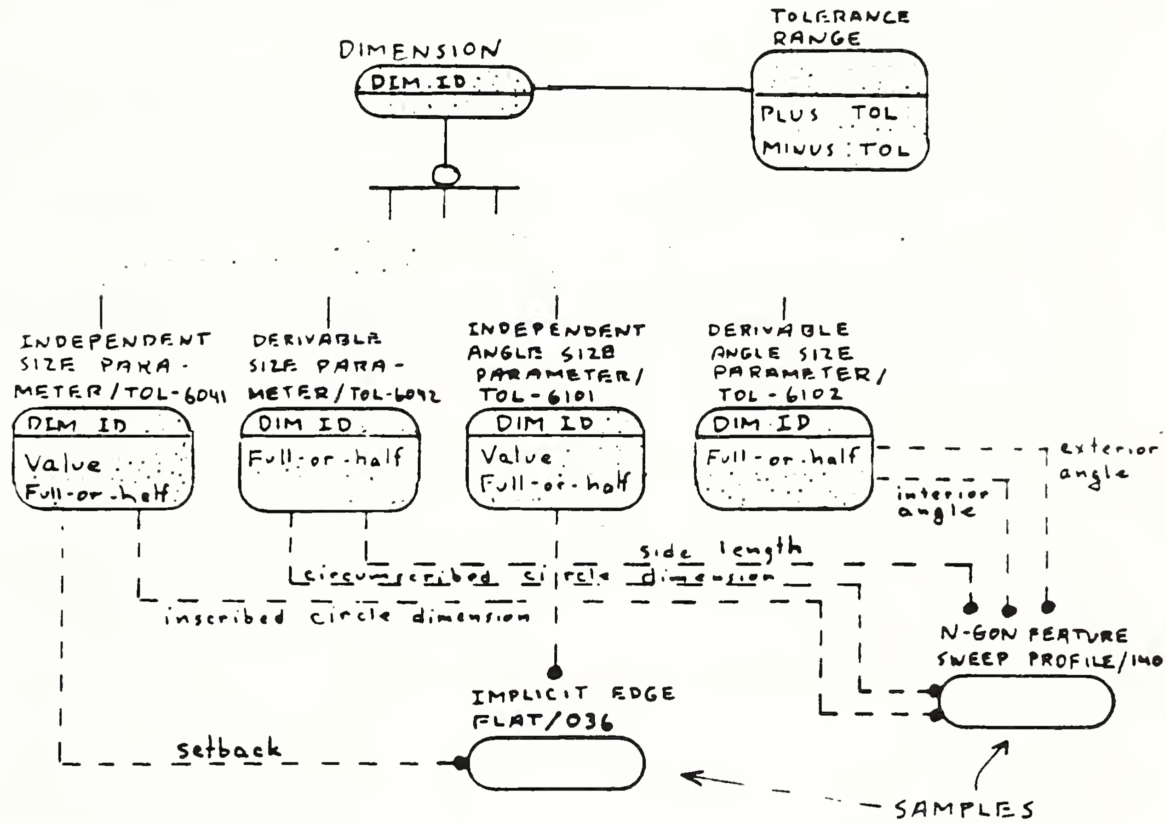


Figure D-58: FFIM Entity/Relationship Diagram

4.4.5 Entity Descriptions

This section contains specifications of the entities of the FFIM. It is primarily an exposition of the IDEF1X model, giving entity definitions, key and non-key attribute definitions, and business rules.

The majority of the business rules state relations between entities. Some, however, state constraints which cannot be conveniently expressed in the IDEF1X language.

The section also gives "in line" EXPRESS for the FFIM. The EXPRESS specification for each entity is given, plus the specification of TYPEs, RULEs, etc. applicable to that entity specification.

The reader will notice that no EXPRESS is given for a number of entities. This is a consequence of the "de-normalization" that typically occurs in a translation from IDEF1X to EXPRESS. In such cases, the information content of the IDEF1X entity is, in EXPRESS, found in one or more other entities. Where this has occurred, a note indicates where the information has been relocated.

It is not intended that the overall EXPRESS model be the sum of the several EXPRESS portions appearing in this section. The complete EXPRESS model for the FFIM appears in subsection 4.7.

Entity Name: FORM FEATURE

Entity Number: 001

A portion of a shape that fits a pattern or stereotype. That is, it is an occurrence of some recognized shape configuration. The stereotype may be precise (constant diameter thru hole) or vague (web).

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)

Other Attributes

FORM FEATURE TYPE
A "vernacular" descriptive name for the feature, e.g., 'pocket', 'bend', 'gear tooth', 'fillet', 'bolt hole circle'. There is no list of acceptable values; the choice is arbitrary.

Business Rules

- A DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 is 0 or 1 FORM FEATUREs/001. This relation provides for enumerative representations of form features (representation by listing the components of the feature, a concept which embraces the common notions of explicit and compound feature representations), since a DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 may be a single AREA or a grouping of DIMENSIONALITY-2 SHAPE ELEMENTs/INT-7.
- A FORM FEATURE/001 is represented by 0, 1, or many IMPLICIT FORM FEATUREs/002.
- A FORM FEATURE/001 is represented as 0 or 1 REPLICATE FORM FEATUREs/067.
- A FORM FEATURE/001 is represented by 0 or 1 IMPLICIT FORM FEATURE PATTERNs/049.

EXPRESS Declaration

```
 ENTITY form_feature;
  feature_type   : STRING(80);
  implicit_reps  : SET [0:#] OF implicit_form_feature;
  pattern_rep    : OPTIONAL implicit_form_feature_pattern;
  replicate_rep  : OPTIONAL replicate_form_feature;
END_ENTITY;
```

Entity Name: IMPLICIT FORM FEATURE

Entity Number: 002

A description of a form feature by parameters and/or geometric data. The description must be sufficient to "realize" the feature; i.e., to compute its shape and location and integrate these into a GEOMETRIC MODEL/INT-23 of the shape which contains the form feature. (For a surfaced wireframe model, for example, this means to determine the faces, loops, edges, and vertices that must be added to/subtracted from the model in order that the GEOMETRIC MODEL/INT-23 "include" the feature. An implicit representation might also be called "constructive" or "descriptive".

Primary Key Attributes

SHAPE ID (FK)
With e following attribute, identifies the represented FORM FEATURE/001.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID
An arbitrary label distinguishing IMPLICIT FORM FEATUREs/002 which represent the same FORM FEATURE/001.

Other Attributes

IMPLICIT FEATURE TYPE (Discriminator)

Business Rules

- A FORM FEATURE/001 is represented by 0, 1, or many IMPLICIT FORM FEATUREs/002.
- An IMPLICIT FORM FEATURE/002 is applied as 1 or many GEOMETRIC MODEL/IMPLICIT FORM FEATURE ASSOCIATIONs/315.
- An IMPLICIT FORM FEATURE/002 may be either an IMPLICIT PASSAGE/003, an IMPLICIT PROTRUSION/004, an IMPLICIT DEPRESSION/005, an IMPLICIT TRANSITION/006, an IMPLICIT DEFORMATION/064, or an IMPLICIT AREA FEATURE/111.
- An IMPLICIT FORM FEATURE/002 is copied by 0, 1, or many REPLICATE FORM FEATUREs/067.
- An IMPLICIT FORM FEATURE/002 is base feature of 0, 1, or many IMPLICIT FORM FEATURE PATTERNs/049.
- An IMPLICIT FORM FEATURE/002 is predecessor in 0, 1, or many IMPLICIT FEATURE PRECEDENCEs/419.
- An IMPLICIT FORM FEATURE/002 is successor in 0, 1, or many IMPLICIT FEATURE PRECEDENCEs/419.

EXPRESS Declaration

```
ENTITY implicit_form_feature
SUPERTYPE OF (implicit_area_feature OR
              implicit_deformation OR
              implicit_depression OR
              implicit_passage OR
              implicit_protrusion OR
              implicit_transition);
END_ENTITY;
```

Entity Name: IMPLICIT PASSAGE

Entity Number: 003

An IMPLICIT FORM FEATURE/002 that is viewed as being "subtracted" from pre-existing shape and which intersects pre-existing shape in two places; i.e., goes through the shape. The feature increases the genus of the shape by 1. Its surface elements generally intersect those of the pre-existing shape at convex solid angles (0 to 180 degrees).

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

FEATURE VOLUME ID (FK) (AK1)

Business Rules

- An IMPLICIT PASSAGE/003 is a type of IMPLICIT FORM FEATURE/002.
- A FEATURE VOLUME/417 defines 0 or 1 IMPLICIT PASSAGEs/003
- An IMPLICIT PASSAGE/003 is bounded by 0,1, or 2 PASSAGE BOUNDs/402.
- An IMPLICIT PASSAGE/003 is blended by 0,1, or 2 PASSAGE BLENDs/152.
- An IMPLICIT PASSAGE/003 is interrupted by 0, 1, or many PASSAGE INTERMEDIATE BOUNDs/186.
- An IMPLICIT PASSAGE/003 is stiffened by 0 or 1 IMPLICIT CUTOUT FLANGES/094.

EXPRESS Declaration

```
ENTITY implicit_passage
SUBTYPE OF (implicit_form_feature);
  definition       : feature_volume;
  end_bounds       : SET [0:2] OF passage_bound;
  boundary_blends  : SET [0:2] OF passage_boundary_blend;
  interruptions    : SET [0:#] OF passage_intermediate_bound;
END_ENTITY;
```

Entity Name: IMPLICIT PROTRUSION

Entity Number: 004

An IMPLICIT FORM FEATURE/002 that extends outward from "the rest of the shape"; i.e., is viewed as being "added" to pre-existing shape. The feature does not change the genus of the shape. Its surface elements generally intersect those of the pre-existing shape at concave solid angles (180 to 360 degrees).

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

FEATURE VOLUME ID (FK) (AK1)

Business Rules

- An IMPLICIT PROTRUSION/004 is a type of IMPLICIT FORM FEATURE/002.
- A FEATURE VOLUME/417 defines 0 or 1 IMPLICIT PROTRUSIONs/004.
- An IMPLICIT PROTRUSION/004 is bounded by 0 or 1 PROTRUSION BOUNDs/404.
- An IMPLICIT PROTRUSION/004 is blended by 0 or 1 PROTRUSION BLENDs/153.

EXPRESS Declaration

```
ENTITY implicit_protrusion
SUBTYPE OF (implicit_form_feature);
  definition : feature_volume;
```

```
  end_bound   : OPTIONAL implicit_feature_bound;
  end_blend   : OPTIONAL implicit_edge_round;
END_ENTITY;
```

Entity Name: IMPLICIT DEPRESSION

Entity Number: 005

An IMPLICIT FORM FEATURE/002 that extends inward from "the rest of the shape"; i.e., is viewed as being "subtracted" from pre-existing shape. The feature does not change the genus of the shape; i.e., does not go through the shape. Its surface elements generally intersect those of the pre-existing shape at convex solid angles (0 to 180 degrees).

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

FEATURE VOLUME ID (FK) (AK1)

Business Rules

- An IMPLICIT DEPRESSION/005 is a type of IMPLICIT FORM FEATURE/002.
- A FEATURE VOLUME/417 defines 0 or 1 IMPLICIT DEPRESSIONs/005.
- An IMPLICIT DEPRESSION/005 is bounded by 0 or 1 DEPRESSION BOUNDs/403.
- An IMPLICIT DEPRESSION/005 is interrupted by 0, 1 or many DEPRESSION INTERMEDIATE BOUNDs/188.
- An IMPLICIT DEPRESSION/005 is blended by 0 or 1 DEPRESSION BLENDs/154.

EXPRESS Declaration

```
ENTITY implicit_depression
SUBTYPE OF (implicit_form_feature);
  definition    : feature_volume;
  end_bound     : OPTIONAL implicit_feature_bound;
  end_blend     : OPTIONAL implicit_edge_blend;
  interruptions : SET [0:*] OF depression_intermediate_bound;
END_ENTITY;
```

Entity Name: IMPLICIT TRANSITION

Entity Number: 006

An implicit representation of a FORM FEATURE/001 which connects two elements of pre-existing shape. As a rule, the transition is regarded as less important than the elements it connects. Fillets and chamfers are examples.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

IMPLICIT TRANSITION TYPE (Discriminator)

Business Rules

- An IMPLICIT TRANSITION/006 is a type of IMPLICIT FORM FEATURE/002.
- An IMPLICIT TRANSITION/006 may be either an IMPLICIT EDGE BLEND/308 or an IMPLICIT CORNER BLEND/309.

EXPRESS Declaration

```
ENTITY implicit_transition
SUPERTYPE OF (implicit_corner_blend OR
              implicit_edge_blend)
SUBTYPE OF (implicit_form_feature);
END_ENTITY;
```

Entity Name: IMPLICIT KNURL

Entity Number: 024

An implicit representation of a scoring pattern on a surface.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

KNURL NUMBER OF TEETH
The number of teeth on the knurling tool to be used.

MAJOR.DIM ID (FK)
The dimension (diameter, preferably, or radius) of the major or outside diameter of the knurling tool.

NOMINAL.DIM ID (FK)
The dimension (diameter, preferably, or radius) of the nominal diameter of the knurling tool.

DEPTH.DIM ID (FK)
The tooth depth dimension of the knurling tool.

FILLET AT ROOT.DIM ID (FK)
The dimension (diameter or, preferably, radius) of the root fillet between teeth.

IMPICIT KNURL TYPE (Discriminator)

Business Rules

- An IMPLICIT KNURL/024 is a type of IMPLICIT AREA FEATURE/111.
- An IMPLICIT KNURL/024 may be either an IMPLICIT STRAIGHT KNURL/354, an IMPLICIT DIAGONAL KNURL/355, or an IMPLICIT DIAMOND KNURL/356.
- The FEATURE APPLICATION AREA(s)/028, if any, associated with an IMPLICIT KNURL/024 must be cylindrical.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 gives major diameter dimension of 0, 1, or many IMPLICIT KNURLs/024.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 gives nominal diameter dimension of 0, 1, or many IMPLICIT KNURLs/024.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 gives tooth depth dimension of 0, 1, or many IMPLICIT KNURLs/024.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 gives fillet at root dimension of 0, 1, or many IMPLICIT KNURLs/024.

EXPRESS Declaration

```
ENTITY implicit_knurl
SUPERTYPE OF (implicit_diagonal_knurl OR
              implicit_diamond_knurl OR
              implicit_straight_knurl)
SUBTYPE OF (implicit_area_feature);
  number_of_teeth    : INTEGER;
  knurl_major_dim    : independent_size_parameter;
  knurl_nominal_dim  : independent_size_parameter;
  tooth_depth        : independent_size_parameter;
  fillet_at_root     : independent_size_parameter;
WHERE
  cylindrical(installation_area);
END_ENTITY;

FUNCTION cylindrical(area:dimensionality_2_shape_element):LOGICAL;
-- determines if an area is cylindrical
END_FUNCTION;
```

Entity Name: IMPLICIT THREAD

Entity Number: 025

An implicit representation of a spiral groove installed on a cylindrical or conical surface.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

THREAD HAND
  Whether the thread is RIGHT handed or LEFT handed.

THREADS PER UNIT
  The number of threads per axial unit of length.

PITCH.DIM ID (FK)
  The dimension (diameter, preferably, or radius) of the pitch circle.

MAJOR.DIM ID (FK)
  The dimension (diameter, preferably, or radius) of the major circle.

MINOR DIM ID (FK)
The dimension (diameter, preferably, or radius) of the minor circle.

THREAD FORM
One of the standard thread forms: SHARPV, UNIFIED, SIXTY DEG, STUB, ACME, SQUARE, STUB ACME, BUTTRESS, KNUCKLE, BRIT STD.

THREAD FIT CLASS
One of the standard Unified National classes of fit: 1A, 2A, 3A, 1B, 2B, 3B.

Business Rules

- An IMPLICIT THREAD/025 is a type of IMPLICIT AREA FEATURE/111.
- The FEATURE APPLICATION AREA(s)/028, if any, associated with an IMPLICIT THREAD/025 must be cylindrical or conical.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 gives pitch circle dimension of 0, 1, or many IMPLICIT THREADs/025.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 gives major circle dimension of 0, 1, or many IMPLICIT THREADs/025.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 gives minor circle dimension of 0, 1, or many IMPLICIT THREADs/025.
- An IMPLICIT THREAD/025 has 0 or 1 FULL THREADING SPECIFICATIONs/421.
- An IMPLICIT THREAD/025 has 0 or 1 THREAD APEXes/426. (The THREAD APEX/426 is present if and only if the thread is tapered. In this case, the thread's pitch diameter, major diameter, and minor diameter are measured at the axial point derivable from the pitch apex and distance of the PITCH CONE/422. The axis is known from the DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 of the parent IMPLICIT AREA FEATURE/111.)
- An IMPLICIT THREAD/025 is timed by 0 or 1 THREAD TIMERs/428.

EXPRESS Declaration

```
ENTITY implicit_thread
SUBTYPE OF (implicit_area_feature);
  major_dim         : independent_size_parameter;
  threads_per_unit  : REAL;
  thread_form       : thread_forms;
  thread_hand       : hands;
  pitch_dim         : OPTIONAL independent_size_parameter;
  minor_dim         : OPTIONAL independent_size_parameter;
  thread_fit_class  : OPTIONAL thread_fit_classes;
  thread_spec       : OPTIONAL full_threading_specification;
  thread_apex       : OPTIONAL pitch_apex;
```

```
  thread_timing     : OPTIONAL thread_timer;
WHERE
  cylindrical(installation_area) OR
  conical(installation_area);
END_ENTITY;

TYPE
hands = ENUMERATION OF (left_hand, right_hand);
thread_fit_classes = ENUMERATION OF (one_a, two_a, three_a,
                                     one_b, two_b, three_b);
thread_forms = ENUMERATION OF (sharpv, unified, sixty_degree_stub, acme,
                               square, stub_acme, buttress, knuckle,
                               brit_std);
END_TYPE;

FUNCTION cylindrical(area:dimensionality_2_shape_element):LOGICAL;
-- determines if an area is cylindrical
END_FUNCTION;

FUNCTION conical(area:dimensionality_2_shape_element):LOGICAL;
-- determines if an area is conical
END_FUNCTION;
```

Entity Name: IMPLICIT FEATURE BOUND

Entity Number: 029

A bound or limit to the extent of an implicit form feature. These are used where implicit feature representations define volumes added to or subtracted from pre-existing material (passages, depressions, protrusions).

They indicate where the pre-existing material's volume and the volume specified by the implicit feature representation intersect and thus serve to reduce the material to be added or subtracted to "realize" the feature. For example, consider a thru hole modeled as an implicit passage. The implicit representation of the hole might be by centerline and diameter, effectively defining an infinite cylindrical volume. IMPLICIT FEATURE BOUNDs/029 could be used to specify entry and exit faces, thereby specifying the limits of the volume to be removed.

Primary Key Attributes

IMPLICIT FEATURE BOUND ID
    An arbitrary label.

Business Rules

- An IMPLICIT FEATURE BOUND/029 consists of 1 or many FEATURE BOUND ELEMENTs/030.
- An IMPLICIT FEATURE BOUND/029 specifies the bound in 0, 1, or many PASSAGE BOUNDs/402.
- An IMPLICIT FEATURE BOUND/029 specifies the bound in 0, 1, or many PROTRUSION BOUNDs/404.
- An IMPLICIT FEATURE BOUND/029 specifies the bound in 0, 1, or many DEPRESSION BOUNDs/403.
- An IMPLICIT FEATURE BOUND/029 specifies the bound in 0, 1, or many PASSAGE INTERMEDIATE BOUNDs/186.
- An IMPLICIT FEATURE BOUND/029 specifies the bound in 0, 1, or many DEPRESSION INTERMEDIATE BOUNDs/188.

EXPRESS Declaration

```
  ENTITY implicit_feature_bound;
   elements : SET [1:#] OF dimensionality_2_shape_element;
 END_ENTITY;
```

Entity Name: FEATURE BOUND ELEMENT

Entity Number: 030

One of the elements comprising an IMPLICIT FEATURE BOUND/029.

Primary Key Attributes

IMPLICIT FEATURE BOUND ID (FK)
Identifies the IMPLICIT FEATURE BOUND/029.

SHAPE ID (FK)
Together with the following attribute, identifies the DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 that is a component of the IMPLICIT FEATURE BOUND/029.

SHAPE ELEMENT ID (FK)

Business Rules

- An IMPLICIT FEATURE BOUND/029 consists of 1 or many FEATURE BOUND ELEMENTs/030.
- A DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 is used as 0, 1, or many FEATURE BOUND ELEMENTs/030.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under IMPLICIT FEATURE BOUND/029.)

Entity Name: **IMPLICIT EDGE FLAT**

Entity Number: **036**

An implicit representation of a ruled surface blend of two surface areas of a shape. The blend surface is set back a constant distance along one of the areas from the intersection of the two surface areas and makes a constant angle with that area. Most often, the blend surface is planar (blending planar surfaces) or conical (blending rotational surfaces).

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

ANGLE.DIM ID (FK)
The constant angular dimension between the flat and the blended surface used for nominal shape definition.

SETBACK.DIM ID (FK)
The constant distance between

(a) The intersection of the two blended surfaces, and
(b) The intersection of the flat and the blended surface used for nominal shape definition.

Business Rules

- An IMPLICIT EDGE FLAT/036 is a type of IMPLICIT EDGE BLEND/308.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is angle of 0, 1, or many IMPLICIT EDGE FLATs/036.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is setback of 0, 1, or many IMPLICIT EDGE FLATs/036.
- ANGLE and SETBACK are measured relative to one of the two elements blended by the chamfer; the following rules tell which, depending on the context of the feature:
  (a) Related to EDGE-BLENDED INTERSECTION/359: ANGLE and SETBACK are relative to the first of the two DIMENSIONALITY-2 SHAPE ELEMENTs/INT-7 associated with the EDGE-BLENDED INTERSECTION/359:

(b) Related to PASSAGE BLEND/152 or DEPRESSION BLEND/151: ANGLE and SETBACK are relative to the boundary element of the passage or depression; i.e., the pre-existing shape.

(c) For any FEATURE SWEEP PROFILE/136 not utilizing GENERAL PROFILE/162: Consider the segments of the profile to be clockwise ordered when looking in the direction of sweep (the view of the illustrations herein). ANGLE and SETBACK are relative to the first of the two profile segments blended.

(d) Related to PROFILE SEQUENCER/165; i.e., a blend between segments of a GENERAL PROFILE/162: ANGLE and SETBACK are relative to the predecessor element in PROFILE SEQUENCER/165.

(e) Used as IN/OUT FEATURE SWEEP WALL/END BLEND/129, ALONG FEATURE SWEEP FLAT END BLEND/161, or AXISYMMETRIC FEATURE SWEEP WALL/END BLEND/131: ANGLE and SETBACK are relative to the swept walls of the feature edge.

EXPRESS Declaration

```
ENTITY implicit_edge_flat
SUBTYPE OF (implicit_edge_blend);
  angle   : independent_angle_size_parameter;
  setback : independent_size_parameter;
END_ENTITY;
```

Entity Name: **IMPLICIT EDGE ROUND**

Entity Number: **037**

A blend of two surface areas of a shape, having a circular cross section of constant radius. The blend surface is tangent to both of the adjacent surface areas. Most often, the blend surface is cylindrical (blending two planar surfaces) or toroidal (blending rotational surfaces).

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

DIM ID (FK)
The dimension (radius, preferably, or diameter) of the round.

Business Rules

- An IMPLICIT EDGE ROUND/037 is a type of IMPLICIT EDGE BLEND/308
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is dimension of 0, 1, or many IMPLICIT EDGE ROUNDs/037.
- An IMPLICIT EDGE ROUND/037 is used as 0, 1, or many PROTRUSION BLENDs/153.
- An IMPLICIT EDGE ROUND/037 is used as 0, 1, or many AXISYMMETRIC FEATURE SWEEP CONICAL END TIP BLENDs/135.

EXPRESS Declaration

```
ENTITY implicit_edge_round
SUBTYPE OF (implicit_edge_blend);
  round_dim : independent_size_parameter;
END_ENTITY;
```

Entity Name: IMPLICIT FORM FEATURE PATTERN

Entity Number: 049

A representation of a FORM FEATURE/001 as an arrangement of identical (except for location/orientation) form features according to some mathematical logic. The pattern is represented by the identification of one of its member features (the "base" feature) and the logic for arranging "copies" of the base feature.

Primary Key Attributes

SHAPE ID (FK)
With the following attribute, identifies the FORM FEATURE/001 represented as a pattern.

SHAPE ELEMENT ID (FK)

Other Attributes

BASE.SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT FORM FEATURE/002 that represents the "base" feature of the pattern.

BASE.SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

PATTERN TYPE (Discriminator)

Business Rules

- A FORM FEATURE/001 is represented by 0 or 1 IMPLICIT FORM FEATURE PATTERNs/049.
- An IMPLICIT FORM FEATURE/002 is base feature of 0, 1, or many IMPLICIT FORM FEATURE PATTERNs/049.
- An IMPLICIT FORM FEATURE PATTERN/049 may be either an IMPLICIT CIRCULAR FORM FEATURE PATTERN/053 or an IMPLICIT ARRAY FORM FEATURE PATTERN/054.

EXPRESS Declaration

```
ENTITY implicit_form_feature_pattern
SUPERTYPE OF (implicit_array_form_feature_pattern OR
              implicit_circular_form_feature_pattern);
  base_form_feature : implicit_form_feature;
END_ENTITY;
```

Entity Name: IMPLICIT CIRCULAR FORM FEATURE

Entity Number: 053

An IMPLICIT FORM FEATURE PATTERN/049 whose component features are arranged in a circular arc pattern; i.e., are equally spaced about a centerline.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

Other Attributes

GEOMETRIC ELEMENT ID (FK)
: Identifies the centerline of the pattern.

NUMBER OF MEMBERS
: The nominal (i.e., includes omitted members) number of features in the pattern.

DIM ID (FK)
: The angular spacing between members of the pattern.

Business Rules

- An IMPLICIT CIRCULAR FORM FEATURE PATTERN/053 is a type of IMPLICIT FORM FEATURE PATTERN/049.
- A LINE/GEO-20 is centerline of 0, 1, or many IMPLICIT CIRCULAR FORM FEATURE PATTERNs/053.
- An IMPLICIT CIRCULAR FORM FEATURE PATTERN/053 has members omitted by 0, 1, or many CIRCULAR PATTERN OMISSIONs/073.
- An IMPLICIT CIRCULAR FORM FEATURE PATTERN/053 has 0, 1, or many CIRCULAR PATTERN OFFSET MEMBERs/074.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is spacing of 0, 1, or many IMPLICIT CIRCULAR FORM FEATURE PATTERNs/053.
- The data "constructs" the pattern as follows. The construction provides a way of referring to individual members of the pattern.
  - Member #1 is the base feature. (See IMPLICIT FORM FEATURE PATTERN/049.)
  - Member #I is a "copy" of the base feature, rotated (I-1)*SPACING about the pattern centerline in the counterclock- wise direction, as viewed when looking in the direction of the centerline.
- NUMBER OF MEMBERS * SPACING $\leq$ 360.

NOTES

- This entity can handle full or partial circle patterns. For a full circle pattern, NUMBER OF MEMBERS * SPACING = 360.

EXPRESS Declaration

```
ENTITY implicit_circular_form_feature_pattern
SUBTYPE OF (implicit_form_feature_pattern);
  centerline        : line;
  number_of_members : INTEGER;
  angular_spacing   : independent_size_parameter;
  omissions         : SET [0:#] OF circular_pattern_omission;
  offsets           : SET [0:#] OF circular_pattern_offset_member;
WHERE
  (number_of_members * angular_spacing.dimension) < 360;
  number_of_members - SIZEOF(omissions) >= 2;
END_ENTITY;

RULE verify_circular_pattern_omissions_and_offsets FOR
  (implicit_circular_form_feature_pattern,
```

```
    circular_pattern_omission,
    circular_pattern_offset_member);
 LOCAL
   i,j:INTEGER;
   omit:circular_pattern_omission;
   offset:circular_pattern_offset_member;
 END_LOCAL;
   REPEAT i := 1 TO SIZEOF(implicit_circular_form_feature_pattern.omissions);
     omit := POSITION(implicit_circular_form_feature_pattern.omissions,i);
     IF (omit.omitted_member_number <= 1) THEN VIOLATION;
     END_IF;
     IF (omit.omitted_member_number>number_of_members) THEN VIOLATION;
     END_IF;
     REPEAT j := 1 TO SIZEOF(implicit_circular_form_feature_pattern.offsets);
       offset := POSITION(implicit_circular_form_feature_pattern.offsets,j);
       IF (offset.offset_member_number >
           implicit_circular_form_feature_pattern.number_of_members)
             THEN VIOLATION;
       END_IF;
       IF (offset.offset_member_number=omit.omitted_member_number)
         THEN VIOLATION;
       END_IF;
     END_REPEAT;
  END_REPEAT;
 END_RULE;
```

Entity Name: **IMPLICIT ARRAY FORM FEATURE**

Entity Number: **054**

An implicit pattern feature whose component features are arranged in a pattern of rows and columns.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)

Other Attributes

NUMBER OF ROWS
The number of rows in the pattern.

NUMBER OF COLUMNS
The number of columns in the pattern.
ARRAY PATTERN TYPE (Discriminator)

Business Rules

- An IMPLICIT ARRAY FORM FEATURE PATTERN/054 is a type of IMPLICIT FORM FEATURE PATTERN/049.
- An IMPLICIT ARRAY FORM FEATURE PATTERN/054 may be either a PARAMETRIC EQUAL SPACING ARRAY PATTERN/055 or a PARALLEL EQUAL SPACING ARRAY PATTERN/056.
- An IMPLICIT ARRAY FORM FEATURE PATTERN/054 has members omitted by 0, 1, or many ARRAY PATTERN OMISSIONs/306.

EXPRESS Declaration

```
ENTITY implicit_array_form_feature_pattern
SUPERTYPE OF (parallel_equal_spacing_array_pattern OR
              parametric_equal_spacing_array_pattern)
SUBTYPE OF (implicit_form_feature_pattern);
  number_of_rows    : INTEGER;
  number_of_columns : INTEGER;
  omissions         : SET [0:#] OF array_pattern_omission;
WHERE
  number_of_rows > 0;
  number_of_columns > 0;
  number_of_rows + number_of_columns > 2;
  SIZEOF(omissions) < (number_of_rows)*(number_of_columns) - 1;
END_ENTITY;
```

Entity Name: **PARAMETRIC EQUAL SPACING ARRAY**

Entity Number: **055**

An array of features arranged on a parametrically represented surface in such a way that the spacing between adjacent features on the same row is given by a U-delta and the spacing between adjacent features on the same column is given by a V-delta.

The location and orientation of each member feature with respect to the surface is the same as that of the base feature. This is made precise in Business Rule 4.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
The surface along which the pattern members are laid out.

UDELTA
Intrarow spacing of pattern members. (If the (i, j) member is at $f(u, v)$, the $(i+1, j)$ member is at $f(u + UDELTA, v)$.

VDELTA
Intracolumn spacing of pattern members. (If the (i, j) member is at $f(u, v)$, the $(i, j+1)$ member is at $f(u, v + VDELTA)$.

Business Rules

- A PARAMETRIC EQUAL SPACING ARRAY PATTERN/055 is a type of IMPLICIT ARRAY FORM FEATURE PATTERN/054.
- A SURFACE/GEO-7 is layout reference for 0, 1, or many PARAMETRIC EQUAL SPACING ARRAY PATTERNs/055.
- A PARAMETRIC EQUAL SPACING ARRAY PATTERN/055 has 0, 1, or many PARAMETRIC ARRAY PATTERN OFFSET MEMBERs/307.
- The features in the pattern are located as follows:
  - Denote the layout surface as $f(u, v)$.
  - The base feature of the pattern must have an LCS; i.e., be a type of IMPLICIT FORM FEATURE/002 with an associated AXIS PLACEMENT/GEO-4.
  - The base feature must be known to correspond to a point (u_1, v_1) in parameter space.
  - Let A_{11} be the AXIS PLACEMENT/GEO-4 defined by $f(u_1, v_1)$, the partial derivative of f with respect to u, and the partial of f with respect to v.
  - Let T be the transformation that takes A_{11} into the LCS of the base feature; i.e., $A_{11} * T = LCS_{11}$.
  - For the i, j member of the pattern, let A_{ij} be the AXIS PLACEMENT/GEO-4 defined by $f(u_1 + (i-1) * UDELTA, v_1 + (j-1) * VDELTA)$, the partial of f with respect to u at that point, and the partial of f with respect to v at that point.
  - Then the location and orientation of the i, j feature of the pattern are established by positioning its LCS at $A_{ij} * T$; i.e., $LCS_{ij} = A_{ij} * T$

EXPRESS Declaration

```
ENTITY parametric_equal_spacing_array_pattern
SUBTYPE OF (implicit_array_form_feature_pattern);
  layout_reference : surface;
  udelta           : REAL;
  vdelta           : REAL;
  offsets          : SET [0:#] OF parametric_array_pattern_offset_member;
END_ENTITY;

RULE verify_parametric_array_pattern_omissions_and_offsets FOR
  (parametric_equal_spacing_array_pattern,
   array_pattern_omission,
   parametric_array_pattern_offset_member);
LOCAL
  i,j:INTEGER;
  omit:array_pattern_omission;
  offset:parametric_array_pattern_offset_member;
END_LOCAL;
  REPEAT i := 1 TO SIZEOF(parametric_equal_spacing_array_pattern.omissions);
    omit := POSITION(parametric_equal_spacing_array_pattern.omissions,i);
    IF (omit.row > parametric_equal_spacing_array_pattern.member_of_rows)
      THEN VIOLATION;
    END_IF;
    IF (omit.column >
      parametric_equal_spacing_array_pattern.member_of_columns)
        THEN VIOLATION;
    END_IF;
    REPEAT j := 1 TO SIZEOF(parametric_equal_spacing_array_pattern.offsets);
      offset := POSITION(parametric_equal_spacing_array_pattern.offsets,j);
      IF (offset.row >
          parametric_equal_spacing_array_pattern.member_of_rows)
            THEN VIOLATION;
      END_IF;
      IF (offset.columns >
          parametric_equal_spacing_array_pattern.member_of_columns)
            THEN VIOLATION;
      END_IF;
      IF ((offset.row=omit.row) AND (offset.column=omit.column))
        THEN VIOLATION;
      END_IF;
    END_REPEAT;
 END_REPEAT;
END_RULE;
```

Entity Name: **PARALLEL EQUAL SPACING ARRAY**

Entity Number: **056**

An array of features which have the same orientation and are equally spaced in the object space.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)

Other Attributes

ROW.GEOMETRIC ENTITY ID (FK)
The translation which takes the i^{th} member of a row into the $(i+1)^{st}$ member of that row.

COLUMN.GEOMETRIC ENTITY ID (FK)
The translation which takes the j^{th} member of a column into the $(j+1)^{st}$ member of that column.

ROW.DIM ID (FK)
The derivable distance between adjacent members in the same row (the length of the intra-row vector).

COLUMN.DIM ID (FK)
The derivable distance between adjacent members in the same column (the length of the intra-column vector).

Business Rules

- A PARALLEL EQUAL SPACING ARRAY PATTERN/056 is a type of IMPLICIT ARRAY FORM FEATURE PATTERN/054.
- A VECTOR WITH MAGNITUDE/GEO-17 gives row translation of 0, 1, or many PARALLEL EQUAL SPACING ARRAY PATTERNs/056.
- A VECTOR WITH MAGNITUDE/GEO-17 gives column translation of 0, 1, or many PARALLEL EQUAL SPACING ARRAY PATTERNs/056.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is the row spacing dimension of 0, 1, or many PARALLEL EQUAL SPACING ARRAY PATTERNs/056.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is the column spacing dimension of 0, 1, or many PARALLEL EQUAL SPACING ARRAY PATTERNs/056.
- A PARALLEL EQUAL SPACING ARRAY PATTERN/056 has 0, 1, or many PARALLEL ARRAY PATTERN OFFSET MEMBERs/305.

EXPRESS Declaration

```
ENTITY parallel_equal_spacing_array_pattern
 SUBTYPE OF (implicit_array_form_feature_pattern);
   row_translation    : vector_with_magnitude;
   column_translation : vector_with_magnitude;
   row_spacing        : derivable_size_parameter;
   column_spacing     : derivable_size_parameter;
   offsets            : SET [0:#] OF
                          parallel_array_pattern_offset_member;
END_ENTITY;

RULE verify_parallel_array_pattern_omissions_and_offsets FOR
  (parallel_equal_spacing_array_pattern,
   array_pattern_omission,
   parallel_array_pattern_offset_member);
LOCAL
  i,j:INTEGER;
  omit:array_pattern_omission;
  offset:parallel_array_pattern_offset_member;
END_LOCAL;
  REPEAT i := 1 TO SIZEOF(parallel_equal_spacing_array_pattern.omissions);
    omit := POSITION(parallel_equal_spacing_array_pattern.omissions,i);
    IF (omit.row <= 1) THEN VIOLATION;
    END_IF;
    IF (omit.column <= 1) THEN VIOLATION;
    END_IF;
    IF (omit.row>parallel_equal_spacing_array_pattern.number_of_rows)
      THEN VIOLATION;
    END_IF;
    IF (omit.column>parallel_equal_spacing_array_pattern.number_of_columns)
      THEN VIOLATION;
    END_IF;
    REPEAT j := 1 TO SIZEOF(parallel_equal_spacing_array_pattern.offsets);
      offset := POSITION(parallel_equal_spacing_array_pattern.offsets,j);
      IF (offset.row >
          parallel_equal_spacing_array_pattern.number_of_rows)
            THEN VIOLATION;
      END_IF;
      IF (offset.columns >
          parallel_equal_spacing_array_pattern.number_of_columns)
            THEN VIOLATION;
      END_IF;
      IF ((offset.row=omit.row) AND (offset.column=omit.column))
```

```
      THEN VIOLATION;
    END_IF;
  END_REPEAT;
END_REPEAT;
END_RULE;
```

Entity Name: **IMPLICIT DEFORMATION**

Entity Number: **064**

An implicit representation of a feature characterized by stretching or bending material.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

IMPLICIT DEFORMATION TYPE (Discriminator)

Business Rules

- An IMPLICIT DEFORMATION/064 is a type of IMPLICIT FORM FEATURE/002.
- An IMPLICIT DEFORMATION/064 may be either an IMPLICIT EMBOSS/081, IMPLICIT TWIST/082, IMPLICIT PARTIAL CUTOUT/083, IMPLICIT BEND/084, or IMPLICIT TUBE DEFORMATION/086.

EXPRESS Declaration

```
ENTITY implicit_deformation
SUPERTYPE OF (implicit_bend OR
              implicit_emboss OR
              implicit_partial_cutout OR
              implicit_tube_deformation OR
              implicit_twist)
SUBTYPE OF (implicit_form_feature);
END_ENTITY;
```

Entity Name: REPLICATE FORM FEATURE

Entity Number: 067

A representation of a form feature as a "copy" of another form feature, but in a different location and perhaps reflected (mirrored). The copied feature necessarily has an implicit representation. The replicate can be "realized" by applying a rigid transformation to its local coordinate system, if it has one, and to any definitional geometric data (points, curves, surfaces, etc.). Features dependent on non-transformable data (e.g., area features, transitions) cannot be replicated.

Primary Key Attributes

REPRESENTED.SHAPE ID (FK)
With the following attribute, identifies the FORM FEATURE/001 represented by replication.

REPRESENTED.SHAPE ELEMENT ID (FK)

Other Attributes

COPIED.SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT FORM FEATURE/002 of which this feature is represented as a replicate.

COPIED.SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

GEOMETRIC ENTITY ID
Identifies the AXIS PLACEMENT/GEO-4 that accomplishes the rigid transformation.

Business Rules

- A FORM FEATURE is represented as 0 or 1 REPLICATE FORM FEATUREs/067.
- An IMPLICIT FORM FEATURE/002 is copied by 0, 1, or many REPLICATE FORM FEATUREs/067.
- An AXIS PLACEMENT/GEO-4 gives the location of 0, 1, or many REPLICATE FORM FEATUREs/067.
- A REPLICATE FORM FEATURE/067 is mirrored by 0 or 1 FORM FEATURE REFLECTIONs/302.

EXPRESS Declaration

```
ENTITY replicate_form_feature;
 copied_feature : implicit_form_feature;
 location       : axis_placement;
 mirror         : OPTIONAL coordinate_enumeration;
END_ENTITY;

TYPE
coordinate_enumeration = ENUMERATION OF (x_coordinate,
                                         y_coordinate,
                                         z_coordinate);
END_TYPE;
```

Entity Name: CIRCULAR PATTERN OMISSION

Entity Number: 073

An indication that one of the members of an IMPLICIT CIRCULAR FORM FEATURE PATTERN/053 is absent.

Primary Key Attributes

SHAPE ID (FK)
: With the following attribute, identifies the IMPLICIT CIRCULAR FORM FEATURE PATTERN/053 from which this member is omitted.

SHAPE ELEMENT ID (FK)

OMITTED MEMBER NUMBER
: The sequence number, per the member numbering scheme given under IMPLICIT CIRCULAR FORM FEATURE PATTERN/053, of the omitted pattern member.

Business Rules

- An IMPLICIT CIRCULAR FORM FEATURE PATTERN/053 has members omitted by 0, 1, or many CIRCULAR PATTERN OMISSIONs/073.
- OMITTED MEMBER NUMBER must be greater than 1 and must not exceed the NUMBER OF MEMBERS attribute of the parent IMPLICIT CIRCULAR FORM FEATURE PATTERN/053.

EXPRESS Declaration

```
ENTITY circular_pattern_omission;
  omitted_member_number : INTEGER;
WHERE
  omitted_member_number > 1;
END_ENTITY;

RULE omitted_member_number_constraint FOR
  (implicit_circular_form_feature_pattern,circular_pattern_omission);
  IF (circular_pattern_omission.omitted_member_number >
    implicit_circular_form_feature_pattern.number_of_members)
      THEN VIOLATION;
  END_IF;
END_RULE;
```

Entity Name: CIRCULAR PATTERN OFFSET MEMBER

Entity Number: 074

An indication that a member of a circular pattern feature is at a location other than that indicated by the pattern rule.

Primary Key Attributes

SHAPE ID (FK)
With the following attribute, identifies the IMPLICIT CIRCULAR FORM FEATURE PATTERN/053 from which this member is offset.

SHAPE ELEMENT ID (FK)

OFFSET MEMBER NUMBER
The sequence number, per the member numbering scheme given under IMPLICIT CIRCULAR FORM FEATURE PATTERN/053, of the offset pattern member.

Other Attributes

OFFSET ANGLE
The angle from the rule-indicated position to the actual position. This is a signed angle, positive or negative according as the delta is in the direction of the pattern or the other direction.

Business Rules

- An IMPLICIT CIRCULAR FORM FEATURE PATTERN/053 has 0, 1, or many CIRCULAR PATTERN OFFSET MEMBERs/074.

NOTES

- As the attributes imply, offsetting is a matter of rotation about the pattern centerline. No other type of relocation is covered.

EXPRESS Declaration

```
ENTITY circular_pattern_offset_member;
  offset_member_number : INTEGER;
  offset_angle         : REAL;
WHERE
  offset_member_number > 1;
  offset_angle <> 0;
END_ENTITY;

RULE offset_member_number_constraint FOR
  (implicit_circular_form_feature_pattern,circular_pattern_offset_member);
  IF (circular_pattern_offset_member.offset_member_number >
    implicit_circular_form_feature_pattern.number_of_members)
      THEN VIOLATION;
  END_IF;
END_RULE;
```

Entity Name: IMPLICIT EMBOSS

Entity Number: 081

An imprinting (rib or recess) that is totally surrounded by part material. Embossing is distinguished from bends and flanges by the fact that embossing is totally surrounded by the part material.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

IMPLICIT EMBOSS TYPE (Discriminator)

Business Rules

- An IMPLICIT EMBOSS/081 is a type of IMPLICIT DEFORMATION/064.
- An IMPLICIT EMBOSS/081 may be either an IMPLICIT V-BEAD/087, IMPLICIT ROUND BEAD/088, IMPLICIT CORNER RIB/089, or IMPLICIT SPHERICAL EMBOSS/400.

EXPRESS Declaration

```
ENTITY implicit_emboss
SUPERTYPE OF (implicit_corner_rib OR
              implicit_round_bead OR
              implicit_v_bead OR
              implicit_spherical_emboss)
SUBTYPE OF (implicit_deformation);
END_ENTITY;
```

Entity Name: IMPLICIT TWIST

Entity Number: 082

A twist in the material about a centerline.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the AXIS PLACEMENT/GEO-4 which locates the feature.

LENGTH.DIM ID (FK)
Length dimension (half or full) of twist region.

ANGLE.DIM ID (FK)
Angle dimension (half or full) of twist.

Business Rules

- An IMPLICIT TWIST/082 is a type of IMPLICIT DEFORMATION/064.
- An IMPLICIT TWIST/082 is applied to a sheet- or plate-like portion of a shape.
- An AXIS PLACEMENT/GEO-4 locates 0, 1, or many IMPLICIT TWISTs/082. This is done as follows. The Z-axis must coincide with the twist axis. The origin must lie on the center of the twist; i.e., the twist region is the z-interval $(-LENGTH/2, LENGTH/2)$ (taking LENGTH to be full length). The XY-plane must be oriented so that the twist takes $+x$ into $+y$ in the positive z halfspace and $+x$ into $-y$ in the negative z halfspace.
- The post-twist shape is calculated as follows. Values are given in local coordinates, but the transformations apply to the entire shape containing the twist. The formulas assume that angular displacement varies linearly with z in the interval $(-LENGTH/2, LENGTH/2)$, that the angular displacement at $z \geq LENGTH/2$ is $ANGLE/2$, and that the angular displacement at $z <= LENGTH/2$ is $-ANGLE/2$.
  - (a) $z' = z$
  - (b) The rotation angle θ at z is
    $2z/LENGTH \times ANGLE/2$ for z in $(-LENGTH/2, LENGTH/2)$
    $ANGLE/2$ for $z > LENGTH/2$
    $-ANGLE/2$ for $z < -LENGTH/2$
  - (c) $x' = x \times \cos(\theta) - y \times \sin(\theta)$
  - (d) $y' = x \times \sin(\theta) + y \times \cos(\theta)$
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is length dimension of 0, 1, or many IMPLICIT TWISTs/082.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is angle dimension of 0, 1, or many IMPLICIT TWISTs/082.

EXPRESS Declaration

```
ENTITY implicit_twist
SUBTYPE OF (implicit_deformation);
  length   : independent_size_parameter;
  angle    : independent_angle_size_parameter;
  location : axis_placement;
END_ENTITY;
```

Entity Name: IMPLICIT PARTIAL CUTOUT

Entity Number: 083

An IMPLICIT DEFORMATION/083 which involves shearing as well as deformation.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

IMPLICIT PARTIAL CUTOUT TYPE (Discriminator)

Business Rules

- An IMPLICIT PARTIAL CUTOUT/083 is a type of IMPLICIT DEFORMATION/064
- An IMPLICIT PARTIAL CUTOUT/083 may be either an IMPLICIT LOUVER/090, IMPLICIT CIRCULAR KNOCKOUT/091, or an IMPLICIT TAB/401.

EXPRESS Declaration

```
ENTITY implicit_partial_cutout
SUPERTYPE OF (implicit_louver OR
              implicit_circular_knockout OR
              implicit_tab)
SUBTYPE OF (implicit_deformation);
END_ENTITY;
```

Entity Name: **IMPLICIT BEND**

Entity Number: **084**

An IMPLICIT DEFORMATION/064 in which deforming occurs along a curve (bend line) and is characterized by radius and angle, which may be constant or vary.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

IMPLICIT BEND TYPE (Discriminator)

Business Rules

- An IMPLICIT BEND/084 is a type of IMPLICIT DEFORMATION/064.
- An IMPLICIT BEND/084 may be either an IMPLICIT GENERAL BEND/092, IMPLICIT CUTOUT FLANGE/094, or IMPLICIT STRAIGHT BEND/095.
- An IMPLICIT BEND/084 is applied to a sheet- or plate-like underlying shape.

EXPRESS Declaration

```
ENTITY implicit_bend
SUPERTYPE OF (implicit_general_bend OR
              implicit_cutout_flange OR
              implicit_straight_bend)
SUBTYPE OF (implicit_deformation);
END_ENTITY;
```

Entity Name: IMPLICIT TUBE DEFORMATION

Entity Number: 086

An implicit representation of a deformation of a tube-like shape.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

IMPLICIT TUBE DEFORMATION TYPE (Discriminator)

Business Rules

- IMPLICIT TUBE DEFORMATION/086 is a type of IMPLICIT DEFORMATION/064.
- An IMPLICIT TUBE DEFORMATION/086 may be either an IMPLICIT TUBE BEND/093, an IMPLICIT TUBE FLARE/098, an IMPLICIT TUBE NECK/099, an IMPLICIT TUBE FLATTENING/100, or an IMPLICIT TUBE ROLL/101.
- The shape with which an IMPLICIT TUBE DEFORMATION/086 is associated must be tubular, except that an IMPLICIT TUBE BEND/093 may be associated with a solid round bar.

EXPRESS Declaration

```
ENTITY implicit_tube_deformation
SUPERTYPE OF (implicit_tube_bend OR
              implicit_tube_flare OR
              implicit_tube_neck OR
              implicit_tube_flattening OR
              implicit_tube_roll)
SUBTYPE OF (implicit_deformation);
END_ENTITY;
```

Entity Name: **IMPLICIT V-BEAD**

Entity Number: **087**

A deformation that has the cross sectional characteristics of a "V".

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the locating BOUNDED CURVE/GEO-22 of the feature.

ANGLE.DIM ID (FK)
The angle dimension (full or semi-angle) of the "V".

HEIGHT.DIM ID (FK)
The height dimension of the bead. The dimension from the undeformed base to the top of the apex round is preferred, though half that dimension is also possible.

APEX ROUND.DIM ID (FK)
The dimension (radius, preferably, or diameter) of the round at the apex of the "V".

BASE BEND.DIM ID (FK)
The dimension (radius, preferably, or diameter) of the round at the base of the bead.

Business Rules

- An IMPLICIT V-BEAD/087 is a type of IMPLICIT EMBOSS/081.

- A BOUNDED CURVE/GEO-22 locates 0, 1, or many IMPLICIT V-BEADs/087. The curve lies on the surface that is displaced to become the inner (concave) portion of the bead, so that points on the curve are displaced to the inner apex of the bead.
- Full bead profile begins and ends at the extrema of the axis BOUNDED CURVE/GEO-22.
- The value of the angle dimension is less than 180 (90 if semi-angle).
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is angle dimension of 0, 1, or many IMPLICIT V-BEADs/087.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is height dimension of 0, 1, or many IMPLICIT V-BEADs/087.
- A BEND DIMENSION/429 is apex round dimension of 0, 1, or many IMPLICIT V-BEADs/087.
- A BEND DIMENSION/429 is base bend dimension of 0, 1, or many IMPLICIT V-BEADs/087.

EXPRESS Declaration

```
ENTITY implicit_v_bead
SUBTYPE OF (implicit_emboss);
  location    : bounded_curve;
  angle       : independent_angle_size_parameter;
  height      : independent_size_parameter;
  width       : independent_size_parameter;
  apex_round  : bend_dimension;
  base_bend   : bend_dimension;
WHERE
  angle.dimension < 180;
END_ENTITY;
```

Entity Name: **IMPLICIT ROUND BEAD**

Entity Number: **088**

An emboss that has the cross sectional characteristics of an arc of a circle.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the locating BOUNDED CURVE/GEO-22 of the feature.

HEIGHT.DIM ID (FK)
The height dimension of the bead. (Though half measurement is possible. full height from undeformed base to top is preferred.)

SIZE.DIM ID (FK)
The size (diameter or radius) of the circular bead cross-section.

BASE BEND.DIM ID (FK)
The dimension (radius, preferably, or diameter) of the bend between the bead and its neighboring, undeformed material.

Business Rules

- An IMPLICIT ROUND BEAD/088 is a type of IMPLICIT EMBOSS/081.
- A BOUNDED CURVE/GEO-22 locates 0, 1, or many IMPLICIT ROUND BEADs/088. The curve lies on the surface that is displaced to become the inner (concave) portion of the bead, so that points on the curve are displaced to the inner apex of the bead.
- Full bead profile begins and ends at the extrema of the axis BOUNDED CURVE/GEO-22.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is height dimension of 0, 1, or many IMPLICIT ROUND BEADs/088.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is size dimension of 0, 1, or many IMPLICIT ROUND BEADs/088.
- A BEND DIMENSION/429 is base bend dimension of 0, 1, or many IMPLICIT ROUND BEADs/088.
- Full height $\leq$ full size/2

EXPRESS Declaration

```
ENTITY implicit_round_bead
SUBTYPE OF (implicit_emboss);
  location     : bounded_curve;
  height       : independent_size_parameter;
  bead_size    : independent_size_parameter;
  base_bend    : bend_dimension;
WHERE
  height.dimension <= bead_size.dimension/2;
END_ENTITY;
```

Entity Name: IMPLICIT CORNER RIB

Entity Number: 089

A stiffening rib that is created by inverting the material, creating a protruding region at a location along a bend. The feature has rectangular surface shape and is planar except for bends at its intersection with the two stiffened surfaces.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the AXIS PLACEMENT/GEO-4 which locates the feature.

WIDTH.DIM ID (FK)
The width dimension of the rib. (While half width may be given, full width is preferred.)

HEIGHT1.DIM ID (FK)
The derivable height of the rib along the first (per the rule below for placement of the feature's local coordinate system) leg of the stiffened bend. (It is possible to give half-height, but full value is recommended.)

HEIGHT2.DIM ID (FK)
The derivable height of the rib along the second (per the rule below for placement of the feature's local coordinate system) leg of the stiffened bend. (It is possible to give half-height, but full value is recommended.)

LENGTH.DIM ID (FK)
The derivable length of the rib. (It is possible to give semi-length but full value is recommended.)

BEND1.DIM ID (FK)
The dimension (radius, preferably, or diameter) of the bend between the feature and the first (per the rule below for placement of the feature's local coordinate system) leg of the stiffened bend.

BEND2.DIM ID (FK)
The bend dimension (radius, preferably, or diameter) between the feature and the second (per the rule below for placement of the feature's local coordinate system) leg of the stiffened bend.

Business Rules

- An IMPLICIT CORNER RIB/089 is a type of IMPLICIT EMBOSS/081.
- An AXIS PLACEMENT/GEO-4 locates 0, 1, or many IMPLICIT CORNER RIBs/089. This is done as follows. The Z-axis is the length centerline of the outer rectangular surface of the rib. The origin is placed at the intersection of the z-axis with the inner surface of one of the legs of the stiffened bend (the "first leg"), with the X-axis lying in that surface. Thus, ignoring bend radii of the feature, the outer surface of the rib is a sweep of the x-interval (-WIDTH/2, WIDTH/2) in the $+z$ direction.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is width dimension of 0, 1, or many IMPLICIT CORNER RIBs/089.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is height1 dimension of 0, 1, or many IMPLICIT CORNER RIBs/089.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is height2 dimension of 0, 1, or many IMPLICIT CORNER RIBs/089.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is length dimension of 0, 1, or many IMPLICIT CORNER RIBs/089.
- A BEND DIMENSION/429 is bend1 dimension of 0, 1, or many IMPLICIT CORNER RIBs/089.
- A BEND DIMENSION/429 is bend2 dimension of 0, 1, or many IMPLICIT CORNER RIBs/089.

EXPRESS Declaration

```
ENTITY implicit_corner_rib
SUBTYPE OF (implicit_emboss);
  stiffens : form_feature;
  location : axis_placement;
  width    : independent_size_parameter;
  height1  : derivable_size_parameter;
  height2  : derivable_size_parameter;
  length   : derivable_size_parameter;
  bend1    : bend_dimension;
  bend2    : bend_dimension;
WHERE
  height1.dimension**2 + height2.dimension**2 = length.dimension**2;
END_ENTITY;
```

Entity Name: IMPLICIT LOUVER

Entity Number: 090

An IMPLICIT PARTIAL CUTOUT/083 for which the shear is a straight line and the deformation to the material causes an opening to be created. Deformation at the shear line is greater than material thickness.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other_Attributes

GEOMETRIC ENTITY ID (FK)
: Identifies the AXIS PLACEMENT/GEO-4 which locates the feature.

LENGTH.DIM ID (FK)
: Length dimension of louver.

WIDTH.DIM ID (FK)
: Width dimension of louver.

HEIGHT.DIM ID (FK)
: Height of louver.

BASE BEND.DIM ID (FK)
: Dimension (radius, preferably, or diameter) of bend at intersection with undeformed material.

INNER BEND.DIM ID (FK)
: Dimension (radius, preferably, or diameter) of bend within louver.

Business_Rules

- An IMPLICIT LOUVER/090 is a type of IMPLICIT PARTIAL CUTOUT/083.
- An AXIS PLACEMENT/GEO-4 locates 0, 1, or many IMPLICIT LOUVERs/090. This is done as follows. The origin is placed at the center of the shear line on the sheet side that is bent inward. The shear line lies on the Z-axis. Deformation is in the $+y$-direction, so that (0,HEIGHT,0) is the maximum height point of the feature and is the displaced position of the local origin. Deformation occurs in the $0 < x <$ WIDTH portion of the XZ-plane.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is length dimension of 0, 1, or many IMPLICIT LOUVERs/090.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is width dimension of 0, 1, or many IMPLICIT LOUVERs/090.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is height dimension of 0, 1, or many IMPLICIT LOUVERs/090.

- A BEND DIMENSION/429 is base bend dimension of 0, 1, or many IMPLICIT LOUVERs/090.
- A BEND DIMENSION/429 is inner bend dimension of 0, 1, or many IMPLICIT LOUVERs/090.

EXPRESS Declaration

```
ENTITY implicit_louver
SUBTYPE OF (implicit_partial_cutout);
  location   : axis_placement;
  length     : independent_size_parameter;
  width      : independent_size_parameter;
  height     : independent_size_parameter;
  base_bend  : bend_dimension;
  inner_bend : bend_dimension;
END_ENTITY;
```

Entity Name: IMPLICIT CIRCULAR KNOCKOUT

Entity Number: 091

A circular IMPLICIT PARTIAL CUTOUT/083 where the deformation to the material causes no opening to be created. Deformation at the shear line is less than material thickness. The shear line has equally spaced gaps of equal length.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
  Identifies the AXIS PLACEMENT/GEO-4 which locates the feature.

SIZE.DIM ID (FK)
  The size (diameter or radius) of the shear circle.

HEIGHT.DIM ID (FK)
  The height dimension of the protrusion of the feature from the material. (Semi-height is possible, but full height is preferred.)

NUMBER OF GAPS

GAP LENGTH.DIM ID (FK)

The space between adjacent shears, measured chordally. (Semi-length is possible, but full length is preferred.)

BEND.DIM ID (FK)

The bend dimension (radius, preferably, or diameter) between the feature and its neighborhood.

Business Rules

- An IMPLICIT CIRCULAR KNOCKOUT/091 is a type of IMPLICIT PARTIAL CUTOUT/083.
- An AXIS PLACEMENT/GEO-4 locates 0, 1, or many IMPLICIT CIRCULAR KNOCKOUTs /091. This is done as follows. The origin is placed at the center of the shear circle, lying on the shear entry surface of the pre-shear shape. The Z-axis points in the direction of material movement. The gaps in the shear line are assumed to be equally spaced, with the X-axis bisecting one of the gaps.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is size dimension of 0, 1, or many IMPLICIT CIRCULAR KNOCKOUTs/091.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is height dimension of 0, 1, or many IMPLICIT CIRCULAR KNOCKOUTs/091.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is gap length dimension of 0, 1, or many IMPLICIT CIRCULAR KNOCKOUTs/091.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT CIRCULAR KNOCKOUTs/091.
- Full height < thickness of sheet-like environment.

EXPRESS Declaration

```
ENTITY implicit_circular_knockout
SUBTYPE OF (implicit_partial_cutout);
  location   : axis_placement;
  size_dim   : independent_size_parameter;
  height     : independent_size_parameter;
  gap_length : independent_size_parameter;
  bend_dim   : bend_dimension;
END_ENTITY;
```

Entity Name: IMPLICIT GENERAL BEND

Entity Number: 092

The general case of IMPLICIT BEND/084. The bend is specified by giving its bendline and the angle and radius at points on the bendline.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the bendline CURVE/GEO-6 of the feature.

GENERAL BEND DIRECTION
Tells whether the bending direction is by the RIGHT or LEFT hand rule with respect to the direction of the bendline CURVE/GEO-6.

Business Rules

- An IMPLICIT GENERAL BEND/092 is a type of IMPLICIT BEND/084.
- A CURVE/GEO-6 is bendline of 0 or 1 IMPLICIT GENERAL BENDs/092. The bendline CURVE/GEO-6 must lie on one side of the pre-bend shape. After bending, it is the intersection curve of the two major surfaces produced from that side by the bend. (Due to bend radius, the bendline is off-part after bending.)
- An IMPLICIT GENERAL BEND/092 is specified at 1 or many BEND POINTs/096. The POINT/GEO-2 underlying each BEND POINT/096 must lie on the CURVE/GEO-6 that is the bendline.
- An IMPLICIT GENERAL BEND/092 has moved side indicated by 0 or 1 GENERAL BEND MOVEMENT INDICATORS/431.

EXPRESS Declaration

```
ENTITY implicit_general_bend
SUBTYPE OF (implicit_bend);
  bend_line      : curve;
  bend_points    : SET [1:#] OF bend_point;
  bend_direction : hands;
  moved_side     : OPTIONAL hands;
END_ENTITY;

TYPE
```

```
hands = ENUMERATION OF (left_hand, right_hand);
END_TYPE;
```

Entity Name: IMPLICIT TUBE BEND

Entity Number: 093

An implicit representation of a bend in a tube or circular bar.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

LOCATION.GEOMETRIC ENTITY ID (FK)
The bend POINT/GEO-2 of the feature.
DIRECTION.GEOMETRIC ENTITY ID (FK)
The VECTOR/GEO-3 that gives bend direction.
SIZE.DIM ID (FK)
The size (diameter or, preferably, radius) dimension of the bend.
ANGLE.DIM ID (FK)
The angle dimension of the bend. (Semi-angle is possible, but full angle is usually preferred.)

Business Rules

- An IMPLICIT TUBE BEND/093 is a type of IMPLICIT TUBE DEFORMATION/086.
- A POINT/GEO-2 locates 0, 1, or many IMPLICIT TUBE BENDs/093.
- The bend locating POINT/GEO-2 must lie on the pre-bend centerline of the tube. After bending, it is the intersection point of the centerlines of the two lengths into which the bend divides the tube. (Due to bend radius, the point is off-centerline after bending.)
- A VECTOR/GEO-3 gives bend of 0, 1, or many IMPLICIT TUBE BENDs/093.
- A BEND DIMENSION/429 is size dimension of 0, 1, or many IMPLICIT TUBE BENDs/093.
- A DERIVABLE ANGLE SIZE PARAMETER/TOL-6102 is angle dimension of 0, 1, or many IMPLICIT TUBE BENDs/093.
- An IMPLICIT TUBE BEND/093 has moved end indicated by 0 or 1 TUBE BEND MOVEMENT INDICATORs/433.

EXPRESS Declaration

```
ENTITY implicit_tube_bend
SUBTYPE OF (implicit_tube_deformation);
  location    : point;
  direction   : vector;
  bend_size   : bend_dimension;
  bend_angle  : derivable_angle_size_parameter;
  moved_end   : tube_bend_moved_end_types;
END_ENTITY;

TYPE
tube_bend_moved_end_types = ENUMERATION OF (tube_forward_end,
                                            tube_rearward_end);
END_TYPE;
```

Entity Name: IMPLICIT CUTOUT FLANGE

Entity Number: 094

An implicit representation of a flange formed by a bend around the periphery of a passage (cutout) feature. The feature has constant setback, bend angle, and bend radius.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

PASSAGE.SHAPE ID (FK)
: With the following two attributes, identifies the cutout (passage) form feature stiffened by the flange.

PASSAGE.SHAPE ELEMENT ID (FK)

PASSAGE.IMPLICIT FORM FEATURE ID (FK)

FLANGE SIDE
: Whether the bend displaces material toward the INITIAL or TERMINAL end of the passage.

SETBACK.DIM ID (FK)
: The distance from the border of the cutout to the bend line of the flange. (Semi-distance is possible, but full is preferred.)

BEND.DIM ID (FK)
The dimension (radius, preferably, or diameter) of the bend.

ANGLE.DIM ID (FK)
The bend angle dimension. (Semi-angle is possible, but full is preferred.)

Business Rules

- An IMPLICIT CUTOUT FLANGE/094 is a type of IMPLICIT BEND/084.
- An IMPLICIT PASSAGE/003 is stiffened by 0 or 1 IMPLICIT CUTOUT FLANGES/094.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is setback dimension of 0, 1, or many IMPLICIT CUTOUT FLANGEs/094.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT CUTOUT FLANGEs/094.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is angle dimension of 0, 1, or many IMPLICIT CUTOUT FLANGEs/094
- Full angle < 180.

EXPRESS Declaration

```
ENTITY implicit_cutout_flange
SUBTYPE OF (implicit_bend);
  stiffens    : UNIQUE implicit_passage;
  flange_side : feature_end_types;
  setback     : independent_size_parameter;
  bend_dim    : bend_dimension;
  bend_angle  : independent_angle_size_parameter;
WHERE
  bend_angle.dimension < 180;
END_ENTITY;

TYPE
feature_end_types = ENUMERATION OF (initial_end, terminal_end);
END_TYPE;
```

Entity Name: **IMPLICIT STRAIGHT BEND**

Entity Number: **095**

An implicit representation of a simple bend having linear bendline, constant angle, and constant radius.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the bendline LINE/GEO-20.

SIZE.DIM ID (FK)
The size dimension (radius, preferably, or diameter) of the bend.

ANGLE.DIM ID (FK)
The angle of the bend. (While semi-angle is possible, full measurement is preferred.)

STRAIGHT BEND DIRECTION
Tells whether the bending direction is by the RIGHT or LEFT hand rule with respect to the direction of the bendline LINE/GEO-20.

Business Rules

- An IMPLICIT STRAIGHT BEND/095 is a type of IMPLICIT BEND/084.
- A LINE/GEO-20 is bendline of 0, 1, or many IMPLICIT STRAIGHT BENDs/095. The bendline LINE/GEO-20 must lie on one side of the pre-bend shape. After bending, it is the intersection curve of the two major surfaces produced from that side by the bend. (Due to bend radius, the bendline is off-part after bending.)
- A BEND DIMENSION/429 is size dimension of 0, 1, or many IMPLICIT STRAIGHT BENDs/095.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is angle dimension of 0, 1, or many IMPLICIT STRAIGHT BENDs/095.
- Full angle < 180.
- An IMPLICIT STRAIGHT BEND/095 has moved side indicated by 0 or 1 STRAIGHT BEND MOVEMENT INDICATORS/432.

EXPRESS Declaration

```
ENTITY implicit_straight_bend
SUBTYPE OF (implicit_bend);
  bend_line      : line;
  bend_dim       : bend_dimension;
  bend_angle     : independent_angle_size_parameter;
  bend_direction : hands;
```

```
  moved_side      : OPTIONAL hands;
WHERE
  bend_angle.dimension < 180;
END_ENTITY;

TYPE
hands = ENUMERATION OF (left_hand, right_hand);
END_TYPE;
```

Entity Name: **BEND POINT**

Entity Number: **096**

A point on the bendline of an IMPLICIT GENERAL BEND/092 at which bend angle and radius are specified.

Primary Key Attributes

SHAPE ID (FK)
: With the two following attributes, identifies the IMPLICIT GENERAL BEND/092 with which this entity is associated.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

GEOMETRIC ENTITY ID (FK)
: Identifies the point on the bendline.

Other Attributes

SIZE.DIM ID (FK)
: The bend size dimension (radius, preferably, or diameter) at the point.

ANGLE.DIMENSION ID (FK)
: The bend angle dimension at the point. (While semi-angle may be used, whole angle is preferred.)

Business Rules

- An IMPLICIT GENERAL BEND/092 is specified at 1 or many BEND POINTs/096.
- A POINT/GEO-2 locates 0, 1, or many BEND POINTs/096.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many BEND POINTs/096.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is angle dimension of 0, 1, or many BEND POINTs/096.
- Full angle < 180.

EXPRESS Declaration

```
ENTITY bend_point;
  location     : point;
  bend_dim     : bend_dimension;
  bend_angle   : independent_angle_size_parameter;
WHERE
  BEND_ANGLE < 180;
END_ENTITY;
```

Entity Name: **IMPLICIT TUBE FLARE**

Entity Number: **098**

An implicit representation of a tapered increase in the diameter of a tube, occurring at an end of the tube.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other_Attributes

FLARED END
Whether the flare occurs at the INITIAL or TERMINAL end of the tube. (It is assumed that the tube has a centerline curve whose direction gives meaning to this attribute.)

OD.DIM ID (FK)
The OD dimension (diameter or radius) at the end of the tube.

LENGTH.DIM ID (FK)
The length dimension of the flare. (It is possible to give semi-length, but full value is preferred.)

ANGLE.DIM ID (FK)
The derivable angle of the flare. (It is possible to give semi-angle, but full value is preferred.)

BEND.DIM ID (FK)
The bend dimension (radius, preferably, or diameter) at the boundary of the feature.

ID.DIM ID (FK)
The derivable ID dimension (radius or diameter) at the end of the tube.

Business Rules

- An IMPLICIT TUBE FLARE/098 is a type of IMPLICIT TUBE DEFORMATION/086.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is end OD dimension of 0, 1, or many IMPLICIT TUBE FLAREs/098.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is length dimension of 0, 1, or many IMPLICIT TUBE FLAREs/098.
- A DERIVABLE ANGLE SIZE PARAMETER/TOL-6102 is angle dimension of 0, 1, or many IMPLICIT TUBE FLAREs/098.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT TUBE FLAREs/098.
- A DERIVABLE SIZE PARAMETER/TOL-6 2 is end ID dimension of 0, 1, or many IMPLICIT TUBE FLAREs/098.

NOTES

- Original tube OD, ID, and wall thickness are presumed to be known from other model information.

EXPRESS Declaration

```
ENTITY implicit_tube_flare
SUBTYPE OF (implicit_tube_deformation);
  flared_end   : feature_end_types;
  end_od       : independent_size_parameter;
  length       : independent_size_parameter;
  angle        : derivable_angle_size_parameter;
  bend_dim     : bend_dimension;
  end_id       : derivable_size_parameter;
END_ENTITY;

TYPE
feature_end_types = ENUMERATION OF (initial_end, terminal_end);
END_TYPE;
```

Entity Name: IMPLICIT TUBE NECK

Entity Number: 099

An implicit representation of a tapered change in the diameter of a tube.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the POINT/GEO-2 that locates the feature.

NECK DIRECTION
(See Business Rules.)

OD.DIM ID (FK)
The OD dimension (diameter or radius) of the neck.

ID.DIM ID (FK)
The derivable ID dimension (diameter or radius) of the neck.

LENGTH.DIM ID (FK)
The length of tube over which the change in diameter occurs. (Half-measure can be given, but full is preferred.)

ANGLE.DIM ID (FK)
The derivable angle dimension of the flare/neck. (Half-measure can be meant, but full is preferred.)

BEND.DIM ID (FK)
The bend dimension (radius, preferably, or diameter) at the boundary between the feature and the larger (by diameter) of the two constant-diameter sections of tube it connects.

Business Rules

- An IMPLICIT TUBE NECK/099 is a type of IMPLICIT TUBE DEFORMATION/086.
- A POINT/GEO-2 locates 0 or 1 IMPLICIT TUBE NECKs/099. The POINT/GEO-2 is located on the tube centerline at the place where diameter begins to increase; i.e., at the intersection of the smaller diameter section of tube and the tapered section (ignoring the bend radius). NECK DIRECTION tells whether the increase in diameter occurs in the POSITIVE or NEGATIVE direction from this POINT/GEO-2. (This assumes that there is a directed centerline curve associated with the tube.)
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is OD dimension of 0, 1, or many IMPLICIT TUBE NECKs/099.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is ID dimension of 0, 1, or many IMPLICIT TUBE NECKs/099.

- An INDEPENDENT SIZE PARAMETER/TOL-6041 is length dimension of 0, 1, or many IMPLICIT TUBE NECKs/099.
- A DERIVABLE ANGLE SIZE PARAMETER/TOL-6102 is angle dimension of 0, 1, or many IMPLICIT TUBE NECKs/099.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT TUBE NECKs/099.

NOTES

- Original tube OD, ID, and wall thickness are presumed to be known from other model information.

EXPRESS Declaration

```
ENTITY implicit_tube_neck
SUBTYPE OF (implicit_tube_deformation);
  location        : point;
  neck_direction  : neck_direction_types;
  outside_dim     : independent_size_parameter;
  inside_dim      : derivable_size_parameter;
  length          : independent_size_parameter;
  angle           : derivable_angle_size_parameter;
  bend_dim        : bend_dimension;
END_ENTITY;

TYPE
neck_direction_types = ENUMERATION OF (neck_positive, neck_negative);
END_TYPE;
```

Entity Name: **IMPLICIT TUBE FLATTENING**

Entity Number: **100**

An implicit representation of a deformation of the end of a tube to an oblong shape.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the AXIS PLACEMENT/GEO-4 that locates the feature.

OUTER MINOR.DIM ID (FK)
The smaller dimension (half or full measure) of the oblong, measured on the outside.

OUTER MAJOR.DIM ID (FK)
The larger dimension (half or full measure) of the oblong, measured on the outside.

LENGTH1.DIM ID (FK)
The length along which the transition from circular tube to oblong shape occurs. (Semi-length is possible, but full measure is preferable.)

LENGTH2.DIM ID (FK)
The derivable length along which the final oblong shape is maintained. (Semi-length is possible, but full measure is preferable.)

BEND.DIM ID (FK)
The dimension (radius,preferably, or diameter) of the bend between the transitional section of the feature and the fully flattened area.

INNER MINOR.DIM ID (FK)
The derivable smaller dimension (half or full measure) of the inside of the oblong.

INNER MAJOR.DIM ID (FK)
The derivable larger dimension (half or full measure) of the inside of the oblong

Business Rules

- An IMPLICIT TUBE FLATTENING/100 is a type of IMPLICIT TUBE DEFORMATION/086.
- An AXIS PLACEMENT/GEO-4 locates 0 or 1 IMPLICIT FLATTENINGs/100. The origin must lie on the centerline of the tube being flattened at the point at which flattening begins to occur. The Z-axis points in the direction of changing shape. That is, the Z-axis points toward the direction where the oblong shape is achieved. The X-axis is parallel to the minor axis of the oblong and the Y-axis to its major axis.
- OUTER MINOR < original tube OD
- OUTER MAJOR > original tube OD
- LENGTH1 < distance, in the $+z$ direction, from the origin to the end of the tube.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is outer minor dimension of 0, 1, or many IMPLICIT TUBE FLATTENINGs/100.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is outer major dimension of 0, 1, or many IMPLICIT TUBE FLATTENINGs/100.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is length1 dimension of 0, 1, or many IMPLICIT TUBE FLATTENINGs/100.

- A DERIVABLE SIZE PARAMETER/TOL-6042 is length2 dimension of 0, 1, or many IMPLICIT TUBE FLATTENINGs/100.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT TUBE FLATTENINGS/100.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is inner minor dimension of 0, 1, or many IMPLICIT TUBE FLATTENINGs/100.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is outer major dimension of 0, 1, or many IMPLICIT TUBE FLATTENINGs/100.

NOTES

- Original tube OD, ID, and wall thickness are presumed to be known from other model information. Other dimensions can be calculated from these, END WIDTH, and LENGTH, assuming
  (a) constant wall thickness, and
  (b) constant circumference, and
  (c) linear variation of all dimensions from local $z = 0$ to $z =$ LENGTH, and
  (d) constant tube length.

EXPRESS Declaration

```
ENTITY implicit_tube_flattening
SUBTYPE OF (implicit_tube_deformation);
  location    : axis_placement;
  outer_major : independent_size_parameter;
  outer_minor : independent_size_parameter;
  length1     : independent_size_parameter;
  length2     : derivable_size_parameter;
  bend_dim    : bend_dimension;
  inner_major : derivable_size_parameter;
  inner_minor : derivable_size_parameter;
END_ENTITY;
```

Entity Name: **IMPLICIT TUBE ROLL**

Entity Number: **101**

An implicit representation of a tube deformation feature where a tube end is expanded in diameter and rolled (curled) back on itself. (This feature is usually made by forcing the tube end into a die with a conical center.)

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

ROLLED END
Whether the flare occurs at the INITIAL or TERMINAL end of the tube. (It is assumed that the tube has a centerline curve whose direction gives meaning to this attribute.)

BEFORE LENGTH.DIM ID (FK)
The length of tubing to be deformed. (Half dimension is possible, but full is preferred.)

TUBE SIZE.DIM ID (FK)
The maximum size (diameter, preferably, or radius) at/near the rolled end.

CURL LENGTH.DIM ID (FK)
The doubled-over length after rolling. (Half dimension is possible, but full is preferred.)

GAP.DIM ID (FK)
The space between the curled end and the uncurled section of the tube. (Half dimension is possible, but full is preferred.)

CURL SIZE.DIM ID (FK)
The curvature dimension (radius, preferably, or diameter) of the curl.

Business Rules

- An IMPLICIT TUBE ROLL/101 is a type of IMPLICIT TUBE DEFORMATION/086.
- TUBE SIZE > size of tube before rolling
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is before length dimension of 0, 1, or many IMPLICIT TUBE ROLLs/101.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is tube size dimension of 0, 1, or many IMPLICIT TUBE ROLLs/101.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is curl length dimension of 0, 1, or many IMPLICIT TUBE ROLLs/101.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is gap dimension of 0, 1, or many IMPLICIT TUBE ROLLs/101.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is curl size dimension of 0, 1, or many IMPLICIT TUBE ROLLs/101.

NOTES

- The rolled end of the tube may not pinch or intersect the tube.
- Original tube OD, ID, and wall thickness are presumed to be known from other model information.

EXPRESS Declaration

```
ENTITY implicit_tube_roll
SUBTYPE OF (implicit_tube_deformation);
  rolled_end    : feature_end_types;
  before_length : independent_size_parameter;
  tube_size     : independent_size_parameter;
  curl_length   : independent_size_parameter;
  gap           : indepe :ent_size_parameter;
  curl_size     : independent_size_parameter;
END_ENTITY;

TYPE
feature_end_types = ENUMERATION OF (initial_end, terminal_end);
END_TYPE;
```

Entity Name: IMPLICIT AREA FEATURE

Entity Number: 111

An implicit representation of one of a class of form features viewed as being installed upon areas of pre-existing shape. Examples are gear teeth, knurls, and threads.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

AREA.SHAPE ID (FK)
With the following attribute, identifies the DIMENSIONALITY-2 SHAPE ELEMENT on which the feature is installed.
AREA.SHAPE ELEMENT ID (FK)
IMPLICIT AREA FEATURE TYPE (Discriminator)

Business Rules

- An IMPLICIT AREA FEATURE/111 is a type of IMPLICIT FORM FEATURE/002.
- A DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 is installation area of 0, 1, or many IMPLICIT AREA FEATUREs/111.
- An IMPLICIT AREA FEATURE/111 may be either an IMPLICIT KNURL/024, an IMPLICIT THREAD/025, an IMPLICIT MARKING/406, or an IMPLICIT COUPLING/415.

NOTES

- Most or all of the feature types in this class are invariably patterns and, in practice, the patterns are described as a whole. That practice is followed here – the entities give a unitary description of the patterns rather than using IMPLICIT FORM FEATURE PATTERN/049.

EXPRESS Declaration

```
ENTITY implicit_area_feature
SUPERTYPE OF (implicit_knurl OR
              implicit_marking OR
              implicit_coupling OR
              implicit_thread)
SUBTYPE OF (implicit_form_feature);
  installation_area : OPTIONAL dimensionality_2_shape_element;
END_ENTITY;
```

Entity Name: **FEATURE SWEEP**

Entity Number: **120**

A procedural definition of a 2 1/2 dimensional shape consisting of a planar profile, a path along which the profile is to be swept, and shape definitions for either or both ends.

The profile is swept normal to the path. The orientation of the path with respect to the profile is achieved by:

(a) Providing a pre-defined orientation for each path type relative to the feature's local coordinate system (LCS). For example, linear sweep paths begin at the LCS origin and extend along the positive Z-axis.

(b) Defining profiles in their own 2-space called AB-space. Each profile type has a pre-defined location and orientation in its AB-space. For example, a rectangular profile has its center at the AB-origin with a specified pair of sides parallel to the A-axis.

(c) Positioning the AB-axes of the profile in the feature's LCS according to conventions for each path type. For a linear path, the profile's A- and B-axes are mapped to the X- and Y-axes of the LCS, respectively.

The shape of FEATURE SWEEPs/120 may be refined via three types of blends: blends within the profile; blends within sweep end shapes, such as a radiused tip on a conical bottomed hole; and blends between the sides and ends of the swept shape.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
The AXIS PLACEMENT/GEO-4 which locates the swept feature. This defines the feature's LCS.

FEATURE SWEEP TYPE (Discriminator)

Business Rules

- A FEATURE SWEEP/120 is a type of FEATURE VOLUME/417.
- A FEATURE SWEEP/120 may be either an ALONG FEATURE SWEEP/121, an AXISYMMETRIC FEATURE SWEEP/122, or an IN/OUT FEATURE SWEEP/123.
- An AXIS PLACEMENT/GEO-4 locates of 0, 1, or many FEATURE SWEEPs/120. It does so, in effect, by transforming the swept profile from the 2-space in which it is defined into its initial sweeping position in global space.

EXPRESS Declaration

```
ENTITY feature_sweep
SUPERTYPE OF (along_feature_sweep OR
              axisymmetric_feature_sweep OR
              in_out_feature_sweep)
SUBTYPE OF (feature_volume);
  location : axis_placement;
END_ENTITY;
```

Entity Name: ALONG FEATURE SWEEP

Entity Number: 121

A FEATURE SWEEP/120 whose path is roughly along the boundary (air/material interface) of pre-existing shape. Some common features defined in this manner are grooves and channels.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

FEATURE SWEEP PATH ID (FK) (AK1)
FEATURE PROFILE ID (FK) (AK1)

Business Rules

- An ALONG FEATURE SWEEP/121 is a type of FEATURE SWEEP/120.
- A FEATURE SWEEP PATH/124 is the path of 0, 1, or many ALONG FEATURE SWEEPs/121.
- An OPEN FEATURE SWEEP PROFILE/142 defines the profile for 0, 1, or many ALONG FEATURE SWEEPs/121.
- An ALONG FEATURE SWEEP/121 ends with 0, 1, or 2 ALONG FEATURE SWEEP ENDs/158.

EXPRESS Declaration

```
ENTITY along_feature_sweep
SUBTYPE OF (feature_sweep);
  sweep_path    : feature_sweep_path;
  sweep_profile : open_feature_sweep_profile;
  sweep_ends    : SET [0:2] OF along_feature_sweep_end;
END_ENTITY;
```

Entity Name: **AXISYMMETRIC FEATURE SWEEP**

Entity Number: **122**

A FEATURE SWEEP/120 that is realized by sweeping a planar curve 360 degrees about a coplanar axis.

The swept curve may be explicitly or implicitly defined. Explicit definition is available via OTHER AXISYMMETRIC FEATURE SWEEP/151, which uses planar curve strings to define arbitrary profiles. Implicit curves are defined by CONSTANT DIAMETER AXISYMMETRIC FEATURE SWEEP/149 and TAPERED AXISYMMETRIC FEATURE SWEEP/150.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

DIM ID (FK)
The length dimension of the swept profile. (Semi-length is possible, but full is preferred.)
AXISYMMETRIC SWEEP TYPE (Discriminator)

Business Rules

- An AXISYMMETRIC FEATURE SWEEP/122 is a type of FEATURE SWEEP/120.
- An AXISYMMETRIC FEATURE SWEEP/122 must be either a CONSTANT DIAMETER AXISYMMETRIC FEATURE SWEEP/149, a TAPERED AXISYMMETRIC FEATURE SWEEP/150, or an OTHER AXISYMMETRIC FEATURE SWEEP/151.
- An AXISYMMETRIC FEATURE SWEEP/122 ends with 0 or 1 AXISYMMETRIC FEATURE SWEEP ENDs/130.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is length dimension of 0, 1, or many LINEAR FEATURE SWEEP PATHs/123.
- The sweep axis coincides with the Z-axis of the feature's local coordinate system.
- The curve may be of any length. However, is is intended that only the portion of the curve defined between $Z = 0$ and $Z =$ LENGTH (full) be operative. (If the curve is not defined throughout the range $0 \leq Z \leq$ LENGTH, then it is assumed to extend to cover the range.)
- All profiles for axisymmetric sweeps are defined in a local AB-space. The local A- and B-axes of the profile are equated with the X- and Z-axes, respectively, of the feature's LCS.
- The curve must not intersect the Z axis in the range $0 < Z <$ LENGTH.

NOTES

- The ends of the swept curve are assumed to be planar, unless an AXISYMMETRIC FEATURE SWEEP END/130 is specified. If an AXISYMMETRIC FEATURE SWEEP END/130 is given, it extends in the $-Z$ direction from $Z = 0$.

EXPRESS Declaration

```
ENTITY axisymmetric_feature_sweep
SUPERTYPE OF (constant_diameter_axisymmetric_feature_sweep OR
              tapered_axisymmetric_feature_sweep OR
```

```
             other_axisymmetric_feature_sweep)
SUBTYPE OF (feature_sweep);
  sweep_length : independent_size_parameter;
  sweep_end    : OPTIONAL axisymmetric_feature_sweep_end;
END_ENTITY;
```

Entity Name: **IN/OUT FEATURE SWEEP**

Entity Number: **123**

A FEATURE SWEEP/120 whose path is roughly a "plunge" into or extrusion from the rest of the shape. Pockets and bosses are typical features defined with this type of sweep.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

FEATURE SWEEP PATH ID (FK) (AK1)
FEATURE PROFILE ID (FK) (AK1)
IN/OUT SWEEP TYPE (Discriminator)

Business Rules

- An IN/OUT FEATURE SWEEP/123 is a type of FEATURE SWEEP/120.
- A FEATURE SWEEP PATH/124 is the path of 0, 1, or many IN/OUT FEATURE SWEEPs/123.
- A CLOSED FEATURE SWEEP PROFILE/137 defines the profile for 0, 1, or many IN/OUT FEATURE SWEEPs/123.
- An IN/OUT FEATURE SWEEP/123 is blended by 0 or 1 IN/OUT FEATURE SWEEP WALL/END BLENDs/129. (This is the round or chamfer at the end of the sweep.)
- An IN/OUT FEATURE SWEEP/123 may be either a CONSTANT PROFILE IN/OUT SWEEP/190, a TAPERED PROFILE IN/OUT SWEEP/192, or a CONTOURED PROFILE IN/OUT SWEEP/193.

NOTES

- Features defined by IN/OUT FEATURE SWEEPs/123 have planar ends.

EXPRESS Declaration

```
ENTITY in_out_feature_sweep
SUPERTYPE OF (constant_profile_in_out_sweep OR
              tapered_profile_in_out_sweep OR
              contoured_profile_in_out_sweep)
SUBTYPE OF (feature_sweep);
  sweep_path      : linear_feature_sweep_path;
  sweep_profile   : closed_feature_sweep_profile;
  wall_end_blend  : OPTIONAL implicit_edge_blend;
END_ENTITY;
```

Entity Name: FEATURE SWEEP PATH

Entity Number: 124

A curve along which a FEATURE SWEEP PROFILE/136 is swept to define the shape of an IMPLICIT FORM FEATURE/002.

Primary Key Attributes

FEATURE SWEEP PATH ID
A unique arbitrary label.

Other Attributes

SWEEP PATH TYPE (Discriminator)

Business Rules

- A FEATURE SWEEP PATH/124 must be either a LINEAR FEATURE SWEEP PATH/125, a CIRCULAR FEATURE SWEEP PATH/126, a SPIRAL FEATURE SWEEP PATH/422, a SURFACE CONFORMING FEATURE SWEEP PATH/423, or an
  OTHER FEATURE SWEEP PATH/424.
- A FEATURE SWEEP PATH/124 is the path of 0, 1, or many ALONG FEATURE SWEEPs/121.
- A FEATURE SWEEP PATH/124 is the path of 0, 1, or many IN/OUT FEATURE SWEEPs/123.

EXPRESS Declaration

```
ENTITY feature_sweep_path
SUPERTYPE OF (circular_feature_sweep_path OR
              spiral_feature_sweep_path OR
              surface_conforming_feature_sweep_path OR
              other_feature_sweep_path OR
              linear_feature_sweep_path);
END_ENTITY;
```

Entity Name: **LINEAR FEATURE SWEEP PATH**

Entity Number: **125**

A FEATURE SWEEP PATH/124 which is a straight line.

Primary Key Attributes

FEATURE SWEEP PATH ID (FK)

Other Attributes

DIM ID (FK)
The length dimension of the sweep path. (Semi-length is possible, but full is preferred.)

Business Rules

- A LINEAR FEATURE SWEEP PATH/125 is a type of FEATURE SWEEP PATH/124.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the path length dimension of 0, 1, or many LINEAR FEATURE SWEEP PATHs/125.
- The sweep is along the $+Z$ axis of the feature's LCS from $Z = 0$ to $Z =$ (full length of path).
- The A- and B-axes of profiles used in linear sweeps are equated with the feature's LCS X- and Y-axes, respectively.
- If the path is that of an ALONG FEATURE SWEEP/121 which has ALONG FEATURE SWEEP ENDs/158 defined, then the ends extend in the $-Z$ direction from Z=0 or the $+Z$ direction from $Z =$ (full length of path).

EXPRESS Declaration

```
ENTITY linear_feature_sweep_path
SUBTYPE OF (feature_sweep_path);
  path_length : independent_size_parameter;
END_ENTITY;
```

Entity Name: **CIRCULAR FEATURE SWEEP PATH**

Entity Number: **126**

A FEATURE SWEEP PATH/124 which is circular.

Primary Key Attributes

FEATURE SWEEP PATH ID (FK)

Other Attributes

SIZE.DIM ID (FK)
The dimension (diameter or radius) of the circular path.

ORIENTATION ANGLE.DIM ID (FK)
An angular dimension (full- or semi-angle) which orients the AB-axes of the sweep's profile with the XZ-axes of the feature's LCS (See Business Rule 4).

CIRCULAR SWEEP PATH TYPE (Discriminator)

Business Rules

- A CIRCULAR FEATURE SWEEP PATH/126 is a type of FEATURE SWEEP PATH/124.
- A CIRCULAR FEATURE SWEEP PATH/126 must be either a PARTIAL CIRCULAR FEATURE SWEEP PATH/127 or a COMPLETE CIRCULAR FEATURE SWEEP PATH/128.
- The sweep is about the LCS Z-axis.
- If the path is a PARTIAL CIRCULAR SWEEP PATH/127, then the full profile interval begins at the XZ plane and runs counterclockwise as viewed from the $+Z$ halfspace. If any ALONG FEATURE SWEEP ENDs/158 are defined, they extend in the clockwise direction from the XZ-plane or the counterclockwise direction from the plane defined by the ORIENTATION ANGLE attribute.

- The AB-space of the profile is positioned and oriented in the LCS XZ-plane by:
  (a) the profile's origin is mapped onto the X-axis at X = RADIUS, and
  (b) the angle from the Z-axis to the profile B-axis equals ORIENTATION ANGLE, measured counterclockwise as viewed from the $+Y$ halfspace. When the angle is 0, the profile A-axis extends in the direction of the positive X-axis.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is size dimension of 0, 1, or many CIRCULAR FEATURE SWEEP PATHs/126.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the orientation angle of 0, 1, or many CIRCULAR FEATURE SWEEP PATHs/126.

EXPRESS Declaration

```
ENTITY circular_feature_sweep_path
SUPERTYPE OF (complete_circular_feature_sweep_path OR
              partial_circular_feature_sweep_path)
SUBTYPE OF (feature_sweep_path);
  path_size         : independent_size_parameter;
  orientation_angle : independent_size_parameter;
END_ENTITY;
```

Entity Name: PARTIAL CIRCULAR FEATURE SWEEP PATH

Entity Number: 127

A CIRCULAR FEATURE SWEEP PATH/126 which is an arc of less than 360 degrees.

Primary Key Attributes

FEATURE SWEEP PATH ID (FK)

Other Attributes

DIM ID (FK)
The swept portion of the circle, given as full angle or semi-angle.

Business Rules

- A PARTIAL CIRCULAR FEATURE SWEEP PATH/127 is a type of CIRCULAR FEATURE SWEEP PATH/126.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is the swept angle of 0, 1, or many PARTIAL CIRCULAR FEATURE SWEEP PATHs/127.

EXPRESS Declaration

```
ENTITY partial_circular_feature_sweep_path
SUBTYPE OF (circular_feature_sweep_path);
  sweep_angle : independent_size_parameter;
WHERE
  -360 < sweep_angle.dimension < 360;
END_ENTITY;
```

Entity Name: COMPLETE CIRCULAR FEATURE SWEEP PATH

Entity Number: 128

A 360 degree CIRCULAR FEATURE SWEEP PATH/126.

Primary Key Attributes

FEATURE SWEEP PATH ID (FK)

Business Rules

- A COMPLETE CIRCULAR FEATURE SWEEP PATH/128 is a type of CIRCULAR FEATURE SWEEP PATH/126.

EXPRESS Declaration

```
ENTITY complete_circular_feature_sweep_path
SUBTYPE OF (circular_feature_sweep_path);
END_ENTITY;
```

Entity Name: IN/OUT FEATURE SWEEP WALL/END BLEND

Entity Number: 129

A blend (round or flat) between the end of a feature defined by an IN/OUT FEATURE SWEEP/122 and the walls/sides of the feature.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An IN/OUT FEATURE SWEEP/123 is blended by 0 or 1 IN/OUT FEATURE SWEEP WALL/END BLENDs/129.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many IN/OUT FEATURE SWEEP WALL/END BLENDs/129.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under IN/OUT FEATURE SWEEP/123.)

Entity Name: **AXISYMMETRIC FEATURE SWEEP END**

Entity Number: **130**

The end shape of an IMPLICIT FORM FEATURE/002 defined by an AXISYMMETRIC FEATURE SWEEP/122. As an example, this entity can be used to model the common end shapes for a hole: flat, conical, and spherical.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

AXISYMMETRIC SWEEP END TYPE (Discriminator)

Business Rules

- An AXISYMMETRIC FEATURE SWEEP/122 ends with 0 or 1 AXISYMMETRIC FEATURE SWEEP ENDs/130.

- An AXISYMMETRIC FEATURE SWEEP END/130 may be either an AXISYMMETRIC FEATURE SWEEP FLAT END/132, an AXISYMMETRIC FEATURE SWEEP SPHERICAL END/133, or an AXISYMMETRIC FEATURE SWEEP CONICAL END/134.
- An AXISYMMETRIC FEATURE SWEEP END/130 is blended by 0 or 1 AXISYMMETRIC FEATURE SWEEP WALL/END BLENDs/131.
- An AXISYMMETRIC FEATURE SWEEP END/130 which is an AXISYMMETRIC FEATURE SWEEP SPHERICAL END/133 must not have an associated AXISYMMETRIC FEATURE SWEEP WALL/END BLEND/131.
- The end meets the swept body of the feature at $Z = 0$. Non-planar ends extend in the $-Z$ direction.

NOTES

- When no AXISYMMETRIC FEATURE SWEEP END/130 is specified for an AXISYMMETRIC FEATURE SWEEP/122, the feature is assumed to have a flat end. A flat end must be modeled explicitly in order to have an associated AXISYMMETRIC FEATURE SWEEP WALL/END BLEND/131.

EXPRESS Declaration

```
ENTITY axisymmetric_feature_sweep_end
SUPERTYPE OF (axisymmetric_feature_sweep_flat_end OR
              axisymmetric_feature_sweep_spherical_end OR
              axisymmetric_feature_sweep_conical_end);
  blend : OPTIONAL implicit_edge_blend;
END_ENTITY;
```

Entity Name: **AXISYMMETRIC FEATURE SWEEP WALL/END BLEND**

Entity Number: **131**

A blend (round or flat) between the end of a feature defined by an AXISYMMETRIC FEATURE SWEEP/122 and the wall(s) of the feature defined by its profile.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An AXISYMMETRIC FEATURE SWEEP END/130 is blended by 0 or 1 AXISYMMETRIC FEATURE SWEEP WALL/END BLENDs/131.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many AXISYMMETRIC FEATURE SWEEP WALL/END BLENDs/131.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under AXISYMMETRIC FEATURE SWEEP END/130.)

Entity Name: **AXISYMMETRIC FEATURE SWEEP FLAT END**

Entity Number: **132**

An indication that an AXISYMMETRIC FEATURE SWEEP END/130 is planar.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Business Rules

- An AXISYMMETRIC FEATURE SWEEP FLAT END/132 is a type of AXISYMMETRIC FEATURE SWEEP END/130.
- The plane of the AXISYMMETRIC FEATURE SWEEP FLAT END is local $Z = 0$.

EXPRESS Declaration

```
ENTITY axisymmetric_feature_sweep_flat_end
SUBTYPE OF (axisymmetric_feature_sweep_end);
END_ENTITY;
```

Entity Name: **AXISYMMETRIC FEATURE SWEEP SPHERICAL END**

Entity Number: **133**

An indication that an AXISYMMETRIC FEATURE SWEEP END/130 is spherical.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Business Rules

- An AXISYMMETRIC FEATURE SWEEP SPHERICAL END/132 is a type of AXISYMMETRIC FEATURE SWEEP END/130.
- The radius of an AXISYMMETRIC FEATURE SWEEP SPHERICAL END/132 is equal to the radius of the parent AXISYMMETRIC FEATURE SWEEP/122 at local $Z = 0$.

EXPRESS Declaration

```
ENTITY axisymmetric_feature_sweep_spherical_end
SUBTYPE OF (axisymmetric_feature_sweep_end);
END_ENTITY;
```

Entity Name: **AXISYMMETRIC FEATURE SWEEP CONICAL END**

Entity Number: **134**

An indication that an AXISYMMETRIC FEATURE SWEEP END/130 is conical.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

ANGLE.DIM ID (FK)
The full angle or semi-angle of the implicit cone.

Business Rules

- An AXISYMMETRIC FEATURE SWEEP CONICAL END/132 is a type of AXISYMMETRIC FEATURE SWEEP END/130.
- An AXISYMMETRIC FEATURE SWEEP CONICAL END/132 is blended at its tip by 0 or 1 AXISYMMETRIC FEATURE SWEEP CONICAL END TIP BLENDs/135.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is the angle dimension of 0, 1, or many AXISYMMETRIC FEATURE SWEEP CONICAL ENDs/134.
- Full ANGLE < 180.

EXPRESS Declaration

```
ENTITY axisymmetric_feature_sweep_conical_end
SUBTYPE OF (axisymmetric_feature_sweep_end);
  end_angle : independent_size_parameter;
  tip_blend : OPTIONAL implicit_edge_round;
WHERE
  end_angle.dimension < 180;
END_ENTITY;
```

Entity Name: AXISYMMETRIC FEATURE SWEEP CONICAL END TIP BLEND

Entity Number: 135

A blend at the tip of an AXISYMMETRIC FEATURE SWEEP CONICAL END/134.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

SHAPE ID (FK)
  With the following two attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An IMPLICIT EDGE ROUND/037 is used as 0,1,or many AXISYMMETRIC FEATURE SWEEP CONICAL END TIP BLENDs/135.
- An AXISYMMETRIC FEATURE SWEEP CONICAL END/132 is blended at its tip by 0 or 1 AXISYMMETRIC FEATURE SWEEP CONICAL END TIP BLENDs/135.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under AXISYMMETRIC FEATURE SWEEP CONICAL END/132.)

Entity Name: **FEATURE SWEEP PROFILE**

Entity Number: **136**

A planar curve or connected set of curves which is swept through space along a FEATURE SWEEP PATH/124 to define an IMPLICIT FORM FEATURE/002. Profiles are defined in a local AB-space which is mapped into the feature's LCS according to conventions based on the type of path. Conventions for the orientation of the profile within the AB-space are specified for the type of profile.

Primary Key Attributes

FEATURE PROFILE ID
An arbitrary label uniquely identifying each profile.

Other Attributes

SWEEP PROFILE TYPE (Discriminator)

Business Rules

- A FEATURE SWEEP PROFILE/136 must be either a CLOSED FEATURE SWEEP PROFILE/137 or an OPEN FEATURE SWEEP PROFILE/142.

EXPRESS Declaration

```
ENTITY feature_sweep_profile
SUPERTYPE OF (closed_feature_sweep_profile OR
              open_feature_sweep_profile);
END_ENTITY;
```

Entity Name: **CLOSED FEATURE SWEEP PROFILE**

Entity Number: **137**

A FEATURE SWEEP PROFILE/136 that is a closed curve or a closed composite curve.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

CLOSED PROFILE TYPE (Discriminator)

Business Rules

- A CLOSED FEATURE SWEEP PROFILE/137 is a type of FEATURE SWEEP PROFILE/136.
- A CLOSED FEATURE SWEEP PROFILE/137 must be either a STANDARD CLOSED FEATURE SWEEP PROFILE/167 or an OTHER CLOSED FEATURE SWEEP PROFILE/168.
- A CLOSED FEATURE SWEEP PROFILE/137 defines the profile for 0, 1, or many IN/OUT FEATURE SWEEPs/123.

EXPRESS Declaration

```
ENTITY closed_feature_sweep_profile
SUPERTYPE OF (other_closed_feature_sweep_profile OR
              standard_closed_feature_sweep_profile)
SUBTYPE OF (feature_sweep_profile);
END_ENTITY;
```

Entity Name: **RECTANGULAR FEATURE SWEEP PROFILE**

Entity Number: **139**

A rectangle used as a CLOSED FEATURE SWEEP PROFILE/136.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

LENGTH.DIM ID (FK)
The full length or semi-length dimension of the rectangle.

WIDTH.DIM ID (FK)
The full width or semi-width dimension of the rectangle.

Business Rules

- A RECTANGULAR FEATURE SWEEP PROFILE/139 is a type of STANDARD CLOSED FEATURE SWEEP PROFILE/167.
- The rectangle is oriented with its center at the origin of the local AB-plane. The sides given by the LENGTH attribute are parallel to the A-axis.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the length of 0, 1, or many RECTANGULAR FEATURE SWEEP PROFILEs/139.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the width of 0, 1, or many RECTANGULAR FEATURE SWEEP PROFILEs/139.

EXPRESS Declaration

```
ENTITY rectangular_feature_sweep_profile
SUBTYPE OF (standard_closed_feature_sweep_profile);
  sweep_length : independent_size_parameter;
  sweep_width  : independent_size_parameter;
END_ENTITY;
```

Entity Name: **N-GON FEATURE SWEEP PROFILE**

Entity Number: **140**

A regular n-gon used as a STANDARD CLOSED FEATURE SWEEP PROFILE/167.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

N-GON NUMBER OF SIDES

INSCRIBED.DIM ID (FK)
The dimension (radius or diameter) of a circle inscribed in the n-gon.

CIRCUMSCRIBED.DIM ID (FK)
The derivable dimension (radius or diameter) of a circle circumscribed about the n-gon.

LENGTH.DIM ID (FK)
The derivable length (half or full dimension) of a side of the n-gon.

INTERIOR ANGLE.DIM ID (FK)
The derivable interior angle or semi-angle between sides.

EXTERIOR ANGLE.DIM ID (FK)
The derivable exterior angle (or semi-angle) between sides. This is the angle between one side and the extension of an adjacent side; i.e., the supplement of the interior angle.

Business Rules

- An N-GON FEATURE SWEEP PROFILE/140 is a type of STANDARD CLOSED FEATURE SWEEP PROFILE/167.
- The N-GON is defined with its center at the local AB-origin. One of its sides is parallel to the A-axis.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the inscribed circle dimension of 0, 1, or many N-GON FEATURE SWEEP PROFILEs/140.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is the circumscribed circle dimension of 0, 1, or many N-GON FEATURE SWEEP PROFILEs/140.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is the side length dimension of 0, 1, or many N-GON FEATURE SWEEP PROFILEs/140.
- A DERIVABLE ANGLE SIZE PARAMETER/TOL-6102 is interior angle dimension of 0, 1, or many N-GON FEATURE SWEEP PROFILEs/140.
- A DERIVABLE ANGLE SIZE PARAMETER/TOL-6102 is exterior angle dimension of 0, 1, or many N-GON FEATURE SWEEP PROFILEs/140.

EXPRESS Declaration

```
ENTITY ngon_feature_sweep_profile
SUBTYPE OF (standard_closed_feature_sweep_profile);
  number_of_sides   : INTEGER;
  inscribed_dim     : independent_size_parameter;
  circumscribed_dim : derivable_size_parameter;
  side_length       : derivable_size_parameter;
  interior_angle    : derivable_angle_size_parameter;
  exterior_angle    : derivable_angle_size_parameter;
END_ENTITY;
```

Entity Name: **STANDARD CLOSED FEATURE SWEEP PROFILE BLEND**

Entity Number: **141**

A blend between the segments of a STANDARD CLOSED FEATURE SWEEP PROFILE/167. The blend applies to all C1 non-continuous vertices.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A STANDARD CLOSED FEATURE SWEEP PROFILE/167 is blended by 0 or 1 STANDARD CLOSED FEATURE SWEEP PROFILE BLENDs/141.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many STANDARD CLOSED FEATURE SWEEP PROFILE BLENDs/141.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under STANDARD CLOSED FEATURE SWEEP PROFILE/167.)

Entity Name: **OPEN FEATURE SWEEP PROFILE**

Entity Number: **142**

A FEATURE SWEEP PROFILE/136 consisting of a curve or composite curve which does not close. The open ends of the profile are assumed to extend to the material/air interface. For example, only the width and blends are specified for a "U"-shaped profile; the lengths of the vertical bars are determined by the profile's placement.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

OPEN PROFILE TYPE (Discriminator)

Business Rules

- An OPEN FEATURE SWEEP PROFILE/142 is a type of FEATURE SWEEP PROFILE/136.
- An OPEN FEATURE SWEEP PROFILE/142 must be either a CIRCULAR ARC FEATURE SWEEP PROFILE/143, a ROUNDED-U FEATURE SWEEP PROFILE/144, a VEE FEATURE SWEEP PROFILE/145, a SQUARE-U FEATURE SWEEP PROFILE/146, a TEE FEATURE SWEEP PROFILE/147, an ELL FEATURE SWEEP PROFILE/148, a LINE PLUS RADIUS FEATURE SWEEP PROFILE/184, a HALF OBROUND FEATURE SWEEP PROFILE/185, or an OTHER OPEN FEATURE SWEEP PROFILE/169.
- An OPEN FEATURE SWEEP PROFILE/142 defines the profile for 0, 1, or many ALONG FEATURE SWEEPs/121.

EXPRESS Declaration

```
ENTITY open_feature_sweep_profile
SUPERTYPE OF (circular_arc_feature_sweep_profile OR
              rounded_u_feature_sweep_profile OR
              vee_feature_sweep_profile OR
              square_u_feature_sweep_profile OR
              tee_feature_sweep_profile OR
              ell_feature_sweep_profile OR
              line_plus_radius_feature_sweep_profile OR
              half_obround_feature_sweep_profile OR
              other_open_feature_sweep_profile)
SUBTYPE OF (feature_sweep_profile);
END_ENTITY;
```

Entity Name: **CIRCULAR ARC FEATURE SWEEP PROFILE**

Entity Number: **143**

An OPEN FEATURE SWEEP PROFILE/142 consisting of a circular arc.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

DIM ID (FK)
The dimension (radius or diameter) of the arc.

Business Rules

- A CIRCULAR ARC FEATURE SWEEP PROFILE/143 is a type of OPEN FEATURE SWEEP PROFILE/142.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is dimension of 0, 1, or many CIRCULAR ARC FEATURE SWEEP PROFILEs/143.
- The center of the defining circle lies on the profile's local AB-origin. The midpoint of the arc is on the negative B-axis.

EXPRESS Declaration

```
ENTITY circular_arc_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  arc_size : independent_size_parameter;
END_ENTITY;
```

Entity Name: **ROUNDED-U FEATURE SWEEP PROFILE**

Entity Number: **144**

An OPEN FEATURE SWEEP PROFILE/142 consisting of two parallel semi-infinite lines connected at their fixed ends by a semi-circle to form a "U" shape. The center of the semi-circle lies on the profile's local origin, and the parallel lines are parallel to the B-axis and in the positive B half-plane.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

DIM ID (FK)
The width (full or half) dimension of the profile.

Business Rules

- A ROUNDED-U FEATURE SWEEP PROFILE/144 is a type of OPEN FEATURE SWEEP PROFILE/142.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is width dimension of 0, 1, or many ROUNDED-U FEATURE SWEEP PROFILEs/144.
- The center of the semi-circle lies on the profile's local origin, and the parallel lines are parallel to the B-axis and lie in the positive B half-plane.

EXPRESS Declaration

```
ENTITY rounded_u_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  sweep_width : independent_size_parameter;
END_ENTITY;
```

Entity Name: VEE FEATURE SWEEP PROFILE

Entity Number: 145

An OPEN FEATURE SWEEP PROFILE/142 in the shape of a "V".

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

ANGLE.DIM ID (FK)
The full angle or semi-angle of the "V".

Business Rules

- A VEE FEATURE SWEEP PROFILE/145 is a type of OPEN FEATURE SWEEP PROFILE/142.
- A VEE FEATURE SWEEP PROFILE/145 is blended by 0 or 1 VEE BLENDs/172.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is angle dimension of 0, 1, or many VEE FEATURE SWEEP PROFILEs/145.
- Full angle < 180.
- The profile is symmetric with respect to the positive B-axis, with the point of the "V" at the origin.

EXPRESS Declaration

```
ENTITY vee_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  vee_angle : independent_size_parameter;
  blend     : OPTIONAL implicit_edge_blend;
WHERE
  vee_angle.dimension < 180;
END_ENTITY;
```

Entity Name: **SQUARE-U FEATURE SWEEP PROFILE**

Entity Number: **146**

An OPEN FEATURE SWEEP PROFILE/142 consisting of two parallel semi-infinite lines connected at their fixed ends by a line segment perpenticular to both.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

DIM ID (FK)
The full width or semi-width of the profile.

Business Rules

- A SQUARE-U FEATURE SWEEP PROFILE/146 is a type of OPEN FEATURE SWEEP PROFILE/142.
- A SQUARE-U FEATURE SWEEP PROFILE/146 is blended by 0 or 1 SQUARE-U BLEND1s/170.
- A SQUARE-U FEATURE SWEEP PROFILE/146 is blended by 0 or 1 SQUARE-U BLEND2s/171.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the width dimension of 0, 1, or many SQUARE-U FEATURE SWEEP PROFILEs/146.
- The profile lies in the positive B half-plane with its line segment on the A-axis. The midpoint of the segment is at the origin.

EXPRESS Declaration

```
ENTITY square_u_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  sweep_width  : independent_size_parameter;
  blend1       : OPTIONAL implicit_edge_blend;
  blend2       : OPTIONAL implicit_edge_blend;
END_ENTITY;
```

Entity Name: **TEE FEATURE SWEEP PROFILE**

Entity Number: **147**

An OPEN FEATURE SWEEP PROFILE/142 in the shape of a "T". It consists of a "stem" and a perpendicular "crossbar". The stem is represented by two parallel semi-infinite lines connected at their fixed ends by the crossbar. The crossbar is a rectangle. The profile is symmetric about the centerline of the stem.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

STEM WIDTH.DIM ID (FK)
The width or half-width dimension of the stem.

CROSSBAR WIDTH.DIM ID (FK)
The width or half-width dimension of the crossbar.

CROSSBAR HEIGHT.DIM ID (FK)
The height or half-height dimension of the stem.

Business Rules

- A TEE FEATURE SWEEP PROFILE/147 is a type of OPEN FEATURE SWEEP PROFILE/142.
- CROSSBAR WIDTH > STEM WIDTH.
- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE STEM/CROSSBAR BLEND1s/173.
- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE STEM/CROSSBAR BLEND2s/174.

- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE CROSSBAR BLEND1s/175.
- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE CROSSBAR BLEND2s/176.
- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE CROSSBAR BLEND3s/177.
- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE CROSSBAR BLEND4s/178.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the stem width dimension of 0, 1, or many TEE FEATURE SWEEP PROFILEs/147.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the crossbar width dimension of 0, 1, or many TEE FEATURE SWEEP PROFILEs/147.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the crossbar height dimension of 0, 1, or many TEE FEATURE SWEEP PROFILEs/147.
- The side of the crossbar opposite the stem lies on the profile's A-axis, the rest of the profile is in the positive B half-plane, and the profile is symmetric with respect to the local B-axis.

EXPRESS Declaration

```
ENTITY tee_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  stem_width           : independent_size_parameter;
  crossbar_width       : independent_size_parameter;
  crossbar_height      : independent_size_parameter;
  stem_crossbar_blend1 : OPTIONAL implicit_edge_blend;
  stem_crossbar_blend2 : OPTIONAL implicit_edge_blend;
  crossbar_blend1      : OPTIONAL implicit_edge_blend;
  crossbar_blend2      : OPTIONAL implicit_edge_blend;
  crossbar_blend3      : OPTIONAL implicit_edge_blend;
  crossbar_blend4      : OPTIONAL implicit_edge_blend;
WHERE
  crossbar_width.dimension > stem_width.dimension;
END_ENTITY;
```

Entity Name: **ELL FEATURE SWEEP PROFILE**

Entity Number: **148**

An OPEN FEATURE SWEEP PROFILE/142 in the shape of an "L". It consists of a "stem" and a perpendicular "endbar". The stem is represented by two parallel semi-infinite lines connected at their fixed ends by the endbar. The endbar is a rectangle.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

STEM WIDTH.DIM ID (FK)
The width or half-width of the stem.

ENDBAR WIDTH.DIM ID (FK)
The width or half-width of the endbar.

ENDBAR HEIGHT.DIM ID (FK)
The height or half-height of the stem.

ELL PROFILE ORIENTATION
An indication of whether the endbar is primarily in the positive or negative A half-plane.

Business Rules

- An ELL FEATURE SWEEP PROFILE/148 is a type of OPEN FEATURE SWEEP PROFILE/142.
- ENDBAR WIDTH > STEM WIDTH.
- An ELL FEATURE SWEEP PROFILE/148 is blended by 0 or 1 ELL STEM/ENDBAR BLENDs/179.
- An ELL FEATURE SWEEP PROFILE/148 is blended by 0 or 1 ELL ENDBAR BLEND1s/180.
- An ELL FEATURE SWEEP PROFILE/148 is blended by 0 or 1 ELL ENDBAR BLEND2s/181.
- An ELL FEATURE SWEEP PROFILE/148 is blended by 0 or 1 ELL ENDBAR BLEND3s/182.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the stem width dimension of 0, 1, or many ELL FEATURE SWEEP PROFILEs/148.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the endbar width dimension of 0, 1, or many ELL FEATURE SWEEP PROFILEs/148.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the endbar height dimension of 0, 1, or many ELL FEATURE SWEEP PROFILEs/148.
- The endbar is in the positive B half-plane with the side opposite the stem lying on the A-axis. The stem is symmetric with respect to the B-axis. The ELL PROFILE ORIENTATION attribute indicates whether the endbar is primarily in the positive or negative A half-plane.

EXPRESS Declaration

```
ENTITY ell_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  stem_width         : independent_size_parameter;
  endbar_width       : independent_size_parameter;
  endbar_height      : independent_size_parameter;
  ell_orientation    : ell_orientation_types;
  stem_endbar_blend  : OPTIONAL implicit_edge_blend;
  endbar_blend1      : OPTIONAL implicit_edge_blend;
  endbar_blend2      : OPTIONAL implicit_edge_blend;
  endbar_blend3      : OPTIONAL implicit_edge_blend;
WHERE
  endbar_width.dimension > stem_width.dimension;
END_ENTITY;

TYPE
ell_orientation_types = ENUMERATION OF (ell_positive, ell_negative);
END_TYPE;
```

Entity Name: CONSTANT DIAMETER AXISYMMETRIC FEATURE SWEEP

Entity Number: 149

An AXISYMMETRIC FEATURE SWEEP/122 whose profile consists of a line parallel to the local profile B-axis. After equating the local B-axis with the feature's Z-axis, the line is rotated about the Z-axis to form a cylinder.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

DIM ID (FK)
The size dimension (diameter or radius) of the sweep.

Business Rules

- A CONSTANT DIAMETER AXISYMMETRIC FEATURE SWEEP/149 is a type of AXISYMMETRIC FEATURE SWEEP/122.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the size dimension of 0, 1, or many CONSTANT DIAMETER AXISYMMETRIC FEATURE SWEEPs/149.

EXPRESS Declaration

```
ENTITY constant_diameter_axisymmetric_feature_sweep
SUBTYPE OF (axisymmetric_feature_sweep);
  sweep_size : independent_size_parameter;
END_ENTITY;
```

Entity Name: **TAPERED AXISYMMETRIC FEATURE SWEEP**

Entity Number: **150**

An AXISYMMETRIC FEATURE SWEEP/122 whose profile consists of a half-line (the half in the $+B$ half-plane) which is not parallel to the local B-axis.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

SIZE AT ORIGIN.DIM ID (FK)
The distance (or twice the distance) from the AB-origin of the implicitly defined profile to the profile, measured along the positive A-axis. This value translates to the radius or diameter of the feature's cross-section in the local XY-plane.

ANGLE.DIM ID (FK)
The full- or semi-angle dimension of the cone formed by the sweep.

SENSE
An indication of whether the cone's diameter increases or decreases as local B increases. Values are "INCREASES" and "DECREASES".

Business Rules

- A TAPERED AXISYMMETRIC FEATURE SWEEP/150 is a type of AXISYMMETRIC FEATURE SWEEP/122.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the size at origin of 0, 1, or many TAPERED AXISYMMETRIC FEATURE SWEEPs/150.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is the angle dimension of 0, 1, or many TAPERED AXISYMMETRIC FEATURE SWEEPs/150.

EXPRESS Declaration

```
ENTITY tapered_axisymmetric_feature_sweep
SUBTYPE OF (axisymmetric_feature_sweep);
  size_at_origin : independent_size_parameter;
  angle_dim      : derivable_size_parameter;
  taper_type     : taper_types;
WHERE
  size_at_origin.dimension >= 0;
  angle_dim.dimension < 90;
END_ENTITY;

TYPE
taper_types = ENUMERATION OF (taper_increasing, taper_decreasing);
END_TYPE;
```

Entity Name: **OTHER AXISYMMETRIC FEATURE SWEEP**

Entity Number: **151**

An AXISYMMETRIC FEATURE SWEEP/122 whose profile is defined by a GENERAL PROFILE/162.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

GENERAL PROFILE ID (FK)

Business Rules

- An OTHER AXISYMMETRIC FEATURE SWEEP/151 is a type of AXISYMMETRIC FEATURE SWEEP/122.
- A GENERAL PROFILE/162 is used by 0, 1, or many OTHER AXISYMMETRIC FEATURE SWEEPs/151.

EXPRESS Declaration

```
ENTITY other_axisymmetric_feature_sweep
SUBTYPE OF (axisymmetric_feature_sweep);
```

```
  sweep_profile : general_profile;
END_ENTITY;
```

Entity Name: **PASSAGE BLEND**

Entity Number: **152**

A blend at one of the openings of an IMPLICIT PASSAGE/003.

Primary Key Attributes

SHAPE ID (FK)
  With the two following attributes, identifies the IMPLICIT PASSAGE/003.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

BLENDED PASSAGE END
  Used to distinguish blends of the same IMPLICIT PASSAGE/003. Valid values are 'ENTRY' and 'EXIT'.

Other Attributes

BLEND.SHAPE ID (FK)
  With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

BLEND.SHAPE ELEMENT ID (FK)

BLEND.IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An IMPLICIT PASSAGE/003 is blended by 0, 1, or 2 PASSAGE BLENDs/152.
- An IMPLICIT EDGE BLEND/308 is used as 0,1, or many PASSAGE BLENDs/152.
- For a swept passage, the defining FEATURE SWEEP/120 provides the directionality needed for 'ENTRY' and 'EXIT' to make sense. For a sweep with a LINEAR FEATURE SWEEP PATH/125, the ENTRY end of the feature is at $z = 0$. The same holds for an AXISYMMETRIC FEATURE SWEEP/122. For a sweep with a PARTIAL CIRCULAR FEATURE SWEEP PATH/127, the ENTRY end of the feature is in the positive XZ quadrant.

EXPRESS Declaration

```
ENTITY passage_boundary_blend;
  blended_end : feature_end_types;
  end_blend   : implicit_edge_blend;
END_ENTITY;

TYPE
feature_end_types = ENUMERATION OF (initial_end, terminal_end);
END_TYPE;
```

Entity Name: PROTRUSION BLEND

Entity Number: 153

A blend at the boundary between an IMPLICIT PROTRUSION/004 and its surrounding geometry.

Primary Key Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT PROTRUSION/004 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

ROUND.SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE ROUND/037 that specifies the blend.

ROUND.SHAPE ELEMENT ID (FK)

ROUND.IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An IMPLICIT PROTRUSION/004 is blended by 0 or 1 PROTRUSION BLENDs/153.
- An IMPLICIT EDGE ROUND/037 is used as 0,1, or many PROTRUSION BLENDs/153.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under IMPLICIT PROTRUSION/004.)

Entity Name: **DEPRESSION BLEND**

Entity Number: **154**

A blend at the opening of an IMPLICIT DEPRESSION/005.

Primary Key Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT DEPRESSION/005.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

BLEND.SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

BLEND.SHAPE ELEMENT ID (FK)

BLEND.IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An IMPLICIT DEPRESSION/005 is blended by 0 or 1 DEPRESSION BLENDs/154.
- An IMPLICIT EDGE BLEND/308 is used as 0,1, or many DEPRESSION BLENDs/154.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under IMPLICIT DEPRESSION/005.)

Entity Name: **ALONG FEATURE SWEEP END**

Entity Number: **158**

The end shape of an IMPLICIT FORM FEATURE/002 defined by an ALONG FEATURE SWEEP/121. It may only be used when the end of the sweep is not at the air/material boundary. As an example, this entity can be used to model the rounded end of a milled slot.

Primary Key Attributes

FEATURE VOLUME ID (FK)

ALONG SWEEP END

Used to distinguish ends of the same ALONG FEATURE SWEEP/121. Valid values are 'INITIAL' and 'TERMINAL'.

Other Attributes

ALONG SWEEP END TYPE (Discriminator)

Business Rules

- An ALONG FEATURE SWEEP/121 ends with 0, 1, or 2 ALONG FEATURE SWEEP ENDs/158.
- An ALONG FEATURE SWEEP END/130 may be either an ALONG FEATURE SWEEP FLAT END/159 or an ALONG FEATURE SWEEP RADIUSED END/160.
- When the sweep does not end at the air/material boundary and no ALONG FEATURE SWEEP END/ 158 is specified, the feature is assumed to have a flat end.

NOTES

- Flat ends must be modeled explicitly when the edge between the end and the rest of the feature is blended.

EXPRESS Declaration

```
ENTITY along_feature_sweep_end
SUPERTYPE OF (along_feature_sweep_flat_end OR
              along_feature_sweep_radiused_end);
  sweep_end : feature_end_types;
END_ENTITY;

TYPE
feature_end_types = ENUMERATION OF (initial_end, terminal_end);
END_TYPE;
```

Entity Name: **ALONG FEATURE SWEEP FLAT END**

Entity Number: **159**

An indication that an ALONG FEATURE SWEEP END/158 is planar.

Primary Key Attributes

FEATURE VOLUME ID (FK)

ALONG SWEEP END (FK)

Whether this is the INITIAL or TERMINAL end of the sweep.

Business Rules

- An ALONG FEATURE SWEEP FLAT END/159 is a type of ALONG FEATURE SWEEP END/158.
- An ALONG FEATURE SWEEP FLAT END/159 is blended by 0 or 1 ALONG FEATURE SWEEP FLAT END BLENDs/161.

EXPRESS Declaration

```
ENTITY along_feature_sweep_flat_end
SUBTYPE OF (along_feature_sweep_end);
  end_blend : OPTIONAL implicit_edge_blend;
END_ENTITY;
```

Entity Name: ALONG FEATURE SWEEP RADIUSED END

Entity Number: 160

An indication that an ALONG FEATURE SWEEP END/158 is defined by a radius. The shape of the end depends on the type of OPEN FEATURE SWEEP PROFILE/142 used.

For all profiles except the ELL FEATURE SWEEP PROFILE/148, the LINE PLUS RADIUS FEATURE SWEEP PROFILE/184 and the HALF OBROUND FEATURE SWEEP PROFILE/185, the shape can be thought of as the result of sweeping the portion of the profile in its positive A half-plane 180 degrees about the local B-axis. For example, the end shape for a TEE FEATURE SWEEP PROFILE/147 consists of two concentric half-cylinders. For an ELL FEATURE SWEEP PROFILE/148, the end shape consists of a semi-infinite cylinder with a radius equal to half the stem width, and another cylinder with a radius equal to half the endbar width. For a LINE PLUS RADIUS FEATURE SWEEP PROFILE/184 and a HALF OBROUND FEATURE SWEEP PROFILE/185, the end shape is obtained by sweeping the defining arcs 90 degrees about a line parallel to thier local B-axes at the arc/line vertices, and sweeping the rest of the profile (i.e. the part above the defining line) a distance equal to the arc radius.

Primary Key Attributes

FEATURE VOLUME ID (FK)

ALONG SWEEP END (FK)

Business Rules

- An ALONG FEATURE SWEEP RADIUSED END/160 is a type of ALONG FEATURE SWEEP END/158.

EXPRESS Declaration

```
ENTITY along_feature_sweep_radiused_end
SUBTYPE OF (along_feature_sweep_end);
END_ENTITY;
```

Entity Name: ALONG FEATURE SWEEP FLAT END BLEND

Entity Number: 161

A blend (round or flat) between an end of a feature defined by an ALONG FEATURE SWEEP/121 and the rest of the feature.

Primary Key Attributes

FEATURE VOLUME ID (FK)

ALONG SWEEP END (FK)

Other Attributes

SHAPE ID (FK)
: With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An ALONG FEATURE SWEEP FLAT END/159 is blended by 0 or 1 ALONG FEATURE SWEEP FLAT END BLENDs/161.
- An IMPLICIT EDGE BLEND/308 is used as 0,1, or many ALONG FEATURE SWEEP FLAT END BLENDs/161.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under ALONG FEATURE SWEEP FLAT END/159.)

Entity Name: **GENERAL PROFILE**

Entity Number: **162**

A sequence of non-overlapping curve segments in a plane, connected end to end. It is modeled by a first curve and some number of 'curve,next-curve' pairs.

Primary Key Attributes

GENERAL PROFILE ID
An arbitrary label uniquely identifying each general profile.

Other Attributes

GEOMETRIC ENTITY ID (FK)
The ID of the first curve of the profile.

GENERAL PROFILE TYPE (Discriminator)

Business Rules

- A GENERAL PROFILE/162 must be either an OPEN GENERAL PROFILE/163 or a CLOSED GENERAL PROFILE/164.
- A CURVE/GEO-6 is first curve of 0, 1, or many GENERAL PROFILEs/162.
- A GENERAL PROFILE/162 has curves after first given by 0, 1, or many PROFILE PAIRs/165.
- A GENERAL PROFILE/162 is used by 0, 1, or many OTHER AXISYMMETRIC FEATURE SWEEPs/151.

EXPRESS Declaration

```
ENTITY general_profile
SUPERTYPE OF (open_general_profile OR
              closed_general_profile);
  first_curve   : curve;
  element_pairs : SET [0:#] OF profile_pair;
WHERE
  sequenced(first_curve,element_pairs);
```

```
END_ENTITY;

TYPE
set_of_profile_pairs = set [0:#] of profile_pair;
END_TYPE;

FUNCTION sequenced(initial:curve;subsequent:set_of_profile_pairs):LOGICAL;
-- determines if a GENERAL PROFILE is made up of correctly ordered
-- initial curve and subsequent curve pairs
END_FUNCTION;
```

Entity Name: OPEN GENERAL PROFILE

Entity Number: 163

A GENERAL PROFILE/162 that does not enclose an area of the plane in which it exists.

Primary Key Attributes

GENERAL PROFILE ID (FK)

Business Rules

- An OPEN GENERAL PROFILE/163 is a type of GENERAL PROFILE/162.
- An OPEN GENERAL PROFILE/163 is used as 0 or 1 OTHER OPEN FEATURE SWEEP PROFILEs/169.

EXPRESS Declaration

```
ENTITY open_general_profile
SUBTYPE OF (general_profile);
WHERE
  NOT closed(element_pairs);
  planar(element_pairs);
END_ENTITY;

TYPE
set_of_profile_pairs = set [0:#] of profile_pair;
END_TYPE;

FUNCTION planar(pairs:set_of_profile_pairs):LOGICAL;
-- determines if a curve string is planar
```

```
END_FUNCTION;

FUNCTION closed(pairs:set_of_profile_pairs):LOGICAL;
-- determines if a curve string is closed
END_FUNCTION;
```

Entity Name: **CLOSED GENERAL PROFILE**

Entity Number: **164**

A GENERAL PROFILE/162 that encloses an area of the plane in which it exists.

Primary Key Attributes

GENERAL PROFILE ID (FK)

Business Rules

- A CLOSED GENERAL PROFILE/164 is a type of GENERAL PROFILE/162.
- A CLOSED GENERAL PROFILE/164 is used as 0 or 1 OTHER CLOSED FEATURE SWEEP PROFILEs/168.

EXPRESS Declaration

```
ENTITY closed_general_profile
SUBTYPE OF (general_profile);
WHERE
    closed(element_pairs);
    planar(element_pairs);
END_ENTITY;

TYPE
set_of_profile_pairs = set [0:#] of profile_pair;
END_TYPE;

FUNCTION planar(pairs:set_of_profile_pairs):LOGICAL;
-- determines if a curve string is planar
END_FUNCTION;

FUNCTION closed(pairs:set_of_profile_pairs):LOGICAL;
-- determines if a curve string is closed
END_FUNCTION;
```

Entity Name: PROFILE PAIR

Entity Number: 165

A predecessor/successor relationship between two CURVEs/GEO-6 of a GENERAL PROFILE/162. The sequence of CURVEs/GEO-6 that constitutes the profile is modeled by an initial curve plus some number of these.

Primary Key Attributes

GENERAL PROFILE ID (FK) (AK1)
PREDECESSOR.GEOMETRIC ENTITY ID (FK)

Other Attributes

SUCCESSOR.GEOMETRIC ENTITY ID (FK) (AK1)

Business_Rules

- A GENERAL PROFILE/162 has curves after first given by 0, 1, or many PROFILE PAIRs/165.
- A CURVE/GEO-6 is predecessor in 0, 1, or many PROFILE PAIRs/165.
- A CURVE/GEO-6 is successor in 0, 1, or many PROFILE PAIRs/165.
- A PROFILE PAIR/165 is blended by 0 or 1 PROFILE PAIR BLENDs/166.

NOTES

- The key and alternate key attributes above assert, in effect, that a given CURVE/GEO-6 can appear at most once in a given GENERAL PROFILE/162.

EXPRESS Declaration

```
ENTITY profile_pair;
  predecessor_curve : curve;
  successor_curve   : curve;
  blend             : OPTIONAL implicit_edge_blend;
END_ENTITY;
```

Entity Name: PROFILE PAIR BLEND

Entity Number: 166

An indication that the implicit edge defined by a PROFILE PAIR 165 is blended by an IMPLICIT EDGE BLEND/308.

Primary Key Attributes

GENERAL PROFILE ID (FK)

GEOMETRIC ENTITY ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A PROFILE PAIR/165 is blended by 0 or 1 PROFILE PAIR BLENDs/166.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many PROFILE PAIR BLENDs/166.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under PROFILE PAIR/165.)

Entity Name: **STANDARD CLOSED FEATURE SWEEP PROFILE**

Entity Number: **167**

A common, "standard" closed profile used to define swept features.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

STANDARD CLOSED PROFILE TYPE (Discriminator)

Business Rules

- A STANDARD CLOSED FEATURE SWEEP PROFILE/167 is a type of CLOSED FEATURE SWEEP PROFILE/137.
- A STANDARD CLOSED FEATURE SWEEP PROFILE/167 may be either a RECTANGULAR FEATURE SWEEP PROFILE/139 or an N-GON FEATURE SWEEP PROFILE/140.
- A STANDARD CLOSED FEATURE SWEEP PROFILE/167 is blended by 0 or 1 STANDARD CLOSED FEATURE SWEEP PROFILE BLENDs/141.

EXPRESS Declaration

```
ENTITY standard_closed_feature_sweep_profile
SUPERTYPE OF (rectangular_feature_sweep_profile OR
              ngon_feature_sweep_profile)
SUBTYPE OF (closed_feature_sweep_profile);
  blend : OPTIONAL implicit_edge_blend;
END_ENTITY;
```

Entity Name: **OTHER CLOSED FEATURE SWEEP PROFILE**

Entity Number: **168**

The use of a CLOSED GENERAL PROFILE/164 as a CLOSED FEATURE SWEEP PROFILE/137.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

GENERAL PROFILE ID (FK)

Business Rules

- An OTHER CLOSED FEATURE SWEEP PROFILE/168 is a type of CLOSED FEATURE SWEEP PROFILE/137.
- A CLOSED GENERAL PROFILE/164 is used as 0 or 1 OTHER CLOSED FEATURE SWEEP PROFILEs/168.

EXPRESS Declaration

```
ENTITY other_closed_feature_sweep_profile
 SUBTYPE OF (closed_feature_sweep_profile);
   sweep_profile : closed_general_profile;
 END_ENTITY;
```

Entity Name: **OTHER OPEN FEATURE SWEEP PROFILE**

Entity Number: **169**

The use of an OPEN GENERAL PROFILE/163 as a OPEN FEATURE SWEEP PROFILE/142.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

GENERAL PROFILE ID (FK)

Business Rules

- An OTHER OPEN FEATURE SWEEP PROFILE/169 is a type of OPEN FEATURE SWEEP PROFILE/142.
- An OPEN GENERAL PROFILE/163 is used as 0 or 1 OTHER OPEN FEATURE SWEEP PROFILEs/169.

EXPRESS Declaration

```
ENTITY other_open_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  sweep_profile : open_general_profile;
END_ENTITY;
```

Entity Name: **SQUARE-U BLEND1**

Entity Number: **170**

A blend between the bottom and side of a SQUARE-U FEATURE SWEEP PROFILE/146. The blend applies to the side of the profile in the negative A half-plane.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A SQUARE-U FEATURE SWEEP PROFILE/146 is blended by 0 or 1 SQUARE-U BLEND1s/170.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many SQUARE-U BLEND1s/170.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under SQUARE-U FEATURE SWEEP PROFILE/146.)

Entity Name: **SQUARE-U BLEND2**

Entity Number: **171**

A blend between the bottom and side of a SQUARE-U FEATURE SWEEP PROFILE/146. The blend applies to the side of the profile in the positive A half-plane.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A SQUARE-U FEATURE SWEEP PROFILE/146 is blended by 0 or 1 SQUARE-U BLEND2s/171.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many SQUARE-U BLEND2s/171.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under SQUARE-U FEATURE SWEEP PROFILE/146.)

Entity Name: **VEE BLEND**

Entity Number: **172**

A blend at the tip of a VEE FEATURE SWEEP PROFILE/145.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A VEE FEATURE SWEEP PROFILE/145 is blended by 0 or 1 VEE BLENDs/172.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many VEE BLENDs/172.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under VEE FEATURE SWEEP PROFILE/145.)

Entity Name: **TEE STEM/CROSSBAR BLEND1**

Entity Number: **173**

A blend between the stem and crossbar of a TEE FEATURE SWEEP PROFILE/147. The blend applies to the side of the profile in the negative *A* half-plane.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE STEM/CROSSBAR BLEND1s/173.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE STEM/CROSSBAR BLEND1s/173.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under TEE FEATURE SWEEP PROFILE/147.)

Entity Name: **TEE STEM/CROSSBAR BLEND2**

Entity Number: **174**

A blend between the stem and crossbar of a TEE FEATURE SWEEP PROFILE/147. The blend applies to the side of the profile in the positive A half-plane.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE STEM/CROSSBAR BLEND2s/174.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE STEM/CROSSBAR BLEND2s/174.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under TEE FEATURE SWEEP PROFILE/147.)

Entity Name: **TEE CROSSBAR BLEND1**

Entity Number: **175**

A blend at the crossbar vertex of a TEE FEATURE SWEEP PROFILE/147 which lies in the negative A, positive B quadrant.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE CROSSBAR BLEND1s/175.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE CROSSBAR BLEND1s/175.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under TEE FEATURE SWEEP PROFILE/147.)

Entity Name: **TEE CROSSBAR BLEND2**

Entity Number: **176**

A blend at the crossbar vertex of a TEE FEATURE SWEEP PROFILE/147 which lies on the negative A-axis.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE CROSSBAR BLEND2s/176.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE CROSSBAR BLEND2s/176.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under TEE FEATURE SWEEP PROFILE/147.)

Entity Name: **TEE CROSSBAR BLEND3**

Entity Number: **177**

A blend at the crossbar vertex of a TEE FEATURE SWEEP PROFILE/147 which lies on the positive A-axis.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE CROSSBAR BLEND3s/177.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE CROSSBAR BLEND3s/177.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under TEE FEATURE SWEEP PROFILE/147.)

Entity Name: **TEE CROSSBAR BLEND4**

Entity Number: **178**

A blend at the crossbar vertex of a TEE FEATURE SWEEP PROFILE/147 which lies in the positive A, positive B quadrant.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A TEE FEATURE SWEEP PROFILE/147 is blended by 0 or 1 TEE CROSSBAR BLEND4s/178.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE CROSSBAR BLEND4s/178.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under TEE FEATURE SWEEP PROFILE/147.)

Entity Name: **ELL STEM/ENDBAR BLEND**

Entity Number: **179**

A blend at the stem/endbar vertex of an ELL FEATURE SWEEP PROFILE/148.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An ELL FEATURE SWEEP PROFILE/148 is blended by 0 or 1 ELL STEM/ENDBAR BLENDs/179.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many ELL STEM/ENDBAR BLENDs/179.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under ELL FEATURE SWEEP PROFILE/148.)

Entity Name: **ELL ENDBAR BLEND1**

Entity Number: **180**

A blend at the endbar vertex of an ELL FEATURE SWEEP PROFILE/148 which lies on the negative A-axis.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An ELL FEATURE SWEEP PROFILE/148 is blended by 0 or 1 ELL ENDBAR BLEND1s/180.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many ELL ENDBAR BLEND1s/180.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under ELL FEATURE SWEEP PROFILE/148.)

Entity Name: **ELL ENDBAR BLEND2**

Entity Number: **181**

A blend at the endbar vertex of an ELL FEATURE SWEEP PROFILE/148 which lies on the positive A-axis.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An ELL FEATURE SWEEP PROFILE/148 is blended by 0 or 1 ELL ENDBAR BLEND2s/181.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many ELL ENDBAR BLEND2s/181.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under ELL FEATURE SWEEP PROFILE/148.)

Entity Name: **ELL ENDBAR BLEND3**

Entity Number: **182**

A blend at the endbar vertex of an ELL FEATURE SWEEP PROFILE/148 which lies in the positive A, positive B quadrant.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

SHAPE ID (FK)
With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An ELL FEATURE SWEEP PROFILE/148 is blended by 0 or 1 ELL ENDBAR BLEND3s/182.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many ELL ENDBAR BLEND3s/182.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under ELL FEATURE SWEEP PROFILE/148.)

Entity Name: **LINE PLUS RADIUS FEATURE SWEEP PROFILE**

Entity Number: **184**

An OPEN FEATURE SWEEP PROFILE/142 consisting of a semi-infinite line and a tangent semi-infinite circular arc connected at their fixed ends.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

DIM ID (FK)
The radius (preferably) or diameter dimension of the circular portion of the profile.

Business Rules

- A LINE PLUS RADIUS FEATURE SWEEP PROFILE/184 is a type of OPEN FEATURE SWEEP PROFILE/142.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the size dimension of 0, 1, or many LINE PLUS RADIUS FEATURE SWEEP PROFILEs/184.
- The intersection point of the arc and line is at the profile's local origin. The line coincides with the local A-axis, and the circular arc extends into the positive B half-plane.

EXPRESS Declaration

```
ENTITY line_plus_radius_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  size_dim : independent_size_parameter;
END_ENTITY;
```

Entity Name: HALF OBROUND FEATURE SWEEP PROFILE

Entity Number: 185

An OPEN FEATURE SWEEP PROFILE/142 consisting of two semi-infinite circular arcs connected at their fixed ends by a line segment.

Primary Key Attributes

FEATURE PROFILE ID (FK)

Other Attributes

CIRCLE.DIM ID (FK)
The common radius (preferably) or diameter dimension of the two circular portions of the profile.

LENGTH.DIM ID (FK)
The length (full or half-length) of the linear portion of the profile.

Business Rules

- A HALF OBROUND FEATURE SWEEP PROFILE/185 is a type of OPEN FEATURE SWEEP PROFILE/142.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the circle dimension of 0, 1, or many HALF OBROUND FEATURE SWEEP PROFILEs/185.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is the length dimension of 0, 1, or many HALF OBROUND FEATURE SWEEP PROFILEs/185.
- The line segment is on the local A-axis with its center at the origin. The arcs are tangent to the segment, and extend into the positive B half-plane.

EXPRESS Declaration

```
ENTITY half_obround_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  circle_dim : independent_size_parameter;
  length_dim : independent_size_parameter;
END_ENTITY;
```

Entity Name: PASSAGE INTERMEDIATE BOUND

Entity Number: 186

A bound or limit to the extent of an IMPLICIT PASSAGE/003, not occurring at the passage ends. This indicates pre-existing air (or a pre-existing air-material interface) contained within the volume implied by the implicit specification of the passage. For example, a hole through both flanges of an I-beam is interrupted by the void between them.

Depending on the context, the interruption may be viewed as resulting from an intervening pre-existing volume or an intervening surface. In the I-beam example, if the void is modeled as a swept form feature then the hole would be viewed as being interrupted by a volume. On the other hand, if the flanges were explicitly modeled, then the hole would be seen as having two planar intermediate bounds.

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT PASSAGE/003.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

PASSAGE INTERMEDIATE BOUND ID
Arbitrary label used to distinguish intermediate bounds of the same passage.

Other Attributes

IMPLICIT FEATURE BOUND ID (FK)

PASSAGE INTERRUPTION COMPLETENESS
An indication of whether the interruption is complete or incomplete. A complete interruption effects the entire profile, rather than only part of it.

PASSAGE BOUND TYPE
An indication of whether the bound is seen as two- or three-dimensional. Values are "START" and "END" for two-dimensional bounds, and "VOLUMETRIC" for three-dimensional bounds. For a two-dimensional bound, the direction of the feature as determined by its path definition is used to label the bound as a start or end.

Business Rules

- An IMPLICIT PASSAGE/003 is interrupted by 0, 1, or many PASSAGE INTERMEDIATE BOUNDs/186.
- An IMPLICIT FEATURE BOUND/029 specifies the bound in 0, 1, or many PASSAGE INTERMEDIATE BOUNDs/186.
- A PASSAGE INTERMEDIATE BOUND/186 is blended by 0 or 1 PASSAGE INTERMEDIATE BOUND BLENDs/187.

EXPRESS Declaration

```
ENTITY passage_intermediate_bound;
  passage_bound_type     : bound_types;
  interruption_complete  : LOGICAL;
  specification          : implicit_feature_bound;
  bound_blend            : OPTIONAL implicit_edge_blend;
END_ENTITY;

TYPE
bound_types = ENUMERATION OF (start_bound,
                              end_bound,
                              volumetric_bound);
END_TYPE;
```

Entity Name: **PASSAGE INTERMEDIATE BOUND BLEND**

Entity Number: **187**

A blend of the intersection edges between an IMPLICIT PASSAGE/003 and a PASSAGE INTERMEDIATE BOUND/186

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT PASSAGE/003.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

PASSAGE INTERMEDIATE BOUND ID (FK)

Other Attributes

BLEND.SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

BLEND.SHAPE ELEMENT ID (FK)

BLEND.IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A PASSAGE INTERMEDIATE BOUND/186 is blended by 0 or 1 PASSAGE INTERMEDIATE BOUND BLENDs/187.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many PASSAGE INTERMEDIATE BOUND BLENDs/187.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under PASSAGE INTERMEDIATE BOUND/186.)

Entity Name: **DEPRESSION INTERMEDIATE BOUND**

Entity Number: **188**

A bound or limit to the extent of an IMPLICIT DEPRESSION/005, not occurring at the open end of the depression. DEPRESSION INTERMEDIATE BOUNDs/188 are analogous to PASSAGE INTERMEDIATE BOUNDs/186.

Primary Key Attributes

SHAPE ID (FK)
: With the following two attributes, identifies the IMPLICIT DEPRESSION/005.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

DEPRESSION INTERMEDIATE BOUND ID
: An arbitrary label used to distinguish intermediate bounds of the same depression.

Other Attributes

IMPLICIT FEATURE BOUND ID (FK)

DEPRESSION INTERRUPTION COMPLETENESS
: An indication of whether the interruption is complete or incomplete. A complete interruption effects the entire profile, rather than only part of it.

DEPRESSION BOUND TYPE
: An indication of whether the bound is seen as two- or three-dimensional. Values are "START" and "END" for two-dimensional bounds, and "VOLUMETRIC" for three-dimensional bounds. For a two-dimensional bound, the direction of the feature as determined by its path definition is used to label the bound as a start or end.

Business Rules

- An IMPLICIT DEPRESSION/005 is interrupted by 0, 1, or many DEPRESSION INTERMEDIATE BOUNDs/188.
- An IMPLICIT FEATURE BOUND/029 specifies the bound in 0, 1, or many DEPRESSION INTERMEDIATE BOUNDs/188.
- A DEPRESSION INTERMEDIATE BOUND/188 is blended by 0 or 1 DEPRESSION INTERMEDIATE BOUND BLENDs/189.

EXPRESS Declaration

```
ENTITY depression_intermediate_bound;
  depression_bound_type : bound_types;
  interruption_complete : LOGICAL;
  specification         : implicit_feature_bound;
  bound_blend           : OPTIONAL implicit_edge_blend;
END_ENTITY;

TYPE
bound_types = ENUMERATION OF (start_bound,
                              end_bound,
                              volumetric_bound);
END_TYPE;
```

Entity Name: **DEPRESSION INTERMEDIATE BOUND BLEND**

Entity Number: **189**

A blend of the intersection edges between an IMPLICIT DEPRESSION/005 and a DEPRESSION INTERMEDIATE BOUND/188

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT DEPRESSION/005.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

DEPRESSION INTERMEDIATE BOUND ID (FK)

Other Attributes

BLEND.SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT EDGE BLEND/308 that specifies the blend.

BLEND.SHAPE ELEMENT ID (FK)

BLEND.IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A DEPRESSION INTERMEDIATE BOUND/188 is blended by 0 or 1 DEPRESSION INTERMEDIATE BOUND BLENDs/189.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many DEPRESSION INTERMEDIATE BOUND BLENDs/189.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under DEPRESSION INTERMEDIATE BOUND/188.)

Entity Name: **CONSTANT PROFILE IN/OUT SWEEP**

Entity Number: **190**

An IN/OUT FEATURE SWEEP/123 with a profile of constant size along its path.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Business Rules

- A CONSTANT PROFILE IN/OUT SWEEP/190 is a type of IN/OUT FEATURE SWEEP/123.

EXPRESS Declaration

```
ENTITY constant_profile_in_out_sweep
SUBTYPE OF (in_out_feature_sweep);
END_ENTITY;
```

Entity Name: **TAPERED PROFILE IN/OUT SWEEP**

Entity Number: **192**

A IN/OUT FEATURE SWEEP/123 with sloped sides; i.e., profile size varies linearly over the length of the sweep.

The dimensions given for the swept profile apply at the start of the sweep; i.e., at local $z = 0$. For $0 < z <$ (sweep length), all dimensions are increased or decreased by $z * \tan(\text{ANGLE})$.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

ANGLE.DIM ID
The angle of inclination controlling the profile. This is the semi-angle between sweep centerline and the lines swept by points of the profile or the meeting angle of lines swept by opposing profile points.

TAPER TYPE
Whether the profile size INCREASES or DECREASES as the profile is swept.

Business Rules

- A TAPERED PROFILE IN/OUT SWEEP/192 is a type of IN/OUT FEATURE SWEEP/123.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is angle dimension of 0, 1, or many TAPERED PROFILE IN/OUT SWEEPSs/192.
- Semi-angle < 90.

EXPRESS Declaration

```
 ENTITY tapered_profile_in_out_sweep
SUBTYPE OF (in_out_feature_sweep);
  semi_angle : independent_size_parameter;
  taper_type : taper_types;
WHERE
    (-90 <=  semi_angle.dimension <=  +90);
END_ENTITY;

TYPE
taper_types = ENUMERATION OF (taper_increasing, taper_decreasing);
```

```
END_TYPE;
```

Entity Name: **CONTOURED PROFILE IN/OUT SWEEP**

Entity Number: **193**

An IN/OUT FEATURE SWEEP/123 whose profile size varies according to a scaling curve.

The curve $f(u)$ must be parameterized over $0 \leq u \leq 1$ with value $f(0) = 1$, and $f(u) > 0$ elsewhere. For any local z from 0 through the LENGTH of the linear path of the IN/OUT FEATURE SWEEP/123, the scaling factor is $f(z/LENGTH)$. The value indicates the size of a cross-section of the feature relative to its defined profile.

In cases where the scaling curve is linear, it is recommended that a TAPERED PROFILE IN/OUT SWEEP/192 be used instead.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Key of the BOUNDED CURVE/GEO-22 which acts as the scaling curve.

Business Rules

- A CONTOURED PROFILE IN/OUT SWEEP/193 is a type of IN/OUT FEATURE SWEEP/123.
- A BOUNDED CURVE/GEO-22 is scaling curve of 0, 1, or many CONTOURED PROFILE IN/OUT SWEEPs/193.

EXPRESS Declaration

```
ENTITY contoured_profile_in_out_sweep
SUBTYPE OF (in_out_feature_sweep);
  scaling_curve : bounded_curve;
END_ENTITY;
```

Entity Name: **FORM FEATURE REFLECTION**

Entity Number: **302**

An indication that a REPLICATE FORM FEATURE/067 is a reflection (mirror image) as well as a relocation of the feature of which it is a "copy". The reflection is performed with respect to a specified coordinate plane of the local coordinate system.

Primary Key Attributes

SHAPE ID (FK)
With the following attribute, the identification of the REPLICATE FORM FEATURE/067 that is reflected.

SHAPE ELEMENT ID (FK)

Other Attributes

REFLECTION COORDINATE
The coordinate (x, y, or z) that is negated in the reflection. For example, if this value is "z", the reflection is with respect to the local XY-plane, taking z into $-z$.

Business Rules

- A REPLICATE FORM FEATURE/067 is mirrored by 0 or 1 FORM FEATURE REFLECTIONs/302.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under REPLICATE FORM FEATURE/067.)

Entity Name: **PARALLEL ARRAY PATTERN OFFSET MEMBER**

Entity Number: **305**

An indication that a member feature of a PARALLEL EQUAL SPACING ARRAY PATTERN/056 is located elsewhere than indicated by the pattern rule.

Primary Key Attributes

SHAPE ID (FK)
With the following attribute, identifies the PARALLEL EQUAL SPACING ARRAY PATTERN from which this member is offset.

SHAPE ELEMENT ID (FK)

PARALLEL OFFSET ROW

PARALLEL OFFSET COLUMN

Other Attributes

GEOMETRIC ENTITY ID (FK)
The identification of the vector which gives the offset location of the member as a delta from its pattern-indicated location.

Business Rules

- A PARALLEL EQUAL SPACING ARRAY PATTERN/056 has 0, 1, or many PARALLEL ARRAY PATTERN OFFSET MEMBERs/305.
- A VECTOR WITH MAGNITUDE/GEO-17 gives offset of 0, 1, or many PARALLEL ARRAY PATTERN OFFSET MEMBERs/305.

EXPRESS Declaration

```
ENTITY parallel_array_pattern_offset_member;
  row    : INTEGER;
  column : INTEGER;
  offset : vector_with_magnitude;
WHERE
  ((row >= 1) OR (column >= 1));
END_ENTITY;
```

Entity Name: **ARRAY PATTERN OMISSION**

Entity Number: **306**

An indication that one feature of a rows-and-columns array of features is absent.

Primary Key Attributes

SHAPE ID (FK)
With the following attribute, identifies the IMPLICIT ARRAY FORM FEATURE PATTERN/054 from which this member is offset.

SHAPE ELEMENT ID (FK)

OMISSION ROW NUMBER
With the following attribute, identifies the omitted member.

OMISSION COLUMN NUMBER

Business Rules

- An IMPLICIT ARRAY FORM FEATURE PATTERN/054 has members omitted by 0, 1, or many ARRAY PATTERN OMISSIONs/306.
- ROW NUMBER ≥ 1
- COLUMN NUMBER ≥ 1
- ROW NUMBER #= 1 or COLUMN NUMBER #= 1.
- ROW NUMBER may not exceed the NUMBER OF ROWS of the parent IMPLICIT ARRAY FORM FEATURE PATTERN/054.
- COLUMN NUMBER may not exceed the NUMBER OF COLUMNS of the parent IMPLICIT ARRAY FORM FEATURE PATTERN/054.

EXPRESS Declaration

```
ENTITY array_pattern_omission;
  row    : INTEGER;
  column : INTEGER;
WHERE
  row >= 1;
  column >= 1;
  (row <> 1) or (column <> 1);
END_ENTITY;
```

Entity Name: PARAMETRIC ARRAY PATTERN OFFSET MEMBER

Entity Number: 307

An indication that a member feature of a PARAMETRIC EQUAL SPACING ARRAY PATTERN/055 is located elsewhere than indicated by the pattern rule.

The orientation of the offset member remains constant, with respect to the layout surface of the parent pattern, during offsetting; hence the global orientation mat change.

Primary Key Attributes

SHAPE ID (FK)
: With the following attribute, identifies the PARAMETRIC EQUAL SPACING ARRAY PATTERN/055 from which this member is offset.

SHAPE ELEMENT ID (FK)

PARAMETRIC OFFSET ROW
: With the following attribute, identifies the offset member of the pattern.

PARAMETRIC OFFSET COLUMN

Other Attributes

UOFFSET
: The u-difference (in parametric space) from the rule-indicated location to the actual location.

VOFFSET
: The v-difference (in parametric space) from the rule-indicated location to the actual location.

Business Rules

- A PARAMETRIC EQUAL SPACING ARRAY PATTERN/055 has 0, 1, or many PARAMETRIC ARRAY PATTERN OFFSET MEMBERs/307.

EXPRESS Declaration

```
ENTITY parametric_array_pattern_offset_member;
  row      : INTEGER;
  column   : INTEGER;
  u_offset : REAL;
  v_offset : REAL;
WHERE
  ((row >=1) OR (column >= 1));
END_ENTITY;
```

Entity Name: **IMPLICIT EDGE BLEND**

Entity Number: **308**

An implicit representation of a blend or transition between two adjacent surface areas of a shape.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

IMPLICIT EDGE BLEND TYPE (Discriminator)

Business Rules

- An IMPLICIT EDGE BLEND/308 is a type of IMPLICIT TRANSITION/006.
- An IMPLICIT EDGE BLEND/308 may be either an IMPLICIT EDGE FLAT/036 or an IMPLICIT EDGE ROUND/037.
- An IMPLICIT EDGE BLEND/308 specifies the blend in 0 or 1 EDGE-BLENDED INTERSECTIONs/359.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many PASSAGE BLENDs/152.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many DEPRESSION BLENDs/154.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many STANDARD CLOSED FEATURE SWEEP PROFILE BLENDs/141.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many SQUARE-U BLEND1s/170.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many SQUARE-U BLEND2s/171.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many VEE BLENDs/172.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE STEM/CROSSBAR BLEND1s/173.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE STEM/CROSSBAR BLEND2s/174.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE CROSSBAR BLEND1s/175.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE CROSSBAR BLEND2s/176.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE CROSSBAR BLEND3s/177.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many TEE CROSSBAR BLEND4s/178.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many ELL STEM/ENDBAR BLENDs/179.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many ELL ENDBAR BLEND1s/180.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many ELL ENDBAR BLEND2s/181.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many ELL ENDBAR BLEND3s/182.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many PROFILE PAIR BLENDs/166.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many IN/OUT FEATURE SWEEP WALL/END BLENDs/129.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many ALONG FEATURE SWEEP FLAT END BLENDs/161.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many PASSAGE INTERMEDIATE BOUND BLENDs/187.

- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many DEPRESSION INTERMEDIATE BOUND BLENDs/189.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many AXISYMMETRIC FEATURE SWEEP WALL/END BLENDs/131.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many FEATURE RULING WALL/END BLENDs/416.

EXPRESS Declaration

```
ENTITY implicit_edge_blend
SUPERTYPE OF (implicit_edge_flat OR
              implicit_edge_round)
SUBTYPE OF (implicit_transition);
END_ENTITY;
```

Entity Name: **IMPLICIT CORNER BLEND**

Entity Number: **309**

An implicit representation of a form feature that smooths or "gradualizes" a corner of a shape.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

SHAPE ID (FK) (AK1)
With the following attribute, identifies the CORNER SHAPE ELEMENT/INT-20.
SHAPE ELEMENT ID (FK) (AK1)
IMPLICIT CORNER BLEND TYPE (Discriminator)

Business Rules

- An IMPLICIT CORNER BLEND/309 is a type of IMPLICIT TRANSITION/006.
- An IMPLICIT CORNER BLEND/309 may be either an IMPLICIT CORNER FLAT/310 or an IMPLICIT OUTSIDE CORNER ROUND/311.
- A CORNER SHAPE ELEMENT/INT-20 is blended by 0 or 1 IMPLICIT CORNER BLENDs/309.

EXPRESS Declaration

```
ENTITY implicit_corner_blend
SUPERTYPE OF (implicit_corner_flat OR
              implicit_outside_corner_round)
SUBTYPE OF (implicit_transition);
END_ENTITY;
```

Entity Name: IMPLICIT CORNER FLAT

Entity Number: 310

An implicit representation of a planar form feature that smooths or "gradualizes" a corner of a shape. The feature specification is in terms of one of the areas meeting at the corner. That area must be planar. Setbacks are given for each of the two edges of that area (thus determining a line in the plane) and an angle between the plane and the flat is given.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

EDGE1.SHAPE ID (FK) (AK1)
With the following attribute, identifies one of the EDGE SHAPE ELEMENTs/INT-15.

EDGE1.SHAPE ELEMENT ID (FK) (AK1)

EDGE2.SHAPE ID (FK) (AK1)
With the following attribute, identifies the other EDGE SHAPE ELEMENT/INT-15.

EDGE2.CORNER.SHAPE ELEMENT ID (FK) (AK1)

SETBACK1.DIM ID (FK)
The setback (measured linearly) from the corner along the first edge. (While half dimension is possible, it seems unuseful.)

SETBACK2.DIM ID (FK)
The setback (measured linearly) from the corner along the second edge. edge. (While half dimension is possible, it seems unuseful.)

ANGLE.DIM ID (FK)
The angle between the planar area and the flat. (While half dimension is possible, it seems unuseful.)

Business Rules

- An IMPLICIT CORNER FLAT/310 is a type of IMPLICIT CORNER BLEND/309
- An EDGE SHAPE ELEMENT/INT-15 is first setback edge of 0, 1, or 2 IMPLICIT CORNER FLATs/310.
- An EDGE SHAPE ELEMENT/INT-15 is second setback edge of 0, 1, or 2 IMPLICIT CORNER FLATs/310.
- The two EDGE SHAPE ELEMENTs/INT-15 must be adjacent boundaries, meeting at the CORNER SHAPE ELEMENT/INT-20 associated with the parent IMPLICIT CORNER BLEND/309 of the feature, of a planar AREA SHAPE ELEMENT/INT-8 which is one of the areas that meet at the CORNER SHAPE ELEMENT/INT-20.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is setback1 dimension of 0, 1, or many IMPLICIT CORNER FLATs/310.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is setback2 dimension of 0, 1, or many IMPLICIT CORNER FLATs/310.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is angle dimension of 0, 1, or many IMPLICIT CORNER FLATs/310.

EXPRESS Declaration

```
ENTITY implicit_corner_flat
SUBTYPE OF (implicit_corner_blend);
  setback_edge1 : edge_shape_element;
  setback_dim1  : independent_size_parameter;
  setback_edge2 : edge_shape_element;
  setback_dim2  : independent_size_parameter;
  angle_dim     : independent_angle_size_parameter;
END_ENTITY;
```

Entity Name: **IMPLICIT OUTSIDE CORNER ROUND**

Entity Number: **311**

An implicit representation of a form feature which is a spherical rounding of a convex corner of a shape. The specification is aimed at simple situations (e.g., a corner of a box) in which the spherical surface is tangent to each edge radiating from the corner.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

DIM ID (FK)
The size dimension (radius, preferably, or diameter) of the spherical feature.

Business Rules

- An IMPLICIT OUTSIDE CORNER ROUND/311 is a type of IMPLICIT CORNER BLEND/309.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is dimension of 0, 1, or many IMPLICIT CORNER ROUNDs/311.

EXPRESS Declaration

```
ENTITY implicit_outside_corner_round
SUBTYPE OF (implicit_corner_blend);
  dimension : independent_size_parameter;
END_ENTITY;
```

Entity Name: **GEOMETRIC MODEL/IMPLICIT FORM FEATURE ASSOCIATIO**

Entity Number: **315**

An indication that a representation of a form feature as an IMPLICIT FORM FEATURE/002 applies to a particular GEOMETRIC MODEL/INT-23.

Primary Key Attributes

SHAPE ID (FK)
Together with the two following attributes, identifies the IMPLICIT FORM FEATURE/002 of interest.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

GEOMETRIC MODEL ID (FK)
The geometric model for which the implicit feature specification is applicable.

Business Rules

- An IMPLICIT FORM FEATURE/002 is applied as 1 or many GEOMETRIC MODEL/ IMPLICIT FORM FEATURE ASSOCIATIONs/315.
- A GEOMETRIC MODEL/INT-23 is augmented by 0, 1, or many GEOMETRIC MODEL-IMPLICIT FORM FEATURE ASSOCIATIONs/315.

EXPRESS Declaration

```
ENTITY geometric_model_implicit_form_feature_assn;
  augmented_geometric_model          : geometric_model;
  augmenting_implicit_form_feature : implicit_form_feature;
END_ENTITY;
```

Entity Name: **IMPLICIT STRAIGHT KNURL**

Entity Number: **354**

An implicit representation of a knurl whose scoring is parallel to the axis of the scored surface.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An IMPLICIT STRAIGHT KNURL/354 is a type of IMPLICIT KNURL/024.

EXPRESS Declaration

```
ENTITY implicit_straight_knurl
SUBTYPE OF (implicit_knurl);
END_ENTITY;
```

Entity Name: **IMPLICIT DIAGONAL KNURL**

Entity Number: **355**

An implicit representation of a knurl whose scoring is spiral about the axis of the scored surface.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

HELIX ANGLE.DIM ID (FK)
The angle the knurl helix makes with the work axis. Half angle is possible, but full angle is preferred.

HELIX HAND
Whether the knurl helix is RIGHT HAND or LEFT HAND, with respect to the direction of the centerline of the DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 on which the feature is installed.

Business Rules

- An IMPLICIT DIAGONAL KNURL/355 is a type of IMPLICIT KNURL/024.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is the helix angle of 0, 1, or many IMPLICIT DIAGONAL KNURLs/355.

EXPRESS Declaration

```
ENTITY implicit_diagonal_knurl
SUBTYPE OF (implicit_knurl);
  helix_angle : independent_size_parameter;
  helix_hand  : hands;
END_ENTITY;

TYPE
hands = ENUMERATION OF (left_hand, right_hand);
END_TYPE;
```

Entity Name: **IMPLICIT DIAMOND KNURL**

Entity Number: **356**

An implicit representation of a knurl whose scoring is doubly spiral (a left and a right hand spiral) about the axis of the scored surface, with equal spacing of the two.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

HELIX ANGLE.DIM ID (FK)
The angle the knurl helix makes with the work axis. Half angle is possible, but full angle is preferred.

Business Rules

- An IMPLICIT DIAMOND KNURL/356 is a type of IMPLICIT KNURL/024.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is helix angle of 0, 1, or many IMPLICIT DIAMOND KNURLs/356.

EXPRESS Declaration

```
ENTITY implicit_diamond_knurl
SUBTYPE OF (implicit_knurl);
  helix_angle : independent_size_parameter;
END_ENTITY;
```

Entity Name: **EDGE-BLENDED INTERSECTION**

Entity Number: **359**

An indication that an IMPLICIT EDGE BLEND/308 applies to the intersection of two DIMENSIONALITY-2 SHAPE ELEMENTs/INT-7.

Primary Key Attributes

BLEND.SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT EDGE BLEND/308.
BLEND.SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

FIRST.SHAPE ID (FK) (AK1)
With the following attribute, identifies the first of the two DIMENSIONALITY-2 SHAPE ELEMENTs/INT-7 whose intersection is blended.
FIRST.SHAPE ELEMENT ID (FK) (AK1)
SECOND.SHAPE ID (FK) (AK1)
With the following attribute, identifies the second of the two DIMENSIONALITY-2 SHAPE ELEMENTs/INT-7 whose intersection is blended.
SECOND.SHAPE ELEMENT ID (FK) (AK1)

Business Rules

- An IMPLICIT EDGE BLEND/308 specifies the blend in 0 or 1 EDGE-BLENDED INTERSECTIONs/359.
- A DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 is the first blended element in 0, 1, or many EDGE-BLENDED INTERSECTIONs/359.
- A DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 is the second blended element in 0, 1, or many EDGE-BLENDED INTERSECTIONs/359.

EXPRESS Declaration

```
ENTITY edge_blended_intersection;
  blend        : implicit_edge_blend;
  first_shape  : dimensionality_2_shape_element;
  second_shape : dimensionality_2_shape_element;
END_ENTITY;
```

Entity Name: IMPLICIT SPHERICAL EMBOSS

Entity Number: 400

An implicit representation of an emboss having spherical displacement of material.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the POINT/GEO-2 which locates the feature.

HEIGHT.DIM ID (FK)
The height dimension of the emboss. May be half or, preferably, full height. Measured from undeformed base of feature.

SIZE.DIM ID (FK)
The diameter or radius of the sphere.

BEND.DIM ID
The dimension (radius, preferably, or diameter) of the bend at the feature's intersection with undeformed material.

Business Rules

- An IMPLICIT SPHERICAL EMBOSS/400 is a type of IMPLICIT EMBOSS/081.
- A POINT/GEO-2 locates 0, 1, or many IMPLICIT SPHERICAL EMBOSSes/400. The POINT/GEO-2 is located on the inward-bent side of the sheetlike geometry, at the location that becomes the apex of the sphere. Material movement is normal to the sheetlike geometry at that location.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is height dimension of 0, 1, or many IMPLICIT SPHERICAL EMBOSSes/400.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is size dimension of 0, 1, or many IMPLICIT SPHERICAL EMBOSSes/400.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT SPHERICAL EMBOSSes/400.
- Sphere radius $\geq$ full feature height.

EXPRESS Declaration

```
ENTITY implicit_spherical_emboss
SUBTYPE OF (implicit_emboss);
  location    : point;
  height      : independent_size_parameter;
  emboss_size : independent_size_parameter;
  bend_dim    : bend_dimension;
END_ENTITY;
```

Entity Name: **IMPLICIT TAB**

Entity Number: **401**

An implicit representation of a partial cutout where the shear is an arc of a circle and the deformation to the material causes an opening to be created.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the AXIS PLACEMENT/GEO-4 that locates the feature.

CHORD.DIM ID (FK)
The length or half-length dimension of the bend line of the cutout, which is a chord of the shear line circular arc.

WIDTH.DIM ID (FK)
The distance or half-distance dimension from the center of the bend line to the edge of the shear line.

HEIGHT.DIM ID (FK)
The maximum distance the feature extends beyond the material. (Full, preferably, or half dimension.)

ANGLE.DIM ID (FK)
The angle the tab makes with unaffected material. (Full, preferably, or semi-angle.)

BEND.DIM ID (FK)
The (radius, preferably, or diameter) dimension of the bend at the base of the tab.

Business Rules

- An IMPLICIT TAB/401 is a type of IMPLICIT PARTIAL CUTOUT/083.
- An AXIS PLACEMENT/GEO-4 locates 0, 1, or many IMPLICIT TABs/401. This is done as follows. The origin must be at the midpoint of the pre-deformation bend line. The bend line lies on the Z-axis. The Y-axis is normal to the sheet and points in the initial direction of material movement. Thus the shear line is the circular arc with end points (0,0,-CHORD) and (0,0,CHORD) and midpoint (WIDTH,0,0). The latter point is displaced to y = HEIGHT.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is chord dimension of 0, 1, or many IMPLICIT TABs/401.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is width dimension of 0, 1, or many IMPLICIT TABs/401.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is height dimension of 0, 1, or many IMPLICIT TABs/401.
- A DERIVABLE ANGLE SIZE PARAMETER/TOL-6102 is angle dimension of 0, 1, or many IMPLICIT TABs/401.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT TABs/401.

EXPRESS Declaration

```
ENTITY implicit_tab
SUBTYPE OF (implicit_partial_cutout);
```

```
   location : axis_placement;
   chord    : independent_size_parameter;
   width    : independent_size_parameter;
   height   : independent_size_parameter;
   angle    : derivable_angle_size_parameter;
   bend_dim : bend_dimension;
 END_ENTITY;
```

Entity Name: **PASSAGE BOUND**

Entity Number: **402**

An indication that an IMPLICIT FEATURE BOUND/029 describes the pre-existing area where an IMPLICIT PASSAGE/003 enters/exits material.

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the bounded IMPLICIT PASSAGE/003.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

BOUNDED PASSAGE END
Whether the bound is regarded as the ENTRY or EXIT area of the feature.

Other Attributes

IMPLICIT FEATURE BOUND ID (FK)

Business Rules

- An IMPLICIT PASSAGE/003 is bounded by 0, 1, or 2 PASSAGE BOUNDs/402.
- An IMPLICIT FEATURE BOUND/029 specifies the bound in 0, 1, or many PASSAGE BOUNDs/402.

EXPRESS Declaration

```
ENTITY passage_bound;
  specification : implicit_feature_bound;
  passage_end   : feature_end_types;
END_ENTITY;
```

```
TYPE
feature_end_types = ENUMERATION OF (initial_end, terminal_end);
END_TYPE;
```

Entity Name: **DEPRESSION BOUND**

Entity Number: **403**

An indication that an IMPLICIT FEATURE BOUND/029 describes the pre-existing area where an IMPLICIT DEPRESSION/005 enters material.

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the bounded IMPLICIT DEPRESSION/005.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other_Attributes

IMPLICIT FEATURE BOUND ID (FK)

Business Rules

- An IMPLICIT DEPRESSION/005 is bounded by 0 or 1 DEPRESSION BOUNDs/403.
- An IMPLICIT FEATURE BOUND/029 specifies the bound in 0, 1, or many DEPRESSION BOUNDs/403.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under IMPLICIT DEPRESSION/005.)

Entity Name: **PROTRUSION BOUND**

Entity Number: **404**

An indication that an IMPLICIT FEATURE BOUND/029 describes the pre-existing area on which an IMPLICIT PROTRUSION/004 is installed.

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the bounded IMPLICIT PROTRUSION/004.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

IMPLICIT FEATURE BOUND ID (FK)

Business Rules

- An IMPLICIT PROTRUSION/004 is bounded by 0 or 1 PROTRUSION BOUNDs/404.
- An IMPLICIT FEATURE BOUND/029 specifies the bound in 0, 1, or many PROTRUSION BOUNDs/404.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under IMPLICIT PROTRUSION/004.)

Entity Name: IMPLICIT MARKING

Entity Number: 406

An implicit representation of the application of a character (or standard symbol) string to an area of a shape. This is a very weak specification; there is no ability to specify size, font, depressed/raised lettering, layout of the characters, etc. Only the characters and, optionally, area can be specified.

Primary Key Attributes

SHAPE ID (FK)

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

MARKED STRING
The character string to be applied.

Business Rules

- An IMPLICIT MARKING/406 is a type of IMPLICIT AREA FEATURE/111.

EXPRESS Declaration

```
ENTITY implicit_marking
SUBTYPE OF (implicit_area_feature);
  marked_string : STRING(80);
END_ENTITY;
```

Entity Name: FEATURE RULING

Entity Number: 411

An implicit representation of a form feature as a ruling of two curves. A FEATURE RULING/411 is used to specify an IMPLICIT PASSAGE/003, IMPLICIT PROTRUSION/004, or IMPLICIT DEPRESSION/005.

A FEATURE RULING/411 may be viewed as a generalization of a FEATURE SWEEP/120; it is used to define added/subtracted volumes over a finite range. The directionality implicit is sweeps is extended to rulings by regarding the first of the two defining curves as the initial end of the ruling and the second as the terminal end.

The curves defining the ruled wall of a FEATURE RULING/411 are required to be CURVE ON SURFACEs/GEO-23. The underlying surfaces of the CURVE ON SURFACEs/GEO-23 specify the ends of the added/subtracted volume.

The curves defining the ruling may be open or closed. If open, it is the creator's responsibility to insure that the ruled surface extends sufficiently far that there is no ambiguity. That is, the ruling lines corresponding to the first and last points on the defining curves must lie in air or on the air/material boundary for a subtracted volume; in material or on the boundary for an added volume.

A FEATURE RULING/411 has a "local" coordinate system. This may be specified by a FEATURE RULING LCS/418, which associates the FEATURE RULING/411 with an AXIS PLACEMENT/GEO-4. If there is no FEATURE RULING LCS/418, the global coordinate system is by default the local system. The defining curves of the ruling are assumed to be specified in the local coordinate system.

Primary Key Attributes

FEATURE VOLUME ID (FK)

Other Attributes

CURVE1.GEOMETRIC ENTITY ID (FK) (AK1)
Identifies the first of the two CURVE ON SURFACEs/GEO-23 which define the ruling.

CURVE2.GEOMETRIC ENTITY ID (FK) (AK1)
Identifies the second of the two CURVE ON SURFACEs/GEO-23 which defines the ruling.

Business Rules

- A FEATURE RULING/411 is a type of FEATURE VOLUME/417.
- A CURVE ON SURFACE/GEO-23 is curve1 of 0, 1, or many FEATURE RULINGs/411.
- A CURVE ON SURFACE/GEO-23 is curve2 of 0, 1, or many FEATURE RULINGs/411.
- A FEATURE RULING/411 is blended by 0, 1, or 2 FEATURE RULING WALL/END BLENDs/416.
- A FEATURE RULING/411 has 0 or 1 FEATURE RULING LCSs/418.

EXPRESS Declaration

```
ENTITY feature_ruling
SUBTYPE OF (feature_volume);
  defining_curve1 : curve_on_surface;
  defining_curve2 : curve_on_surface;
  blend           : SET [0:2] OF feature_ruling_wall_end_blend;
  location        : OPTIONAL axis_placement;
END_ENTITY;
```

Entity Name: **IMPLICIT COUPLING**

Entity Number: **415**

An implicit representation of a toothed feature installed at the end of a cylindrical shape whose purpose is to fix axially adjacent components with respect to each other. The feature's teeth have sides with an arc shape. One coupling in a pair will have convex (barrel shaped) teeth; the other, concave.

Primary Key Attributes

SHAPE ID (FK)
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

COUPLING SHAPE
: Whether teeth are CONVEX or CONCAVE.

COUPLING NUMBER OF TEETH

PRESSURE ANGLE.DIM ID (FK)
: The angle between the tapered mating walls of teeth and the radial direction of the cylindrical shape. (Half-angle is possible, but discouraged.)

WHOLE DEPTH.DIM ID (FK)
: The tooth depth dimension. (Half-dimension is possible, but discouraged.)

CHAMFER DEPTH.DIM ID (FK)
: The dimension from the top of a tooth to the tapered contact surface of the tooth. (Half-dimension is possible, but discouraged.)

ROOT FILLET.DIM ID (FK)
: The dimension (radius, preferably, or diameter) of the fillet between tooth and slot bottom.

RADIUS TO POINT OF TANGENCY.DIM ID (FK)
: The dimension (radius, preferably, or diameter) of the grinding wheel at the pitch line, which establishes the curvature of the teeth sides.

Business Rules

- An IMPLICIT COUPLING/415 is a type of IMPLICIT AREA FEATURE/111.
- An INDEPENDENT ANGLE SIZE PARAMETER/TOL-6101 is pressure angle dimension of 0, 1, or many IMPLICIT COUPLINGs/415.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is whole depth dimension of 0, 1, or many IMPLICIT COUPLINGs/415.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is chamfer depth dimension of 0, 1, or many IMPLICIT COUPLINGs/415.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is root fillet dimension of 0, 1, or many IMPLICIT COUPLINGs/415.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is radius to point of tangency dimension of 0, 1, or many IMPLICIT COUPLINGs/415.
- An IMPLICIT COUPLING/415 is timed by 0 or 1 COUPLING TIMERs/420.
- The FEATURE APPLICATION AREA/028 on which an IMPLICIT COUPLING/415 is installed must be a open planar face of an axisymmetric geometry, adjacent to an internal and an external diameter.

NOTES

- The outside diameter, inside diameter, and face width associated with an IMPLICIT COUPLING/415 are presumed to be available from other data in the model of shape.

EXPRESS Declaration

```
ENTITY implicit_coupling
SUBTYPE OF (implicit_area_feature);
  coupling_shape            : coupling_shape_types;
  number_of_teeth           : INTEGER;
  pressure_angle            : independent_size_parameter;
  whole_depth               : independent_size_parameter;
  chamfer_depth             : independent_size_parameter;
  root_fillet               : independent_size_parameter;
  dim_to_point_of_tangency  : independent_size_parameter;
  coupling_timing           : OPTIONAL coupling_timer;
END_ENTITY;

TYPE
coupling_shape_types = ENUMERATION OF (concave_coupling, convex_coupling);
END_TYPE;
```

Entity Name: **FEATURE RULING WALL/END BLEND**

Entity Number: **416**

A blend (round or chamfer) between the ruled surface wall of a FEATURE RULING/411 and one of its end surfaces. (The ruling curves are CURVE ON SURFACEs/GEO-23. The end surfaces are the surfaces on which the ruling curves lie.)

Primary Key Attributes

FEATURE VOLUME ID (FK)
: Identifies the parent FEATURE RULING/411.

BLENDED RULING END
: Whether the blend is at the INITIAL (curve1) or TERMINAL (curve2) end of the ruling.

Other Attributes

SHAPE ID (FK)
: With the two following attributes, identifies the IMPLICIT EDGE BLEND/308 that applies at the intersection of the wall and end.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Business Rules

- A FEATURE RULING/411 is blended by 0, 1, or 2 FEATURE RULING WALL/END BLENDs/416.
- An IMPLICIT EDGE BLEND/308 is used as 0, 1, or many FEATURE RULING WALL/END BLENDs/416.

EXPRESS Declaration

```
ENTITY feature_ruling_wall_end_blend;
  blend_end  : feature_end_types;
  blend      : implicit_edge_blend;
END_ENTITY;

TYPE
feature_end_types = ENUMERATION OF (initial_end, terminal_end);
END_TYPE;
```

Entity Name: FEATURE VOLUME

Entity Number: 417

A volume added to or subtracted from pre-existing shape. Used to specify implicit definitions of form features modeled as increments/decrements of material.

Primary Key Attributes

FEATURE VOLUME ID
An arbitrary label.

Other Attributes

FEATURE VOLUME TYPE (Discriminator)

Business Rules

- A FEATURE VOLUME/417 may be either a FEATURE SWEEP/120 or a FEATURE RULING/411.
- A FEATURE VOLUME/417 defines 0 or 1 IMPLICIT PASSAGEs/003.
- A FEATURE VOLUME/417 defines 0 or 1 IMPLICIT PROTRUSIONs/004.
- A FEATURE VOLUME/417 defines 0 or 1 IMPLICIT DEPRESSIONS/005.

EXPRESS Declaration

```
ENTITY feature_volume
SUPERTYPE OF (feature_sweep OR
              feature_ruling);
END_ENTITY;
```

Entity Name: **FEATURE RULING LCS**

Entity Number: **418**

The use of an AXIS PLACEMENT/GEO-4 to define a local coordinate system for a FEATURE RULING/411.

Primary Key Attributes

FEATURE VOLUME ID
An arbitrary label.

Other Attributes

GEOMETRIC ENTITY ID
The identity of the AXIS PLACEMENT/GEO-4.

Business Rules

- A FEATURE RULING/411 has 0 or 1 FEATURE RULING LCSs/418.
- An AXIS PLACEMENT/GEO-4 is used as 0, 1, or many FEATURE RULING LCSs/418.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under FEATURE RULING/411.)

Entity Name: **IMPLICIT FEATURE PRECEDENCE**

Entity Number: **419**

An indication of existence precedence between two implicit feature representations This entity gives a predecessor/successor relation between implicitly represented form features. It implies that the predecessor feature is a part of the pre-existing shape to which the successor is an increment.

Note that, functionally, the IMPLICIT FEATURE BOUND/029 mechanism partially overlaps this entity in the sense that a "bounder"/"boundee" relationship implies a predecessor/successor relation. Only one of the two modeling tactics should be used for a given case.

Primary Key Attributes

PREDECESSOR.SHAPE ID (FK)
: With the two following attributes, identifies the predecessor IMPLICIT FORM FEATURE/002.

PREDECESSOR.SHAPE ELEMENT ID (FK)

PREDECESSOR.IMPLICIT FORM FEATURE ID (FK)

SUCCESSOR.SHAPE ID (FK)
: With the two following attributes, identifies the successor IMPLICIT FORM FEATURE/002.

SUCCESSOR.SHAPE ELEMENT ID (FK)

SUCCESSOR.IMPLICIT FORM FEATURE ID (FK)

Business Rules

- An IMPLICIT FORM FEATURE/002 is predecessor in 0, 1, or many IMPLICIT FEATURE PRECEDENCEs/419.
- An IMPLICIT FORM FEATURE/002 is successor in 0, 1, or many IMPLICIT FEATURE PRECEDENCEs/419.

EXPRESS Declaration

```
ENTITY implicit_feature_precedence;
  predecessor_feature : implicit_form_feature;
  successor_feature   : implicit_form_feature;
END_ENTITY;
```

Entity Name: **COUPLING TIMER**

Entity Number: **420**

The use of a POINT/GEO-2 to control the rotational placement of an IMPLICIT COUPLING/415. The POINT/GEO-2 lies on any radial (with respect to the coupling's axis of rotation) line passing through the middle of any tooth of the feature.

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT COUPLING/415.
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the POINT/GEO-2.

Business Rules

- An IMPLICIT COUPLING/415 is timed by 0 or 1 COUPLING TIMERs/420.
- A POINT/GEO-2 is used as 0, 1, or many COUPLING TIMERs/420.

EXPRESS Declaration

```
ENTITY coupling_timer;
  timing_point : point;
END_ENTITY;
```

Entity Name: FULL THREADING SPECIFICATION

Entity Number: 421

A specification of the area of full threading of an IMPLICIT THREAD/025. This entity is used when the DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 of (the parent IMPLICIT AREA FEATURE/111 of) an IMPLICIT THREAD/025 is not fully threaded along its entire area. The full threading area is specified via a POINT/GEO-2 on the axis of the threaded DIMENSIONALITY-2 SHAPE ELEMENT/INT-7. The DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 must be fully threaded on the side of that point indicated by the attribute FULL THREADING DIRECTION. If FULL THREADING DIRECTION is 'UPTO', full threading occurs in the negative direction along the axis from the point; if 'BEYOND', positive.

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT THREAD/025.
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
Identifies the POINT/GEO-2.

FULL THREADING DIRECTION
Whether full threading occurs UP TO or BEYOND the area indicated by the POINT/GEO-2.

DIM ID (FK)
The derivable length/depth dimension of full threading. While semi-depth can be given, full dimension seems best.)

Business Rules

- An IMPLICIT THREAD has 0 or 1 FULL THREADING SPECIFICATIONs/421.
- A POINT/GEO-2 is used as 0, 1, or many FULL THREADING SPECIFICATIONs/421. The POINT/GEO-2 should lie on the axis of the cylindrical or conical DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 on which the parent IMPLICIT THREAD/025 is installed. If it does not, a perpendicular from the POINT/GEO-2 to the axis will yield a POINT/GEO-2 satisfying the criterion.
- A DERIVABLE SIZE PARAMETER/TOL-6042 is full thread length/depth dimension of 0, 1, or many FULL THREADING SPECIFICATIONs/421.

EXPRESS Declaration

```
ENTITY full_threading_specification;
  reference_point           : point;
  full_threading_direction  : threading_direction_types;
  full_thread_length        : derivable_size_parameter;
END_ENTITY;

TYPE
threading_direction_types = ENUMERATION OF (thread_upto, thread_beyond);
END_TYPE;
```

Entity Name: **SPIRAL FEATURE SWEEP PATH**

Entity Number: **422**

A FEATURE SWEEP PATH/124 which is a spiral. The spiral may be of constant diameter or tapered. The latter is the case if and only if there is an associated SPIRAL TAPER/427 entity.

Primary Key Attributes

FEATURE SWEEP PATH ID (FK)

Other Attributes

LENGTH.DIM ID (FK)
The length dimension, measured along its centerline, of the sweep path. (Half dimension can be specified, but full is preferred.)

SIZE.DIM ID (FK)
The diameter or radius dimension of the spiral. In the case of a tapered spiral, this is measured at local $z = 0$.

SPIRAL PATH HAND
Whether the spiral is LEFT or RIGHT handed, with respect to the Z-axis of its LCS.

TURNS PER UNIT
The number of times, per unit length along the centerline, that the spiral revolves about its centerline.

Business Rules

- A SPIRAL FEATURE SWEEP PATH/422 is a type of FEATURE SWEEP PATH/124.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is length dimension of 0, 1, or many SPIRAL FEATURE SWEEP PATHs/422.
- An INDEPENDENT SIZE PARAMETER/TOL-6041 is size dimension of 0, 1, or many SPIRAL FEATURE SWEEP PATHs/422.
- The swept profile is positioned as follows by the AXIS PLACEMENT/GEO-4 of the FEATURE SWEEP/120 which uses the spiral as sweep path:
  (a) AB-plane at initial position of profile; i.e., start of sweep.
  (b) AB-plane perpendicular to spiral.
  (c) B-axis intersecting centerline.
- A SPIRAL FEATURE SWEEP PATH/422 has 0 or 1 SPIRAL TAPERs/427.

EXPRESS Declaration

```
ENTITY spiral_feature_sweep_path
SUBTYPE OF (feature_sweep_path);
  spiral_path_length : independent_size_parameter;
  spiral_path_size   : independent_size_parameter;
  spiral_path_hand   : hands;
  turns_per_unit     : REAL;
  spiral_path_taper  : OPTIONAL spiral_taper;
```

```
END_ENTITY;

TYPE
hands = ENUMERATION OF (left_hand, right_hand);
END_TYPE;
```

Entity Name: **SURFACE CONFORMING FEATURE SWEEP PATH**

Entity Number: **423**

A FEATURE SWEEP PATH/124 that is a curve on a surface. The swept profile maintains constant position and attitude with respect to the surface as it is swept along the path.

Primary Key Attributes

FEATURE SWEEP PATH ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
The CURVE ON SURFACE/GEO-23 that serves as the path.

Business Rules

- A SURFACE CONFORMING FEATURE SWEEP PATH/423 is a type of FEATURE SWEEP PATH/124.
- A CURVE ON SURFACE/GEO-23 is used as 0, 1, or many SURFACE CONFORMING FEATURE SWEEP PATHs/423. This is done as follows:
  - Denote the surface on which the path lies as $f(u, v)$.
  - Let T be the transformation accomplished by the AXIS PLACEMENT/GEO-4 of the FEATURE SWEEP/120; i.e., T transforms the swept profile from its AB-space into the desired position (in global space) relative to the initial point of the CURVE ON SURFACE/GEO-23. (Note: T must be such that the initial point of the CURVE ON SURFACE/GEO-23 lies in the plane of the transformed profile.)
  - For any point $f(u, v)$ on the CURVE ON SURFACE/GEO-23, T transforms the profile into its position with respect to the AXIS PLACEMENT/GEO-4 defined by $f(u, v)$, the partial derivative of f with respect to u at $f(u, v)$, and the partial derivative of f with respect to v at $f(u, v)$.

EXPRESS Declaration

```
ENTITY surface_conforming_feature_sweep_path
SUBTYPE OF (feature_sweep_path);
  reference_surface : curve_on_surface;
END_ENTITY;
```

Entity Name: **OTHER FEATURE SWEEP PATH**

Entity Number: **424**

The use of an arbitrary CURVE/GEO-6 as a FEATURE SWEEP PATH/124. The position and orientation of the swept profile are controlled by the path CURVE/GEO-6 and an orienting CURVE/GEO-6.

Primary Key Attributes

FEATURE SWEEP PATH ID (FK)

Other Attributes

PATH.GEOMETRIC ENTITY ID (FK)
The CURVE/GEO-6 that serves as the path.

ORIENT.GEOMETRIC ENTITY ID (FK)
The CURVE/GEO-6 that controls orientation of the swept profile along the path.

Business Rules

- An OTHER FEATURE SWEEP PATH/424 is a type of FEATURE SWEEP PATH/124.
- A CURVE/GEO-6 is path of 0, 1, or many OTHER FEATURE SWEEP PATHs/424.
- A CURVE/GEO-6 orients the swept profile of of 0, 1, or many OTHER FEATURE SWEEP PATHs/424.
- The position and orientation of the swept profile are determined as follows:
  (a) Denote the path CURVE/GEO-6 as $f(u)$ and the orienting CURVE/GEO-6 as $g(u)$.
  (b) Let T be the transformation accomplished by the AXIS PLACEMENT/GEO-4 of the FEATURE SWEEP/120; i.e., T transforms the swept profile from its AB-space into the desired position (in global space) relative to the initial point of the CURVE/GEO-6. (Note: T must be such that the initial point of the CURVE/GEO-6 lies in the plane of the transformed profile.)
  (c) For any point $f(u)$ on the path CURVE/GEO-6, T transforms the profile into its position with respect to the AXIS PLACEMENT/GEO-4 defined by $f(u)$, the tangent vector of f at u, and the tangent vector of g at u.

EXPRESS Declaration

```
ENTITY other_feature_sweep_path
SUBTYPE OF (feature_sweep_path);
  path_def    : curve;
  orientation : curve;
END_ENTITY;
```

Entity Name: PITCH APEX

Entity Number: 425

The definition of a pitch cone for an IMPLICIT AREA FEATURE/111. The specification assumes that a directed axis is known from the DIMENSIONALITY-2 SHAPE ELEMENT/INT-7 on which the feature is installed.

Primary Key Attributes

PITCH APEX ID
An arbitrary label distinguishing these entities.

Other Attributes

GEOMETRIC ENTITY ID (FK)
The POINT/GEO-2 that is the apex of the cone.

PITCH DISTANCE
The distance along the axis from the apex to the axial point at which diametral dimensions (e.g., pitch diameter, major diameter, minor diameter) are given.

PITCH DIRECTION
Whether the direction from the apex to the axial point is the same as or the opposite of the axis direction.

Business Rules

- A POINT/GEO-2 is used as 0, 1, or many PITCH APEXes/425.
- A PITCH APEX/425 is used as 0, 1, or many THREAD APEXes/426.

EXPRESS Declaration

```
ENTITY pitch_apex;
  apex_point              : point;
```

```
  pitch_distance          : REAL;
  pitch_in_axis_direction : LOGICAL;
END_ENTITY;
```

Entity Name: **THREAD APEX**

Entity Number: **426**

The definition of a pitch cone for an IMPLICIT THREAD/025.

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT THREAD/025.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

PITCH APEX ID
The pitch apex's key.

Business Rules

- An IMPLICIT THREAD/025 has 0 or 1 THREAD APEXes/426.
- A PITCH APEX/425 is used as 0, 1, or many THREAD APEXes/426.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under IMPLICIT THREAD/025.)

Entity Name: **SPIRAL TAPER**

Entity Number: **427**

A specification of the taper of a SPIRAL FEATURE SWEEP PATH/422 by giving the location of the spiral's apex. The apex point is located by giving its local z-coordinate. A SPIRAL FEATURE SWEEP PATH/422 is tapered if and only if this entity is associated with it.

Primary Key Attributes

FEATURE SWEEP PATH ID (FK)

Other Attributes

APEX Z-COORDINATE
The local z-coordinate of the spiral apex.

DIM ID (FK)
The taper angle dimension (full or half angle) of the sweep.

Business Rules

- A SPIRAL FEATURE SWEEP PATH/422 has 0 or 1 SPIRAL TAPERs/427.
- A DERIVABLE ANGLE SIZE PARAMETER/TOL-6102 is taper angle dimension of 0, 1, or many SPIRAL TAPERs/427.

EXPRESS Declaration

```
ENTITY spiral_taper;
  apex_z_coord : REAL;
  angle_dim    : derivable_angle_size_parameter;
END_ENTITY;
```

Entity Name: **THREAD TIMER**

Entity Number: **428**

A specification of the rotational placement of an IMPLICIT THREAD/025 on the threaded (cylindrical or conical) surface by identifying a pressure point (a point on the pitch spiral).

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT THREAD/025.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

GEOMETRIC ENTITY ID (FK)
The pressure point's identification.

Business Rules

- An IMPLICIT THREAD/025 is timed by 0 or 1 THREAD TIMERs/428.
- A POINT/GEO-2 is pressure point in 0, 1, or many THREAD TIMERs/428.

EXPRESS Declaration

```
ENTITY thread_timer;
  pressure_point : point;
END_ENTITY;
```

Entity Name: BEND DIMENSION

Entity Number: 429

The dimension of a bend, plus an indication whether the dimension applies to the concave side, convex side, or center of the bend.

Primary Key Attributes

DIM ID (FK)
The ID of the INDEPENDENT SIZE PARAMETER/TOL-6041 which gives the dimension value.

Other Attributes

BEND MEASUREMENT
Whether the dimension is specified for the CONCAVE side of the bend, the CONVEX side, or CENTER.

Business Rules

- An INDEPENDENT SIZE PARAMETER/TOL-6041 is used as 0 or 1 BEND DIMENSIONs/429.
- A BEND DIMENSION/429 is base bend dimension of 0, 1, or many IMPLICIT V-BEADs/087.
- A BEND DIMENSION/429 is apex round dimension of 0, 1, or many IMPLICIT V-BEADs/087.
- A BEND DIMENSION/429 is base bend dimension of 0, 1, or many IMPLICIT ROUND BEADs/088.

- A BEND DIMENSION/429 is bend1 dimension of 0, 1, or many IMPLICIT CORNER RIBs/089.
- A BEND DIMENSION/429 is bend2 dimension of 0, 1, or many IMPLICIT CORNER RIBs/089.
- A BEND DIMENSION/429 is base bend dimension of 0, 1, or many IMPLICIT LOUVERs/090.
- A BEND DIMENSION/429 is inner bend dimension of 0, 1, or many IMPLICIT LOUVERs/090.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT CIRCULAR KNOCKOUTs/091.
- A BEND DIMENSION/429 is size dimension of 0, 1, or many IMPLICIT TUBE BENDs/093.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT CUTOUT FLANGEs/094.
- A BEND DIMENSION/429 is size dimension of 0, 1, or many IMPLICIT STRAIGHT BENDs/095.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many BEND POINTs/096.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT TUBE FLAREs/098.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT TUBE NECKs/099.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT TUBE FLATTENINGs/100.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT SPHERICAL EMBOSSes/400.
- A BEND DIMENSION/429 is bend dimension of 0, 1, or many IMPLICIT TABs/401.

EXPRESS Declaration

```
ENTITY bend_dimension;
  bend_dim   : independent_size_parameter;
  bend_msrmt : bend_measurement_types;
END_ENTITY;

TYPE
bend_measurement_types = ENUMERATION OF (concave_measurement,
                                         convex_measurement,
                                         center_measurement);
END_TYPE;
```

Entity Name: **BEND MOVEMENT INDICATOR**

Entity Number: **430**

For a bend described using a bendline, an indication of the side of the bendline on which material is regarded (for computational purposes, at least) as to be moved (or having been moved) by the bend. Material on the other side of the bendline is regarded as static.

The purpose of this indication is to constrain calculation of post-bend shape (if the bend is not realized in the geometric model) or pre-bend shape (if the bend is realized in the geometric model).

Primary Key Attributes

BEND MOVEMENT INDICATOR ID

Other Attributes

MOVED SIDE
Whether material on the LEFT or RIGHT side of the bendline, with respect to the direction of that curve, is to be/was moved.

Business Rules

- A BEND MOVEMENT INDICATOR/430 is used as 0, 1, or many GENERAL BEND MOVEMENT INDICATORs/431.
- A BEND MOVEMENT INDICATOR/430 is used as 0, 1, or many STRAIGHT BEND MOVEMENT INDICATORs/432.
- In order for the left/right indication to be meaningful, the bendline curve must be known, explicitly or implicitly, to lie on a surface.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under GENERAL BEND MOVEMENT INDICATOR/431 and STRAIGHT BEND MOVEMENT INDICATOR/432.

Entity Name: **GENERAL BEND MOVEMENT INDICATOR**

Entity Number: **431**

The use of a BEND MOVEMENT INDICATOR/430 it indicate material movement for an IMPLICIT GENERAL BEND/092.

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT GENERAL BEND/092.
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

BEND MOVEMENT INDICATOR ID (FK)

Business Rules

- An IMPLICIT GENERAL BEND/092 has moved side indicated by 0 or 1 GENERAL BEND MOVEMENT INDICATORS/431.
- A BEND MOVEMENT INDICATOR/430 is used as 0, 1, or many GENERAL BEND MOVEMENT INDICATORs/431.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under IMPLICIT GENERAL BEND/092.)

Entity Name: **STRAIGHT BEND MOVEMENT INDICATOR**

Entity Number: **432**

The use of a BEND MOVEMENT INDICATOR/430 it indicate material movement for an IMPLICIT STRAIGHT BEND/095.

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT STRAIGHT BEND/095.
SHAPE ELEMENT ID (FK)
IMPLICIT FORM FEATURE ID (FK)

Other Attributes

BEND MOVEMENT INDICATOR ID (FK)

Business Rules

- An IMPLICIT STRAIGHT BEND/095 has moved side indicated by 0 or 1 STRAIGHT BEND MOVEMENT INDICATORs/432.
- A BEND MOVEMENT INDICATOR/430 is used as 0, 1, or many STRAIGHT BEND MOVEMENT INDICATORs/432.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under IMPLICIT STRAIGHT BEND/095.)

Entity Name: **TUBE BEND MOVEMENT INDICATOR**

Entity Number: **433**

An indication of which material is viewed as being moved (or having been moved) by an IMPLICIT TUBE BEND/093.

The purpose of this indication is to constrain calculation of post-bend shape (if the bend is not realized in the geometric model) or pre-bend shape (if the bend is realized in the geometric model).

Primary Key Attributes

SHAPE ID (FK)
With the following two attributes, identifies the IMPLICIT TUBE BEND/093.

SHAPE ELEMENT ID (FK)

IMPLICIT FORM FEATURE ID (FK)

Other Attributes

MOVED END
Whether movement of material is on the FORWARD or REARWARD side, with respect to the direction of the centerline of the tube, of the POINT/GEO-2 that locates the bend.

Business Rules

- An IMPLICIT TUBE BEND/093 has moved end indicated by 0 or 1 TUBE BEND MOVEMENT INDICATORs/433.

EXPRESS Declaration

(In the de-normalized EXPRESS model, this entity's information is found under IMPLICIT TUBE BEND/093.)

4.5 Issue Log

Following is a log of technical issues raised during development of the FFIM. Each issue is described, the solutions considered are stated and discussed, and the resolution is given.

All issues have been resolved. However, many of the resolutions are identified as specific to this version of the draft proposal, with future reconsideration intended.

It should be noted that model evolution tends to make portions of an issue log obsolete and/or difficult to decipher. Issues are stated and considered in the context of the model as it exists at the time. Subsequent model changes can render an issue obsolete or change the context in which the issue is discussed. Some effort has been made to keep the issues log sensible in the context of the current model, but complete success is not possible.

ISSUE FF-1
INITIATION DATE: 1987
INITIATOR: Several
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: FORM FEATURE/001, IMPLICIT FORM FEATURE/002
FORM FEATURE COMPOSITION/319
DESCRIPTION Should it be allowed for a feature to have both an implicit and an enumerative representation with respect to the same geometric model?

ISSUE OPTIONS & EVIDENCE:

Option 1: Yes.
Pro: Situations can be foreseen where both are desirable.
Con: Opens the door to inconsistent data.
Option 2: No.
Pro: Prevents inconsistency. Conversion between representations is theoretically possible.

Con: A broad conversion capability is a huge requirement. For enumerative-to-implicit, it is probably beyond the state of the art.

OPTION PROPOSED: Option 1
EXPLANATION: The practical argument outweighed the inconsistency danger. Model exchangers may, as a matter of convention, bilaterally adopt Option 2 if they wish.
DECISION: Adopt Option 1
DECISION DATE: 1987
ACTION: The FFIM is in accord with Option 1.

ISSUE FF-2
INITIATION DATE: 1987
INITIATOR: Several
STATUS: Resolved for this version of the draft proposal
To be reconsidered for future versions. RELATED ISSUES:
ENTITY REFERENCES: IMPLICIT FORM FEATURE/002, GEOMETRIC MODEL/IMPLICIT FORM FEATURE ASSOCIATION/315
DESCRIPTION The FFIM requires that an IMPLICIT FORM FEATURE/002 be applied in association with a geometric model. This precludes representing a shape wholly by implicit features. Should shape representation entirely by implicit features be permitted?

ISSUE OPTIONS & EVIDENCE:

Option 1: Retain the requirement for association with a geometric model.
Pro: Caution and inertia are the primary arguments for this option. The FFIM for this version of the draft proposal was developed with the fundamental assumption that an implicit feature representation is a "delta" to pre-existing shape. The Version 1 schedule is too tight confidently revise the FFIM in this respect.
Con: See "Pro" under Option 2.
Option 2: Revise the model to permit shape representation entirely by implicit form features.
Pro: There seems to be a trend toward representation by features. The FFIM clearly has entities which could be used to establish initial shape; i.e., the first instance of pre-existing shape.
Con: See "Pro" under Option 1.

CURRENT STATUS:
OPTION PROPOSED: Option 1
EXPLANATION: There isn't time to implement Option 2 with confidence that the FFIM will consistently reflect a new attitude in this fundamental matter.
DECISION: Continue to Reflect Option 1 in this Version of the FFIM.
Revisit the issue at the earliest opportunity.
DECISION DATE: Early 1988
ACTION: None required at present.

ISSUE FF-3
INITIATION DATE: December, 1987
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES: FF-7
ENTITY REFERENCES: FORM FEATURE/001
DESCRIPTION Should the FORM FEATURE TYPE attribute of FORM FEATURE/001 be an unconstrained alphanumeric string or from an enumerated set?

ISSUE OPTIONS & EVIDENCE:

Option 1: Unconstrained alphanumeric string

Pro: It is impossible to predict all desirable values. Feature nomenclature is not standardized and won't be any time soon. Model exchangers can bilaterally agree, to the extent needed and desired, on the nomenclature to be used and the implications thereof.

Con: Data is not information unless its meaning is agreed upon; bilateral agreement is much less useful than standardization.

Option 2: Enumerated set

Pro: Much of the utility of form features is in the classification information implicit in nomenclature. True standardization requires that the meaning of a given value of FORM FEATURE TYPE be known. Limiting the values to an enumerated set will give impetus to standardization.

Con: (i) There'll be a lot of RFC's to process. (ii) There'll be no way to insure that the nomenclature is consistent with feature shape. We'd be elevating the meaning of possibly inconsistent data.

Option 3: Make FORM FEATURE TYPE an entity. Use it as follows:

```
              FORM FEATURE
                  TYPE                          FORM FEATURE
              ------------                      ------------
              :          :                      :          :
              :          :--------------------0:          :
              :          :                      :          :
              ------------                      ------------
                    :
                    0
         ======================
         :                    :
         :                    :
 STANDARD FORM        OTHER FORM
 FEATURE TYPE         FEATURE TYPE
 -------------        -------------
 :           :        :           :
 :-----------:        :-----------:
 :(Enumerated:        :(Alpha     :
 : attribute):        : string)   :
 -------------        -------------
```

Pro: It's a compromise.

Con: (i) It's a compromise. (ii) Any change to be incorporated in this Version detracts from work on polishing and integrating the FFIM.

CURRENT STATUS:	FFIM in accord with Option 1.
OPTION PROPOSED:	Option 1 for this version. (Option 3 once a starter set of 'standard' feature type names is established.)
EXPLANATION:	There's no information value in using an enumerated set unless there are definitions or specifications which give meaning to the values in the set.
DECISION:	Option 1.
DECISION DATE:	July 14^{th}, 1988
ACTION:	None required.

ISSUE FF-4
INITIATION DATE: 1987
INITIATOR: Several
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: FORM FEATURE/001, DIMENSIONALITY-2 SHAPE ELEMENT
DESCRIPTION Categorizing FORM FEATURE/001 as a DIMENSIONALITY-2 SHAPE ELEMENT is distasteful to some. Some feature classes have obvious volumetric associations; every class in the FFIM except deformations can be viewed as an added/subtracted volume. The attitude also makes it difficult to reconcile the FFIM to CSG modeling contexts.

ISSUE OPTIONS & EVIDENCE:

Option 1: Retain the questioned categorization.

Pro: (i) Apparently every form feature can be associated with a portion of its part's "skin". Not every feature has volume associations. (ii) The idea of volume is present where appropriate within the FFIM. No clear present advantage is seen to changing attitude. (ii) In terms of the sorts of nonshape properties to be attached to them, form features seem to fit best into the DIMENSIONALITY-2 SHAPE ELEMENT category.

Con: This option does appear to make it difficult to fit features into a CSG setting in a natural way.

Option 2: Avoid categorization by dimensionality. Make FORM FEATURE/001 an immediate category of SHAPE ELEMENT.

Pro: Offers flexibility. Avoids committment where there is doubt.

Con: Introduces vagueness. Would force integration work to deal with form features as a separate problem.

Option 3: Introduce a FEATURE VOLUME entity into the FFIM and modify the model to elaborate and use it as appropriate.

Pro: Would make the volumetric associations of some features explicit and evident. Might make a "sharper" model for some readers/users.

Con: Doesn't wholly address the objection, since the categorization of FORM FEATURE/001 as a DIMENSIONALITY-2 SHAPE ELEMENT would remain. Wouldn't make a real change inside the FFIM, either, since the concept of volume is present where a need is recognized.

CURRENT STATUS:
OPTION PROPOSED: Option 3

EXPLANATION:	See "pro" for Option 3. Also, FEATURE VOLUME pulls together the FFIM's two methods of defining features by added/subtracted material - sweeps and rulings. Finally, FEATURE VOLUME can be linked to DIMENSIONALITY-3 SHAPE ELEMENT when/if there is value in doing so.
DECISION	Categorize FORM FEATURE/001 as a DIMENSIONALITY-2 SHAPE ELEMENT. Create FEATURE VOLUME entity and integrate into FFIM. Leave open future possibility of linkage between FEATURE VOLUME and DIMENSIONALITY-3 SHAPE ELEMENT.
DECISION DATE:	May 29^{th}, 1988
ACTION:	Model modified to include FEATURE VOLUME/419.

ISSUE FF-5

INITIATION DATE: 1987
INITIATOR: Several
STATUS: Resolved for this Version
RELATED ISSUES:
ENTITY REFERENCES: FORM FEATURE/001 and/or IMPLICIT FORM FEATURE/002
DESCRIPTION There should be a means to support "user defined" features.

ISSUE OPTIONS & EVIDENCE:

Option 1: Don't do it.
Pro: This sort of thing seems beyond the scope and capability of this Version, which is limited to schema specification.
Con: Given the acknowledged company- and application-dependence of features and the impossibility of including all desired features in the FFIM, there is apparent value in the ability to go beyond the features supported by the FFIM.
Option 2: Provide some means of exchanging schema extensions as well as models using the extensions.
Pro:
Con:
Option 3: Provide a form feature specification language within the schema. Presumably this language would provide for exchange of de facto schema extensions and algorithms which deal with these extensions.
Pro:
Con:
Option 4: Provide for exchange of user-defined features accompanied by "black box" software with known functionality; e.g., evaluating a feature in a BREP context.
Pro:
Con:

CURRENT STATUS:
OPTION PROPOSED: Option 1
EXPLANATION: None of the other options seems possible for this version of the draft proposal.
DECISION: Adopt Option 1 for this Version 1. Revisit the issue when this standards functionality might be extensible in the desired direction.

DECISION DATE: 1987
ACTION: None required

ISSUE FF-6

INITIATION DATE: February, 1988
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: Many involved in implicit form feature representations.

DESCRIPTION In accord with directed policy, the FFIM identifies a minimal set of attributes for the entities involved in implicit feature representation. There are other, derivable dimensional characteristics which may be toleranced. The topical model for shape tolerance information provides no way to tolerance these derivable dimensions.

ISSUE OPTIONS & EVIDENCE:

Option 1: The tolerances model should provide an entity named DERIVABLE DIMENSION, or something similar, which (i) has no attribute giving dimension value and (ii) can be toleranced. The entity would be used as a resource by the FFIM. The diagram below suggests the approach:

```
DERIVABLE
DIMENSION                            TOLERANCE
------------                        ------------
:          :                        :          :
:----------:--------------------0:----------:
:(No value :                        :Plus Tol  :
:attribute):                        :Minus Tol :
------------                        ------------
     :
     :is diagonal of
     :
     0
RECTANGLE
------------
:          :
:----------:
:Diagonal. :
:Dimension :
:  ID (FK) :
------------
```

Pro: Solves the problem.
Con: None known.

Option 2: Provide the required capability within the FFIM.
Pro: Solves the problem.
Con: Violates the scope charters. The FFIM is supposed to be limited to nominal shape. The tolerances model is supposed to provide for tolerances on nominal shape.

CURRENT STATUS: The FFIM assumes that the STIM is consistent with Option 1.

OPTION PROPOSED:	Option 1. (Requires action by tolerances committee)
EXPLANATION:	Solves the problem within PDES committee charters.
DECISION:	Option 1
DECISION DATE:	July 13th (Tolerances Committee) and 14th (Form Features Committee), 19
ACTION:	The FFIM was adapted to the details (entity and key attribute names) of the STIM.

ISSUE	FF-7
INITIATION DATE:	February, 1988
INITIATOR:	Several
STATUS:	Resolved for PDES Version 1. Should be revisited later.
RELATED ISSUES:	FF-3, FF-13
ENTITY REFERENCES:	All entities involved in implicit feature representation.
DESCRIPTION	A number of people felt that the FFIM would/should be devoted to establishing standard meanings for and specifications of terms like "hole" and "pocket". It doesn't do this. Instead, it can be viewed as a schema for constructive geometry. (This criticism seems aimed at the passages/protrusions/depressions portion of the FFIM. Deformations, transitions, and area features seem to have the desired quality.)

ISSUE OPTIONS & EVIDENCE:

Option 1: Take no action. Regard the FFIM as (i) a resource model which can be drawn upon by application-oriented models that need to make "hole" and "pocket" tangible or (ii) a start toward explicating "hole" and "pocket" within a later version of the FFIM.

Pro: There's no realistic alternative for Version 1. And it's debatable whether the FFIM has strayed from its proper objective.

Con: Could leave expectations of some disappointed.

Option 2: Augment the FFIM with some entities like "hole" and "pocket".

Pro: Would start in the direction that some feel is proper.

Con: It's probably impossible to do this in a quality way for Version 1.

CURRENT STATUS:	
OPTION PROPOSED:	Option 1, for the present.
EXPLANATION:	The FFIM may prove to have neglected a necessary objective. But a profound redirection or expansion should not be undertaken until that is clear.
DECISION:	For Version 1, do not change approach or open new area.
DECISION DATE:	March, 1988
ACTION:	None at present.

ISSUE FF-8

INITIATION DATE: April, 1988
INITIATOR: Bruce Bailey
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: IMPLICIT FORM FEATURE/002, DIMENSIONALITY-2 SHAPE ELEMENT, IMPLICIT FEATURE BOUND/029
DESCRIPTION An implicitly represented feature often cannot be "understood" (e.g., evaluated) until one or more other implicitly representation feature is "understood". The FFIM offers no clearcut way to recognize such dependencies. (Issue raised with respect to GMAP, but is equally applicable to this standard.)

ISSUE OPTIONS & EVIDENCE:

Option 1: Consider that there is no problem.
Pro: The FFIM provides a variety of ways in which one feature may be "used" in the definition of a second. This referencing implies the ordering desired.
Con: As practical matters: (i) it is tedious to infer order by examining all possible references, (ii) the references are often optional, not mandatory, and (iii) it seems unlikely that all cases of order-inferring relations have been covered.
Option 2: Provide a general means of partially ordering implicit representations as a supplement to the specific relations implying precedence.
Pro: Provides needed precedence information when specific relations (i) aren't applicable or (ii) are unnecessarily verbose.
Con: Overlapping capabilities and perhaps vagueness of meaning.

CURRENT STATUS: Option 2 is in the current FFIM.
OPTION PROPOSED: Option 2
EXPLANATION: The general precedence capability seems useful. The dangers seem minimal.
DECISION: Option 2
DECISION DATE: July 14^{th}, 1988
ACTION: None required

ISSUE FF-9

INITIATION DATE:	1987
INITIATOR:	Mark Dunn
STATUS:	Resolved
RELATED ISSUES:	
ENTITY REFERENCES:	IMPLICIT AREA FEATURE/111, FEATURE APPLICATION AREA/6
DESCRIPTION	Should it be required or optional that an IMPLICIT AREA FEATURE/111 be identified with the DIMENSIONALITY-2 SHAPE ELEMENT on which it is installed?

ISSUE OPTIONS & EVIDENCE:

Option 1: Required
Pro: If the DIMENSIONALITY-2 SHAPE ELEMENT is not known, the location of the feature is unknown and the feature fails the basic test for implicit representations.
Con: See "Pro" for Option 2.
Option 2: Optional
Pro: Useful information is given even if all desirable information isn't.
Con: See "Pro" for Option 1.

CURRENT STATUS:	
OPTION PROPOSED:	Option 1
EXPLANATION:	Option 1 is consistent with FFIM approach.
DECISION:	Option 1
DECISION DATE:	May 6^{th}, 1988
ACTION:	FFIM has been revised to reflect Option 1.

ISSUE	FF-10
INITIATION DATE:	1987
INITIATOR:	Mark Dunn
STATUS:	Resolved
RELATED ISSUES:	
ENTITY REFERENCES:	Many involved in implicit feature representation.
DESCRIPTION	Should an implicit feature representation be uniquely located or allowed multiple locations? The FFIM could be designed (i) to distinguish carefully between the intrinsic shape of an implicit feature and its location and (ii) to allow repeated use of the intrinsic shape at different locations. For example, a single IMPLICIT EDGE BLEND/037 could be associated with many EDGE BLENDED INTERSECTIONs/359.

ISSUE OPTIONS & EVIDENCE:

Option 1: Single location.

Pro: (i) Gives the FFIM a clear conceptual viewpoint: an IMPLICIT XXXXX is a single occurrence of an XXXXX. (ii) For features located via an AXIS PLACEMENT/GEO-4, REPLICATE FORM FEATURE/067 provides the ability to 'reuse' an implicit representation.

Con: See "Pro" for Option 2.

Option 2: Multiple locations

Pro: Introduces an efficiency that may never emerge on path to implementation.

Con: (i) Efficiency is irrelevant in conceptual model. (ii) Model provides for replication, which is terse and aimed at "same feature, different location" situation.

CURRENT STATUS:	FFIM is designed in spirit of Option 1.
OPTION PROPOSED:	Option 1
EXPLANATION:	Weight of arguments above
DECISION:	Option 1, with the reservation that the FFIM should not be made more cumbersome in order to preclude 'multiple use'.
DECISION DATE:	July 14^{th}, 1988
ACTION:	None required

ISSUE FF-11

INITIATION DATE:	May, 1987
INITIATOR:	Mark Dunn
STATUS:	Resolved
RELATED ISSUES:	FF-9
ENTITY REFERENCES:	IMPLICIT THREAD/025, IMPLICIT KNURL/024
DESCRIPTION	The FFIM provides no way to "time" threads and knurls; i.e., control their rotational location. This is done for gears, splines, and curvic couplings.

ISSUE OPTIONS & EVIDENCE:

Option 1: Provide entities (similar to GEAR TIMER/357, SPLINE TIMER/358, and COUPLING TIMER/420) for timing threads and knurls.

Pro: Location of threads and knurls is not fully known without these – features are "free to rotate". With "Z" relations, timing will not be required when not of interest.

Con: It seems unlikely anyone would want to time knurls. Need to time threads hasn't been asserted.

Option 2: Do not provide for thread and knurl timing.

Pro: See "Con" under Option 1.

Con: See "Pro" under Option 1.

Option 3: Provide for thread timing, but not knurl timing.

Pro: The function of knurls is to provide texture or roughness to surfaces, so it seems unlikely that timing is needed. It seems reasonable that someone would need to time threads.

Con: Argument is based on instinct rather than knowledge.

CURRENT STATUS:	There is no provision for timing threads and knurls.
OPTION PROPOSED:	Option 3
EXPLANATION:	A poll revealed that threads are sometimes timed, but disclosed no need to time knurls.
DECISION:	Option 3
DECISION DATE:	July 14^{th}, 1988
ACTION:	Add thread timing capability to FFIM.

ISSUE	FF-12
INITIATION DATE:	1987
INITIATOR:	Mark Dunn
STATUS:	Resolved
RELATED ISSUES:	
ENTITY REFERENCES:	IMPLICIT DEFORMATION/064 and descendents
DESCRIPTION	Deformations can make major, non-localized changes in a shape. (Consider a bend in mid-sheet.) This being so, there is danger in allowing implicit representation of deformations in an environment where the features may or may not be "realized" in the geometric model. And there is danger in allowing stack-up of implicit representations.

ISSUE OPTIONS & EVIDENCE:

Option 1: Impose constraints that reduce the potential for confusion.
Pro: Constraints could prevent some poor practice and potential confusion.
Con: Any general constraint that has been considered would outlaw tactics that are desirable in cases. A lot of specific constraints could be more confusing than the problem they aim to prevent. Enforcability of non-simple constraints seems unlikely, anyway.

Option 2: Depend on user good sense and communication.
Pro: No other course seems workable.
Con:

CURRENT STATUS:	
OPTION PROPOSED:	Option 2
EXPLANATION:	No better course is known.
DECISION:	Option 2
DECISION DATE:	1987
ACTION:	None required.

ISSUE FF-13

INITIATION DATE: 1987
INITIATOR: Several
STATUS: Resolved
RELATED ISSUES: FF-7
ENTITY REFERENCES: GENERAL PROFILE/162, GENERAL FEATURE SWEEP PATH/???, FEATURE RULING/415.
DESCRIPTION Should the FFIM limit itself to a limited set of "popular" feature shapes or provide a powerful means of identifying, specifying, and type-labeling portions of shape?

ISSUE OPTIONS & EVIDENCE:

Option 1: Limited set.
Pro: Clarity of form features in PDES. Small set of clearly useful entities. Start on firm ground and grow as (and if) experience indicates.
Con: There is widespread sentiment that general constructs are needed in a constructive form features (rather than geometric modeling) context. Each of the questionable entities has proponents.

Option 2: Include general entities.
Pro: The general capabilities are used in current CAD practice.
Con: Yes, but in a form features environment? Is the sweeps and rulings portion of the FFIM significantly different from a trimmed surfaces geometric model with type-labels on surfaces; i.e., from a surfaced wireframe with enumerative features?

CURRENT STATUS:
OPTION PROPOSED: Option 2
EXPLANATION: The powerful entities are clearly needed somewhere. In some contexts, at least, there is clear value in having them in an implicit features mechanism.
DECISION: Adopt Option 2
DECISION DATE: May 5^{th}, 1988
ACTION: None required.

ISSUE	FF-14
INITIATION DATE:	1987
INITIATOR:	Several
STATUS:	Resolved for Version 1.
RELATED ISSUES:	FF-7
ENTITY REFERENCES:	FORM FEATURE COMPOSITION/319
DESCRIPTION	With FORM FEATURE COMPOSITION/319, The FFIM offers a general way to model compound features. But the model doesn't have entities specifically for common compounds; e.g., threaded hole.

ISSUE OPTIONS & EVIDENCE:

Option 1: Specifically include common compounds; e.g., have a THREADED HOLE entity.

Pro: It would be convenient if there were entities in the schema that, for example, linked hole and thread entities according to stated conventions to make a threaded hole.

Con: The FFIM's structure isn't congenial to this sort of entity association. There's no hole entity, only a way to specify a sweep and call it a hole.

Option 2: Don't include common compounds.

Pro: The FFIM is a tool kit for specifying portions of shape and applying creator-chosen type-labels to those portions of shape. The compounding capability of FORM FEATURE COMPOSITION/319 is consistent with that trait. The creator can group a hole and a thread and call the grouping a "threaded hole".

Con:

CURRENT STATUS:

OPTION PROPOSED:	Option 2
EXPLANATION:	The issue here is essentially the same as Issue FF-7, and should be resolved in the same way.
DECISION:	Option 2
DECISION DATE:	1987
ACTION:	None needed.

ISSUE	FF-15
INITIATION DATE:	April, 1988
INITIATOR:	Kim Perlotto
STATUS:	Resolved
RELATED ISSUES:	
ENTITY REFERENCES:	IMPLICIT GEAR/026, IMPLICIT SPLINE/027, and descendents.
DESCRIPTION	The initiator, who is also author of the questioned portion of the FFIM, feels that portion is shaky in many of its details.

ISSUE OPTIONS & EVIDENCE:

Option 1:	Reexamine gears and splines in detail, with expert assistance.
Pro:	Good coverage of implicit gears and splines is very important to the FFIM. Explicit representations of these features are impractical.
Con:	
Option 2:	Delete implicit gears and splines from model.
Pro:	Better absent than wrong.
Con:	
Option 3:	Leave as-is.
Pro:	Needed improvements are more likely to be identified and made if the current material is retained.
Con:	

CURRENT STATUS:	Suspect material has been removed from FFIM.
OPTION PROPOSED:	Option 1 if possible. If not, get PDES management decision between option 2 and option 3.
EXPLANATION:	Obvious
DECISION:	Option 2 (omission) for the present. Pursue Option 1 and, if successful, try to get quality material into PDES Version 1.
DECISION DATE:	July 14, 1988
ACTION:	FFIM Intensify effort to seek expert help in sharpening gear and spline material.

ISSUE FF-16
INITIATION DATE: 1987
INITIATOR: Several
STATUS: Resolved for PDES Version 1.
RELATED ISSUES:
ENTITY REFERENCES:
DESCRIPTION The FFIM covers three classes of volumetric features - passages, depressions, protrusions. Three additional classes would fill out this set neatly: voids, doubly connected protrusions (e.g., cup handles), and connectors (e.g., the bar of a barbell). Should these be added?

ISSUE OPTIONS & EVIDENCE:

Option 1: Yes.
Pro: Features of these sorts exist. The FFIM's volumetric feature sector (sweeps and rulings) would support the additional classes readily.
Con:
Option 2: No
Pro: The additional classes are clearly less useful than those already covered. It seems wise to await experience with the classes covered before extending.
Con:

CURRENT STATUS:
OPTION PROPOSED: Option 2.
EXPLANATION: See "Pro" under Option 2.
DECISION: Option 2
DECISION DATE: 1987
ACTION: None required.

ISSUE FF-17

INITIATION DATE: 1987
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: IMPLICIT FORM FEATURE/002, GEOMETRIC MODEL/IMPLICIT FORM FEATURE ASSOCIATION/315
DESCRIPTION Should the cardinality of the relation between these two entities be "1 or many" or "0, 1, or many"?

ISSUE OPTIONS & EVIDENCE:

Option 1: "1 or many"
Pro: The attitude of the FFIM is that an implicit representation of a portion of a model of product shape. The representation should be fully understandable in the context of the overall model of shape. This understandability is not present when the feature is modeled in isolation.
Con:
Option 2: "0, 1, or many"
Pro: There is value in knowing that a feature is present and knowing some of its attributes, even if there is no geometric model of the product. For example, classification coding could use the isolated feature data.
Con:

CURRENT STATUS:
OPTION PROPOSED: Option 2
EXPLANATION: This option seems consistent with the spirit of PDES. Providing for odds and ends of product data isn't the goal.
DECISION: Option 2
DECISION DATE: 1987
ACTION: FFIM has "1 or many" relation.

ISSUE FF-18
INITIATION DATE: 1987
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: IMPLICIT KNURL/024
DESCRIPTION Knurls are modeled by describing the knurling tool. Is this good?

ISSUE OPTIONS & EVIDENCE:

Option 1: Yes.
Pro: This seems a practical approach. (i) Knurls are on the borderline of 'microshape'. It seems unlikely that there would be much interest in computing their shape. (ii) If desired, it doesn't seem difficult to compute nominal knurl shape from tool data plus work blank data.
Con: In response: (i) a description of the knurl itself would be more consistent with the FFIM's approach and (ii) it is doubtful that tool parameters produce knurl parameters in a simple way; e.g., the groove depth of the knurl is probably less than the tooth depth of the tool, with the difference dependent on several factors.
Option 2: No.
Pro: See above.
Con: See above.

CURRENT STATUS:
OPTION PROPOSED: Option 1
EXPLANATION: Pragmatism. Tool data seems the more useful and easily obtained data.
DECISION: Option 1
DECISION DATE: 1987
ACTION: FFIM employs Option 1.

ISSUE FF-19

INITIATION DATE: 1987
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES: FF-9
ENTITY REFERENCES: IMPLICIT KNURL/024
DESCRIPTION Should work blank diameter be an attribute of IMPLICIT KNURL/024?

ISSUE OPTIONS & EVIDENCE:

Option 1: Yes.
Pro: It is useful data which may not be available elsewhere.
Con: It is redundant with the FEATURE APPLICATION AREA/028 of the parent IMPLICIT AREA FEATURE/111, if present.
Option 2: No
Pro: The attribute limits knurls to cylindrical surfaces. There is no obvious reason for that limitation.
Con:

CURRENT STATUS:
OPTION PROPOSED: Option 2
EXPLANATION: The resolution of issue FF-9 requires that the area on which the knurl is applied be known. Hence the attribute in question adds no value.
DECISION: Option 2
DECISION DATE: May 19th, 1988
ACTION: FFIM modified (attribute deleted).

ISSUE FF-20

INITIATION DATE: 1987
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: IMPLICIT EDGE ROUND/037
DESCRIPTION Should the entity have an attribute indicating whether the round is convex or concave?

ISSUE OPTIONS & EVIDENCE:

Option 1: Yes.
Pro: The information is clearly of interest.
Con: The information is redundant, being determinable from context.
Option 2: No
Pro: See above.
Con: See above.

CURRENT STATUS:
OPTION PROPOSED: Option 2
EXPLANATION: The entity (or its parent, IMPLICIT EDGE BLEND/308) is used in two circumstances: (i) to blend two identified DIMENSIONALITY-2 SHAPE ELEMENTs and (ii) in a variety of special situations. In the first case, analysis of the intersection of the DIMENSIONALITY-2 SHAPE ELEMENTs would seem to readily determine whether it is concave or convex. In the second, the knowledge is immediately available from context.
DECISION: Option 2
DECISION DATE: 1987
ACTION: Attribute is not in FFIM.

ISSUE # FF-21

INITIATION DATE: 1987

INITIATOR: Peter Wilson

STATUS: Resolved

RELATED ISSUES:

ENTITY REFERENCES: IMPLICIT EDGE FLAT/036

DESCRIPTION The majority of chamfers are apparently symmetric; i.e., have the same setback and angle with respect to each of the two blended surfaces. Should this popular, simple case be handled separately from the general case?

ISSUE OPTIONS & EVIDENCE:

Option 1: Just handle the general case.

Pro: The general case is relatively uncomplicated, so there's no motive for splitting into two cases.

Con: Perhaps the general case isn't complex in terms of entities and attributes, but there's a tedious set of conventions for establishing the reference "surface" of the feature depending on context. These conventions imply programming that isn't trivial.

Option 2: Handle only the symmetric case.

Pro: Covers the majority of cases in a very simple way.

Con: Excludes a lot of instances; e.g., countersinks other than 45 degrees.

Option 3: Provide distinct entities for symmetric and asymmetric edge flats.

Pro: Handles simple cases simply while providing for the more complex cases.

Con: The worst of both worlds. Would lengthen the FFIM without avoiding the need for the conventions mentioned above.

CURRENT STATUS:

OPTION PROPOSED: Option 1

EXPLANATION: Option 3 has no merit, except perhaps that it would facilitate easy partial implementation of edge flats. Option 2 would exclude many instances that aren't hard to handle.

DECISION: Option 1

DECISION DATE: 1987

ACTION: The FFIM incorporates Option 1.

ISSUE FF-22

INITIATION DATE: May 19th, 1988
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: IMPLICIT FORM FEATURE/002 and descendents.
DESCRIPTION The FFIM policy is that there should be enough information in an implicit representation to 'compute' shape with the feature included. Does this imply an obligation to exhibit logic/mathematics for doing so?

ISSUE OPTIONS & EVIDENCE:

Option 1: Yes, at least in non-trivial cases.
Pro: (i) There is danger of misinterpretation and inconsistent use of PDES if this is not done. (ii) Doing so enforces discipline on the FFIM, insuring that it is true to its policy.
Con:
Option 2: No
Pro: (i)The great majority of the implied 'algorithms' are obvious, at least in outline. (ii) The 'computable' criterion is more a matter of intellectual honesty than practical necessity. Implicit representations are often useful without being explicited. (Anyone who needs a Brep of a gear to make it is in trouble.) Hopefully, this will become more true as time passes. (iii) Industry often designs, makes, and inspects features without specifying or being able to compute all the details of shape.
Con:

CURRENT STATUS: There is little of the suggested material in the FFIM.
OPTION PROPOSED: Include such material, on a need basis, as an aspect of model refinement or when review uncovers ambiguity. (See IMPLICIT TWIST/082 as an example.) Employ illustrations heavily to avoid ambiguity. Do not, though, institute a general requirement to exhibit the logic/mathematics of realizing all implicit features.
EXPLANATION: Unnecessary material would greatly increase the already fearsome length of the FFIM. And it's probably not possible to find the resources to compose such material in a timely manner.
DECISION: Option 2
DECISION DATE: July 14^{th}, 1988
ACTION: None required

ISSUE FF-23
INITIATION DATE: 1988
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: Many descendents of IMPLICIT DEFORMATION/064.
DESCRIPTION Assuming constant material thickness, a bend will have different radii on its convex and concave sides. Yet it is practice to specify only one radius. How to handle this?

ISSUE OPTIONS & EVIDENCE:

Option 1: For any bend radius specification, provide a CONCAVE/CONVEX switch.
Pro: A general solution.
Con: Probably calls for data where data is not needed or wanted in the great majority of cases.
Option 2: Adopt a general convention that all bend radii are measured on the concave side of the bend. (Concave is chosen on the rationale that formtools positioned on the concave side of the bend are the primary determinants of bend radius.)
Pro:
Con:
Option 3: Handle on a case-by-case basis. For each use of a bend radius attribute, do one of three things: (i) state that radius is measured on concave side, (ii) state that radius is measured on convex side, or (iii) provide switch.
Pro:
Con:
Option 4: Say nothing about the matter in the FFIM.
Pro: The ideal solution, assuming that (i) the difference between the two radii is industrially insignificant or (ii) the measured radius is known from context.
Con: It's hard to believe that one of the two conditions above will apply in every case.
Option 5: Make it a user option to specify concave/convex or say nothing. This could be done as follows.

```
BEND RADIUS                          IMPLICIT XXX
------------                        ------------
:----------:     is radius of      :          :
:(Value)   :-------------------0:          :
:          :                       :          :
------------                        ------------
      :
      :
      :Z
      0
 BEND RADIUS
 MEASUREMENT
 SIDE
 ------------
 :----------:
 :Concave or:
 :  Convex  :
 ------------
```

Pro: Maximum flexibility solution. Makes the FFIM totally non-commital.

Con: A copout if there is a more constrained treatment that works

CURRENT STATUS: The FFIM employs Option 2.

OPTION PROPOSED: Option 1, with the additional choice that the dimension be measured at bend centersurface/centerline.

EXPLANATION: Polling indicates that practice varies depending on company, process, and other factors. It's best to leave maximum flexibility.

DECISION: Option 1, with the addition indicated under 'OPTION PROPOSED'.

DECISION DATE: July 14^{th}, 1988

ACTION: Create entity BEND RADIUS which gives the desired information. Use that entity as the source of all bend radius dimensions.

ISSUE FF-24

INITIATION DATE: March 29th, 1988

INITIATOR: Mark Dunn

STATUS: Resolved

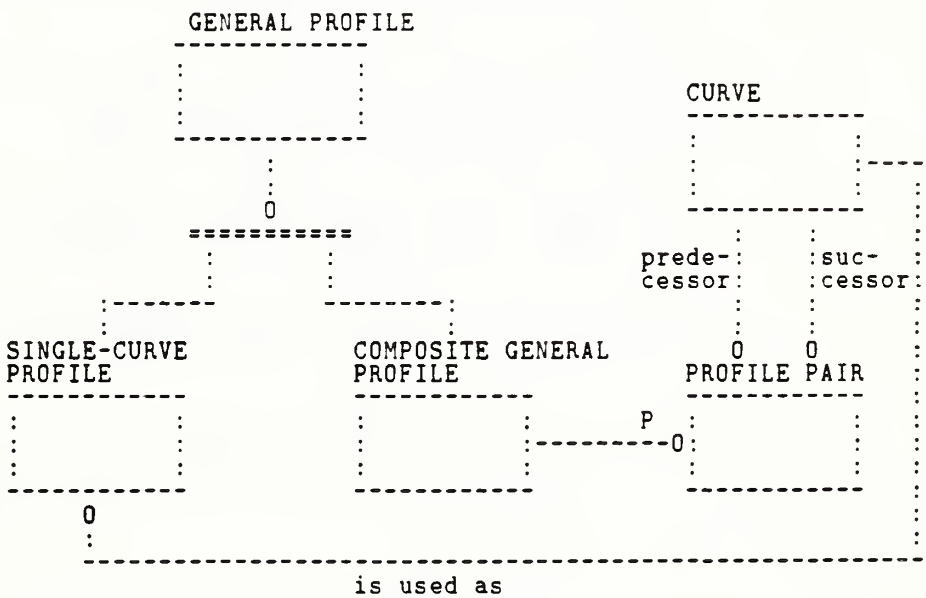
RELATED ISSUES:

ENTITY REFERENCES: GENERAL PROFILE/162, PROFILE PAIR/165, PROFILE BLEND/166

DESCRIPTION The function of this entity is to provide for arbitrary profiles in sweeping. As presently modeled, it does not allow for a single curve to be used as a general profile, only sequences of two or more curves. (In order to provide for blending of profile segments, the FFIM built its own composite curve mechanism. A GENERAL PROFILE/162 is a set of PROFILE PAIRs/165 which give a sequence of CURVEs.)

ISSUE OPTIONS & EVIDENCE:

Option 1: Modify the model as follows to allow for single-curve profiles.



Pro: Does the job.

Con: Verbose. Unnecessary if Geometry people would produce a composite curve with accessible segment intersections.

Option 2: Modify the model as follows to allow for single-curve profiles.

```
 CURVE                                    GENERAL PROFILE
------------                             ------------
:          :   is first                  :          :
:          :-------------------------0:          :
:          :   curve of                  :          :
------------                             ------------
  :      :                                     :
  :      :                                     :has curves after
  :      :                                     :first given by
  :      :                                     :
  :      :                                     0
  :      :                               PROFILE PAIR
  :      :is successor in               ------------
  :      --------------------------------0:          :
  :is predecessor in                     :          :
  ---------------------------------------0:          :
                                         ------------
```

Pro: Does the job tersely.

Con: A bit more obscure than Option 1. May be shaky at some normalization level (big deal!). Unnecessary if Geometry people would produce composite curve with accessible segment intersections.

Option 3: Leave as-is until/unless it is clear that integration with Geometry won't render the issue moot.

Pro: Ideal solution if integration effort produces solution.

Con: May merely be postponement of dealing with problem.

Option 4: Leave as-is.

Pro: It's not clear that there's much demand for single-curve profiles not available as 'standard' profiles. If this must be done, the curve could be handled as two segments.

Con: Splitting a curve in two in order to fit into the schema is objectionable.

CURRENT STATUS:	FFIM is as in DESCRIPTION.
OPTION PROPOSED:	Option 2, if integration work doesn't solve the problem.
EXPLANATION:	Seems the best way to handle the situation within the FFIM.
DECISION:	Option 2
DECISION DATE:	July 14^{th},1988
ACTION:	FFIM modified as indicated under Option 2.

ISSUE FF-25

INITIATION DATE:	1987
INITIATOR:	Mark Dunn
STATUS:	Resolved
RELATED ISSUES:	FF-26
ENTITY REFERENCES:	IMPLICIT CORNER FLAT/310
DESCRIPTION	The flat is merely modeled by identifying a plane. It seems preferable to give a 'local' specification in terms of the things blended, as is done with IMPLICIT EDGE FLAT/036.

ISSUE OPTIONS & EVIDENCE:

Option 1: Provide local specification.
Pro: Seems compatible with 'local' nature of the feature.
Con: There's no obvious local definition.
Option 2: Leave as-is.
Pro: Generality.
Con: Seems only a small step removed from trimmed surface representation.

CURRENT STATUS:	As in DESCRIPTION.
OPTION PROPOSED:	Option 1, in a way that reduces the number of cases covered. A flat will be specified by (i) identifying a planar surface at the corner, (ii) two setbacks, one along each edge emanating from the corner, and (iii) angle with the planar surface.
EXPLANATION:	See "Pro" under Option 1. The loss of generality seems less important than fitting the specification to the situation.
DECISION:	Option 1, as described under OPTION PROPOSED.
DECISION DATE:	July 14^{th}, 1988
ACTION:	Modify FFIM as described under OPTION PROPOSED.

ISSUE FF-26

INITIATION DATE: 1987
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES: FF-25
ENTITY REFERENCES: IMPLICIT CORNER ROUND/311
DESCRIPTION The round is merely modeled by identifying a sphere. It seems preferable to give a 'local' specification in terms of the things blended, as is done with IMPLICIT EDGE ROUND/037.

ISSUE OPTIONS & EVIDENCE:

Option 1: Provide local specification.
Pro: Seems compatible with 'local' nature of the feature.
Con: There's no obvious local definition.
Option 2: Leave as-is.
Pro: Generality.
Con: Seems only a small step removed from trimmed surface representation.

CURRENT STATUS: As in DESCRIPTION.
OPTION PROPOSED: Option 1
EXPLANATION: Des. for local specification.
DECISION: Option 1
DECISION DATE: July 14^{th}, 1988
ACTION: Give a local specification for a simple outside corner round. Omit inside corner rounding, which is usually a consequence of the fillets on the edges emanating from the vertex.

ISSUE FF-27

INITIATION DATE: April 29th, 1988

INITIATOR: Mark Dunn

STATUS: Resolved

RELATED ISSUES:

ENTITY REFERENCES: IMPLICIT MARKING/406

DESCRIPTION The entity provides for only an indication of the desired characterstring and the area to be marked. There's nothing on font, character size, raised/lowered requirements, etc.

ISSUE OPTIONS & EVIDENCE:

Option 1: Make no change unless some other model provides for the information needed.

Pro: Provides a needed capability, the ability to indicate that there is to be textual data marked on a specified area of the part.

Con: Stretches the scope of the FFIM.

Option 2: If another committee produces something usable that provides additional data, link to it. If not, do nothing.

Pro: Text is going too far from the originally intended scope. There has been enough text exchange activity that original work by the Form Features group should be unnecessary.

Con: Doesn't insure needed coverage. Large lettering, in high or low relief, can be a major aspect of the shape of some products.

Option 3: Retain the current entity for unconstrained annotation on products. Cover big lettering as swept volumes, augmented by text strings that 'translate' the swept profile geometry into text.

Pro: Extends coverage to large lettering without major expansion into text standardization. There would be decent coverage of what are probably the two priority cases: non-aesthetic marking for identification and similar purposes and lettering that is a factor in product shape.

Con: A simple indication that a given sweep produces a given text is not fully 'sensible'.

Option 4: Delete the current entity for unconstrained annotation on products. Cover big lettering as swept volumes, augmented by text strings that 'translate' the swept profile geometry into text.

Pro: Covers lettering that is a factor in product shape. Omits marking that is not in scope of FFIM, which is aimed at product shape.

Con: It's not clear some other committee will cover the needed marking requirements.

CURRENT STATUS: See DESCRIPTION.

OPTION PROPOSED: Option 1

EXPLANATION: Provides a needed capability without going overboard into a major new area which would better be explored by another committee.

DECISION: Option 1

SECTION 4: FORM FEATURES INFORMATION MODEL

DECISION DATE:	July 14^{th}, 1988
ACTION:	None required

ISSUE FF-28

INITIATION DATE:	1987
INITIATOR:	Mark Dunn
STATUS:	Resolved
RELATED ISSUES:	
ENTITY REFERENCES:	IMPLICIT FEATURE BOUND/029
DESCRIPTION	Should the 'bound' concept be extended to permit bounding of explicit features?

ISSUE OPTIONS & EVIDENCE:

Option 1: No.

Pro: (i) If a feature is explicitly represented, its bounds are easily determinable. (ii) There's a vagueness in explicit features in the sense that there's no precise information on the 'pattern' to which the feature corresponds. To associate bounds with an explicit representation would compound the vagueness; there'd be an uncertain bounding role as well as an uncertain bounded configuration. The data would not be 'information' because its meaning would be unclear.

Con: See 'pro' for Option 2.

Option 2: Yes.

Pro: There's insufficient evidence to assert that the bounding data would be trivially redundant and/or incomprehensible. This is probably true for 'higher level' geometric models (e.g., Brep), but not at all clear for less intelligent ones (e.g., wireframe). Bound data might prove useful in communicating the 'sense' of explicit features, especially if information on the nature of the bound were provided for.

Con: See 'pro' for Option 1.

CURRENT STATUS:

OPTION PROPOSED:	Option 1
EXPLANATION:	Intuitively, the argument for Option 1 seems valid. It seems unwise to provide for information of dubious value and potential mischief unless a clear need is recognized.
DECISION:	Option 1
DECISION DATE:	1987
ACTION:	FFIM provides only for declaration of bounds of implicit features. This is done in such a way that the relationship of the feature and its bound is stereotyped.

ISSUE FF-29
INITIATION DATE: 1987
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: IMPLICIT BEND/084 and categories.
DESCRIPTION Two issues re IMPLICIT BLENDs/084. First, what information should be modeled to show the direction of the bend? That is, which of the two possible bend directions applies? Second, if the bend is to be explicited, on which side of the bendline should material be displaced? (Alternately, if the bend is already explicited, which side moved?)

ISSUE OPTIONS & EVIDENCE:

Option 1: Indicate bending direction by a switch telling whether the bendline is on the concave or convex side of the bend. Optionally indicate displaced side by indicating left/right side of bendline curve.
Pro: Terse. Does the job.
Con: Requires that bendline direction be carefully chosen. A left/right indicator is meaningful only if the bendline is embedded in a surface. Needs convention to be applicable to IMPLICIT CUTOUT FLANGE/094.
Option 2: Indicate bending direction by a switch telling whether the left- or right-hand rule applies. Optionally indicate displaced side by indicating left/right side of bendline curve.
Pro: Allows bendline, which has to exist before bend can exist, to have either possible direction.
Con: Terse. Again, the left/right indicator is meaningful only if the bendline is embedded in a surface. Needs convention to be applicable to IMPLICIT CUTOUT FLANGE/094.

CURRENT STATUS: The FFIM uses a vector to give direction and has no way of indicating the displaced side, except that it is implicit that the material near the cutout is displaced for an IMPLICIT CUTOUT FLANGE/094.
PROPOSED: Option 2
EXPLANATION: Less interdependence of information.
DECISION: Option 2, with appropriate special handling of IMPLICIT CUTOUT FLANGE/094.
DECISION DATE: July 14^{th}, 1988
ACTION: Model modified accordingly.

ISSUE FF-30

INITIATION DATE: June 1st, 1988
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUES:
ENTITY REFERENCES: IMPLICIT CORNER RIB/089
DESCRIPTION Should the feature be classified as an IMPLICIT EMBOSS/081 or an IMPLICIT PARTIAL CUTOUT/083?

ISSUE OPTIONS & EVIDENCE:

Option 1: Emboss.
Pro:
Con:
Option 2: Partial cutout
Pro:
Con:

CURRENT STATUS: IMPLICIT CORNER RIB/089 is a category of IMPLICIT EMBOSS/081 because this is the way it's classified in the CAM-I feature hierarchy.

OPTION PROPOSED: Option 2
EXPLANATION: No compelling reason to change.
DECISION: Option 2
DECISION DATE: July 14th, 1988
ACTION: None required

ISSUE FF-31
INITIATION DATE: 1987
INITIATOR: Mark Dunn
STATUS: Resolved
RELATED ISSUE: FF-22
ENTITY REFERENCES: IMPLICIT LOUVER/090
DESCRIPTION The data for this entity are not sufficient to determine feature shape, unless there are underlying assumptions.

ISSUE OPTIONS & EVIDENCE:

Option 1: Add parameters as necessary, so that feature shape is fully specified.
Pro: Keeps FFIM consistent with ideal of fully specifying feature shape.
Con: This seems to be a case where fully detailed shape specification is not usual and requiring necessary data would be undesirable. (See discussion of Issue FF-22.)

Option 2: Keep as-is.
Pro: See 'Con' above.
Con: See 'Pro' above.

CURRENT STATUS: Option 2
OPTION PROPOSED: Option 2
EXPLANATION: Weight of evidence
DECISION: Option 2
DECISION DATE: July 14th, 1988
ACTION: None required

ISSUE FF-32

INITIATION DATE: 1987

INITIATOR: Mark Dunn

STATUS: Resolved for PDES Version 1.

RELATED ISSUES:

ENTITY REFERENCES:

DESCRIPTION Should the FFIM provide feature-specific entities for explicit representations, as it does for implicit? As an example, there might be an EXPLICIT RECTANGULAR POCKET entity as follows:

```
DIMENSIONALITY-2
SHAPE ELEMENT
...............
:             :.........................
:             :   is wall1 of          :
:             :                        :
:             :...................     :
:.............:   is wall2 of    :     :
  : : : : : : :                  :     :
  : : : : : : :is                :     :
  : : : : : : :wall3             0     0
  : : : : : : :of               EXPLICIT RECT-
  : : : : : : :                 ANGULAR POCKET
  : : : : : : :                 ...............
  : : : : : : :................0:             :
  : : : : : :                   :             :
  : : : : : :..................0:             :
  : : : : :    is wall4 of      :             :
  : : : : :....................0:.............:
  : : :        is bottom of       :    :    :
  : : :                           :    :    :
  : : :........     ..............:    :    :.......
  : :       is:     :has           .....:          :has
  : :     used:     :              :  has          :
  : :       as:     :Z             :Z              :Z
  : :         0     0              0               0
  : :     EXPLICIT RECT-    EXPLICIT RECT-    EXPLICIT RECT-
  : :     ANGULAR POCKET    ANGULAR POCKET    ANGULAR POCKET
  : :     ENTRY ROUND       WALL/WALL FILLET  WALL/BOTTOM
  : :     ..............    ..............    FILLET
  : :     :            :    :            :    ...............
  : :     :            :    :            :    :             :
  : :     :............:    :............:    :             :
  : :                             0           :.............:
  : :.............................:                 0
  :          is used as                             :
  :.................................................:
                   is used as
```

ISSUE OPTIONS & EVIDENCE:

Option 1: Yes.

Pro:	Intuitively, it seems that stereotyping of oft-occurring configurations in geometric models would be useful. Models like the one above would 'translate' geometric model entities into a recognized pattern. Determination of parameters of interest (e.g., the length, width, and depth of the rectangular pocket) would be facilitated.
Con:	Explicit representations are less stereotype-able than implicit ones because they are more susceptible to variation due to the geometric environment of the feature. This doesn't mean that they're undesirable, but rather that they will present a tougher modeling problem.
Option 2:	No
Pro:	See above.
Con:	See above.

CURRENT STATUS:	
OPTION PROPOSED:	Option 2 ('No') for this Version. Reopen the question in future.
EXPLANATION:	It isn't possible to address an entire new area with confidence in time for Version 1. The FFIM now provides for explicit representations in a non-feature-specific way. That will have to do until the matter can be given the attention it deserves.
DECISION:	Option 2
DECISION DATE:	1987
ACTION:	None required.

4.6 SML MODEL

(This appendix is not available at this time.)

4.7 EXPRESS MODEL

```
SCHEMA form_features;
 EXPORT EVERYTHING;
 ASSUME (integration,geometric_models,curves_and_surfaces,tolerances);
 USES(
   point,
   line,
   vector,
   vector_with_magnitude,
   surface,
   axis_placement,
   curve,
   bounded_curve,
   curve_on_surface,
   geometric_model,
   dimensionality_2_shape_element,
   edge_shape_element,
   independent_size_parameter,
   independent_angle_size_parameter,
   derivable_size_parameter,
   derivable_angle_size_parameter);


 TYPE

 bound_types = ENUMERATION OF (start_bound,
                               end_bound,
                               volumetric_bound);

 coordinate_enumeration = ENUMERATION OF (x_coordinate,
                                          y_coordinate,
                                          z_coordinate);

 neck_direction_types = ENUMERATION OF (neck_positive, neck_negative);

 ell_orientation_types = ENUMERATION OF (ell_positive, ell_negative);

 taper_types = ENUMERATION OF (taper_increasing, taper_decreasing);

 feature_end_types = ENUMERATION OF (initial_end, terminal_end);

 hands = ENUMERATION OF (left_hand, right_hand);
```

```
thread_fit_classes = ENUMERATION OF (one_a, two_a, three_a,
                                     one_b, two_b, three_b);

thread_forms = ENUMERATION OF (sharpv, unified, sixty_degree_stub, acme,
                               square, stub_acme, buttress, knuckle,
                               brit_std);

threading_direction_types = ENUMERATION OF (thread_upto, thread_beyond);

coupling_shape_types = ENUMERATION OF (concave_coupling, convex_coupling);

bend_measurement_types = ENUMERATION OF (concave_measurement,
                                         convex_measurement,
                                         center_measurement);

tube_bend_moved_end_types = ENUMERATION OF (tube_forward_end,
                                            tube_rearward_end);

set_of_profile_pairs = set [0:#] of profile_pair;

END_TYPE;



FUNCTION cylindrical(area:dimensionality_2_shape_element):LOGICAL;
-- determines if an area is cylindrical
END_FUNCTION;

FUNCTION conical(area:dimensionality_2_shape_element):LOGICAL;
-- determines if an area is conical
END_FUNCTION;

FUNCTION planar(pairs:set_of_profile_pairs):LOGICAL;
-- determines if a curve string is planar
END_FUNCTION;

FUNCTION sequenced(initial:curve;subsequent:set_of_profile_pairs):LOGICAL;
-- determines if a GENERAL PROFILE is made up of correctly ordered
-- initial curve and subsequent curve pairs
END_FUNCTION;

FUNCTION closed(pairs:set_of_profile_pairs):LOGICAL;
-- determines if a curve string is closed
END_FUNCTION;
```

```
--FF-001
ENTITY form_feature;
  feature_type  : STRING(80);
  implicit_reps : SET [0:#] OF implicit_form_feature;
  pattern_rep   : OPTIONAL implicit_form_feature_pattern;
  replicate_rep : OPTIONAL replicate_form_feature;
END_ENTITY;


--FF-002
ENTITY implicit_form_feature
SUPERTYPE OF (implicit_area_feature OR
              implicit_deformation OR
              implicit_depression OR
              implicit_passage OR
              implicit_protrusion OR
              implicit_transition);
END_ENTITY;


--FF-315
ENTITY geometric_model_implicit_form_feature_assn;
  augmented_geometric_model         : geometric_model;
  augmenting_implicit_form_feature : implicit_form_feature;
END_ENTITY;


--FF-419
ENTITY implicit_feature_precedence;
  predecessor_feature : implicit_form_feature;
  successor_feature   : implicit_form_feature;
END_ENTITY;


--FF-029
ENTITY implicit_feature_bound;
  elements : SET [1:#] OF dimensionality_2_shape_element;
END_ENTITY;


--FF-003
ENTITY implicit_passage
SUBTYPE OF (implicit_form_feature);
  definition      : feature_volume;
```

```
  end_bounds       : SET [0:2] OF passage_bound;
  boundary_blends  : SET [0:2] OF passage_boundary_blend;
  interruptions    : SET [0:#] OF passage_intermediate_bound;
END_ENTITY;


--FF-402
ENTITY passage_bound;
  specification : implicit_feature_bound;
  passage_end   : feature_end_types;
END_ENTITY;


--FF-152
ENTITY passage_boundary_blend;
  blended_end : feature_end_types;
  end_blend   : implicit_edge_blend;
END_ENTITY;


--FF-186
ENTITY passage_intermediate_bound;
  passage_bound_type    : bound_types;
  interruption_complete : LOGICAL;
  specification         : implicit_feature_bound;
  bound_blend           : OPTIONAL implicit_edge_blend;
END_ENTITY;


--FF-004
ENTITY implicit_protrusion
SUBTYPE OF (implicit_form_feature);
  definition : feature_volume;
  end_bound  : OPTIONAL implicit_feature_bound;
  end_blend  : OPTIONAL implicit_edge_round;
END_ENTITY;
```

```
--FF-005
ENTITY implicit_depression
SUBTYPE OF (implicit_form_feature);
  definition    : feature_volume;
  end_bound     : OPTIONAL implicit_feature_bound;
  end_blend     : OPTIONAL implicit_edge_blend;
  interruptions : SET [0:#] OF depression_intermediate_bound;
END_ENTITY;


--FF-188
ENTITY depression_intermediate_bound;
  depression_bound_type  : bound_types;
  interruption_complete  : LOGICAL;
  specification          : implicit_feature_bound;
  bound_blend            : OPTIONAL implicit_edge_blend;
END_ENTITY;


--FF-006
ENTITY implicit_transition
SUPERTYPE OF (implicit_corner_blend OR
              implicit_edge_blend)
SUBTYPE OF (implicit_form_feature);
END_ENTITY;


--FF-308
ENTITY implicit_edge_blend
SUPERTYPE OF (implicit_edge_flat OR
              implicit_edge_round)
SUBTYPE OF (implicit_transition);
END_ENTITY;


--FF-359
ENTITY edge_blended_intersection;
  blend        : implicit_edge_blend;
  first_shape  : dimensionality_2_shape_element;
  second_shape : dimensionality_2_shape_element;
END_ENTITY;
```

```
--FF-036
ENTITY implicit_edge_flat
SUBTYPE OF (implicit_edge_blend);
  angle   : independent_angle_size_parameter;
  setback : independent_size_parameter;
END_ENTITY;


--FF-037
ENTITY implicit_edge_round
SUBTYPE OF (implicit_edge_blend);
  round_dim : independent_size_parameter;
END_ENTITY;


--FF-309
ENTITY implicit_corner_blend
SUPERTYPE OF (implicit_corner_flat OR
              implicit_outside_corner_round)
SUBTYPE OF (implicit_transition);
END_ENTITY;


--FF-310
ENTITY implicit_corner_flat
SUBTYPE OF (implicit_corner_blend);
  setback_edge1 : edge_shape_element;
  setback_dim1  : independent_size_parameter;
  setback_edge2 : edge_shape_element;
  setback_dim2  : independent_size_parameter;
  angle_dim     : independent_angle_size_parameter;
END_ENTITY;


--FF-311
ENTITY implicit_outside_corner_round
SUBTYPE OF (implicit_corner_blend);
  dimension : independent_size_parameter;
END_ENTITY;


--FF-064
ENTITY implicit_deformation
SUPERTYPE OF (implicit_bend OR
```

```
              implicit_emboss OR
              implicit_partial_cutout OR
              implicit_tube_deformation OR
              implicit_twist)
SUBTYPE OF (implicit_form_feature);
END_ENTITY;


--FF-429
ENTITY bend_dimension;
  bend_dim    : independent_size_parameter;
  bend_msrmt : bend_measurement_types;
END_ENTITY;


--FF-081
ENTITY implicit_emboss
SUPERTYPE OF (implicit_corner_rib OR
              implicit_round_bead OR
              implicit_v_bead OR
              implicit_spherical_emboss)
SUBTYPE OF (implicit_deformation);
END_ENTITY;


--FF-087
ENTITY implicit_v_bead
SUBTYPE OF (implicit_emboss);
  location    : bounded_curve;
  angle       : independent_angle_size_parameter;
  height      : independent_size_parameter;
  width       : independent_size_parameter;
  apex_round : bend_dimension;
  base_bend   : bend_dimension;
WHERE
  angle.dimension < 180;
END_ENTITY;


--FF-088
ENTITY implicit_round_bead
SUBTYPE OF (implicit_emboss);
  location     : bounded_curve;
  height       : independent_size_parameter;
```

```
  bead_size    : independent_size_parameter;
  base_bend    : bend_dimension;
WHERE
  height.dimension <= bead_size.dimension/2;
END_ENTITY;


--FF-089
ENTITY implicit_corner_rib
SUBTYPE OF (implicit_emboss);
  stiffens : form_feature;
  location : axis_placement;
  width    : independent_size_parameter;
  height1  : derivable_size_parameter;
  height2  : derivable_size_parameter;
  length   : derivable_size_parameter;
  bend1    : bend_dimension;
  bend2    : bend_dimension;
WHERE
  height1.dimension**2 + height2.dimension**2 = length.dimension**2;
END_ENTITY;


--FF-400
ENTITY implicit_spherical_emboss
SUBTYPE OF (implicit_emboss);
  location    : point;
  height      : independent_size_parameter;
  emboss_size : independent_size_parameter;
  bend_dim    : bend_dimension;
END_ENTITY;


--FF-082
ENTITY implicit_twist
SUBTYPE OF (implicit_deformation);
  length   : independent_size_parameter;
  angle    : independent_angle_size_parameter;
  location : axis_placement;
END_ENTITY;
```

```
--FF-083
ENTITY implicit_partial_cutout
SUPERTYPE OF (implicit_louver OR
              implicit_circular_knockout OR
              implicit_tab)
SUBTYPE OF (implicit_deformation);
END_ENTITY;


--FF-090
ENTITY implicit_louver
SUBTYPE OF (implicit_partial_cutout);
  location   : axis_placement;
  length     : independent_size_parameter;
  width      : independent_size_parameter;
  height     : independent_size_parameter;
  base_bend  : bend_dimension;
  inner_bend : bend_dimension;
END_ENTITY;


--FF-091
ENTITY implicit_circular_knockout
SUBTYPE OF (implicit_partial_cutout);
  location   : axis_placement;
  size_dim   : independent_size_parameter;
  height     : independent_size_parameter;
  gap_length : independent_size_parameter;
  bend_dim   : bend_dimension;
END_ENTITY;


--FF-401
ENTITY implicit_tab
SUBTYPE OF (implicit_partial_cutout);
  location : axis_placement;
  chord    : independent_size_parameter;
  width    : independent_size_parameter;
  height   : independent_size_parameter;
  angle    : derivable_angle_size_parameter;
  bend_dim : bend_dimension;
END_ENTITY;
```

```
--FF-084
ENTITY implicit_bend
SUPERTYPE OF (implicit_general_bend OR
              implicit_cutout_flange OR
              implicit_straight_bend)
SUBTYPE OF (implicit_deformation);
END_ENTITY;


--FF-092
ENTITY implicit_general_bend
SUBTYPE OF (implicit_bend);
  bend_line      : curve;
  bend_points    : SET [1:#] OF bend_point;
  bend_direction : hands;
  moved_side     : OPTIONAL hands;
END_ENTITY;


--FF-096
ENTITY bend_point;
  location    : point;
  bend_dim    : bend_dimension;
  bend_angle  : independent_angle_size_parameter;
WHERE
  BEND_ANGLE < 180;
END_ENTITY;


--FF-094
ENTITY implicit_cutout_flange
SUBTYPE OF (implicit_bend);
  stiffens    : UNIQUE implicit_passage;
  flange_side : feature_end_types;
  setback     : independent_size_parameter;
  bend_dim    : bend_dimension;
  bend_angle  : independent_angle_size_parameter;
WHERE
  bend_angle.dimension < 180;
END_ENTITY;


--FF-095
ENTITY implicit_straight_bend
```

```
SUBTYPE OF (implicit_bend);
  bend_line      : line;
  bend_dim       : bend_dimension;
  bend_angle     : independent_angle_size_parameter;
  bend_direction : hands;
  moved_side     : OPTIONAL hands;
WHERE
  bend_angle.dimension < 180;
END_ENTITY;


--FF-086
ENTITY implicit_tube_deformation
SUPERTYPE OF (implicit_tube_bend OR
              implicit_tube_flare OR
              implicit_tube_neck OR
              implicit_tube_flattening OR
              implicit_tube_roll)
SUBTYPE OF (implicit_deformation);
END_ENTITY;


--FF-093
ENTITY implicit_tube_bend
SUBTYPE OF (implicit_tube_deformation);
  location   : point;
  direction  : vector;
  bend_size  : bend_dimension;
  bend_angle : derivable_angle_size_parameter;
  moved_end  : tube_bend_moved_end_types;
END_ENTITY;


--FF-098
ENTITY implicit_tube_flare
SUBTYPE OF (implicit_tube_deformation);
  flared_end  : feature_end_types;
  end_od      : independent_size_parameter;
  length      : independent_size_parameter;
  angle       : derivable_angle_size_parameter;
  bend_dim    : bend_dimension;
  end_id      : derivable_size_parameter;
END_ENTITY;
```

```
--FF-099
ENTITY implicit_tube_neck
SUBTYPE OF (implicit_tube_deformation);
  location       : point;
  neck_direction : neck_direction_types;
  outside_dim    : independent_size_parameter;
  inside_dim     : derivable_size_parameter;
  length         : independent_size_parameter;
  angle          : derivable_angle_size_parameter;
  bend_dim       : bend_dimension;
END_ENTITY;


--FF-100
ENTITY implicit_tube_flattening
SUBTYPE OF (implicit_tube_deformation);
  location    : axis_placement;
  outer_major : independent_size_parameter;
  outer_minor : independent_size_parameter;
  length1     : independent_size_parameter;
  length2     : derivable_size_parameter;
  bend_dim    : bend_dimension;
  inner_major : derivable_size_parameter;
  inner_minor : derivable_size_parameter;
END_ENTITY;


--FF-101
ENTITY implicit_tube_roll
SUBTYPE OF (implicit_tube_deformation);
  rolled_end    : feature_end_types;
  before_length : independent_size_parameter;
  tube_size     : independent_size_parameter;
  curl_length   : independent_size_parameter;
  gap           : independent_size_parameter;
  curl_size     : independent_size_parameter;
END_ENTITY;


--FF-111
ENTITY implicit_area_feature
SUPERTYPE OF (implicit_knurl OR
              implicit_marking OR
```

```
              implicit_coupling OR
              implicit_thread)
SUBTYPE OF (implicit_form_feature);
  installation_area : OPTIONAL dimensionality_2_shape_element;
END_ENTITY;


--FF-024
ENTITY implicit_knurl
SUPERTYPE OF (implicit_diagonal_knurl OR
              implicit_diamond_knurl OR
              implicit_straight_knurl)
SUBTYPE OF (implicit_area_feature);
  number_of_teeth   : INTEGER;
  knurl_major_dim   : independent_size_parameter;
  knurl_nominal_dim : independent_size_parameter;
  tooth_depth       : independent_size_parameter;
  fillet_at_root    : independent_size_parameter;
WHERE
  cylindrical(installation_area);
END_ENTITY;


--FF-354
ENTITY implicit_straight_knurl
SUBTYPE OF (implicit_knurl);
END_ENTITY;


--FF-355
ENTITY implicit_diagonal_knurl
SUBTYPE OF (implicit_knurl);
  helix_angle : independent_size_parameter;
  helix_hand  : hands;
END_ENTITY;


--FF-356
ENTITY implicit_diamond_knurl
SUBTYPE OF (implicit_knurl);
  helix_angle : independent_size_parameter;
END_ENTITY;
```

```
--FF-025
ENTITY implicit_thread
SUBTYPE OF (implicit_area_feature);
  major_dim         : independent_size_parameter;
  threads_per_unit  : REAL;
  thread_form       : thread_forms;
  thread_hand       : hands;
  pitch_dim         : OPTIONAL independent_size_parameter;
  minor_dim         : OPTIONAL independent_size_parameter;
  thread_fit_class  : OPTIONAL thread_fit_classes;
  thread_spec       : OPTIONAL full_threading_specification;
  thread_apex       : OPTIONAL pitch_apex;
  thread_timing     : OPTIONAL thread_timer;
WHERE
  cylindrical(installation_area) OR
  conical(installation_area);
END_ENTITY;


--FF-421
ENTITY full_threading_specification;
  reference_point          : point;
  full_threading_direction : threading_direction_types;
  full_thread_length       : derivable_size_parameter;
END_ENTITY;


--FF-428
ENTITY thread_timer;
  pressure_point : point;
END_ENTITY;


--FF-425
ENTITY pitch_apex;
  apex_point              : point;
  pitch_distance          : REAL;
  pitch_in_axis_direction : LOGICAL;
END_ENTITY;
```

```
--FF-406
ENTITY implicit_marking
SUBTYPE OF (implicit_area_feature);
  marked_string : STRING(80);
 END_ENTITY;


--FF-415
ENTITY implicit_coupling
SUBTYPE OF (implicit_area_feature);
  coupling_shape            : coupling_shape_types;
  number_of_teeth           : INTEGER;
  pressure_angle            : independent_size_parameter;
  whole_depth               : independent_size_parameter;
  chamfer_depth             : independent_size_parameter;
  root_fillet               : independent_size_parameter;
  dim_to_point_of_tangency  : independent_size_parameter;
  coupling_timing           : OPTIONAL coupling_timer;
END_ENTITY;


--FF-420
ENTITY coupling_timer;
  timing_point : point;
END_ENTITY;


--FF-049
ENTITY implicit_form_feature_pattern
SUPERTYPE OF (implicit_array_form_feature_pattern OR
              implicit_circular_form_feature_pattern);
  base_form_feature : implicit_form_feature;
END_ENTITY;


--FF-053
ENTITY implicit_circular_form_feature_pattern
SUBTYPE OF (implicit_form_feature_pattern);
  centerline        : line;
  number_of_members : INTEGER;
  angular_spacing   : independent_size_parameter;
  omissions         : SET [0:#] OF circular_pattern_omission;
  offsets           : SET [0:#] OF circular_pattern_offset_member;
WHERE
```

```
  (number_of_members * angular_spacing.dimension) < 360;
  number_of_members - SIZEOF(omissions) >= 2;
END_ENTITY;

RULE verify_circular_pattern_omissions_and_offsets FOR
  (implicit_circular_form_feature_pattern,
   circular_pattern_omission,
   circular_pattern_offset_member);
LOCAL
  i,j:INTEGER;
  omit:circular_pattern_omission;
  offset:circular_pattern_offset_member;
END_LOCAL;
  REPEAT i := 1 TO SIZEOF(implicit_circular_form_feature_pattern.omissions);
    omit := POSITION(implicit_circular_form_feature_pattern.omissions,i);
    IF (omit.omitted_member_number<=1) THEN VIOLATION;
    END_IF;
    IF (omit.omitted_member_number>number_of_members) THEN VIOLATION;
    END_IF;
    REPEAT j := 1 TO SIZEOF(implicit_circular_form_feature_pattern.offsets);
      offset := POSITION(implicit_circular_form_feature_pattern.offsets,j);
      IF (offset.offset_member_number >
          implicit_circular_form_feature_pattern.number_of_members)
            THEN VIOLATION;
      END_IF;
      IF (offset.offset_member_number=omit.omitted_member_number)
        THEN VIOLATION;
      END_IF;
    END_REPEAT;
 END_REPEAT;
END_RULE;



--FF-073
ENTITY circular_pattern_omission;
  omitted_member_number : INTEGER;
WHERE
  omitted_member_number > 1;
END_ENTITY;

RULE omitted_member_number_constraint FOR
  (implicit_circular_form_feature_pattern,circular_pattern_omission);
  IF (circular_pattern_omission.omitted_member_number >
```

```
    implicit_circular_form_feature_pattern.number_of_members)
      THEN VIOLATION;
  END_IF;
END_RULE;



--FF-074
ENTITY circular_pattern_offset_member;
  offset_member_number : INTEGER;
  offset_angle         : REAL;
WHERE
  offset_member_number > 1;
  offset_angle <> 0;
END_ENTITY;

RULE offset_member_number_constraint FOR
  (implicit_circular_form_feature_pattern,circular_pattern_offset_member);
  IF (circular_pattern_offset_member.offset_member_number >
    implicit_circular_form_feature_pattern.number_of_members)
      THEN VIOLATION;
  END_IF;
END_RULE;


--FF-054
ENTITY implicit_array_form_feature_pattern
SUPERTYPE OF (parallel_equal_spacing_array_pattern OR
              parametric_equal_spacing_array_pattern)
SUBTYPE OF (implicit_form_feature_pattern);
  number_of_rows    : INTEGER;
  number_of_columns : INTEGER;
  omissions         : SET [0:#] OF array_pattern_omission;
WHERE
  number_of_rows > 0;
  number_of_columns > 0;
  number_of_rows + number_of_columns > 2;
  SIZEOF(omissions) < (number_of_rows)*(number_of_columns) - 1;
END_ENTITY;
```

```
--FF-306
ENTITY array_pattern_omission;
  row    : INTEGER;
  column : INTEGER;
WHERE
  row >= 1;
  column >= 1;
  (row <> 1) or (column <> 1);
END_ENTITY;


--FF-055
ENTITY parametric_equal_spacing_array_pattern
SUBTYPE OF (implicit_array_form_feature_pattern);
  layout_reference : surface;
  udelta           : REAL;
  vdelta           : REAL;
  offsets          : SET [0:#] OF parametric_array_pattern_offset_member;
END_ENTITY;

RULE verify_parametric_array_pattern_omissions_and_offsets FOR
  (parametric_equal_spacing_array_pattern,
   array_pattern_omission,
   parametric_array_pattern_offset_member);
LOCAL
  i,j:INTEGER;
  omit:array_pattern_omission;
  offset:parametric_array_pattern_offset_member;
END_LOCAL;
  REPEAT i := 1 TO SIZEOF(parametric_equal_spacing_array_pattern.omissions);
    omit := POSITION(parametric_equal_spacing_array_pattern.omissions,i);
    IF (omit.row > parametric_equal_spacing_array_pattern.member_of_rows)
      THEN VIOLATION;
    END_IF;
    IF (omit.column > parametric_equal_spacing_array_pattern.member_of_columns)
      THEN VIOLATION;
    END_IF;
    REPEAT j := 1 TO SIZEOF(parametric_equal_spacing_array_pattern.offsets);
      offset := POSITION(parametric_equal_spacing_array_pattern.offsets,j);
      IF (offset.row >
          parametric_equal_spacing_array_pattern.member_of_rows)
            THEN VIOLATION;
      END_IF;
      IF (offset.columns >
```

```
          parametric_equal_spacing_array_pattern.member_of_columns)
            THEN VIOLATION;
        END_IF;
        IF ((offset.row=omit.row) AND (offset.column=omit.column))
          THEN VIOLATION;
        END_IF;
      END_REPEAT;
   END_REPEAT;
  END_RULE;



  --FF-307
  ENTITY parametric_array_pattern_offset_member;
    row      : INTEGER;
    column   : INTEGER;
    u_offset : REAL;
    v_offset : REAL;
  WHERE
    ((row >=1) OR (column >= 1));
  END_ENTITY;


  --FF-056
  ENTITY parallel_equal_spacing_array_pattern
  SUBTYPE OF (implicit_array_form_feature_pattern);
    row_translation    : vector_with_magnitude;
    column_translation : vector_with_magnitude;
    row_spacing        : derivable_size_parameter;
    column_spacing     : derivable_size_parameter;
    offsets            : SET [0:#] OF
                           parallel_array_pattern_offset_member;
  END_ENTITY;

  RULE verify_parallel_array_pattern_omissions_and_offsets FOR
    (parallel_equal_spacing_array_pattern.
     array_pattern_omission,
     parallel_array_pattern_offset_member);
  LOCAL
    i,j:INTEGER;
    omit:array_pattern_omission;
    offset:parallel_array_pattern_offset_member;
  END_LOCAL;
    REPEAT i := 1 TO SIZEOF(parallel_equal_spacing_array_pattern.omissions);
```

```
    omit := POSITION(parallel_equal_spacing_array_pattern.omissions,i);
    IF (omit.row<=1) THEN VIOLATION;
    END_IF;
    IF (omit.column<=1) THEN VIOLATION;
    END_IF;
    IF (omit.row>parallel_equal_spacing_array_pattern.number_of_rows)
      THEN VIOLATION;
    END_IF;
    IF (omit.column>parallel_equal_spacing_array_pattern.number_of_columns)
      THEN VIOLATION;
    END_IF;
    REPEAT j := 1 TO SIZEOF(parallel_equal_spacing_array_pattern.offsets);
      offset := POSITION(parallel_equal_spacing_array_pattern.offsets,j);
      IF (offset.row >
          parallel_equal_spacing_array_pattern.number_of_rows)
            THEN VIOLATION;
      END_IF;
      IF (offset.columns >
          parallel_equal_spacing_array_pattern.number_of_columns)
            THEN VIOLATION;
      END_IF;
      IF ((offset.row=omit.row) AND (offset.column=omit.column))
        THEN VIOLATION;
      END_IF;
    END_REPEAT;
 END_REPEAT;
END_RULE;



--FF-305
ENTITY parallel_array_pattern_offset_member;
  row    : INTEGER;
  column : INTEGER;
  offset : vector_with_magnitude;
WHERE
  ((row >= 1) OR (column >= 1));
END_ENTITY;
```

```
--FF-067
ENTITY replicate_form_feature;
  copied_feature : implicit_form_feature;
  location       : axis_placement;
  mirror         : OPTIONAL coordinate_enumeration;
END_ENTITY;



--FF-417
ENTITY feature_volume
SUPERTYPE OF (feature_sweep OR
              feature_ruling);
END_ENTITY;


--FF-411
ENTITY feature_ruling
SUBTYPE OF (feature_volume);
  defining_curve1 : curve_on_surface;
  defining_curve2 : curve_on_surface;
  blend           : SET [0:2] OF feature_ruling_wall_end_blend;
  location        : OPTIONAL axis_placement;
END_ENTITY;

--FF-416
ENTITY feature_ruling_wall_end_blend;
  blend_end : feature_end_types;
  blend     : implicit_edge_blend;
END_ENTITY;


--FF-120
ENTITY feature_sweep
SUPERTYPE OF (along_feature_sweep OR
             axisymmetric_feature_sweep OR
             in_out_feature_sweep)
SUBTYPE OF (feature_volume);
  location : axis_placement;
END_ENTITY;


--FF-422
ENTITY spiral_feature_sweep_path
```

```
SUBTYPE OF (feature_sweep_path);
  spiral_path_length : independent_size_parameter;
  spiral_path_size   : independent_size_parameter;
  spiral_path_hand   : hands;
  turns_per_unit     : REAL;
  spiral_path_taper  : OPTIONAL spiral_taper;
END_ENTITY;


--FF-427
ENTITY spiral_taper;
  apex_z_coord : REAL;
  angle_dim    : derivable_angle_size_parameter;
END_ENTITY;


--FF-423
ENTITY surface_conforming_feature_sweep_path
SUBTYPE OF (feature_sweep_path);
  reference_surface : curve_on_surface;
END_ENTITY;


--FF-424
ENTITY other_feature_sweep_path
SUBTYPE OF (feature_sweep_path);
  path_def    : curve;
  orientation : curve;
END_ENTITY;


--FF-121
ENTITY along_feature_sweep
SUBTYPE OF (feature_sweep);
  sweep_path    : feature_sweep_path;
  sweep_profile : open_feature_sweep_profile;
  sweep_ends    : SET [0:2] OF along_feature_sweep_end;
END_ENTITY;
```

```
--FF-158
ENTITY along_feature_sweep_end
SUPERTYPE OF (along_feature_sweep_flat_end OR
              along_feature_sweep_radiused_end);
  sweep_end : feature_end_types;
END_ENTITY;


--FF-159
ENTITY along_feature_sweep_flat_end
SUBTYPE OF (along_feature_sweep_end);
  end_blend : OPTIONAL implicit_edge_blend;
END_ENTITY;


--FF-160
ENTITY along_feature_sweep_radiused_end
SUBTYPE OF (along_feature_sweep_end);
END_ENTITY;


--FF-122
ENTITY axisymmetric_feature_sweep
SUPERTYPE OF (constant_diameter_axisymmetric_feature_sweep OR
              tapered_axisymmetric_feature_sweep OR
              other_axisymmetric_feature_sweep)
SUBTYPE OF (feature_sweep);
  sweep_length : independent_size_parameter;
  sweep_end    : OPTIONAL axisymmetric_feature_sweep_end;
END_ENTITY;


--FF-149
ENTITY constant_diameter_axisymmetric_feature_sweep
SUBTYPE OF (axisymmetric_feature_sweep);
  sweep_size : independent_size_parameter;
END_ENTITY;


--FF-150
ENTITY tapered_axisymmetric_feature_sweep
SUBTYPE OF (axisymmetric_feature_sweep);
  size_at_origin : independent_size_parameter;
```

```
  angle_dim      : derivable_size_parameter;
  taper_type     : taper_types;
WHERE
  size_at_origin.dimension >= 0;
  angle_dim.dimension < 90;
END_ENTITY;


--FF-151
ENTITY other_axisymmetric_feature_sweep
SUBTYPE OF (axisymmetric_feature_sweep);
  sweep_profile : general_profile;
END_ENTITY;


--FF-130
ENTITY axisymmetric_feature_sweep_end
SUPERTYPE OF (axisymmetric_feature_sweep_flat_end OR
              axisymmetric_feature_sweep_spherical_end OR
              axisymmetric_feature_sweep_conical_end);
  blend : OPTIONAL implicit_edge_blend;
END_ENTITY;


--FF-132
ENTITY axisymmetric_feature_sweep_flat_end
SUBTYPE OF (axisymmetric_feature_sweep_end);
END_ENTITY;


--FF-133
ENTITY axisymmetric_feature_sweep_spherical_end
SUBTYPE OF (axisymmetric_feature_sweep_end);
END_ENTITY;


--FF-134
ENTITY axisymmetric_feature_sweep_conical_end
SUBTYPE OF (axisymmetric_feature_sweep_end);
  end_angle : independent_size_parameter;
  tip_blend : OPTIONAL implicit_edge_round;
WHERE
  end_angle.dimension < 180;
END_ENTITY;
```

```
--FF-123
ENTITY in_out_feature_sweep
SUPERTYPE OF (constant_profile_in_out_sweep OR
              tapered_profile_in_out_sweep OR
              contoured_profile_in_out_sweep)
SUBTYPE OF (feature_sweep);
  sweep_path     : linear_feature_sweep_path;
  sweep_profile  : closed_feature_sweep_profile;
  wall_end_blend : OPTIONAL implicit_edge_blend;
END_ENTITY;


--FF-190
ENTITY constant_profile_in_out_sweep
SUBTYPE OF (in_out_feature_sweep);
END_ENTITY;


--FF-192
ENTITY tapered_profile_in_out_sweep
SUBTYPE OF (in_out_feature_sweep);
  semi_angle : independent_size_parameter;
  taper_type : taper_types;
WHERE
    (-90 <= semi_angle.dimension <= +90);
END_ENTITY;


--FF-193
ENTITY contoured_profile_in_out_sweep
SUBTYPE OF (in_out_feature_sweep);
  scaling_curve : bounded_curve;
END_ENTITY;


--FF-124
ENTITY feature_sweep_path
SUPERTYPE OF (circular_feature_sweep_path OR
              spiral_feature_sweep_path OR
              surface_conforming_feature_sweep_path OR
              other_feature_sweep_path OR
              linear_feature_sweep_path);
```

```
END_ENTITY;


--FF-125
ENTITY linear_feature_sweep_path
SUBTYPE OF (feature_sweep_path);
  path_length : independent_size_parameter;
END_ENTITY;


--FF-126
ENTITY circular_feature_sweep_path
SUPERTYPE OF (complete_circular_feature_sweep_path OR
              partial_circular_feature_sweep_path)
SUBTYPE OF (feature_sweep_path);
  path_size         : independent_size_parameter;
  orientation_angle : independent_size_parameter;
END_ENTITY;


--FF-127
ENTITY partial_circular_feature_sweep_path
SUBTYPE OF (circular_feature_sweep_path);
  sweep_angle : independent_size_parameter;
WHERE
  -360 < sweep_angle.dimension < 360;
END_ENTITY;


--FF-128
ENTITY complete_circular_feature_sweep_path
SUBTYPE OF (circular_feature_sweep_path);
END_ENTITY;


--FF-136
ENTITY feature_sweep_profile
SUPERTYPE OF (closed_feature_sweep_profile OR
              open_feature_sweep_profile);
END_ENTITY;
```

```
--FF-137
ENTITY closed_feature_sweep_profile
SUPERTYPE OF (other_closed_feature_sweep_profile OR
              standard_closed_feature_sweep_profile)
SUBTYPE OF (feature_sweep_profile);
END_ENTITY;


--FF-167
ENTITY standard_closed_feature_sweep_profile
SUPERTYPE OF (rectangular_feature_sweep_profile OR
              ngon_feature_sweep_profile)
SUBTYPE OF (closed_feature_sweep_profile);
  blend : OPTIONAL implicit_edge_blend;
END_ENTITY;


--FF-139
ENTITY rectangular_feature_sweep_profile
SUBTYPE OF (standard_closed_feature_sweep_profile);
  sweep_length : independent_size_parameter;
  sweep_width  : independent_size_parameter;
END_ENTITY;


--FF-140
ENTITY ngon_feature_sweep_profile
SUBTYPE OF (standard_closed_feature_sweep_profile);
  number_of_sides    : INTEGER;
  inscribed_dim      : independent_size_parameter;
  circumscribed_dim  : derivable_size_parameter;
  side_length        : derivable_size_parameter;
  interior_angle     : derivable_angle_size_parameter;
  exterior_angle     : derivable_angle_size_parameter;
END_ENTITY;


--FF-168
ENTITY other_closed_feature_sweep_profile
SUBTYPE OF (closed_feature_sweep_profile);
  sweep_profile : closed_general_profile;
END_ENTITY;
```

```
--FF-142
ENTITY open_feature_sweep_profile
SUPERTYPE OF (circular_arc_feature_sweep_profile OR
              rounded_u_feature_sweep_profile OR
              vee_feature_sweep_profile OR
              square_u_feature_sweep_profile OR
              tee_feature_sweep_profile OR
              ell_feature_sweep_profile OR
              line_plus_radius_feature_sweep_profile OR
              half_obround_feature_sweep_profile OR
              other_open_feature_sweep_profile)
SUBTYPE OF (feature_sweep_profile);
END_ENTITY;


--FF-143
ENTITY circular_arc_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  arc_size : independent_size_parameter;
END_ENTITY;


--FF-144
ENTITY rounded_u_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  sweep_width : independent_size_parameter;
END_ENTITY;


--FF-145
ENTITY vee_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  vee_angle : independent_size_parameter;
  blend     : OPTIONAL implicit_edge_blend;
WHERE
  vee_angle.dimension < 180;
END_ENTITY;


--FF-146
ENTITY square_u_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  sweep_width  : independent_size_parameter;
```

```
  blend1       : OPTIONAL implicit_edge_blend;
  blend2       : OPTIONAL implicit_edge_blend;
END_ENTITY;


--FF-147
ENTITY tee_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  stem_width            : independent_size_parameter;
  crossbar_width        : independent_size_parameter;
  crossbar_height       : independent_size_parameter;
  stem_crossbar_blend1 : OPTIONAL implicit_edge_blend;
  stem_crossbar_blend2 : OPTIONAL implicit_edge_blend;
  crossbar_blend1       : OPTIONAL implicit_edge_blend;
  crossbar_blend2       : OPTIONAL implicit_edge_blend;
  crossbar_blend3       : OPTIONAL implicit_edge_blend;
  crossbar_blend4       : OPTIONAL implicit_edge_blend;
WHERE
  crossbar_width.dimension > stem_width.dimension;
END_ENTITY;


--FF-148
ENTITY ell_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  stem_width         : independent_size_parameter;
  endbar_width       : independent_size_parameter;
  endbar_height      : independent_size_parameter;
  ell_orientation    : ell_orientation_types;
  stem_endbar_blend : OPTIONAL implicit_edge_blend;
  endbar_blend1      : OPTIONAL implicit_edge_blend;
  endbar_blend2      : OPTIONAL implicit_edge_blend;
  endbar_blend3      : OPTIONAL implicit_edge_blend;
WHERE
  endbar_width.dimension > stem_width.dimension;
END_ENTITY;


--FF-184
ENTITY line_plus_radius_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  size_dim : independent_size_parameter;
END_ENTITY;
```

```
--FF-185
ENTITY half_obround_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  circle_dim : independent_size_parameter;
  length_dim : independent_size_parameter;
END_ENTITY;


--FF-169
ENTITY other_open_feature_sweep_profile
SUBTYPE OF (open_feature_sweep_profile);
  sweep_profile : open_general_profile;
END_ENTITY;


--FF-162
ENTITY general_profile
SUPERTYPE OF (open_general_profile OR
              closed_general_profile);
  first_curve   : curve;
  element_pairs : SET [0:#] OF profile_pair;
WHERE
  sequenced(first_curve,element_pairs);
END_ENTITY;


--FF-165
ENTITY profile_pair;
  predecessor_curve : curve;
  successor_curve   : curve;
  blend             : OPTIONAL implicit_edge_blend;
END_ENTITY;


--FF-163
ENTITY open_general_profile
SUBTYPE OF (general_profile);
WHERE
  NOT closed(element_pairs);
  planar(element_pairs);
END_ENTITY;
```

```
 --FF-164
 ENTITY closed_general_profile
 SUBTYPE OF (general_profile);
 WHERE
     closed(element_pairs);
     planar(element_pairs);
 END_ENTITY;
END_SCHEMA;
```

SECTION 4: FORM FEATURES INFORMATION MODEL

4.8 NIAM MODEL

This section gives the FFIM in the diagrammatic NIAM modeling language. The NIAM model is broken into sixteen diagrams. (A single-sheet NIAM model exists and is available upon request.)

The following table serves as an index to the NIAM model and a cross reference between the IDEF1X and NIAM. It gives, for each IDEF1X entity, the NIAM diagram in which the corresponding construct appears. Note that, due to differences between IDEF1X and NIAM, several IDEF1X entities have no corresponding NIAM construct. These are indicated in the table. In most such cases, a relation between two diagrammed entities corresponds to the IDEF1X entity. Where this is so, a parenthetical note identifies those two entities. For example, [134,37] indicates that the NIAM relation between entities 134 and 37 corresponds to the IDEF1X entity.

NUMBER	NAME	DIAGRAM(S)
001	FORM FEATURE	1,2
002	IMPLICIT FORM FEATURE	2
003	IMPLICIT PASSAGE	2,5,8
004	IMPLICIT PROTRUSION	2,9
005	IMPLICIT DEPRESSION	2,9
006	IMPLICIT TRANSITION	2,15
024	IMPLICIT KNURL	16
025	IMPLICIT THREAD	16
029	IMPLICIT FEATURE BOUND	1,8,9
030	FEATURE BOUND ELEMENT	1
036	IMPLICIT EDGE FLAT	15
037	IMPLICIT EDGE ROUND	9,12,15
049	IMPLICIT FORM FEATURE PATTERN	2,3
053	IMPLICIT CIRCULAR FORM FEATURE PATTERN	3
054	IMPLICIT ARRAY FORM FEATURE PATTERN	3
055	PARAMETRIC EQUAL SPACING ARRAY PATTERN	3
056	PARALLEL EQUAL SPACING ARRAY PATTERN	3
064	IMPLICIT DEFORMATION	2,4
067	REPLICATE FORM FEATURE	2
073	CIRCULAR PATTERN OMISSION	3
074	CIRCULAR PATTERN OFFSET MEMBER	3
081	IMPLICIT EMBOSS	4,7
082	IMPLICIT TWIST	4
083	IMPLICIT PARTIAL CUTOUT	4
084	IMPLICIT BEND	4,5
086	IMPLICIT TUBE DEFORMATION	4,6
087	IMPLICIT V-BEAD	7
088	IMPLICIT ROUND BEAD	7
089	IMPLICIT CORNER RIB	7
090	IMPLICIT LOUVER	4
091	IMPLICIT CIRCULAR KNOCKOUT	4
092	IMPLICIT GENERAL BEND	5
093	IMPLICIT TUBE BEND	6
094	IMPLICIT CUTOUT FLANGE	5
095	IMPLICIT STRAIGHT BEND	5
096	BEND POINT	5
098	IMPLICIT TUBE FLARE	6
099	IMPLICIT TUBE NECK	6
100	IMPLICIT TUBE FLATTENING	6

Table 3: Entity Pool - Numeric Order

NUMBER	NAME	DIAGRAM(S)
101	IMPLICIT TUBE ROLL	6
111	IMPLICIT AREA FEATURE	2,16
120	FEATURE SWEEP	10,11
121	ALONG FEATURE SWEEP	14
122	AXISYMMETRIC FEATURE SWEEP	11,12
123	IN/OUT FEATURE SWEEP	11,13
124	FEATURE SWEEP PATH	11
125	LINEAR FEATURE SWEEP PATH	11
126	CIRCULAR FEATURE SWEEP PATH	11
127	PARTIAL CIRCULAR FEATURE SWEEP PATH	11
128	COMPLETE CIRCULAR FEATURE SWEEP PATH	11
129	IN/OUT FEATURE SWEEP WALL/END BLEND	13 [123,308]
130	AXISYMMETRIC FEATURE SWEEP END	12
131	AXISYMMETRIC FEATURE SWEEP WALL/END BLEND	12 [130,308]
132	AXISYMMETRIC FEATURE SWEEP FLAT END	12
133	AXISYMMETRIC FEATURE SWEEP SPHERICAL END	12
134	AXISYMMETRIC FEATURE SWEEP CONICAL END	12
135	AXISYMMETRIC FEATURE SWEEP CONICAL END TIP BLEND	12 [134,37]
136	FEATURE SWEEP PROFILE	–
137	CLOSED FEATURE SWEEP PROFILE	13
139	RECTANGULAR FEATURE SWEEP PROFILE	13
140	N-GON FEATURE SWEEP PROFILE	13
141	STANDARD CLOSED FEATURE SWEEP PROFILE BLEND	–
142	OPEN FEATURE SWEEP PROFILE	14
143	CIRCULAR ARC FEATURE SWEEP PROFILE	14
144	ROUNDED-U FEATURE SWEEP PROFILE	14
145	VEE FEATURE SWEEP PROFILE	14
146	SQUARE-U FEATURE SWEEP PROFILE	14
147	TEE FEATURE SWEEP PROFILE	14
148	ELL FEATURE SWEEP PROFILE	14
149	CONSTANT DIAMETER AXISYMMETRIC FEATURE SWEEP	12
150	TAPERED AXISYMMETRIC FEATURE SWEEP	12
151	OTHER AXISYMMETRIC FEATURE SWEEP	12
152	PASSAGE BLEND	8
153	PROTRUSION BLEND	9
154	DEPRESSION BLEND	9

Table 3: Entity Pool – Numeric Order (Continued)

NUMBER	NAME	DIAGRAM(S)
158	ALONG FEATURE SWEEP END	14
159	ALONG FEATURE SWEEP FLAT END	14
160	ALONG FEATURE SWEEP RADIUSED END	14
161	ALONG FEATURE SWEEP FLAT END BLEND	14 [159,308]
162	GENERAL PROFILE	12,13
163	OPEN GENERAL PROFILE	13,14
164	CLOSED GENERAL PROFILE	13
165	PROFILE PAIR	13
166	PROFILE PAIR BLEND	13
167	STANDARD CLOSED FEATURE SWEEP PROFILE	13
168	OTHER CLOSED FEATURE SWEEP PROFILE	13
169	OTHER OPEN FEATURE SWEEP PROFILE	14
170	SQUARE-U BLEND1	14 [146,308]
171	SQUARE-U BLEND2	14 [146,308]
172	VEE BLEND	14 [145,308]
173	TEE STEM/CROSSBAR BLEND1	14 [147,308]
174	TEE STEM/CROSSBAR BLEND2	14 [147,308]
175	TEE CROSSBAR BLEND1	14 [147.308]
176	TEE CROSSBAR BLEND2	14 [147,308]
177	TEE CROSSBAR BLEND3	14 [147,308]
178	TEE CROSSBAR BLEND4	14 [147,308]
179	ELL STEM/ENDBAR BLEND	14 [148,308]
180	ELL ENDBAR BLEND1	14 [148,308]
181	ELL ENDBAR BLEND2	14 [148,308]
182	ELL ENDBAR BLEND3	14 [148,308]
184	LINE PLUS RADIUS FEATURE SWEEP PROFILE	14
185	HALF OBROUND FEATURE SWEEP PROFILE	14
186	PASSAGE INTERMEDIATE BOUND	8
187	PASSAGE INTERMEDIATE BOUND BLEND	8
188	DEPRESSION INTERMEDIATE BOUND	9
189	DEPRESSION INTERMEDIATE BOUND BLEND	9
190	CONSTANT PROFILE IN/OUT SWEEP	13
192	TAPERED PROFILE IN/OUT SWEEP	13
193	CONTOURED PROFILE IN/OUT SWEEP	13
302	FORM FEATURE REFLECTION	2
305	PARALLEL ARRAY PATTERN OFFSET MEMBER	3
306	ARRAY PATTERN OMISSION	3
307	PARAMETRIC ARRAY PATTERN OFFSET MEMBER	3

Table 3: Entity Pool – Numeric Order (Continued)

NUMBER	NAME	DIAGRAM(S)
308	IMPLICIT EDGE BLEND	8,9,10,12,13,14 15
309	IMPLICIT CORNER BLEND	15
310	IMPLICIT CORNER FLAT	15
311	IMPLICIT OUTSIDE CORNER ROUND	15
315	GEOMETRIC MODEL/IMPLICIT FORM FEATURE ASSOCIATION	2
354	IMPLICIT STRAIGHT KNURL	16
355	IMPLICIT DIAGONAL KNURL	16
356	IMPLICIT DIAMOND KNURL	16
359	EDGE-BLENDED INTERSECTION	15
400	IMPLICIT SPHERICAL EMBOSS	7
401	IMPLICIT TAB	4
402	PASSAGE BOUND	8
403	DEPRESSION BOUND	9
404	PROTRUSION BOUND	9
406	IMPLICIT MARKING	16
411	FEATURE RULING	10
415	IMPLICIT COUPLING	16
416	FEATURE RULING WALL/END BLEND	10
417	FEATURE VOLUME	8,9,10
418	FEATURE RULING LCS	10
419	IMPLICIT FEATURE PRECEDENCE	2,10
420	COUPLING TIMER	16
421	FULL THREADING SPECIFICATION	16
422	SPIRAL FEATURE SWEEP PATH	11
423	SURFACE CONFORMING FEATURE SWEEP PATH	11
424	OTHER FEATURE SWEEP PATH	11
425	PITCH APEX	16
426	THREAD APEX	16
427	SPIRAL TAPER	11
428	THREAD TIMER	16
429	BEND DIMENSION	4,5,6,7
430	BEND MOVEMENT INDICATOR	5
431	GENERAL BEND MOVEMENT INDICATOR	5
432	STRAIGHT BEND MOVEMENT INDICATOR	5
433	TUBE BEND MOVEMENT INDICATOR	6
GEO-2	POINT	5,6,7,16
GEO-3	VECTOR	6

Table 3: Entity Pool – Numeric Order (Continued)

NUMBER	NAME	DIAGRAM(S)
GEO-4	AXIS PLACEMENT	2,4,6,7,10
GEO-6	CURVE	5,11,13
GEO-7	SURFACE	3
GEO-17	VECTOR WITH MAGNITUDE	3
GEO-20	LINE	3,5
GEO-22	BOUNDED CURVE	7,13
GEO-23	CURVE ON SURFACE	10,11
INT-7	DIMENSIONALITY-2 SHAPE ELEMENT	1,15,16
INT-8	AREA SHAPE ELEMENT	1
INT-11	ZONE SHAPE ELEMENT	1
INT-15	EDGE SHAPE ELEMENT	15
INT-20	CORNER SHAPE ELEMENT	15
INT-23	GEOMETRIC MODEL	2
INT-103	ZONE SHAPE ELEMENT COMPONENT	1
TOL-6041	INDEPENDENT SIZE PARAMETER	4
TOL-6042	DERIVABLE SIZE PARAMETER	-
TOL-6101	INDEPENDENT ANGLE SIZE PARAMETER	-
TOL-6102	DERIVABLE ANGLE SIZE PARAMETER	-

Table 3: Entity Pool - Numeric Order (Continued)

5 PRODUCT STRUCTURE & CONFIGURATION MANAGEMENT INFORMATION MODEL

5.1 Purpose & Scope

5.1.1 Purpose

The purpose of the data model described herein is to portray the significant information required to manage the configuration (or structure) of an industrial product.

The information is independent of the type of product. Whether it be strictly mechanical, electrical, architectural, or of some other type, the information contained in the model remains unchanged.

This model tends to portray the information in the product development life cycle near the completion of engineering detail design. This is often referred to as "engineering release." The model is therefore independent of any particular design or analysis function.

5.1.2 Scope and Viewpoint

The Product Structure Configuration Management data model is limited primarily to engineering data. It is assumed, however, that the product described will be manufactured (and probably supported by a logistics organization throughout its life).

The viewpoint represented in this data model is primarily that of configuration management of product structures. Engineering design and management are also strongly represented, however. The particular terms may vary depending on the nature of the enterprise, but the function is that of managing the way in which deliverable products are designed and configured.

The scope of this model for the short term is restricted by the following:

1. Product Structure is restricted to the relationship between physical components and assemblies.

2. Configuration Management applies only to Product Structure, i.e. manages the way assemblies and components are configured. This is initially limited to managing the way products are "intended" to be built, not how they were actually built. It is further restricted to address the configuration of specific units of product (serial or lot numbers), and not time ranges of products.

3. Product Structure will support more than one Bill-Of-Material view.

4. Product Structure and Configuration Management will support material information only with respect to stock items and make-from issues. This means that the current model will not include material property data. Material property data is currently being addressed by a separate committee within PDES.

5. The primary goal is to capture the "as designed" information. This is initially limited to mechanical product issues. There has not been an attempt to integrate this model with work done by the AEC or Electrical committees.

6. Currently this model does not explicitly address:

   - Physical interface information, such as mating conditions, fastening, and joining requirements.
   - Deformable conditions and degrees-of-freedom.
   - Part sourcing requirements.
   - Part family coding and/or classifications.
   - Pre-release, preliminary design requirements.
   - Change order information.

It is planned that the model contained herein develop into a "resource" model, satisfying the needs of many application areas. It is also intended to be a "conceptual" model, in that it attempts to record the minimal set of entities required to capture the desired information. It does **NOT** include those entities and relationships whose sole purpose is to make the model easier to use or more efficient at data storage, i.e. those desired at the "implementation" level, but unnecessary at the "conceptual" level.

This was done in order to minimize the labor involved in the future integration of the model with other committees. Implementation, although important, can only be satisfactorily achieved after this integration process has occurred.

When integration has proceeded to a level deemed sufficient, an "implementation" level model will be produced which will include those entities and relationships that optimize the performance or data storage requirements of the model.

5.1.3 Fundamental Concepts and Assumptions

This data model contains underlying ideas found in the "real world" of configuration management. The ideas are important to the focus of the model, and they are represented either in the entities or the relationships of the model.

- CM is interested in products (items) which are intended to be built and which may or may not be sold. This implies that the configuration of parts supplied by a vendor is not usually managed by CM. For example, if a certain motor is supplied by a vendor, the individual parts of the motor are not usually under CM.

- The enterprise and the customer agree on the identity of certain configuration items (CI's). These are usually high level assemblies which act as focal points for managing the effectivity of lower level parts and assemblies. In other words, effectivity is managed by the configuration item. In some businesses, these configuration items are equated to models of product.

- An enterprise often makes minor changes to a product for marketing purposes. General "product lines" which include one or more (model) variations of a product are therefore maintained, often simultaneously.

5.1.4 Abbreviations and Acronyms

The following abbreviations and acronyms are used throughout this document.

CI	Configuration Item
CM	Configuration Management
CR	Change Request
PDD	Product Definition Data
PMD	Product Management Data
PS	Product Structure
PSCM	Product Structure Configuration Management

SECTION 5: PSCM INFORMATION MODEL

5.2 PSCM Planning Model

This section describes the planning model for the Product Structure Configuration Management model. This planning model serves as a high level overview of the coverage of the model, and as a starting point for the development of the detailed PSCM reference models.

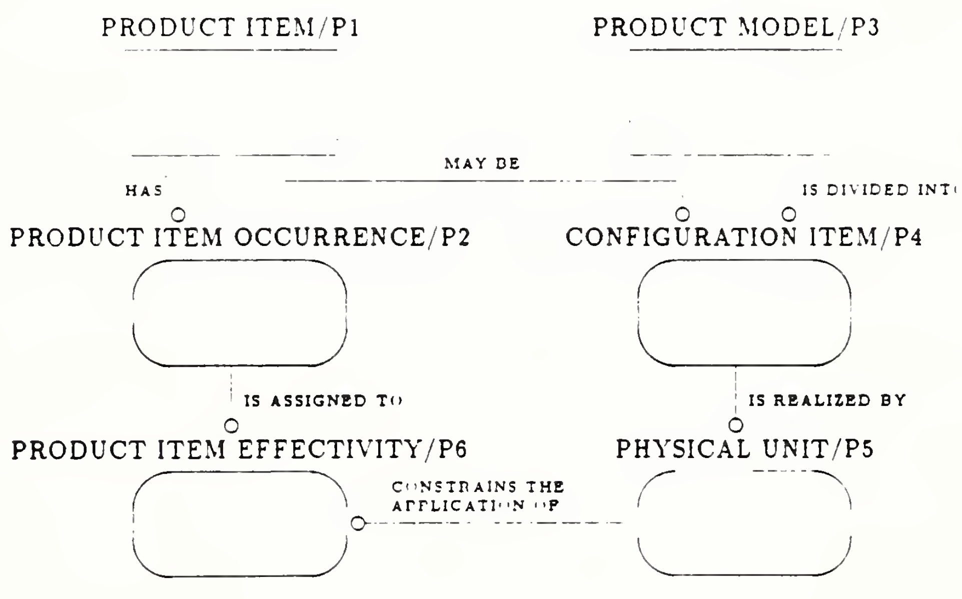
5.2.1 Entity Pool

The following entities constitute the Planning Model Entity Pool.

Planning Model Entity Pool

NUMBER	NAME
P1	Product Item
P2	Product Item Occurrence
P3	Product Model
P4	Configuration Item
P5	Physical Unit
P6	Product Item Effectivity

5.2.2 IDEF1x Diagram



5.2.3 Entity Glossary and Business Rules

Product Item/P1

The design of something produced which is, or is intended to be, a part of a *Physical Unit*, or is consumed in the production of a *Physical Unit*. Everything from paint, bar stock, purchased screws, and resistors to the highest defined assembly may appear as an instance of this entity.

Every *Product Item/P1*:
- has zero, one, or many *Product Item Occurrence/P2*(s).
- may be a *Configuration Item/P4*.

Product Item Occurrence/P2

Information about an occurrence of a *Product Item* in the context of any higher level *Product Item* (assembly).

Every *Product Item Occurrence/P2*:
- belongs to one *Product Item/P1*.
- is assigned to zero, one, or many *Product Item Effectivity*(s).

Product Model/P3

A variant within the product representing some significant change in mission or function. Product and model are essentially marketing ideas, but are used to carry through to the item's identification.

Every *Product Model/P3*:
- is divided into zero, one, or many *Configuration Item/P4*(s).

Configuration Item/P4

A *Configuration Item* is a *Product Item* which is sold or planned to be sold in and of itself. In this sense, it is a concept that is driven by marketing concerns. The enterprise and the customer or audience come to an agreement on how this thing is to be partitioned at a very high level. All *Configuration Items* are the top level or some portion of a *Product Model* which is sold. A *Configuration Item* must be managed separately because it is something which is sold in and of itself. *Configuration Items* are also sold as part of higher level *Configuration Items*.

Every *Configuration Item/P4*:
- is a division of one *Product Model/P3*.
- is a *Product Item/P1*.
- is realized by zero, one, or many *Physical Unit/P5*(s).

Physical Unit/P5

A physical manifestation of a *Configuration Item* which is specified for use in a *Product Item Effectivity*.

Every *Physical Unit*:

- is a realization of one *Configuration Item/P4*.
- constrains the application of zero, one, or many *Product Item Effectivity*(s).

Product Item Effectivity/P6

Information that a particular *Planned Product Item* is supposed to appear in specific units of a deliverable product (*Configuration Items*).

Every *Product Item Effectivity/P6*:

- is constrained by one *Physical Unit/P5*.
- has assigned one *Product Item Occurrence/P2*.

SECTION 5: PSCM INFORMATION MODEL

5.3 PSCM Reference Models

The information contained in this model has several uses. These may be categorized in several broad areas. This division of data is useful to show particular viewpoints or functional areas of the entire data model. Some of the information in the Product Structure Configuration Management model is related to the designed structure of the product, some is concerned with the overall configuration management of the product. The data in each of these groups forms a reference model (or sub-model) of the Product Structure Configuration Management data. The reference models are listed below.

1. Product Item
2. Product Item Usage
3. Product Configuration

These reference models are described on the following pages.

5.3.1 Entity Pool

The following entities appear within the PSCM reference model and are defined on the following pages.

Entity Pool in Numeric Order

NUMBER	NAME	DIAGRAM
1	Product Item	FEO/1
2	Product Item Version	FEO/1, FEO/3
4	Product Item Version Functional Definition	FEO/1, FEO/2
5	Product Item Usage Traversal	FEO/2, FEO/3
6	Design Change Sequence	FEO/1
10	Product Model	FEO/3
11	Configuration Item	FEO/3
13	Built Physical Unit	FEO/3
14	Planned Physical Unit	FEO/3
15	Traversal Effectivity	FEO/3
16	Geometric Product Item Usage Traversal	FEO/2
17	Make From Usage Option	FEO/2
18	Assembly Component Usage Traversal	FEO/2
19	Assembly Component Usage Traversal Substitute	FEO/2
20	Next Assembly Usage Occurrence	FEO/2
21	Higher Assembly Usage Traversal	FEO/2
22	Discrete Physical Unit	FEO/3
23	Physical Unit Lot	FEO/3
24	Physical Unit Open Ended Range	FEO/3
26	Higher Assembly Usage Traversal Step	FEO/2
27	Product Item Version Definition Shape	FEO/1
28	Product Item Version Definition Material	FEO/1
29	Product Item Version Contract	FEO/1
30	Contract	FEO/1
31	Make From Usage Option Group	FEO/2

Entity Pool in Alphabetic Order

NUMBER	NAME	DIAGRAM
18	Assembly Component Usage Traversal	FEO/2
19	Assembly Component Usage Traversal Substitute	FEO/2
13	Built Physical Unit	FEO/3
11	Configuration Item	FEO/3
30	Contract	FEO/1
6	Design Change Sequence	FEO/1
22	Discrete Physical Unit	FEO/3
16	Geometric Product Item Usage Traversal	FEO/2
21	Higher Assembly Usage Traversal	FEO/2
26	Higher Assembly Usage Traversal Step	FEO/2
17	Make From Usage Option	FEO/2
31	Make From Usage Option Group	FEO/2
20	Next Assembly Usage Occurrence	FEO/2
23	Physical Unit Lot	FEO/3
24	Physical Unit Open Ended Range	FEO/3
14	Planned Physical Unit	FEO/3
1	Product Item	FEO/1
5	Product Item Usage Traversal	FEO/2, FEO/3
2	Product Item Version	FEO/1, FEO/3
29	Product Item Version Contract	FEO/1
28	Product Item Version Definition Material	FEO/1
27	Product Item Version Definition Shape	FEO/1
4	Product Item Version Functional Definition	FEO/1, FEO/2
10	Product Model	FEO/3
15	Traversal Effectivity	FEO/3

The entities below are referenced within the PSCM Reference Model but are defined within other PDES reference models:

NUMBER	NAME
FEM-1	FEM
FEM-2	FEM Product Item Version Definition
GEO-4	Axis Placement
GEO-??	Shape Model
INT-1	Shape
MAT-1	Material Property
RA-4	Product Item Version Approval
RA-6	Product Item Version Security Classification
RA-7	Organization
RA-8	Date Time

SECTION 5: PSCM INFORMATION MODEL

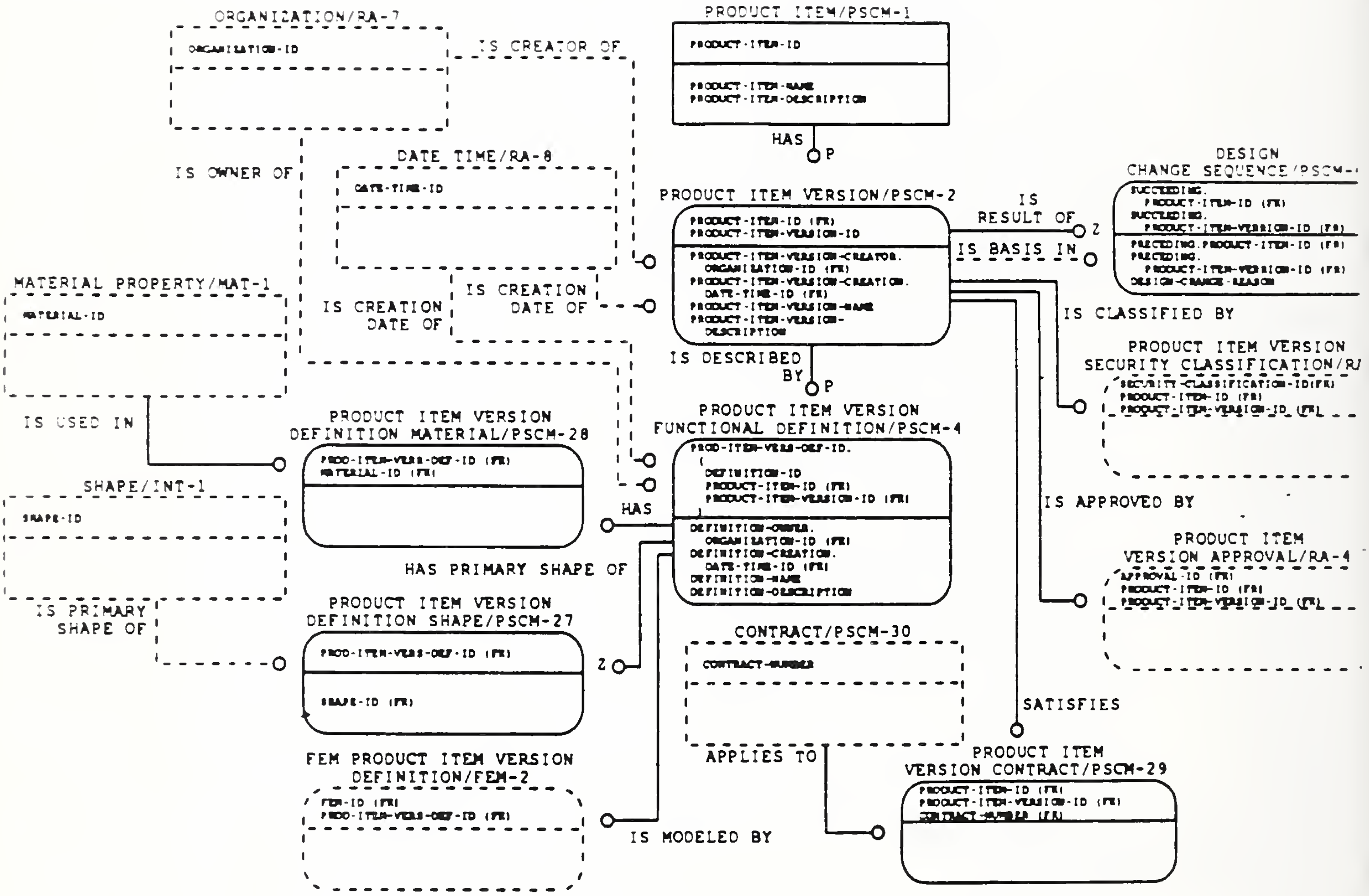


Figure D-59: Product Item Diagram (FEO/1)

SECTION 5: PSCM INFORMATION MODEL

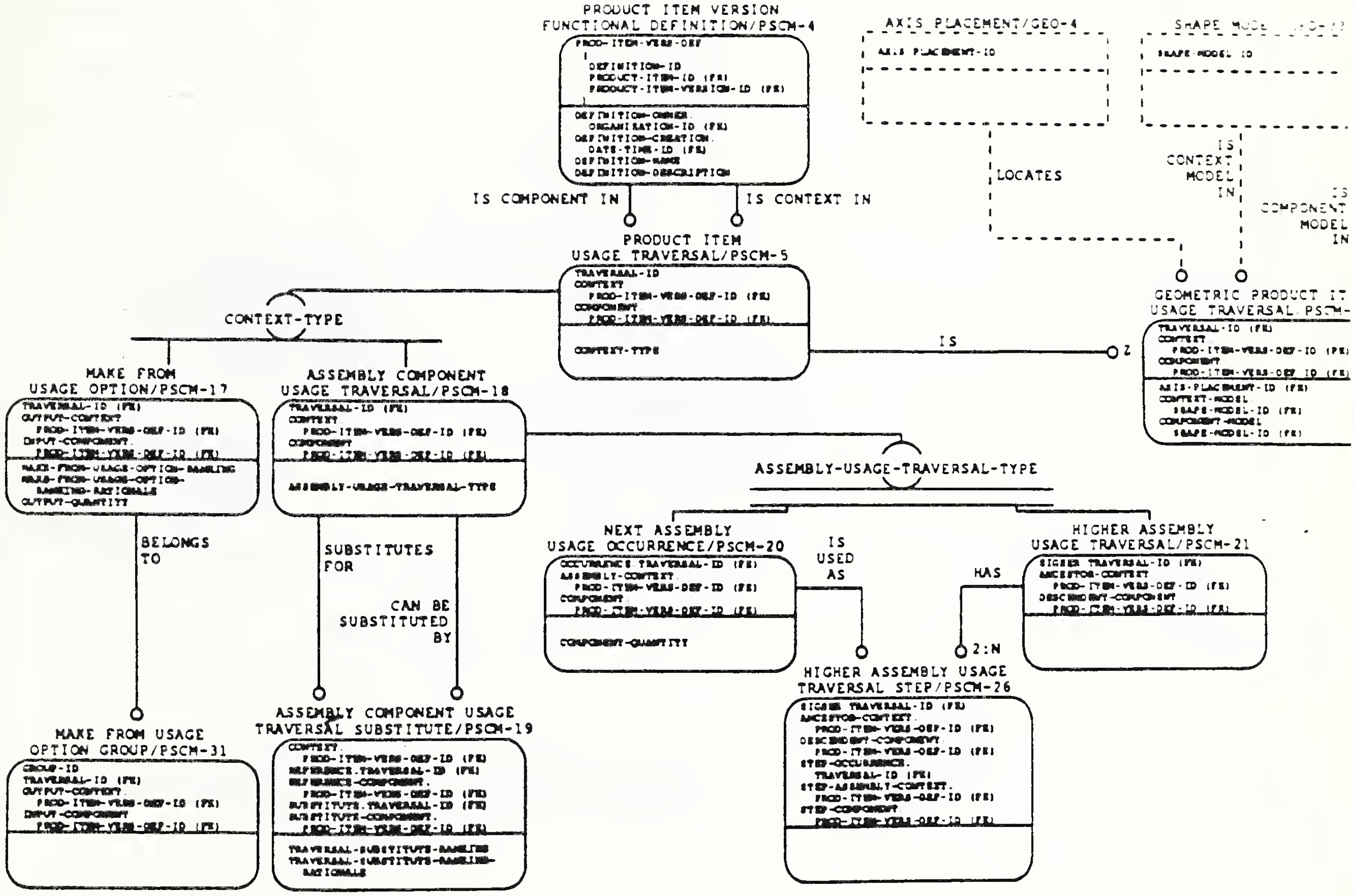


Figure D-60: Product Item Usage Diagram (FEO/2)

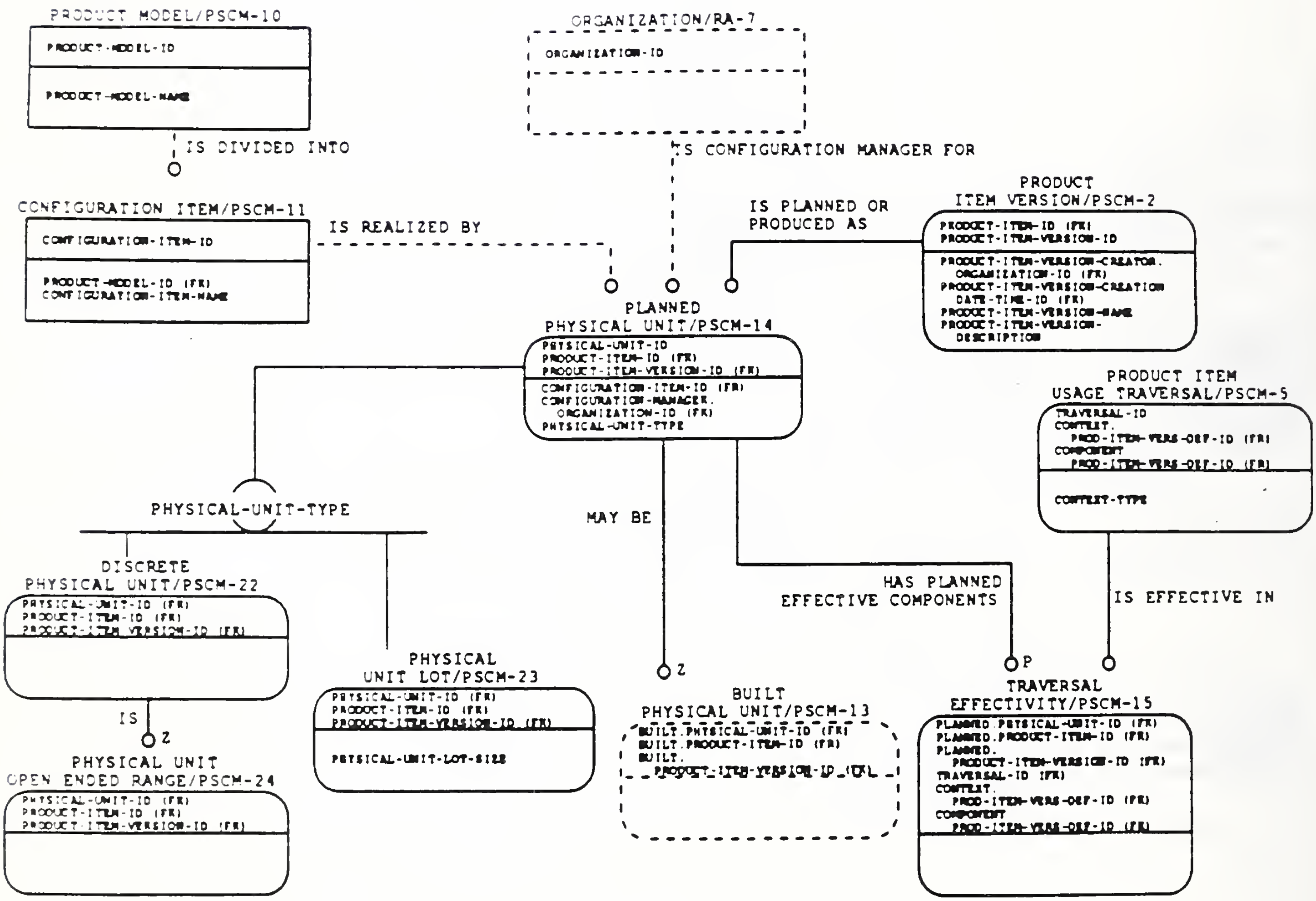


Figure D-61: Product Configuration Diagram (FEO/3)

5.3.2 Entity Glossary and Business Rules

Entity Name: **Product Item**

Entity Number: **PSCM-1**

The generic description of one or more versions of a (thing) that is, or is intended to be, produced, or is consumed in a production process.

This entity identifies those *Product Item Versions* which fulfill the generic description of the *Product Item*. Although one *Product Item Version* may fulfill the generic requirements of more than one *Product Item*, common practice dictates that each is restricted so as to belong to one and only one *Product Item*.

This entity represents a prevalent concept that there are deviations in the defined characteristics of *Product Item Versions* that are such that they represent variants of the same *Product Item*.

Every *Product Item* has at least one *Product Item Version* and may have more.

It is important to note that the rules for determining if variations to a design are "small" enough to constitute membership within the same *Product Item* are company specific. A common rule is that "small" variations do not effect the fit, form, and function of the *Product Item* (all versions of a *Product Item* are interchangeable throughout all usages of it within higher assemblies). This determination is subjective, however, and varies greatly among companies and departments. It is also highly life-cycle dependent, as the creator normally does not know at the time of creation all of the higher assembly usages in which the item may be applied, and therefore its degree of interchangeability.

As such, this entity only captures information that at the time of creation the variations were deemed "small", not that they are still considered to be "small".

Primary Key Attributes

Product-Item-ID
The unique identification label for the *Product Item*, such as a part number, stock item number, etc.

Other Attributes

Product-Item-Name
The name of the *Product Item*.

Product-Item-Description
A description of the *Product Item*.

Business Rules

Every *Product Item/1*:

- has one or more *Product Item Version/2*(s).

Express Declaration

```
*)
 TYPE
   product_item_versions   = LIST [1:#] OF product_item_version;
 END_TYPE;

 ENTITY product_item;
   product_item_id          : UNIQUE STRING;
   product_item_name        : STRING;
   product_item_description : STRING;
   versions                 : product_item_versions;
 WHERE

  --Constrain the 'product_item_version_id's to be unique within 'versions'.

   (version_id_unique_in(versions)=TRUE);

 END_ENTITY;
(*
```

This function controls the uniqueness of the 'Product Item Version ID' within a list of *Product Item Versions* for a *Product Item*. The required constraint is that the 'Product Item Version ID's must be unique across all *Product Item Versions* contained within a Versions list for a given *Product Item*. This function is TRUE if the uniqueness constraint is met, otherwise it is FALSE.

```
*)
 FUNCTION version_id_unique_in(versions: product_item_versions): LOGICAL;
 LOCAL
   k1,k2,k3 : NUMBER;
   v1,v2    : product_item_version;
 END_LOCAL;
   k1 := SIZEOF(versions);
   REPEAT k2 := 1 TO k1-1;
    v1 := POSITION(versions,k2);
    REPEAT k3 := k2+1 TO k1;
     v2 := POSITION(versions,k3);
     IF (v1.product_item_version_id = v2.product_item_version_id) THEN
      RETURN(FALSE);
     END_IF;
    END_REPEAT;
   END_REPEAT;
   RETURN(TRUE);
 END_FUNCTION;
(*
```

Entity Name: **Product Item Version**

Entity Number: **PSCM-2**

A specific description of an existing concept which defines the characteristics of a *Product Item*. This description is commonly referred to as the "design", and aggregates information for a (thing) that is, or is intended to be, produced, or is consumed in a production process (e.g. a part, stock material item, etc.)

Although *Product Item Versions* may share common information (especially those referred to by the same *Product Item*), each is a unique description for a achieving a particular concept, be it small or large; each, by itself, is considered a separate design.

Primary Key Attributes

Product-Item-ID (FK)

Product-Item-Version-ID
The unique identification label for the *Product Item Version*, such as a part version number.

Other Attributes

Product-Item-Version-Name
The name of the *Product Item Version*.

Product-Item-Version-Description
A description of the *Product Item Version*.

Product-Item-Version-Creator.Organization-ID (FK)
The identification of the creating organization (company, department, section, person, title, and/or project) for the *Product Item Version*.

Product-Item-Version-Creation.Date-Time-ID (FK)
The date and time of the creation of the *Product Item Version*.

Business Rules

Every *Product Item Version/2*:

- is described by one or more *Product Item Version Functional Definition/4*(s).
- is the result of zero or one *Design Change Sequence/6*.
- is the basis in zero, one, or many *Design Change Sequence/6*(s).
- is classified by zero, one, or many *Product Item Version Security Classification/RA-6*(s).
- satisfies zero, one, or many *Product Item Version Contract/29*(s).
- is approved by zero, one, or many *Product Item Version Approval/RA-4*(s).

- is planned or produced as zero, one, or many *Planned Physical Unit/14*(s).
- belongs to one *Product Item/1*.
- is created by one *Organization/RA-7*.
- is created at one *Date Time/RA-8*.

Express Declaration

Product Item Version Contract, *Product Item Version Approval*, and *Product Item Version Security Classification* have been incorporated here as attributes `contract_numbers`, `approvals`, and `security_classifications`, respectively.

```
*)
 ENTITY product_item_version;
   product_item_version_id          : STRING;
   product_item_version_name        : STRING;
   product_item_version_description : STRING;
   product_item_version_creator     : organization;
   product_item_version_creation    : date_time;
   contract_numbers                 : OPTIONAL SET [1:#] OF STRING;
   approvals                        : OPTIONAL SET [1:#] OF approval;
   security_classifications         : OPTIONAL SET [1:#] OF
                                        security_classification;
   definitions                      : SET [1:#] OF
                                        product_item_version_functional_definition;
 WHERE
   MEMBER(product_item, 1, 1);
 END_ENTITY;
(*
```

Entity Name: **Product Item Version Functional Definition**

Entity Number: **PSCM-4**

A unique description of the underlying relationships and information defining *Product Item Versions*.

Functional Definitions may be defined for many reasons, but probably the most common is for different organizational definitions of a *Product Item Version*'s parts list (bill-of-material). For example, the component parts list of a multi-level assembly may vary considerably depending on the particular purpose or organization for which it is intended, such as between "as designed", "as planned", or "as serviced".

Functional Definitions may also be defined to provide for restricted or expanded definitions. Consider a primary design definition of an assembly which includes *Assembly Component Usage Traversal Substitutes*. The use of any particular substitute is considered to have no appreciable effect on the final product; the substitutes are interchangeable. However, different applications (analysis, inspection, process planning, etc.) may have greatly different results based on which substitute is used. *Functional Definition* allows (but does not require) each application to construct it's own definitions for the assembly, which may have fewer or more *Assembly Component Usage Traversal Substitutes* allowed, and to relate the results of the application to those restricted or expanded definitions.

Each *Product Item Version* must be associated with at least one *Functional Definition* (usually a definition created by an "engineering design" organization), and may be associated with more than one. This supplies a three-part product identification: A *Product Item* has one or more *Product Item Versions*, each of which has one or more *Functional Definitions*.

Primary Key Attributes

Product-Item-ID (FK)
Product-Item-Version-ID (FK)

Definition-ID
The unique identification of the *Product Item Version Functional Definition*.

Other Attributes

Definition-Owner.Organization-ID (FK)
The identification of the owning organization (company, department, section, person, title, and/or project) for the *Product Item Version Functional Definition*.

Definition-Name
The name of the *Product Item Version Functional Definition*.

Definition-Description
A description of the *Product Item Version Functional Definition*.

Definition-Creation.Date-Time-ID (FK)
The date and time of the creation of the *Product Item Version Functional Definition*.

Business Rules

Every *Product Item Version Functional Definition/4*:

- is the context in zero, one, or many *Product Item Usage Traversal/5*(s).
- is the component in zero, one, or many *Product Item Usage Traversal/5*(s).
- has zero, one, or many *Product Item Version Definition Material/28*(s).
- has a primary shape of zero or one *Product Item Version Definition Shape/27*.
- is modeled by zero, one, or many *FEM Product Item Version Definition/FEM-2*(s).
- is a description of one *Product Item Version/2*.
- is owned by one *Organization/RA-7*.
- is created at one *Date-Time/RA-8*.

Express Declaration

Product Item Version Definition Shape and *Product Item Version Definition Material* have been incorporated here as attributes `primary_shape` and `materials`, respectively.

```
*)
 ENTITY product_item_version_functional_definition;
   version                  : product_item_version;
   definition_owner         : organization;
   definition_name          : STRING;
   definition_description   : STRING;
   definition_creation      : date_time;
   primary_shape            : OPTIONAL shape;
   materials                : OPTIONAL SET [1:#] OF material_property;
 WHERE
   MEMBER(product_item_version, 1, 1);
 END_ENTITY;
(*
```

Entity Name: **Product Item Usage Traversal**

Entity Number: **PSCM-5**

A *Product Item Usage Traversal* is the use of an instance of a *Product Item Version Functional Definition* in the context of a higher level *Product Item Version Functional Definition* (i.e. *Assembly Component Usage Traversal*), or in the context of an output *Product Item Version Functional Definition* (i.e., *Make From Usage Option*).

A *Product Item Version Functional Definition* may be used many times within a given context *Product Item Version Functional Definition*. This entity allows for the unique identification of each individual usage of interest.

Product Item Usage Traversal serves to uniquely identify instances of a component or output *Product Item Version Functional Definition* for purposes of associating information unique to those instances, such as location information, effectivity, and/or functional requirements.

Because these traversals may be established for functional reasons, when location may or may not be known, a *Product Item Usage Traversal* is not constrained to always have a geometric relationship (location and orientation) relative to its context.

Primary Key Attributes

Context.Definition-ID (FK)
Context.Product-Item-ID (FK)
Context.Product-Item-Version-ID (FK)
Component.Definition-ID (FK)
Component.Product-Item-ID (FK)
Component.Product-Item-Version-ID (FK)

Traversal-ID
: The unique identification of the usage of the component relative to the context. This must be unique across all usages of a component *Product Item Version Functional Definition* within the given context.

Other Attributes

Context-Type
: The type of context. Identifies the traversal as being either an *Assembly Component Usage Traversal* or a *Make From Usage Option*.

Business Rules

Every *Product Item Usage Traversal/5*:

- may be either a *Make From Usage Option/17* or an *Assembly Component Usage Traversal/18*.
- is effective in zero, one, or many *Traversal Effectivity/15*(s).
- is zero or one *Geometric Product Item Usage Traversal/16*.

- identifies one *Product Item Version Functional Definition/4* as being the context of one component *Product Item Version Functional Definition/4*.

Express Declaration

The 'Context' and 'Component' identifier attributes have been moved down from this entity to it's sub-types in order to capture the semantics of the role names given them within the sub-type entities.

```
*)
 ENTITY product_item_usage_traversal
          SUPERTYPE OF (make_from_usage_option OR
                        assembly_component_usage_traversal);
 END_ENTITY;
(*
```

Entity Name: **Design Change Sequence**

Entity Number: **PSCM-6**

A description of the association of two *Product Item Versions* in which one (the preceding) was the basis for the creation or definition of the other (the succeeding).

The preceding and succeeding *Product Item Versions* can be versions of the same *Product Item* or of different *Product Items*.

A *Product Item Version* does not always have a preceding *Product Item Version* from which its design is derived, as in the case of an "initial" design.

This entity differs from *Make From Usage Option* in that a design change captures information regarding the change of the description (design), not information regarding changes to manifestations (*Planned* or *Built Physical Units*) of the design.

Primary Key Attributes

Succeeding.Product-Item-ID (FK)
Succeeding.Product-Item-Version-ID (FK)

Other Attributes

Preceding.Product-Item-ID (FK)
Preceding.Product-Item-Version-ID (FK)

Design-Change-Reason
A description of the reason for the design change.

Business Rules

Every *Design Change Sequence/6*:

- identifies one *Product Item Version/2* as being the basis for one resulting *Product Item Version/2*.

Express Declaration

```
*)
 ENTITY design_change_sequence;
   succeeding            : UNIQUE product_item_version;
   preceding             : product_item_version;
   design_change_reason : STRING;
 WHERE
   (preceding <> succeeding);
 END_ENTITY;
(*
```

Entity Name: Product Model

Entity Number: PSCM-10

An identification of a collection of specific product features that are associated with marketing requirements.

Models of a product are essentially marketing ideas, but are used to carry through the *Planned* and *Built Physical Unit*'s identification.

Each manifestation (*Planned Physical Unit*) of a *Product Item Version* is associated with one and only one *Product Model* (via the *Configuration Item*), although another manifestation (*Planned Physical Unit*) for the same *Product Item Version* may be associated with a different *Product Model*.

Primary Key Attributes

Product-Model-ID
The unique identification label for the *Product Model*, such as a sales model number.

Other Attributes

Product-Model-Name
The name of the *Product Model*.

Business Rules

Every *Product Model/10*:

- is divided into zero, one, or many *Configuration Item/11*(s).

Express Declaration

```
*)
 ENTITY product_model;
   product_model_id   : UNIQUE STRING;
   product_model_name : STRING;
 END_ENTITY;
(*
```

Entity Name: Configuration Item

Entity Number: PSCM-11

A *Configuration Item* is the identification of a portion of a *Product Model* for the purpose of managing it's configuration.

All configuration management ("as planned", "as built", etc.) is tracked against these *Configuration Items*.

A *Configuration Item* can be an entire *Product Model* or some portion thereof. It may be sold as part of other *Configuration Items* or by itself.

Primary Key Attributes

Configuration-Item-ID
  The unique identification label of the *Configuration Item*.

Other Attributes

Product-Model-ID (FK)

Configuration-Item-Name
  The name of the *Configuration Item*.

Business Rules

Every *Configuration Item/11*:

- is realized by zero, one, or many *Planned Physical Unit/14*(s).
- is a division of one *Product Model/10*.

Express Declaration

```
*)
 ENTITY configuration_item;
   configuration_item_id   : UNIQUE STRING;
   product_model_id        : product_model;
   configuration_item_name : STRING;
 END_ENTITY;
(*
```

Entity Name: Built Physical Unit

Entity Number: **PSCM-13**

A *Planned Physical Unit* which has been manufactured, as opposed to a "concept" of the unit before it is built.

This entity is intended to allow for tracking the actual production or component usage of the unit as it was built, which may or may not be the same as the way it was intended (planned) to be built.

Note:

The current scope of this model is limited to *Planned Physical Units* only. *Built Physical Unit* has not been addressed in detail. It has been added as a "shadow" entity in order to emphasize that there are other types of physical units besides "planned", and to clarify the current scope of the model. It also serves to show where *Built Physical Unit* may eventually fit into the model.

Primary Key Attributes

Built.Physical-Unit-ID (FK)
Built.Product-Item-ID (FK)
Built.Product-Item-Version-ID (FK)

Other Attributes

None

Business Rules

Every *Built Physical Unit/13*:

- is a *Planned Physical Unit/14*.

Express Declaration

As stated in the above note, this entity has not been included within the current scope of the PSCM model. The Express declaration for this entity has been intentionally omitted.

Entity Name: **Planned Physical Unit**

Entity Number: **PSCM-14**

The identification of an intended physical manifestation of a *Product Item Version* for the purpose of planning the production of one or more *Built Physical Units*.

A *Planned Physical Unit* identifies a single unit of product (such as a single serial or lot number) for which a given configuration is intended. This intended configuration is typically, but not always, specified by a "Configuration Control Board" or equivalent organization after the design phase of a product is complete.

Although all *Planned Physical Units* are intended to be produced, some may never be.

A *Planned Physical Unit* is associated with one and only one *Configuration Item*. It may be expressed as a single item (*Discrete Physical Unit*) or as a group of items (*Physical Unit Lot*).

Primary Key Attributes

Product-Item-ID (FK)
Product-Item-Version-ID (FK)

Physical-Unit-ID
The unique identification label for the *Planned Physical Unit*, such as a serial or lot number.

Other Attributes

Configuration-Item-ID (FK)

Configuration-Manager.Organization-ID (FK)
The identification of the configuration manager (company, department, section, person, title, and/or project) responsible for the *Planned Physical Unit*.

Physical-Unit-Type
Identifies the *Planned Physical Unit* as a *Discrete Physical Unit* or a *Physical Unit Lot*.

Business Rules

Every *Planned Physical Unit/14*:

- may be zero or one *Built Physical Unit/13*.
- may be either a *Discrete Physical Unit/22* or a *Physical Unit Lot/23*.
- is produced or planned for one *Product Item Version/2*.
- is the realization of one *Configuration Item/11*.
- has planned effective components of one or more *Traversal Effectivity/15*(s).
- has one managing *Organization/RA-7* for it's configuration.

Express Declaration

This entity has been split into two entities to capture an existence constraint The required constraint is that the 'Product Item Version ID' of the referenced *Product Item Version* and the 'Physical Unit ID' form a unique identification. Multiple *Planned Physical Units* referencing the same *Product Item Version* cannot have the same 'Physical Unit ID'. However, multiple *Planned Physical Units* can have the same 'Physical Unit ID' if they reference different *Product Item Versions*.

To satisfy this constraint, *Planned Physical Unit One* has been created which contains only the attributes for which the uniqueness applies. *Planned Physical Unit* then references it with a UNIQUE qualifier, which guarantees that the attribute pair is unique for all *Planned Physical Units*.

The entity *Traversal Effectivity* has been incorporated into *Planned Physical Unit* as attribute `effective_traversals`.

```
*)
 ENTITY planned_physical_unit_one;
   physical_unit_id : STRING;
   design           : product_item_version;
 END_ENTITY;


 ENTITY planned_physical_unit
          SUPERTYPE OF (discrete_physical_unit OR
                        physical_unit_lot);
   physical_unit_1        : UNIQUE planned_physical_unit_one;
   configuration_item_id : configuration_item;
   configuration_manager : organization;
   effective_traversals  : SET [1:#] OF product_item_usage_traversal;
 END_ENTITY;
(*
```

Entity Name: Traversal Effectivity

Entity Number: PSCM-15

Information that a distinct *Product Item Usage Traversal* (either an *Assembly Component Usage Traversal* or a *Make From Usage Option*) is supposed to appear in specific *Planned Physical Units*.

Product Item Versions normally allow for many optional or substitute components, all of which fulfill the design's objective. *Traversal Effectivity* designates which optional or substitute components are to be used when producing a physical manifestation of the description. As such, this entity allows for capturing a more precise intended configuration (assembly/component or make-from usage) for a *Planned Physical Unit* than that allowed within the description of the *Product Item Version*'s definition.

Primary Key Attributes

Planned.Physical-Unit-ID (FK)
Planned.Product-Item-ID (FK)
Planned.Product-Item-Version-ID (FK)
Traversal-ID (FK)
Context.Definition-ID (FK)
Context.Product-Item-ID (FK)
Context.Product-Item-Version-ID (FK)
Component.Definition-ID (FK)
Component.Product-Item-ID (FK)
Component.Product-Item-Version-ID (FK)

Other Attributes

None

Business Rules

Every *Traversal Effectivity/15*:

- identifies one *Product Item Usage Traversal/5* which is effective in one *Planned Physical Unit/14*.

Express Declaration

The entity *Traversal Effectivity* has been incorporated into *Planned Physical Unit* as attribute **effective traversals**.

Entity Name: **Geometric Product Item Usage Traversal**

Entity Number: **PSCM-16**

A *Product Item Usage Traversal* in which the component has a unique geometric orientation or spatial relationship relative to its context.

Not all *Product Item Usage Traversals* are required to be *Geometric Product Item Usage Traversals*, as they may have been identified for functional reasons where location may not be known.

At the present time, this entity allows for the capture of at most one location for a component relative to the context. This can accommodate static, fixed locations for non-deformable items only. This restriction may require modification in the future in order to accommodate multiple degrees of freedom and deformable components.

Any type of *Product Item Usage Traversal* may be a *Geometric Product Item Usage Traversal*. This does allow for conflicting location information between multi-level traversals (*Higher Assembly Usage Traversals*) and single-level traversals (*Next Assembly Usage Occurrences*).

Primary Key Attributes

Traversal-ID (FK)
Context.Definition-ID (FK)
Context.Product-Item-ID (FK)
Context.Product-Item-Version-ID (FK)
Component.Definition-ID (FK)
Component.Product-Item-ID (FK)
Component.Product-Item-Version-ID (FK)

Other Attributes

Axis-Placement-ID (FK)
The location of the *Shape Model* representing the component *Product Item Version Functional Definition* relative to the *Shape Model* representing the context *Product Item Version Functional Definition*.

The *Shape Models* of the context and component must be associated to the correct *Product Item Version Functional Definitions* through *Product Item Version Definition Shape*.

Context-Model.Shape-Model-ID (FK)
The identification of the *Shape Model* representing the context.

Component-Model.Shape-Model-ID (FK)
The identification of the *Shape Model* representing the component.

Business Rules

Every *Geometric Product Item Usage Traversal/16*:

- is a *Product Item Usage Traversal/5*.

- is located by one *Axis Placement/GEO-4*.
- identifies one component *Shape Model/GEO-??* as located relative to one context *Shape Model/GEO-??*.

Express Declaration

```
*)
 ENTITY geometric_product_item_usage_traversal;
   traversal         : UNIQUE product_item_usage_traversal;
   location          : axis_placement;
   context_model     : shape_model;
   component_model   : shape_model;
 WHERE
   (context_model <> component_model);
 END_ENTITY;
(*
```

Entity Name: Make From Usage Option

Entity Number: PSCM-17

Establishes a relationship that a *Product Item Version* (the input) within a given *Functional Definition* can be physically transformed into another *Product Item Version* (the output).

In reality, a make-from relationship states that any physical manifestation of one design (the output) can be manufactured from any physical manifestation of another design (the input). This relationship is independent of any particular instances of the physical manifestations, however, and so is established between the designs (*Product Item Version Functional Definitions*).

The 'Definition ID' for the *Product Item Versions* is necessary because a single *Make From Usage Option* within one *Functional Definition* may actually be seen as several *Make From Usage Options* or sequences of options within another. The 'Traversal ID' is necessary since one input *Product Item Version* may be used to generate more than one instance of the output *Product Item Version*, and each instance (occurrence) of the output may need to be separately identified (such as for location or processing differences).

The input *Product Item Versions* are those typically called "stock" items, although they are not restricted to this. The resulting *Product Item Version* can be either a version of a new *Product Item* or a version of the input *Product Item*.

As an example, consider the case of a shaft which can be machined from either a casting or a forging. All three (the shaft, the forging, and the casting) are separate instances of *Product Item Versions*, and two instances of *Make From Usage Option* exist, one between the output shaft and the input forging, the other between the output shaft and the input casting.

This entity is a subtype of *Product Item Usage Traversal*, where the traversal is limited to a single step through a make-from tree (the output item relative to the immediately preceding input item), as opposed to a general multi-level traversal down an assembly tree (*Assembly Component Usage Traversal*). Multiple-level make-from traversals (the output relative to inputs prior to the immediately preceding) are not supported.

Primary Key Attributes

Traversal-ID (FK)
Output-Context.Definition-ID (FK)
Output-Context.Product-Item-ID (FK)
Output-Context.Product-Item-Version-ID (FK)
Input-Component.Definition-ID (FK)
Input-Component.Product-Item-ID (FK)
Input-Component.Product-Item-Version-ID (FK)

Other Attributes

Make-From-Usage-Option-Ranking

A ranking of the preference for use of the input *Product Item Version*. This is a positive integer value interpreted as follows: A low value indicates a high preference for the input *Product Item Version Functional Definition*, and a high value indicates a low preference.

Make-From-Usage-Option-Ranking-Rationale

The description of the rationale used for the ranking, such as "Cost" or "High Lead Time".

Output-Quantity

The quantity of physical manifestations of the output *Product Item Version* which will be produced from a single physical manifestation of the input *Product Item Version*. This is typically one, although it may be more (such as cutting one forging into two shafts).

Business Rules

Every *Make From Usage Option/17*:

- is a *Product Item Usage Traversal/5*.
- belongs to zero, one, or many *Make From Usage Option Group/31*(s).

Express Declaration

The 'Input' component and 'Output' context identifiers have been moved into this entity from its supertype (see *Product Item Usage Traversal*).

```
*)
 ENTITY make_from_usage_option
          SUBTYPE OF (product_item_usage_traversal);
   input                                : product_item_version_functional_definition;
   output                               : product_item_version_functional_definition;
   make_from_option_ranking             : INTEGER;
   make_from_option_ranking_rationale   : STRING;
   output_quantity                      : INTEGER;
 WHERE
   (make_from_option_ranking  > 0);
   (output_quantity > 0);
   (input.version <> output.version);
 END_ENTITY;
(*
```

Entity Name: **Assembly Component Usage Traversal**

Entity Number: **PSCM-18**

A *Product Item Usage Traversal* in which the context *Product Item Version* is any higher level (parent or ancestor) node within an assembly hierarchy.

This entity is a super-type of both single-level traversals (normal assembly-component relationships), and multi-level traversals (ancestor-descendent relationships).

Primary Key Attributes

Traversal-ID (FK)
Context.Definition-ID (FK)
Context.Product-Item-ID (FK)
Context.Product-Item-Version-ID (FK)
Component.Definition-ID (FK)
Component.Product-Item-ID (FK)
Component.Product-Item-Version-ID (FK)

Other Attributes

Assembly-Usage-Traversal-Type
The type of traversal. Identifies the traversal as being either a *Next Assembly Usage Occurrence* or a *Higher Assembly Usage Traversal.*

Business Rules

Every *Assembly Component Usage Traversal/18*:

- is either a *Next Assembly Usage Occurrence/20* or a *Higher Assembly Usage Traversal/21.*
- can be substituted by zero, one, or many *Assembly Component Usage Traversal Substitute/19*(s).
- can substitute for zero, one, or many *Assembly Component Usage Traversal Substitute/19*(s).
- is a *Product Item Usage Traversal/5.*

Express Declaration

The 'Ancestor' context and 'Descendent' component identifiers have been moved into this entity from it's supertype (see *Product Item Usage Traversal*).

```
*)
 ENTITY assembly_component_usage_traversal
          SUPERTYPE OF (next_assembly_usage_occurrence OR
                        higher_assembly_usage_traversal)
          SUBTYPE OF (product_item_usage_traversal);
```

```
  ancestor           : product_item_version_functional_definition;
  descendent         : product_item_version_functional_definition;
 WHERE
  (descendent.version <> ancestor.version);
 END_ENTITY;
(*
```

Entity Name: **Assembly Component Usage Traversal Substitute**

Entity Number: **PSCM-19**

Identifies an *Assembly Component Usage Traversal* for which the component may be used to take the place of another within the context of the same higher level *Product Item Version Functional Definition*.

This substitution only allows for the use of one instance of a component for another (either another instance of the same component or that of a different component) within the given context *Product Item Version Functional Definition*. It does not allow for the substitution within any other context.

The context *Product Item Version Functional Definitions* for the reference and substitute must be the same.

The instance of the substitute component is not required to have the same spatial relationship relative to the context, nor the same quantity.

This entity allows one-way substitution only: If, within a given context, A is a substitute for B, then B is not to be assumed a substitute for A, unless explicitly stated so in another instance of this entity.

Assembly Component Usage Traversal Substitute establishes mutual exclusion between the reference and substitute components. If A is a substitute for B within context C, then either A or B must be used, but not both, within an eventual *Built Physical Unit* for C.

This entity also may be used to eliminate version "roll-up", i.e. the re-identification of all higher level assemblies when a new version of a lower level component is created. For instance, if A_2 is a new version of A_1, but is considered to be "form, fit, and function" interchangeable with A_1, then A_2 can be captured as a substitute for A_1 (possibly with a higher ranking), without modifying the 'Product Item ID' or 'Product Item Version ID' of the context *Product Item Version* in which both may be used. This substitution, however, is only valid within the given *Functional Definition* of the context (e.g. A_2 may be deemed as equivalent within the "as designed" definition of an assembly, but may or may not be deemed equivalent within an "assembly planning" definition).

Primary Key Attributes

Context.Definition-ID (FK)
Context.Product-Item-ID (FK)
Context.Product-Item-Version-ID (FK)
Reference.Traversal-ID (FK)
Reference-Component.Definition-ID (FK)
Reference-Component.Product-Item-ID (FK)
Reference-Component.Product-Item-Version-ID (FK)
Substitute.Traversal-ID (FK)
Substitute-Component.Definition-ID (FK)
Substitute-Component.Product-Item-ID (FK)
Substitute-Component.Product-Item-Version-ID (FK)

Other Attributes

Traversal-Substitute-Ranking

A ranking of the preference for use of the *Assembly Component Usage Traversal Substitute*. This is a positive integer value interpreted as follows: A low value indicates a high preference for the substitute *Product Item Version Functional Definition*, and a high value indicates a low preference.

Traversal-Substitute-Ranking-Rationale

A description of the rationale used for the ranking, such as "Cost" or "High Lead Time".

Business Rules

Every *Assembly Component Usage Traversal Substitute/19*:

- identifies one *Assembly Component Usage Traversal/18* as being a substitute for another *Assembly Component Usage Traversal/18*.

Express Declaration

```
*)
 ENTITY assembly_component_usage_traversal_substitute;
   reference                               : assembly_component_usage_traversal;
   substitute                              : assembly_component_usage_traversal;
   traversal_substitute_ranking            : INTEGER;
   traversal_substitute_ranking_rationale  : STRING;
 WHERE
   (traversal_substitute_ranking > 0);
   (reference.ancestor = substitute.ancestor);
   (reference.descendent <> substitute.descendent);
 END_ENTITY;
(*
```

Entity Name: **Next Assembly Usage Occurrence**

Entity Number: **PSCM-20**

The representation of an organization of a product which relates a *Product Item Version Functional Definition* to its immediate parent within an assembly hierarchy.

This entity is used to derive the typical indented parts list for a product. By sequentially tracing through the context of these occurrences (D is a component in context B, B is component in context G, etc.), or through the components (G is the context of component B, B is the context of component D, etc.), it is possible to derive the normal indented parts list for a product. It is also possible to determine both where an item is used and what is used within it.

For explicit locations of each instance of a component relative to it's context assembly, an instance of this entity must exist for each located instance of the component.

For simple parts list information, the quantity is the count of components within the given context. Because each instance of the component is not normally separately identified, location information may not exist.

This model does not prohibit the inclusion of redundant information; both non-geometric, quantified occurrences for typical indented parts list information, and explicit occurrences for each instance of a component may exist, although the former may be derived from the latter.

Primary Key Attributes

Occurrence.Traversal-ID (FK)
Assembly-Context.Definition-ID (FK)
Assembly-Context.Product-Item-ID (FK)
Assembly-Context.Product-Item-Version-ID (FK)
Component.Definition-ID (FK)
Component.Product-Item-ID (FK)
Component.Product-Item-Version-ID (FK)

Other Attributes

Component-Quantity
The number of instances of the component which belong to this occurrence. For parts list information, this is normally the count of components within the assembly. For explicit location information, this quantity is normally one.

Business Rules

Every *Next Assembly Usage Occurrence/20*:

- is an *Assembly Component Usage Traversal/18*.
- can be used as zero, one, or many *Higher Assembly Usage Traversal Step/26*(s).

Express Declaration

```
*)
 ENTITY next_assembly_usage_occurrence
           SUBTYPE OF (assembly_component_usage_traversal);
   component_quantity : INTEGER;
 WHERE
   (component_quantity > 0);

 -- Future work needs to enforce an acyclic graph structure here!!!

 END_ENTITY;
(*
```

Entity Name: **Higher Assembly Usage Traversal**

Entity Number: **PSCM-21**

The unique identification of a *Product Item Version Functional Definition* used within the context of a higher level parent than the immediate parent within an assembly hierarchy.

The most common use of this entity is to capture assembly requirements at some higher level of assembly, where the identification of instances of some (much) lower level component is required. For example, part A in assembly B must mate within a specified tolerance with part C in assembly D, when assemblies B and D are used in assembly E. To uniquely identify the A and C within assembly E, two instances of *Higher Assembly Usage Traversal* would exist with assembly E as the context; one to uniquely identify the lower level part A, the other for lower level part C.

This higher parent assembly is viewed as the top node of the assembly tree within the given context; any relationships or properties associated with the traversal are not dependent on any higher level of assembly than the given context.

Primary Key Attributes

Higher.Traversal-ID (FK)
Ancestor-Context.Definition-ID (FK)
Ancestor-Context.Product-Item-ID (FK)
Ancestor-Context.Product-Item-Version-ID (FK)
Descendent-Component.Definition-ID (FK)
Descendent-Component.Product-Item-ID (FK)
Descendent-Component.Product-Item-Version-ID (FK)

Other Attributes

None

Business Rules

Every *Higher Assembly Usage Traversal/21*:

- is an *Assembly Component Usage Traversal/18.*
- has two or more *Higher Assembly Usage Traversal Step/26*(s).

Express Declaration

The entity *Higher Assembly Usage Traversal Step* has been incorporated here as the attribute **traversal steps**.

```
*)
 TYPE
   traversal_steps = LIST [2:#] OF next_assembly_usage_occurrence;
 END_TYPE;
```

```
ENTITY higher_assembly_usage_traversal
         SUBTYPE OF (assembly_component_usage_traversal);
  steps : traversal_steps;
WHERE

   -- Insure a well-ordered traversal list.

  (ordered_traversal(ancestor,descendent,steps) = TRUE);

END_ENTITY;
(*
```

This function constrains the ordering of the **traversal_steps** within a *Higher Assembly Usage Traversal*. The ordering must be such that:

- **ancestor** is the ancestor within the first step.
- **descendent** must be the descendent within the last step.
- No ancestor or descendent may appear more than once throughout the entire traversal (among all steps).
- The steps must be contiguous, i.e. for all i between zero and sizeof(steps), $step_{(i)}$.descendent = $step_{(i+1)}$.ancestor.

This function is **TRUE** if the traversal steps conform to the above criteria, otherwise it is **FALSE**.

```
*)
FUNCTION ordered_traversal(
           ancestor   : product_item_version_functional_definition;
           descendent : product_item_version_functional_definition;
           steps      : traversal_steps): LOGICAL;
LOCAL
  k1,k2,k3   : NUMBER;
  s1,s2      : next_assembly_usage_occurrence;
END_LOCAL;
  k1 := SIZEOF(steps);
  s1 := POSITION(steps,1);          -- validate start, end of traversal
  s2 := POSITION(steps,k1);
  IF (ancestor <> s1.ancestor) OR (descendent <> s2.descendent) THEN
   RETURN(FALSE);
  END_IF;
  REPEAT k2 := 1 TO k1-1;           -- insure unique ancestors, descendents
   s1 := POSITION(steps,k2);
   REPEAT k3 := k2+1 TO k1;
    s2 := POSITION(steps,k3);
    IF (s1.ancestor.version = s2.ancestor.version) OR
       (s1.descendent.version = s2.descendent.version) THEN
     RETURN(FALSE);
    END_IF;
```

```
    END_REPEAT;
   END_REPEAT;
   REPEAT k2 := 2 TO k1;              -- insure contiguous traversal steps
    s1 := POSITION(steps,k2-1);
    s2 := POSITION(steps,k2);
    IF (s1.descendent.version <> s2.ancestor.version) THEN
     RETURN(FALSE);
    END_IF;
   END_REPEAT;
   RETURN(TRUE);
 END_FUNCTION;
(*
```

Entity Name: **Discrete Physical Unit**

Entity Number: **PSCM-22**

Identifies a *Planned Physical Unit* as a single, discrete instance of an intended manifestation of a *Product Item Version*.

Discrete Physical Units are not divisible into non-distinct members, as are *Physical Unit Lots*.

This entity is normally used for "serialized" items, where each instance of an item is given a unique serial number.

Primary Key Attributes

Physical-Unit-ID (FK)
Product-Item-ID (FK)
Product-Item-Version-ID (FK)

Other Attributes

None

Business Rules

Every *Discrete Physical Unit/22*:

- is zero or one *Physical Unit Open Ended Range/24*.
- is a *Planned Physical Unit/14*.

Express Declaration

Physical Unit Open Ended Range has been incorporated here as attribute open_ended_range. This attribute is TRUE if the *Discrete Physical Unit* is an *Open Ended Range*, FALSE if it is not.

```
*)
 ENTITY discrete_physical_unit
          SUBTYPE OF (planned_physical_unit);
   open_ended_range : LOGICAL;
 END_ENTITY;
(*
```

Entity Name: Physical Unit Lot

Entity Number: PSCM-23

A *Planned Physical Unit* which is a "lot"; a collection of more than one intended physical manifestations, where each member of the lot is identifiable as belonging to the lot, but each is not discernible from others within the same lot.

Lots are used where *Planned Physical Units* are intended to be produced in "batches", and where characteristics which vary between the lots (such as a production method used or facility where created) are of importance, but not between individual members of a lot.

Primary Key Attributes

Physical-Unit-ID (FK)
Product-Item-ID (FK)
Product-Item-Version-ID (FK)

Other Attributes

Physical-Unit-Lot-Size
The quantity of members within the lot.

Business Rules

Every *Physical Unit Lot/23*:

- is a *Planned Physical Unit/14*.

Express Declaration

```
*)
 ENTITY physical_unit_lot
           SUBTYPE OF (planned_physical_unit);
   physical_unit_lot_size : INTEGER;
 WHERE
   (physical_unit_lot_size > 0);
 END_ENTITY;
(*
```

Entity Name: **Physical Unit Open Ended Range**

Entity Number: **PSCM-24**

Identifies a *Discrete Physical Unit* which is the beginning of a range for which *Traversal Effectivity* is to apply.

This allows for an "open-ended" range of *Discrete Physical Units* for which the end of the range is unknown, such as "for serial numbers 12 and on".

Primary Key Attributes

Physical-Unit-ID (FK)
Product-Item-ID (FK)
Product-Item-Version-ID (FK)

Other Attributes

None

Business Rules

Every *Physical Unit Open Ended Range/24*:

- is a *Discrete Physical Unit/22.*

Express Declaration

This entity has been incorporated into *Discrete Physical Unit* as attribute open_ended range.

Entity Name: **Higher Assembly Usage Traversal Step**

Entity Number: **PSCM-26**

The identification of the traversal steps (*Next Assembly Usage Occurrences*) required to establish a unique (much) lower level instance of a component for a *Higher Assembly Usage Traversal*.

To identify a unique instance of a lower level component, the entire assembly traversal, starting at the given context and proceeding down to the instance of the component desired, must exist within instances of *Next Assembly Usage Occurrence*. This entity then accumulates the steps required to give a unique traversal down the assembly tree.

For example, if two A's are used in B, and B is used in C, *Next Assembly Usage Occurrences* of BA_1, BA_2, and CB_1 would exist (subscripts denote the 'Traversal ID'). A *Higher Assembly Usage Traversal* to uniquely identify the first A in ancestor C would accumulate steps of CB_1 and BA_1.

A *Higher Assembly Usage Traversal* must be associated with at least two *Higher Assembly Usage Traversal Steps*.

Primary Key Attributes

Higher.Traversal-ID (FK)
Ancestor-Context.Definition-ID (FK)
Ancestor-Context.Product-Item-ID (FK)
Ancestor-Context.Product-Item-Version-ID (FK)
Descendent-Component.Definition-ID (FK)
Descendent-Component.Product-Item-ID (FK)
Descendent-Component.Product-Item-Version-ID (FK)
Step-Occurrence.Traversal-ID (FK)
Step-Assembly-Context.Definition-ID (FK)
Step-Assembly-Context.Product-Item-ID (FK)
Step-Assembly-Context.Product-Item-Version-ID (FK)
Step-Component.Definition-ID (FK)
Step-Component.Product-Item-ID (FK)
Step-Component.Product-Item-Version-ID (FK)

Other Attributes

None

Business Rules

Every *Higher Assembly Usage Traversal Step/26*:

- identifies one *Next Assembly Usage Occurrence/20* as being a traversal step for one *Higher Assembly Usage Traversal/21*.

Express Declaration

Higher Assembly Usage Traversal Step has been incorporated into *Higher Assembly Usage Traversal* as attribute traversal_steps.

Entity Name: **Product Item Version Definition Shape**

Entity Number: **PSCM-27**

Identifies the primary *Shape* associated with a particular *Product Item Version Functional Definition*.

Not all *Product Item Version Functional Definitions* are required to have a primary *Shape*, and can have no more than one primary *Shape*.

This entity captures only the primary shape for a *Product Item Version Functional Definition*. For alternate *Shape/INT-1*s, see the Shape reference model.

The *Shape* is currently related to *Product Item Version Functional Definition*, rather than the *Product Item Version*. This allows for (but does not require) a separate *Shape* (and related information, such as *Axis Placements*) to exist for each definition of the *Product Item Version*. For example, the "as designed" and "as serviced" definitions for an assembly may vary considerably in the way in which the components are assembled. These differences may not only include the assembly tree structure, but also related assembly requirements and restricted location information.

Primary Key Attributes

Definition-ID (FK)
Product-Item-ID (FK)
Product-Item-Version-ID (FK)

Other Attributes

Shape-ID (FK)

Business Rules

Every *Product Item Version Definition Shape/27*:

- identifies one *Shape/INT-1* as the primary shape for one *Product Item Version Functional Definition/4*.

Express Declaration

This entity has been incorporated into *Product Item Version Functional Definition* as attribute `primary_shape`.

Entity Name: **Product Item Version Definition Material**

Entity Number: **PSCM-28**

Identifies a *Material Property* that is used within a *Product Item Version Functional Definition*.

Not all *Product Item Version Functional Definitions* are required to be associated with a *Material Property*; the components within an assembly may be associated with a *Material Property*, but the assembly itself may not be.

They may also be associated with many (for non-homogeneous material composition).

Note:

Although this entity allows for multiple *Material Propertys* to be associated with a *Product Item Version Functional Definition*, at the current time it does not allow for the specification of the location of each within the *Shape* of the *Product Item Version Functional Definition*.

Primary Key Attributes

Definition-ID (FK)
Product-Item-ID (FK)
Product-Item-Version-ID (FK)
Material-ID (FK)

Other Attributes

None

Business Rules

Every *Product Item Version Definition Material/28*:

- identifies one *Material Property/MAT-1* as used in one *Product Item Version Functional Definition/4*.

Express Declaration

This entity has been incorporated into *Product Item Version Functional Definition* as attribute `materials`.

Entity Name: **Product Item Version Contract**

Entity Number: **PSCM-29**

The identification of the *Contract* under which a *Product Item Version* is created.

Primary Key Attributes

Product-Item-ID (FK)
Product-Item-Version-ID (FK)
Contract-Number (FK)

Other Attributes

None

Business Rules

Every *Product Item Version Contract/29*:

- identifies one *Product Item Version/4* as satisfying one *Contract/30.*

Express Declaration

This entity has been incorporated into *Product Item Version* as attribute contracts.

Entity Name: Contract

Entity Number: **PSCM-30**

An agreement or order for the procurement of supplies or services.

Note:

This entity is not included within the scope of the current PSCM model. However, it is recognized that a relationship does exist between it and the PSCM model. It has been added as a "shadow" entity in order to emphasize the bounds of the current scope and to show where it may eventually exist within the model.

Primary Key Attributes

Contract-Number

The unique identification label of the *Contract*.

Other Attributes

None

Business Rules

Every *Contract/30*:

- applies to zero, one, or many *Product Item Version Contract/29*(s).

Express Declaration

As stated above, this entity is not included within the current scope of the PSCM model. The primary key of 'Contract Number', however, is necessary within the model and has been incorporated into *Product Item Version* as attribute **contracts**.

Entity Name: Make From Usage Option Group

Entity Number: **PSCM-31**

A collection of *Make From Usage Option* instances which delimits one possible combination of the outputs that can be made from a single physical manifestation of an input.

The input *Product Item Versions* for all of the instances of *Make From Usage Option* within a single group must be the same.

For instance, an input bar stock item D can be cut twice so to create three output items, x, y, and z. *Make From Usage Option* instances Dx_1, Dy_1, and Dz_1 (subscripts denote the 'Traversal ID') would all exist and be gathered into one *Make From Usage Option Group*.

Multiple groups are possible for the same input item.

For example, the input bar stock item D may also be cut twice to produce two output items x and output item t, which would be collected into a group of Dx_1, Dx_2, and Dt_1. If each x has not been separately identified (a *Make From Usage Option* instance of Dx_n with a quantity of "2" exists), the group would contain instances of Dx_n and Dt_1.

Make From Usage Option instances may be shared between multiple groups.

Primary Key Attributes

Group-ID (FK)
Traversal-ID (FK)
Output-Context.Definition-ID (FK)
Output-Context.Product-Item-ID (FK)
Output-Context.Product-Item-Version-ID (FK)
Input-Component.Definition-ID (FK)
Input-Component.Product-Item-ID (FK)
Input-Component.Product-Item-Version-ID (FK)

Other Attributes

None

Business Rules

Every *Make From Usage Option Group/31*:

- identifies one member *Make From Usage Option/17.*

Express Declaration

```
*)
 TYPE
   make_from_group_members = LIST [1:#] OF make_from_usage_option;
 END_TYPE;
```

```
 ENTITY make_from_usage_option_group;
   members : make_from_group_members;
 WHERE                                  --Restrict all 'Input's to be the same
   (same_input_in(members) = TRUE);
 END_ENTITY;
(*
```

This function insures that the 'Input' attributes are the same across all members of a *Make From Option Usage Group*, and that the 'Output' attributes are different. This function is `TRUE` if the constraints are met, otherwise it is `FALSE`.

```
*)
 FUNCTION same_input_in(members: make_from_group_members): LOGICAL;
 LOCAL
   k1,k2,k3 : NUMBER;
   m1,m2    : make_from_usage_option;
 END_LOCAL;
   m1 := POSITION(members,1);
   k1 := SIZEOF(members);
   REPEAT k2 := 2 TO k1;                 -- All inputs must be the same
    m2 := POSITION(members,k2);
    IF (m1.input <> m2.input) THEN
     RETURN(FALSE);
    END_IF;
   END_REPEAT;
   REPEAT k2 := 1 TO k1-1;               -- All outputs must be different
    m1 := POSITION(members,k2);
    REPEAT k3 := k2+1 TO k1;
     m2 := POSITION(members,k3);
     IF (m1.output = m2.output) THEN
      RETURN(FALSE);
     END_IF;
    END_REPEAT;
   END_REPEAT;
   RETURN(TRUE);
 END_FUNCTION;
(*
```

5.4 Approvals Reference Model

This model is a **very** limited view of approvals and security classifications. It is currently restricted to the requirements found in the PSCM and FEM application models, and represents a synthesis of the related information from both.

Two notes on this model deserve attention:

- An overall problem exists in identifying information subsets for which approvals and classifications apply (see issue PSCM-29). This is particularly difficult when dealing with "aggregation" entities. Both the Mechanical Product Definition Committee and the PDES Logical Layer Integration Committee have agreed that some mechanism for identifying information subsets should be found before devoting much effort to the expansion of this model.

- The security classifications contained here only reflect limited information for military-type security levels, and do not address various company internal or proprietary classifications.

This model needs considerable work in the future. It is intended that it eventually be extracted from the PSCM document and be made into a separate application model.

5.4.1 Entity Pool

The following entities appear within the Approvals reference model and are defined on the following pages.

Entity Pool in Numeric Order

NUMBER	NAME	DIAGRAM
RA-1	Approval	FEO/4
RA-2	Release Approval	FEO/4
RA-3	FEM Approval	FEO/4
RA-4	Product Item Version Approval	FEO/4
RA-5	Security Classification	FEO/4
RA-6	Product Item Version Security Classification	FEO/4
RA-7	Organization	FEO/4
RA-8	Date Time	FEO/4

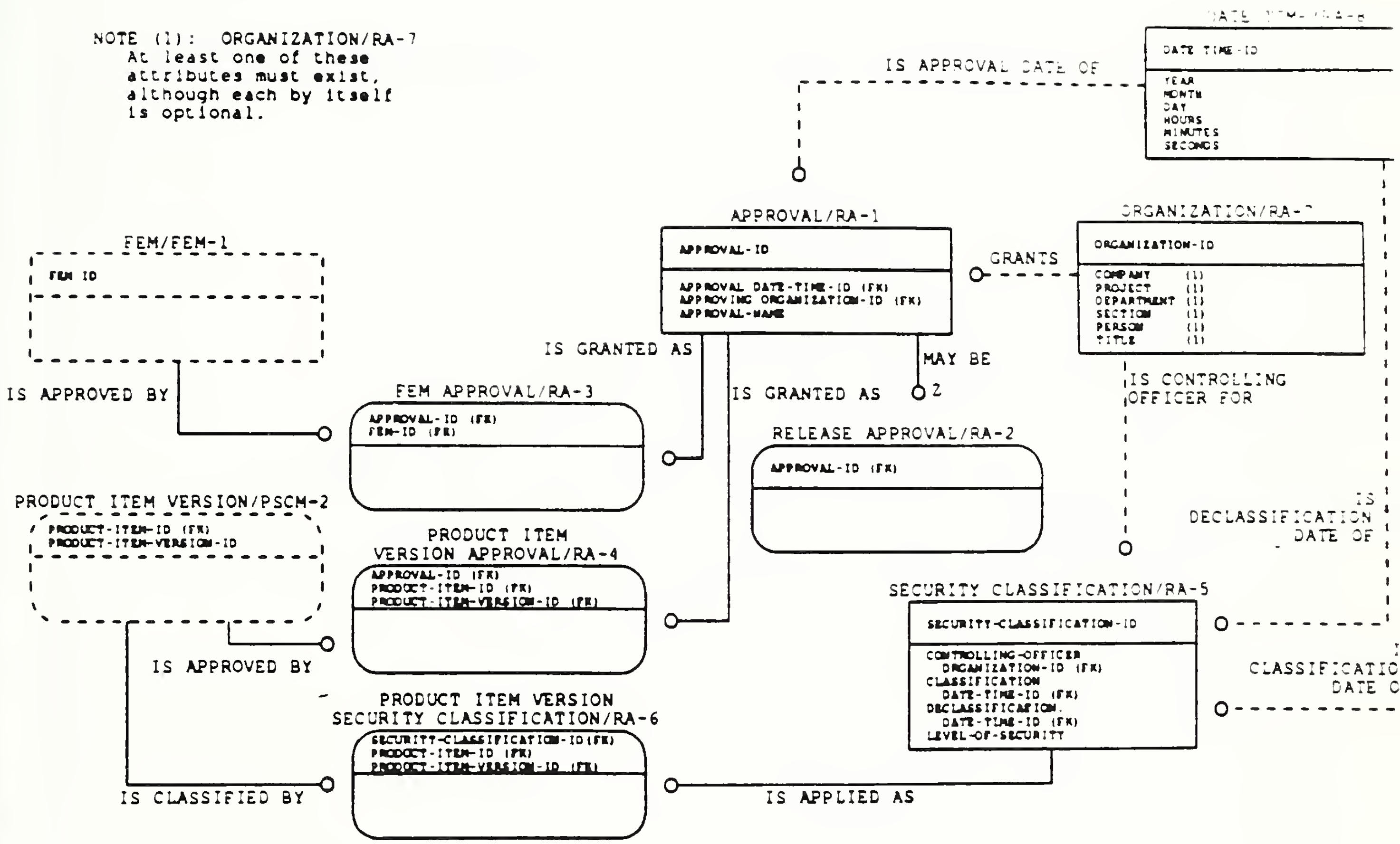
Entity Pool in Alphabetic Order

NUMBER	NAME	DIAGRAM
RA-1	Approval	FEO/4
RA-8	Date Time	FEO/4
RA-3	FEM Approval	FEO/4
RA-7	Organization	FEO/4
RA-4	Product Item Version Approval	FEO/4
RA-6	Product Item Version Security Classification	FEO/4
RA-2	Release Approval	FEO/4
RA-5	Security Classification	FEO/4

The entities below are referenced within the Approvals and Security Classifications reference model but are defined within other PDES reference models:

NUMBER	NAME
FEM-1	FEM
PSCM-2	Product Item Version

5.4.2 IDEF1x Diagram



APPROVALS AND SECURITY CLASSIFICATIONS

5.4.3 Entity Glossary and Business Rules

Entity Name: **Approval**

Entity Number: **RA-1**

Identifies an instance of a specific formal or official status granted to the definition of a "thing".

Primary Key Attributes

Approval-ID
The unique identification of the *Approval*.

Other Attributes

Approval.Date-Time-ID (FK)
The date and time on which the *Approval* was granted.

Approval-Name
The name of the *Approval*, e.g. "Design Approval", "Flutter Analysis Approval", etc..

Approving.Organization-ID (FK)
The identification of the approving organization (company, department, section, person, and/or project).

Business Rules

Every *Approval/RA-1*:

- may be zero or one *Release Approval/RA-2*.
- is granted as zero, one, or many *FEM Approval/RA-3*(s).
- is granted as zero, one, or many *Product Item Version Approval/RA-4*(s).
- is granted by one *Organization/RA-7*.
- is granted at one *Date Time/RA-8*.

Express Declaration

Approval/RA-1
Release Approval has been incorporated here as attribute `release`. This attribute is `TRUE` if the *Approval* is a *Release Approval*, `FALSE` if it is not.

```
*)
 ENTITY approval;
   approval_date          : date_time;
   approval_name          : STRING;
   approving_organization : organization;
   release                : LOGICAL;
 END_ENTITY;
(*
```

Entity Name: **Release Approval**

Entity Number: **RA-2**

An *Approval* in which a responsible organization has designated the definition of a "thing" as being "complete", i.e. correct and sufficient to allow for subsequent use or production.

Primary Key Attributes

Approval-ID (FK)

Other Attributes

None

Business Rules

Every *Release Approval/RA-2*:

- is an *Approval/RA-1*.

Express Declaration

This entity has been incorporated into *Approval* as attrbute `release`.

Entity Name: **FEM Approval**

Entity Number: **RA-3**

Identifies an *Approval* as granted to the definition of a Finite Element Model.

A *FEM* is not required to have any *Approvals* granted to it, and it may have many granted to it (either simultaneously or over time).

Each *FEM Approval*, however, must have one *FEM* to which it applies.

Primary Key Attributes

Approval-ID (FK)
FEM-ID (FK)

Other Attributes

None

Business Rules

Every *FEM Approval/RA-3*:

- identifies one *FEM/FEM-1* which is approved by one *Approval/RA-1*.

Express Declaration

This entity has been incorporated into *FEM* as attribute approval_ref.

Entity Name: **Product Item Version Approval**

Entity Number: **RA-4**

Identifies an *Approval* granted to the definition of a *Product Item Version*.

A *Product Item Version* is not required to have any *Approvals* granted to it, and it may have many granted to it (either simultaneously or over time).

Each *Product Item Version Approval*, however, must have one *Product Item Version* to which it applies.

Primary Key Attributes

Approval-ID (FK)
Product-Item-ID (FK)
Product-Item-Version-ID (FK)

Other Attributes

None

Business Rules

Every *Product Item Version Approval/RA-4*:

- identifies one *Product Item Version/PSCM-2* which is approved by one *Approval/RA-1*.

Express Declaration

This entity has been incorporated into *Product Item Version* as attribute approvals.

Entity Name: **Security Classification**

Entity Number: **RA-5**

Identifies a particular security classification state assigned to a "thing".

Primary Key Attributes

Security-Classification-ID
The unique identification of the *Security Classification*.

Other Attributes

Level-Of-Security
The level of security assigned. Allowed values are "Unclassified", "Confidential", "Secret", and "Top Secret".

Classification.Date-Time-ID (FK)
The date and time upon which the *Security Classification* became effective.

Declassification.Date-Time-ID (FK)
The intended date and time upon which the classification is be removed. This is not the date and time when it actually was removed. This date and time must be later than the date and time at which the classification became effective.

Controlling-Officer.Organization-ID (FK)
The identification of the primary controlling officer (company, department, section, person, and/or project) for the classification.

Business_Rules

Every *Security Classification/RA-5*:

- is applied as zero, one, or many *Product Item Version Security Classification/RA-6*(s).
- has one *Date Time/RA-8* at which it became effective.
- has one intended *Date Time/RA-8* for declassification.
- has one controlling *Organization/RA-7*.

Express Declaration

```
*)
 TYPE
   security_level       = ENUMERATION OF
                           (unclassified,confidential,secret,top_secret);
   date_time_comparator = ENUMERATION OF
                           (before, equal, after);
```

```
 END_TYPE;

 ENTITY security_classification;
   level                  : security_level;
   classification_date    : date_time;
   declassification_date  : date_time;
   controlling_officer    : organization;
 WHERE

 -- Constrain the declassification date to be later than the classification date.

   (compare_date_time(classification_date, declassification_date) = after);

 END_ENTITY;
(*
```

Given two dates, this function determines whether the second is before, equal to, or after the first.

```
*)
 FUNCTION compare_date_time( start_date : date_time;
                             end_date   : date_time) : date_time_comparator;

   IF (end_date.year    < start_date.year)    THEN RETURN(before); END_IF;
   IF (end_date.year    > start_date.year)    THEN RETURN(after) ; END_IF;
   IF (end_date.month   < start_date.month)   THEN RETURN(before); END_IF;
   IF (end_date.month   > start_date.month)   THEN RETURN(after) ; END_IF;
   IF (end_date.day     < start_date.day)     THEN RETURN(before); END_IF;
   IF (end_date.day     > start_date.day)     THEN RETURN(after) ; END_IF;
   IF (end_date.hours   < start_date.hours)   THEN RETURN(before); END_IF;
   IF (end_date.hours   > start_date.hours)   THEN RETURN(after) ; END_IF;
   IF (end_date.minutes < start_date.minutes) THEN RETURN(before); END_IF;
   IF (end_date.minutes > start_date.minutes) THEN RETURN(after) ; END_IF;
   IF (end_date.seconds < start_date.seconds) THEN RETURN(before); END_IF;
   IF (end_date.seconds > start_date.seconds) THEN
    RETURN(after) ;
   ELSE
    RETURN(equal);
   END_IF;
 END_FUNCTION;
(*
```

Entity Name: **Product Item Version Security Classification**

Entity Number: **RA-6**

Identifies the *Product Item Version* for which a *Security Classification* is assigned.

A *Product Item Version* is not required to have any security classifications assigned to it, and it may have many assigned to it over time.

Each *Product Item Version Security Classification*, however, must have one *Product Item Version* to which it is assigned.

Primary Key Attributes

Security-Classification-ID (FK)
Product-Item-ID (FK)
Product-Item-Version-ID (FK)

Other Attributes

None

Business Rules

Every *Product Item Version Security Classification/RA-6*:

- identifies one *Security Classification/RA-5* applied to one *Product Item Version/PSCM-2*.

Express Declaration

Product Item Version Security Classification has been incorporated into *Product Item Version* as attribute `security_classifications`.

Entity Name: Organization

Entity Number: RA-7

The identification of a unit of organization (company, project, department, section, and/or person).

Primary Key Attributes

Organization-ID
The unique identification for the *Organization*.

Other Attributes

Company
The name of an association of persons for carrying on a commercial or industrial enterprise.

Project
The name of an undertaking to perform or resolve certain tasks or problems.

Department
The name of a major administrative division within a company.

Section
The name of a major administrative division within a department.

Person
The name of an individual.

Title
The official title of the *Organization*.

Note:

At least one of the above attributes must exist for each instance of *Organization*, although each attribute is optional. If more than one of the attributes are present, they are assumed to be within the same organizational unit, e.g. if 'Company' and 'Department' are present, the 'Department' is assumed to be a division within 'Company'.

Business Rules

Every *Organization/RA-7*:

- grants zero, one, or more *Approval/RA-1*(s).
- is the controlling officer for zero, one, or more *Security Classification/RA-5*(s).

Express Declaration

```
*)
 ENTITY organization;
   person     : OPTIONAL STRING;
   company    : OPTIONAL STRING;
   department : OPTIONAL STRING;
   section    : OPTIONAL STRING;
   project    : OPTIONAL STRING;
   title      : OPTIONAL STRING;
 WHERE
   (person     <> NULL) OR
   (company    <> NULL) OR
   (department <> NULL) OR
   (section    <> NULL) OR
   (project    <> NULL) OR
   (title      <> NULL);
 END_ENTITY;
(*
```

Entity Name: Date Time

Entity Number: RA-8

The date and time in the western calendar and defined by a 24 hour clock.

Primary Key Attributes

Date-Time-ID
The unique identification of the *Date Time*.

Other Attributes

Year
The number representing the year. A negative value for the year is to be interpreted as BC. All positive years are AD.

Month
The number representing the month of the year. This number must have a value of at least one and at most twelve.

Day
The number representing the day of the month. This number must have a value of at least one and at most thirty-one.

Hours
The number representing the hour of the day. This number must have a value of at least zero and less than twenty-four.

Minutes
The number representing the minute of the hour. This number must have a value of at least zero and less than sixty.

Seconds
The number representing the seconds of the minute. This number must have a value of at least zero and less than sixty.

Business_Rules

Every *Date Time/RA-8*:

- is the approval date for zero, one, or many *Approval/RA-1*(s).
- is the classification date for zero, one, or many *Security Classification/RA-5*(s).
- is the declassification date for zero, one, or many *Security Classification/RA-5*(s).

Express Declaration

```
*)
 ENTITY date_time;
   year    : INTEGER;
   month   : INTEGER;
   day     : INTEGER;
   hours   : INTEGER;
   minutes : INTEGER;
   seconds : REAL;
 WHERE
   (1 <= month  <= 12);
   (1 <= day    <= 31);
   (0 <= hours   < 24);
   (0 <= minutes < 60);
   (0 <= seconds < 60);
 END_ENTITY;
(*
```

5.5 IDEF0 Activity Model

The Product Structure Configuration Management model represents a significant amount of information which is recorded during the design planning and processes. The particular viewpoint of these processes as understood by the developers of the model is represented in the following activity model. The model portrays some of the design activities, the information required and produced by those activities, the constraints which guide them, and the mechanisms which make them happen. The model is represented in the IDEF0 notation.

The following list gives the outline of the activity model:

- A-0 Context of Develop and Produce Product
- A0 Develop and Produce Product
  - A1 Manage Product Development
    - A11 Define Tasks
    - A12 Determine Resource Requirements
    - A13 Develop Schedules
    - A14 Develop Budgets
    - A15 Develop Management Conformance Criteria
  - A2 Design Product
    - A21 Develop Conceptual Design
    - A22 Develop Preliminary Design
    - A23 Develop Detail Design
  - A3 Manufacture Product
  - A4 Provide for Product Logistics

This version of the Product Structure Configuration Management model is primarily concerned with activities A2, A21, A22, and A23.

A1 - Manage Product Development
Management of product development is concerned with accomplishing the contracted statement of work by applying the enterprise's resources in a timely manner. During this activity, knowledge of the intended product grows as requirements are analyzed and research is undertaken. The product structure emerges as major assemblies are decomposed.

A2 - Design Product
The product design activity is comprised of three major areas into which the activity can be divided: conceptual, preliminary, and detail design.

A21 - Develop Conceptual Design
The formulation of concepts is based on a combination of customer, market, or mission requirements, basic research data, and exploratory and advanced development programs. Here candidate configurations are chosen for possible preliminary design development.

A22 - Develop Preliminary Design
Preliminary design begins with the definition of several candidate configurations that are all

able to meet the program technical objectives. The actual design is not started, however, until feasible concepts have been defined sufficiently to constrain the scope of subsequent engineering activity. The design problem is approached with advanced analytical methods so that there might be increased confidence in the final selected design.

A23 - Develop Detail Design

During detail design every detail part, assembly, and sub-assembly is defined in its entirety. This activity occurs when the preliminary designs have explored representative design areas to a depth where there are no significant problems, mysteries, or voids remaining. When risk of design change is small and the market prospects are high, management decides to proceed, and the detail design commences.

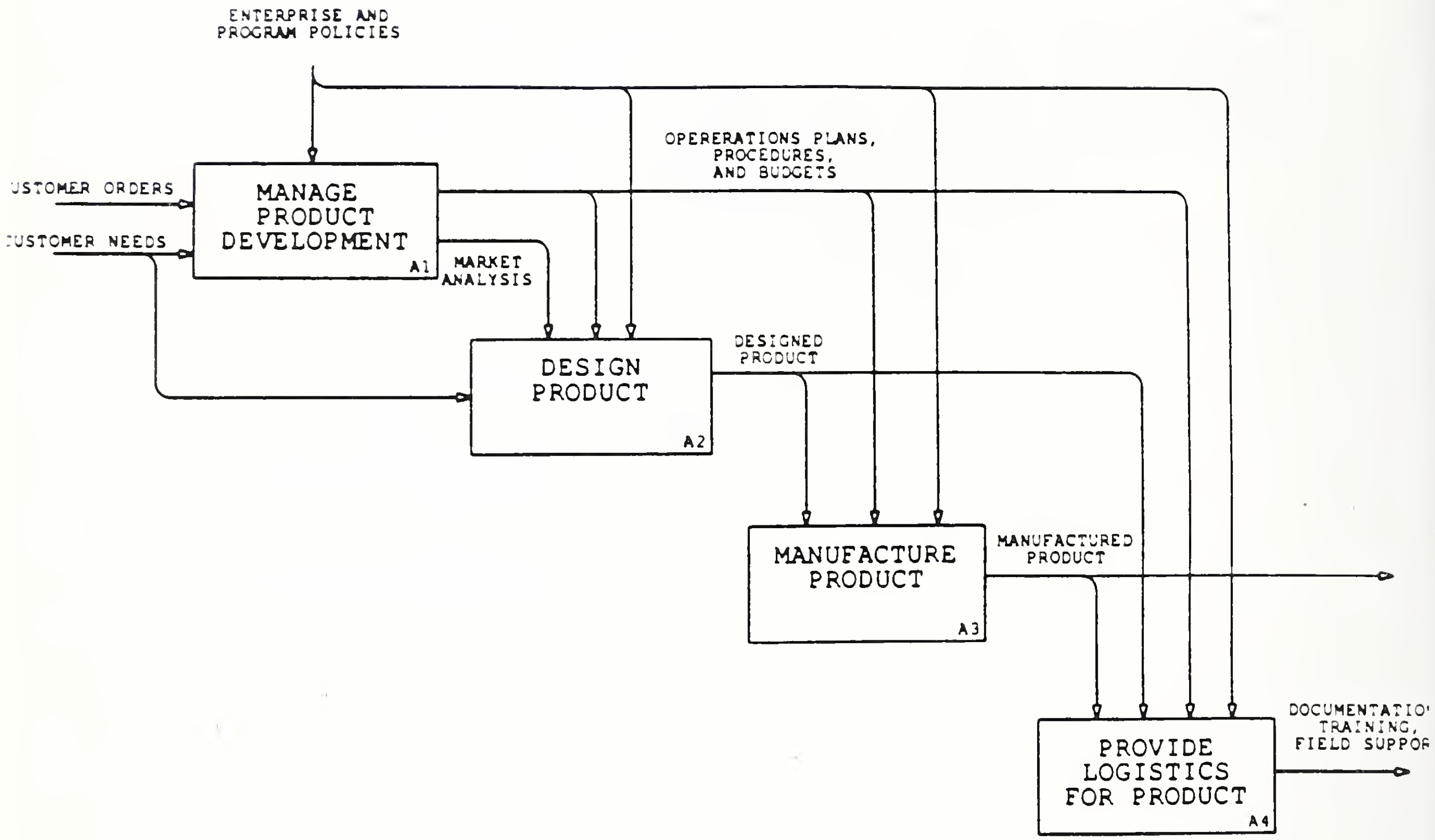
DEVELOP AND PRODUCE PRODUCT DEFINITION DATA A0

Viewpoint: Product Structure Configuration Management

Scope: Limited to alpha-numeric product definition data from the point of conceptual design to release

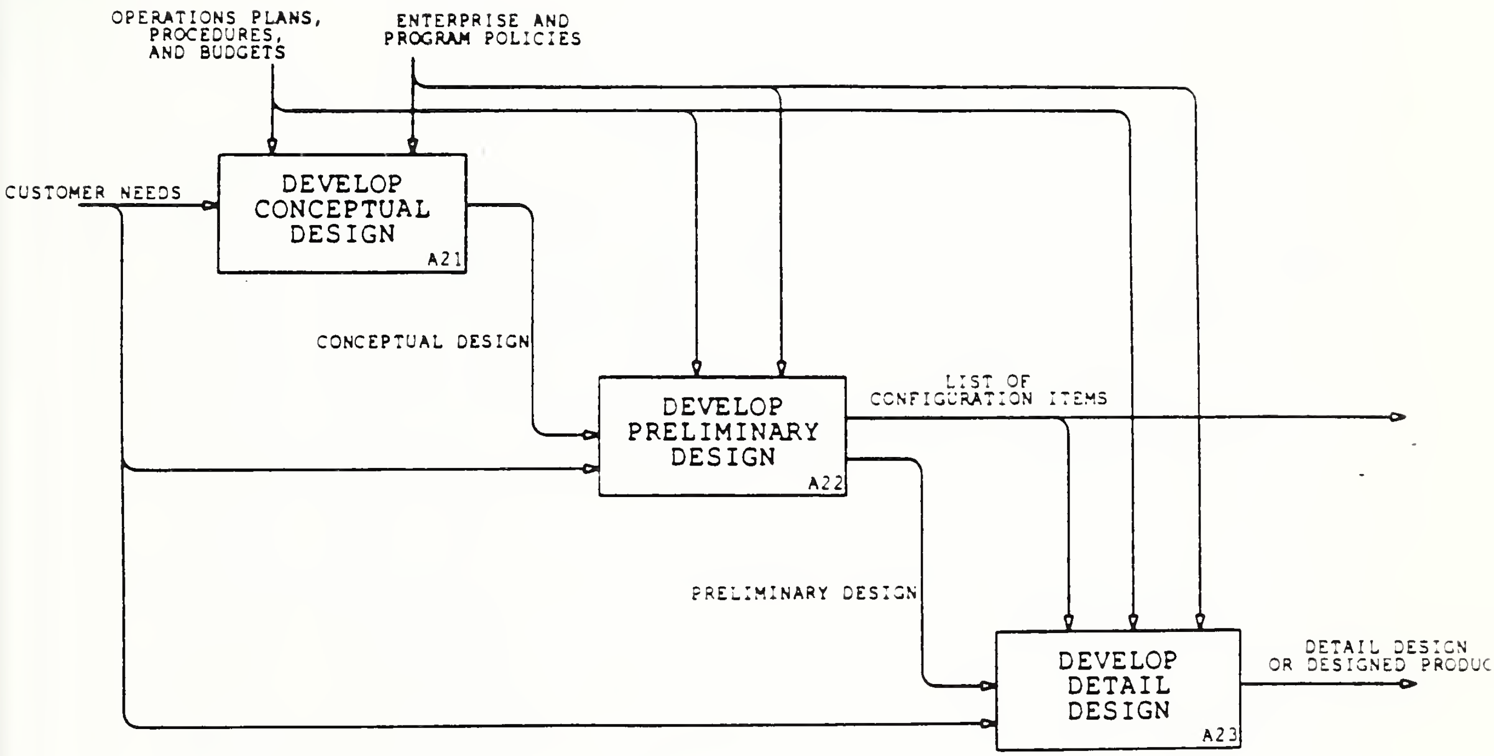
Purpose: To provide a data model which expresses data requirements for Product Structure Configuration Management

CONTEXT OF DEVELOP AND PRODUCE PRODUCT DEFINITION DATA



DEVELOP AND PRODUCE PRODUCT DEFINITION DATA

SECTION 5: PSCM INFORMATION MODEL



DESIGN PRODUCT

SECTION 5: PSCM INFORMATION MODEL

5.6 Issues Log

This Appendix records some of the more important issues which arose during the development of the Product Structure Configuration Management data model.

The initial creation of this model was the result of integrating several representative models into one. As such, many of the earlier issues do not conform to the usual issues found in data models. Many of the issues begin with "How do we model ..." instead of specific instances of problems. In addition, outstanding issues against those models which were represented in the initial effort have not been included.

LOG of OPEN and RESOLVED ISSUES

Issue	Title	Date Described	Date Resolved
PSCM-1	Multiple level relationships	08/21/87	08/21/87
PSCM-2	Explicit Product Item Version	08/21/87	08/21/87
PSCM-3	Component usage within assemblies	08/21/87	08/21/87
PSCM-4	Bar stock vs. Product Item	08/21/87	08/21/87
PSCM-5	Occurrence identification	08/21/87	08/21/87
PSCM-6	Multiple Occurrence context need	08/21/87	08/21/87
PSCM-7	Separate A/C Usage and Occurrence	08/21/87	09/24/87
PSCM-8	Specific/Non-specific Occurrences	08/21/87	09/24/87
PSCM-9	Explicit Make-From entity	08/21/87	08/21/87
PSCM-10	Make-From modeling	08/21/87	09/24/87
PSCM-11	Allowed levels of substitution	08/21/87	09/24/87
PSCM-12	Substitution modeling	08/21/87	09/24/87
PSCM-13	Dependent substitutions	08/21/87	in-work
PSCM-14	Identification of Product	08/21/87	08/21/87
PSCM-15	Multiple bill of material scope	08/21/87	08/21/87
PSCM-16	Multiple BOM view modeling	08/21/87	08/21/87
PSCM-17	Occurrence cross reference	08/21/87	01/13/88
PSCM-18	Effectivity scope	08/21/87	08/21/87
PSCM-19	Configuration Item modeling	08/21/87	09/24/87
PSCM-20	Physical Unit modeling	08/21/87	09/24/87
PSCM-21	Authorized effectivity modeling	08/21/87	09/24/87
PSCM-22	Physical Unit/Version relationship	08/21/87	09/24/87
PSCM-23	Conflicting effectivity	08/21/87	01/13/88
PSCM-24	Release and Approvals	08/21/87	09/24/87
PSCM-25	Security modeling	08/21/87	09/24/87

LOG of OPEN and RESOLVED ISSUES (continued)

Issue	Title	Date Described	Date Resolved
PSCM-26	Inclusion of general notes	08/21/87	08/21/87
PSCM-27	Redundant Effectivity/Model relation	08/21/87	09/24/87
PSCM-28	Design change "result of" cardinality	08/21/87	01/13/88
PSCM-29	Item information identification	08/21/87	not-in-work
PSCM-30	Multiple entities w/change history	10/28/87	01/13/88
PSCM-31	Assembly component usage type/role	10/27/87	in-work
PSCM-32	Intended or planned physical unit	10/28/87	01/13/88
PSCM-33	Range ends for physical units	10/28/87	in-work
PSCM-34	Range ends for physical unit lots	10/28/87	01/13/88
PSCM-35	Redundant keys for A/C Substitution	10/28/87	01/13/88
PSCM-36	Bad Structure/Item intersection	02/21/88	04/27/88
PSCM-37	Assembly use of functional definitions	02/21/88	02/22/88
PSCM-38	Contract numbers required	03/31/88	03/31/88
PSCM-39	Higher assembly use as traversal steps	02/21/88	03/31/88
PSCM-40	Equivalent Product Item dependency	03/11/88	in-work
PSCM-41	Multiple Make From identification	03/11/88	03/29/88
PSCM-42	Physical Unit both planned and built	03/11/88	03/31/88
PSCM-43	Applicable Effectivity not yet captured	03/11/88	in-work
PSCM-44	Explicit Structure identification	01/19/88	in-work
PSCM-45	Shape related to Product Item Version	01/19/88	07/12/88
PSCM-46	Design Change Sequence not normalized	03/31/88	in-work
PSCM-47	Standard part information requirements	01/19/88	not-in-work
PSCM-48	Shape/FEM/Traversal relationship missing	06/15/88	not-in-work
PSCM-49	Product Model subtype of Product Item	06/15/88	in-work
PSCM-50	More enumerations for security levels	07/11/88	in-work
PSCM-51	Design activity codes for organizations	07/11/88	in-work
PSCM-52	Cardinality of security classifications	07/11/88	in-work
PSCM-53	In-Service Bill of Material required	07/05/88	not-in-work
PSCM-54	Explicit constraints for traversal steps	07/05/88	in-work
PSCM-55	Redundancy of open ended ranges	07/05/88	in-work
PSCM-56	Location information for traversal	07/11/88	in-work
PSCM-57	Effective dates required	07/11/88	in-work
PSCM-58	Organizational titles required	07/11/88	07/12/88

ISSUE # PSCM-1 When faced with relationships which can be assigned to multiple levels of identification, what is to be done?

INITIATION DATE: 08/21/87

INITIATOR: Mechanical Product Definition Committee

STATUS: Resolved

RELATED ISSUES: PSCM-10, PSCM-11

DESCRIPTION: In the model there is a hierarchy of identification: *Product Item*, *Product Item Version*, *Product Item Version Structure*, etc.. A relationship such as part substitution could be applied to all levels of identification.

ISSUE OPTIONS & EVIDENCE:

Option 1: Apply the relationship only to the lowest level.

Pro: Again, this is an implementation issue. Within a "conceptual" model, things like substitution have only to be applied to the lowest level of identification in order to capture the necessary information. A given implementation may choose to organize this differently for performance or efficiency reasons.

Con: This will force the users of the model to populate the database with far more instances than would otherwise be necessary. For instance, if substitution is applied at the *Product Item Version* level, but one *Product Item* can be substituted for another for all versions, there will have to be instances of the substitution for all versions of both items.

Option 2: Apply the relationship to all levels.

Pro: This option allows the user to apply the substitution only where it is needed.

Con: Massive redundancy is likely to erupt if this alternative is adopted.

The problem with applying relationships like substitution at higher levels of identification is that an assertion made at one point in time may not be true in lower levels of identification at other points in time. For instance, if a substitution is asserted at the *Product Item* level when there are only three versions of the given item, future versions of that item may not be adequate substitutes for all versions of the other item.

OPTION PROPOSED: Number 1

EXPLANATION: The primary intent of this effort is to create a "conceptual" model for integration purposes. An implementation level model will be produced after the integration effort with other committees.

DECISION:

DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-2	Should there be an explicit *Product Item Version* in the model?
INITIATION DATE:	08/21/87
INITIATOR:	Mechanical Product Definition Committee
STATUS:	Resolved
RELATED ISSUES:	
DESCRIPTION:	There has been pitched debate about what a "version" is and what purpose it serves. Many firms have collections of parts which share certain characteristics. If the similarities are dominant, then instead of re-identifying the part, the new part is simply given a new version number. In many industries, the criterion for a new version/new part number decision is whether or not the new part has the same form, fit and function as the old one. But the criterion for versioning are not universally observed even within a single industry, much less across many firms.

ISSUE OPTIONS & EVIDENCE:

Option 1: No explicit *Product Item Version* in the model.

Pro: To prevent "roll up", a "usage" entity could be used to relate a given assembly to all of the possible alternative versions of each of the components. To create the typical indented parts list, the effectivity needs to be consulted to ascertain which of the alternatives were used on a given *Physical Unit* (which is necessary whether versioning is used or not).

Con: Many corporations have the notion of part versions very strongly entrenched in their thinking. If the Mechanical Products group declines to model versions, users will have a difficult time relating to the model.

Option 2: An explicit *Product Item Version* in the model.

Pro: One purpose of the version is to prevent "roll up" throughout an assembly tree. Without the capability for versions, any change to a part that is tracked results in a new part number. When assemblies call out their components, the complete identification of the component has to be used. If one of the components changes in the slightest, then that component is re-identified, causing a change to the identification of the assemblies using that part. This "roll up" continues throughout the assembly tree until the highest level of assembly is re-identified.

Another purpose of the version is to establish part interchangeability. Again, it is generally the case that any version of a part may be substituted for any version of the same part in all usages. This, however, only applies for those items which are likely to be spared. The reason is that the non-spared items do not have to be replaced in service. Therefore, the "form, fit and function" rule does not have to be followed as rigorously. That, in turn, means that a non-spared part could change significantly and not be completely re-identified. That, in turn, reduces the amount of paper work on the part of the customer, without having to worry about repairs to the product not working because the wrong version of some replacement part was supplied.

Con: The interchangeability of parts is much more complicated than the above argument implies. What the designer considers to be interchangeable, the planner or analyst may not. Part interchangeability should be modeled explicitly.

Since there is not general agreement concerning what constitutes a version and what makes a whole new part, the notion of version should be abandoned. There is no way that any relationships or attributes can be ascribed to *Product Item* since there is no way to be sure that an attribute or relationship is constant across all versions of a given item.

Versioning is generally used for three purposes:

1. Interchangeability (all versions of the same item are assumed to be interchangeable).

2. To denote that one thing is a design change of another.

3. To prevent "roll up", i.e., data reduction.

(1) and (2) are required (and must be modeled) not just between versions of the same item, but between versions of different items as well. Having both a version entity (for versions of the same item) and similar relationships for versions between different items is redundant. Specific relationships that accommodate versions between items can also accommodate versions between the same item, but not vice-versa. (3) is an implementation method, which should not be included in a "conceptual" model.

Some companies do not have any notion of version in their business.

OPTION PROPOSED: Number 2

EXPLANATION: It was resolved to explicitly include a version entity for aid in understanding the model.

Those companies which do not use versioning can map their *Product Item* directly to *Product Item Version*. This should be trivial in that almost nothing is related to *Product Item* directly.

DECISION:

DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-3 How is the usage of components within assemblies modeled?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-5
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1:

```
                   Product Item
                        |      |
                        |      |has
                      is|      o p
              component|  Product Item Version
                      in|      |                  |
                        |      |is assembly of    |
                        o      o                  |
            Product Item A/C Usage               |
                        |                         |
                        o                         o
          Product Item A/C Usage Restriction
```

To handle those cases when it is not acceptable for a given assembly to use one or more versions of its components, an explicit entity which is a cross reference between *Product Item Version* and Product Item A/C Usage establishes a restriction of use of a given version of a component.

The usage entity in this model allows for any version of the components to be used, unless restrictions exist.

Pro: It is rather rare in most aerospace companies that the version of the components is at all an issue in defining the assembly. To have to explicitly state every single version of a component which can ever be used in a given assembly is an enormous amount of data.

Con: While the above may be true, this is an implementation issue. The intent of the present modeling effort is to arrive at a "conceptual" model.

Pro: If it is accurate to say that all versions of the component are valid for use on the assembly, then there should be some means of capitalizing on that.

Option 2: Have a unique instance of Product Item A/C Usage for each version of each component which is valid for use in a given assembly.

```
                 Product Item
                       |
                       |has
                       o p
            Product Item Version
                   |      |
 is component in|      |is assembly of
                   o      o
              Product Item A/C Usage
```

The 'Product Item Version ID' is carried down for both the component and the next assembly into Product Item A/C Usage. This means that there is an explicit relationship between each version of the component which can go into a given version of the assembly. There is no need for another entity to restrict which versions of the component may appear in a given version of the assembly.

Pro: By the very nature of the issue, to adopt a model which allows for restrictions on which versions of a component can be used in a version of the next assembly violates the definition of version. Versioning is not a universal concept. It would seem more "conceptual" to simply not allow versions, re-identify the part completely when a change is made, and link assemblies to the exact components. This obviates the need for component version restrictions.

This alternative preserves us from creating an unnecessary entity.

OPTION PROPOSED: Number 2
EXPLANATION:
DECISION:
DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-4 Is bar stock a *Product Item*?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: No, it is a separate concept and therefore a separate entity.

Pro: Bar stock is not generally produced by the same enterprise as that which creates the final product.

Con: The above is true, but the bar stock has to be made by someone.

Option 2: Yes, it is a *Product Item*.

Pro: Everything that we do with conventional *Product Items* we do to bar stock. It has effectivity so that the lot number of the stock that went into a given *Physical Unit* is noted, there is versioning of the stock, and so forth. Therefore, bar stock should be included as a valid instance of *Product Item*.

OPTION PROPOSED: Number 2
EXPLANATION: The definition of *Product Item* was expanded to include those things consumed in the process of producing *Physical Units*.
DECISION:
DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-5 How are separate occurrences to be identified?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: By location within the next assembly.
Option 2: By location within the "end" item (whatever that means).
Option 3: By location within some intermediate context.
Option 4: By what it attaches to.
Option 5: By where it occurs in the assembly process plan.

OPTION PROPOSED: Number 3
EXPLANATION: For the model, there will be a unique identifier for an instance of a component relative to a context item. The exact means of identification is not so important as the fact that there is some unique label for the occurrence. Because location within the context is critical to mechanical applications, this will be included.
DECISION:
DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-6 Does there need to be more than one occurrence for a single part?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-5, PSCM-39
DESCRIPTION: Assume the following assembly tree:

```
          A
         / \
        /   \
       B     C
      / \   / \
     D   E F   B
              / \
             D   E
```

Is it inherently redundant to have an instance of *Product Item Usage Occurrence* which puts D in the context of B as well as putting D in the context of A? The reader must remember that the location and orientation of the D in each context may be an attribute of *Product Item Usage Occurrence*.

ISSUE OPTIONS & EVIDENCE:

Option 1: No, it is redundant and should not be allowed for.

Pro: It is possible, by having skip level occurrences, to have conflicting locations for the same instance of a given *Product Item*. The "conceptual" approach would be to only allow occurrences of the lowest level (detail) parts only within the context of the highest level of assembly.

Option 2: Yes, it is not redundant.

Pro: If we had perfect computers with infinite arithmetic precision, and they were really fast, then it would be redundant. But it would be almost impossible to derive the location of D in A from the location and orientation of D in B and B in A.
Effectivity will be a cross reference between occurrences and the *Physical Unit*. Configuration items sometimes assemble into configuration items. This means that there have to be multiple instances of *Product Item Usage Occurrence* for the same *Product Item Version*. Each configuration item would be a different context for the occurrence.
Occurrences only between lowest level (detail) parts and highest level assemblies forces all assembly information to only exist within the context of the highest level of assembly. Nearly every lower level assembly (whether it is a configuration item or not) includes occurrence information as well. This information (assembly tolerances, instructions, etc.) is not dependent on the highest level of assembly which it eventually will be used in.

OPTION PROPOSED: Number 2
EXPLANATION: *Product Item Usage Occurrence* will be free to allow multiple instances for the same *Product Item Version*.

DECISION:
DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-7 Do we need to have separate entities for Product Item A/C Usage and for *Product Item Usage Occurrence*?

INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-3, PSCM-39
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: No, use one entity for both.

Con: Since we have already agreed that the context of *Product Item Usage Occurrence* need not be the next assembly (Issue PSCM-6), it would be impossible to generate a typical indented BOM.

Option 2: Yes, keep them separate.

Con: The two entities have the same keys. That would make any normalization attempts combine the two of them together.

Option 3: Make them categories of the same entity:

```
                 Product Item Usage Occurrence
              -┬---------------|----------┬-
    Component Type                   Next Assembly Type
      -┬---|---┬-                 -┬------|-┬--------┬-
  Specific   Non-Specific     Next     Top  Intermediate
```

The specific component is used when there is a single occurrence instance having a location referred to. The Non-Specific instances (typically for lots of two or more) has no location, only a quantity. The next assembly type categorization is used to distinguish the skip level occurrences versus those occurrences whose context is the next assembly. To produce an indented parts list, the Next Assembly occurrences of *Product Item Usage Occurrence* would be consulted.

OPTION PROPOSED: Number 3
EXPLANATION: The following modifications were made to the 'Next Assembly Type':

```
        Assembly Usage Occurrence Type
      -┬----------|----------┬-
  Next Assembly     Higher Assembly
```

The change in categorization of next, top, and intermediate was necessary in that a "next" could also be a "top". The name of the type was changed for consistency.

DECISION:
DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-8 Do we need to have "specific" and "non-specific" categorizations of *Product Item Usage Occurrence*?

INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-7
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: Yes.
Option 2: No.
Pro: A "lot" can have a location if there is some fixed relationship between the members of that lot. Further, even if there is only one of a given *Product Item Version* which is occurring in a given context, there is a quantity of "one" associated with it.

OPTION PROPOSED: Number 2
EXPLANATION: Because there isn't anything (no relationship or attribute) that we wish to say about *Product Item Usage Occurrence* and "non-specific" occurrence, the categorization between specific and non-specific was removed. The relationship to Specific Product Item Occurrence was given a cardinality of zero or one. Quantity was made an attribute of *Product Item Usage Occurrence*. For clarification, Specific Product Item Occurrence was renamed to *Geometric Product Item Usage Occurrence*.
DECISION:
DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-9 Should there be an explicit Make From entity?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION: A "make from" is to establish that one item is created from another by some means, usually a manufacturing process.

ISSUE OPTIONS & EVIDENCE:

Option 1: Yes, there should be an explicit Make From entity.

Pro: It is possible for one item to be made into multiple copies of another. There will be a quantity attribute for "make from" which indicates how many items are output from processing the input item. The attributes are slightly different in that a "make from" has an output quantity, while *Product Item Usage Occurrence* has an input (component) quantity.

An explicit "make from" will make the intent of the model more obvious.

Option 2: No, it can be handled via *Product Item Usage Occurrence*.

Pro: If the number of components of a given assembly is one, then it could be assumed that the relationship is a "make from".

OPTION PROPOSED: Number 1
EXPLANATION:
DECISION:
DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-10 Where does Make From hang from?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-1, PSCM-9
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: *Product Item*
Option 2: *Product Item Version*
Pro: The way in which a product is fabricated should not be dependent upon the assembly in which it is used. However, the input *Product Item Version* can definitely vary across versions of the output *Product Item.*
Option 3: Product Item A/C Usage
Option 4: All of the above
Con: See the resolution of Issue # PSCM-1.

OPTION PROPOSED: Number 2
EXPLANATION: Two deficiencies were noted with the above:

1. It is necessary to establish effectivity for make-from situations.
2. It is sometimes necessary to address specific instances of output items where a make-from results in more than one output item.

In order to avoid duplication, *Make From Usage Option* was made a subtype of *Product Item Usage Occurrence*. The categorization was changed to the following:

```
      Product Item Usage Occurrence
          -T--------|------T-
           |               |
        Make From      A/C Usage Occurrence
      Usage Option    ---------|----------
                       |                 |
                 Next Assembly     Higher Assembly
```

The 'Component Quantity' attribute was moved to *A/C Usage Occurrence*, while *Make From Usage Option* retains the attribute of 'Output Quantity.'

DECISION:
DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-11 To what entity (entities) do substitutions attach to?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-1
DESCRIPTION: This issue was used as an example in Issue # PSCM-1. Many of the arguments raised for this issue are recorded there. The resolution of this issue was highly dependent on the resolution for that issue.

ISSUE OPTIONS & EVIDENCE:

Option 1: *Product Item*, *Product Item Version*, *Product Item Version Structure*, and *Product Item Usage Occurrence*.

Pro: The model will be much more explicit and easy to use. Checks can be installed at the different levels to avoid conflicting statements.

Con: This alternative has the potential for conflicting substitution statements between the different levels of substitution.

Option 2: Only *Product Item Occurrence*.

Con: This allows substitution if the item of interest has an occurrence relative to some context, but doesn't allow for a detail part to be a substitute for another detail part when that part is not used in an assembly (is sold by itself).

Option 3: *Product Item Version* and *Product Item Usage Occurrence*.

OPTION PROPOSED: Number 3
EXPLANATION: The two types of substitution allow for conflicts and redundancy. The definition of *Product Item Version Substitute* was changed to avoid the problem: It does not infer substitution across all usages of the version, but only when the two versions are deemed substitutes independently of any usage context.

DECISION:
DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-12 How are substitutions to be modeled?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1:

```
                 Product Item
                      |
                      o
             Product Item Version
             | |         |  | |
             o o         |  | |
        Product Item     |  o o
      Usage Occurrence   |  Version Substitute
                  |      |
                  o      o
           Occurrence Substitute
```

Con: This option is ambiguous and insufficient for downstream applications to interpret without manual intervention. A substitute occurrence may have a different orientation or quantity than that which it is substituted for. It also opens the door for more redundancy problems, in that any relationship which may be applied to the occurrence will by necessity also have to be applied to the occurrence substitute.

Option 2:

```
                 Product Item
                      |
                      o
              Product Item Version
              | |          | |
              o o          | |
         Product Item      o o
       Usage Occurrence  Version Substitute
              | |
              o o
         Occurrence Substitute
```

Con: When specifying substitutions for occurrences, it is not necessary to create an occurrence instance for the substitute itself. This option would force a great deal of unnecessary effort on the designer.

OPTION PROPOSED: Number 2

EXPLANATION: For *Product Item Version Substitute*, both 'Product Item Version ID's will be role named so that the substitution can be between any two versions of any two *Product Items*.

For Occurrence Substitute, the context assembly identifier will not be role-named within the occurrence substitute, forcing the substitution to be applied only within the same context assembly.

There was discussion concerning one way substitutions versus two way substitutions. A one way substitution indicates that part A can be used in place of part B, but not necessarily vice versa. The model implies that the instance of Occurrence Substitute will be one way since the identifiers for the substitute and that which it substitutes for are role named differently. If two way substitution is desired, there must be two instances of Occurrence Substitute, one for each direction of the substitution.

Make From Usage Option was made a type of occurrence (see related issues). This entity infers its own method of substitution. Occurrence Substitute is intended only for assembly/component situations. It was moved to relate to the occurrence sub-type *Assembly Component Usage Occurrence*, and renamed to *Assembly Component Usage Occurrence Substitute*.

DECISION:

DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-13 The model does not adequately deal with substitutions which depend on one another.
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: In-work
RELATED ISSUES: PSCM-12
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: Have entities which indicate an order to the substitutions.

```
        Occurrence Substitution
          |    |      |    |
          o    o      o    o
      Substitution  Substitution
       Dependence     Exclusion
```

The "Substitution Dependence" would indicate that if a given substitution is made, another has to have occurred before that substitution. The "Substitution Exclusion" instance establishes that once a given substitution is made, it is not authorized to make some other specific substitution.

OPTION PROPOSED:
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:
This issue was only glanced at in passing. Although important, the committee has decided to defer the issue until sufficient time is allocated to examine it in more detail.

ISSUE # PSCM-14 How is the "Product" to be identified?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION: By "product" we mean something that is marketed. This is distinct from any physical manifestations of the product which are delivered to the customer.

ISSUE OPTIONS & EVIDENCE:

Option 1: Put all of the levels of identification of the product into one entity called *Product Model.*

Pro: By merely assigning the product a unique identifier, companies can map there own hierarchy of identification to it.

Con: This makes the model more difficult to read since it is not as specific.

Option 2: Call out specific levels of identification of the product.

Con: Not all corporations identify their products in the same fashion. One may have a break-down of product, model, model series, model series unit, and another may use a totally different breakdown.

Option 3: Make the product a type of *Product Item* and use the *Product Item Usage Occurrence* to handle any necessary indenture in the product.

Con: At some companies, most, if not all, of the products sold do not have part numbers. There is a great deal of information which is kept about parts and assemblies which is not trapped with the "end" item.

OPTION PROPOSED: Number 1
EXPLANATION:
DECISION:
DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-15 Will this model support multiple bill of material (BOM) views?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION: Imagine that there is a bucket of parts which have to be properly assembled to create a final product. A person is sat down in front of the parts and told to assemble them. The person has nothing to aid them but a "blow up" of the final product. This person is systematic and creates a set of lowest level assemblies. Then these are assembled into higher assemblies, and so forth until finally he has the finished product with all of the parts consumed. The assembly hierarchy which that person used is one BOM view. Another person could come to the same bucket of parts and assemble them in a totally different sequence, and still arrive at the same final product. The assembly tree that they created in the process would be another BOM view.

ISSUE OPTIONS & EVIDENCE:

Option 1: No.
Pro: One of the members asserted that they are investigating having no explicit assembly tree. They may be able to assemble their products by establishing assembly procedures which explicitly cite the components which are to be put together. There is no need to have explicit assembly structures except within these procedures. After all, a BOM view is merely an alternative grouping of parts.
Con: Not everyone can do their job this way. Further, it is often the case that important information is tied to the usage of a part in a given assembly.

Option 2: Yes.
Pro: In several of the participant's shops, the assemblies, in several BOM views, are released. The BOM view is therefore necessary.
Con: The only BOM views that most of the participants were familiar with were the "as designed" and the "as planned". Since it had been resolved that the MP PSCM scope did not include manufacturing, BOM view was clearly out of scope.
Pro: There are multiple BOM views even within Engineering in some shops. In addition, successful resolution of this issue now will save a great deal of work later.

OPTION PROPOSED: Number 2
EXPLANATION: There will be multiple BOM views allowed in the model.
DECISION:
DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-16 How is the BOM view to be modeled?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-15
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1:

```
 Product Item Version
         |    |
         o    o
   Item Occurrence      BOM View
         |                 |
         o                 o
 Item Occurrence In BOM View
```

Option 2:

```
          Product Item
                |
                o
     Product Item Version
            |      |
            o      o
 Product Item Usage Occurrence
```

In this approach, the BOM view, since it has no attributes and no keys which are inherited, gets sucked into the *Product Item Version* entity. The BOM view then just appears as an attribute. A different BOM view results in a different version.

Con: This model implies that the *Product Item Version* has to be re-identified for each BOM view that it participates in. That, in turn, implies that the data dependent upon the *Product Item Version* has to be duplicated, when it does not really depend upon which BOM view it is a part of.

Option 3:

```
     BOM View     Product Item Version
         |           |
         o           o
     Product Item Version BOM
         |           |
         o           o
 Product Item Usage Occurrence
```

In this model, the identifier of the "BOM view" is not role named in the *Product Item Usage Occurrence*. This insures that the context and the component come from the same "BOM view".

Pro: This approach has the appeal that the *Product Item Version* can be used in multiple "BOM views", but the usage still reflects the notion of the same parts going together in a multitude of assembly trees.

Con: One of the participants asserted that they had no use for "BOM view" at all and for them this alternative was onerous because it forced them to define a "BOM view" instance before they could establish an instance of *Product Item Usage Occurrence* which was necessary before they could carry any effectivity.

Pro: It was pointed out that there need only be one BOM view defined for the entire enterprise for this model to function. If the enterprise attached no importance to the notion of "BOM view", they were free to virtually ignore it.

Option 4:

```
            BOM View       Product Item
                |              |
                o              o
        Product Item Version IN BOM View
                |              |
                o              o
         Product Item Usage Occurrence
```

Con: In this scenario, each time a *Product Item* appears in a different "BOM view", all of the information which is dependent upon the *Product Item Version* in BOM view has to be replicated. It was established that such important data as geometry did not depend upon which "BOM view" the *Product Item Version* was appearing in.

OPTION PROPOSED: Number 3
EXPLANATION: It was also resolved that BOM View would be called *Structure*.
DECISION:
DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-17 Given that a single manifestation of a *Product Item Version* can be in multiple *Structures*, does there need to be an explicit cross reference between occurrences?

INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-16, PSCM-39
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: No.
Option 2: Yes:

```
Product Item Version       BOM View
               |                 |
               o                 o
     Product Item Version Structure
               |                 |
               o                 o
       Product Item Usage Occurrence
               |                 |
               o                 o
 Structure Occurrence Cross Reference
```

Con: This is redundant information since it should be obvious from the location that the two occurrences are the same.

Pro: The same logic applies here that does to having the same occurrence appearing in multiple contexts within the same *Structure*. There is no means of physically implementing the computations necessary without some round off errors calling into question the fact that two occurrences are in reality the same. Occurrences are also not required to have locations (they may be defined for functional reasons). There is no other way to establish this fact.

OPTION PROPOSED: Number 2
EXPLANATION: The arguments against an *Occurrence Cross Reference* are basically implementation methods for avoiding it. Conceptually, *Occurrence Cross References* do exist.

DECISION:
DECISION DATE: 01/13/88

ACTION:

ISSUE # PSCM-18 What sort of effectivity should be addressed?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: Shelf life effectivity. This is a statement that a given physical instance of *Product Item* is authorized for use in assembly during a given time span. This prevents articles which have exceeded their shelf life from being consumed in the production of some *Physical Unit*.

Con: This is outside of the scope of this model.

Option 2: Authorized effectivity. This is a statement that a *Product Item Version* (or occurrences thereof) can be used for a given *Physical Unit*. The authorized effectivity does not state what actually went out the door, but only what was authorized to go out the door.

Pro: Authorized effectivity is the only one which is clearly within our scope.

Con: Let's just tackle one at a time.

Option 3: Applicable effectivity. This is a statement of what actually was used in the construction of an instance of *Physical Unit*.

Pro: This is by far the most important to the enterprise.

Con: Applicable effectivity is too engrossed in the manufacturing arena.

OPTION PROPOSED: Number 2
EXPLANATION: Other types of effectivity will be included with future expansions to the model.
DECISION:
DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-19 How is *Configuration Item* to be modeled?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-27
DESCRIPTION: A *Configuration Item* is a *Product Item* which is managed separately. It may be sold as part of other *Configuration Items* or by itself. *Physical Units* are made according to the specifications of a given *Configuration Item*. One *Configuration Item* can have many *Physical Units* which correspond to that *Configuration Item* but are not exact copies of one another. For instance, on the MD80, there are a variety of options for the passenger seats. But regardless of which option is selected, it is still an MD80. A *Product Model*, on the other hand, is something with a set of capabilities and requirements which is sold to a customer.

ISSUE OPTIONS & EVIDENCE:

Option 1: A configuration item is just a special case of *Product Item* or maybe *Product Item Version*.

Option 2:

```
Product Model     Product Item
      |                |
      o p              o z
   Configuration Item
```

Option 3:

```
        Product Model
              .
              o
   Configuration Item      Product Item
              .                  |
              o z                o z
       Product Configuration Item
```

Pro: Not all *Configuration Items* are *Product Items*.

Option 4: Do not model a configuration item.

Pro: Some companies do not have the notion of *Configuration Items* at all. Effectivity is applied to the *Product Model* instead.

OPTION PROPOSED: Number 3
EXPLANATION: See the resolution of the Issue # PSCM-27.
DECISION:
DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-20 How is *Physical Unit* to be modeled?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-27
DESCRIPTION: An instance of *Physical Unit* documents an individual customer deliverable. That individual customer deliverable may be a lot of two or more, but with the restriction that there is no means of distinguishing one member of the lot from another.

ISSUE OPTIONS & EVIDENCE:

Option 1:

```
Configuration Item
        |
        | manages
        o
  Physical Unit
```

Note: The fact that the cardinality from *Configuration Item* to *Physical Unit* is 0, 1, or many is important. The design process may lead to a product which is totally described, but never sold.

It was recognized by the group that the *Physical Unit* is a build of a *Configuration Item*. Or, in other words, the customer purchases a product which is completely described by the data related to an instance of a *Configuration Item*. But what gets delivered is something described by an instance of *Physical Unit*. The reason that they are not one and the same is that *Physical Units* may be sold to different customers for different prices, delivered at different dates, manufactured at different facilities, planned differently, etc. Another dimension of the difference between the *Physical Unit* and the *Configuration Item* is that the authorized options are documented relative to the *Configuration Item*, but what actually went into the item delivered to the customer is documented relative to the *Physical Unit*.

OPTION PROPOSED: Number 1
EXPLANATION: See the resolution of the Issue # PSCM-27.
DECISION:
DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-21 How is authorized effectivity to be modeled?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-27
DESCRIPTION: Authorized effectivity is an expression of what is allowed to go into a given *Physical Unit*. This is very much like the substitution issue where the relationship can be tied to a variety of levels of identification of the part.

ISSUE OPTIONS & EVIDENCE:

Option 1: As a cross reference entity between *Physical Unit* and *Product Item Version*.
Option 2: As a cross reference entity between a *Physical Unit* and *Product Item Usage Occurrence*.
Con: Some companies do not express authorized effectivity at the occurrence level. This option causes many more effectivity instances in the data base.
Pro: The above argument is an implementation issue.
Option 3: Both of the above.
Con: This causes redundancy and allows possible conflicting effectivity.

OPTION PROPOSED: Number 2
EXPLANATION: See the resolution of the Issue # PSCM-27.
DECISION:
DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-22 What is the relationship between *Physical Unit* and *Product Item Version*?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-27
DESCRIPTION: The *Product Item Version* represents the design side of the house, while the *Physical Unit* is the physical side. If there is a design for the *Physical Unit*, then there must be a corresponding instance of *Product Item Version* which is the design of that *Physical Unit*.

ISSUE OPTIONS & EVIDENCE:

Option 1: *Physical Unit* is just another category of *Product Item Version*.

Pro: When a design is actually built, there is no tremendous difference between the design and the build unit as far as the information is concerned. One can talk about versions of the *Physical Unit* that would result as the product is maintained in service. There are physical occurrences within the *Physical Unit*.

Option 2: There is a many to many relationship between *Physical Unit* and *Product Item Version*.

Pro: The 'Product Item ID' and 'Product Item Version ID' migrating down into the *Occurrence Effectivity* from *Physical Unit* can be used to make sure that the context for the instances of *Product Item Usage Occurrence* which are used for establishing effectivity is the same as the instance of *Product Item Version* corresponding to the *Physical Unit*.

Con: The occurrences mentioned in the previous argument do not have to always be in the context of the "end" item just so they can be used to express effectivity.

Option 3: For every *Physical Unit* of a given configuration item there is one and only one *Product Item Version* which represents the design of that *Physical Unit*.

Pro: A given *Physical Unit* has only one design. There is no reason to have more than one design which leads to the same *Physical Unit*.

OPTION PROPOSED: Number 3
EXPLANATION: This option provides the most flexibility. See the resolution of the Issue # PSCM-27.

DECISION:
DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-23 If *Configuration Items* assemble into *Configuration Items*, how is conflicting effectivity to be avoided?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-6
DESCRIPTION: In some companies, effectivity is expressed in a hierarchy; at some point in the assembly tree, the authorized *Product Item Version* for all of the nodes below that in the tree is expressed. Then, for all of the nodes below that, their effectivity, if expressed at all, must be a subset of the effectivity which was expressed at the higher level. If no effectivity is expressed at a given node, then it inherits whatever effectivity that its parent had. In this manner, one can manage to keep a *Configuration Item* which assembles into another from expressing an effectivity which conflicts with the effectivity of the *Configuration Item* which it assembles into. This issue was addressed within the McDonnell Douglas Product Definition Data corporate project and it was resolved that the "higher" configuration item could express effectivity in terms of lower configuration items and detail parts, but not in terms of the detail parts making up the lower level configuration items.

ISSUE OPTIONS & EVIDENCE:

Option 1: Allow effectivity to be specified for lowest level *Product Item Versions* to highest level *Product Item Versions* for a *Configuration Item*.

Pro: This would disallow conflicting effectivity to be specified for sub-assemblies and detail items within higher assemblies for the *Configuration Item*.

Con: For many companies, effectivity for sub-assemblies within higher assemblies is used most often, with the detail items rarely of concern.

Option 2: Leave it as it is.

Pro: The methods used to avoid conflicting effectivity are very company specific. Depending on the organization, effectivity may be applied between major component assemblies, assemblies and sub-assemblies, assemblies and details, or all of these.

OPTION PROPOSED: Number 2
EXPLANATION: We cannot restrict the entity-relationship model directly to avoid conflicting effectivity. Rules can be established, however, to avoid this (similar to *Assembly Component Usage Occurrence*, where we apply rules to avoid recursion within an assembly tree). The documentation will be modified to describe these rules.
DECISION:
DECISION DATE: 01/13/88

ACTION:

ISSUE # PSCM-24 Should the model include Release or Approvals?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-29, PSCM-30
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: Yes.

Pro: Release is an essential part of the design process. Therefore, even given the limited scope of this model, there should be an attempt to handle the information associated with a design release.

Approvals exist for just about every aspect of the information associated with the *Product Item*. If a new version is identified every time a change occurs to a *Product Item*, then the approval could be hung from the version. What exactly was being approved would have to be inferred from the cross reference from that particular version. Any instance of any entity that contained that version's identifier would be assumed to have its classification.

Con: The approach described above would not handle the problem of tracking who is needed for a formal approval of a given type of release.

Option 2: No.

Option 3: Just model a very limited scope of approvals: the drawing release.

Pro: To model all of the possible approvals would clutter the model to the point where it would no longer be understandable.

Con: Since drawing release is not the entire story, it may miss important information. Better to acknowledge that we do not know how to model approvals.

OPTION PROPOSED: Number 3
EXPLANATION: For consistency with the approach taken for security, it was resolved to include a very limited portion for released versions of *Product Items*. This has flaws: See Issue # PSCM-29.

DECISION:
DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-25 How is security to be modeled?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES: PSCM-29, PSCM-30
DESCRIPTION: Much like approvals, security can become very pervasive throughout the model.

ISSUE OPTIONS & EVIDENCE:

Option 1: Nothing at all.
Pro: If we are going to be realistic, there should be a security reference model, just like there should be one for approvals. Many companies have no need for security classifications.
Option 2: Try to handle all of the various types of security
Pro: Security is an essential part of the product description and is necessary for indicating who the intended audience is for the information.
Option 3: For now, model just a simple form of security:

```
             Product Item
                  |
                  o z
     Classified Product Item
```

Classified Product Item has the attribute 'Level Of Security'. Some *Product Item Versions* may not have any security classification; if so, there is no instance of Classified Product Item for that *Product Item*.
Pro: At least this would give a flavor of what security would involve and handle the most pressing needs.

OPTION PROPOSED: Number 3
EXPLANATION: The classification was moved to relate to *Product Item Version* instead of *Product Item*. This was done because one criteria for security is the design responsibility, which is normally associated at the version level.
DECISION:
DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-26 Should "General Notes" be included in the model?
INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: No.
Pro: Notes are places where information hides in most models which allow for general notes. The blob of text which comprises the note is not machine interpretable. We should try to find out what information is going into the note and try to model that explicitly.

Option 2: Yes.
Pro: Notes will be with us for a long time. We might as well allow for the PDES exchange mechanism to handle them as they are, and try to break notes down to their constituent parts later.

OPTION PROPOSED: Number 1
EXPLANATION: It was resolved to not explicitly model notes
DECISION:
DECISION DATE: 08/21/87

ACTION:

ISSUE # PSCM-27	The effectivity portion of the model has redundancies and incorrect relationships.
INITIATION DATE:	08/21/87
INITIATOR:	Mechanical Product Definition Committee
STATUS:	Resolved
RELATED ISSUES:	PSCM-19, PSCM-20, PSCM-21, PSCM-22
DESCRIPTION:	Two shortcomings exist with the model for effectivity:

1. Some redundant relationships occur between effectivity and physical unit.

2. The model restricts each *Product Item* to be related to zero or one Product Configuration Item, which in turn is restricted (indirectly) to be related to only one *Product Model*. In many companies, one *Product Item* may be used for several different *Product Models*.

ISSUE OPTIONS & EVIDENCE:

Option 1: Leave as is:

```
          Product Model     Product Item
               .               |      |
               o               |      o p
     Configuration Item        | Product Item Version
      |            .           |             .   |  |
      |            o z         o z           .   |  |
      | Product Configuration Item           .   |  |
      |                                      .   |  |
      |       ........................           |  |
      o       o                                  o  o
     Physical Unit    Product Item Usage Occurrence
              |              |
              o p            o
         Occurrence Effectivity
```

Option 2:

```
        Product Model        Product Item
              .                    |
              o                    o p
    Configuration Item    Product Item Version
         .     |-----------|       |    |
         .     |                   |    |
         o     o                   o    o
    Physical Unit    Product Item Usage Occurrence
               |             |
               o p           o
         Occurrence Effectivity
```

Pro: This option eliminates the redundancy of the first, while also allowing for one *Product Item* to be used in several *Product Models*.

OPTION PROPOSED: Number 2

EXPLANATION: The previous restriction of allowing each *Product Item* to be associated to only one *Product Model* is a company specific way of doing business, not universally accepted.

DECISION:

DECISION DATE: 09/24/87

ACTION:

ISSUE # PSCM-28 Within *Design Change Sequence*, is the cardinality for the 'Is result of' relationship really one only, instead of zero or one?

INITIATION DATE: 08/21/87
INITIATOR: Mechanical Product Definition Committee
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION:

ISSUE OPTIONS & EVIDENCE:

Option 1: Leave as is.

Option 2: Make it one only; the preceding 'Product Item ID' and preceding 'Product Item Version ID' become attributes of *Product Item Version*, and *Design Change Sequence* is removed.

Pro: Every item conceptually has something which it is designed from.

Con: An initial design may have no other design from which it is derived.

Option 3: Leave the entity-relationship model as it is, but modify the documentation to state that if the *Product Item Version* is an initial design, the "result of" cardinality may be zero.

OPTION PROPOSED: Number 3

EXPLANATION: The documentation will be modified to state that if the *Product Item Version* is an initial design, the "result of" cardinality may be zero. This may have been handled via sub-typing of *Product Item Version*, although the definition of what is an initial design is not clear. The first version of a *Product Item* is not always an initial design, as it may have come about due to a change in another *Product Item*'s "form, fit, and function." An error was also noted in the model diagram for the "is basis in" relationship. This will be modified to correctly show it as an independent relationship.

DECISION:
DECISION DATE: 01/13/88

ACTION:

ISSUE # PSCM-29 How do we identify the information within the model which is part of the design, as opposed to that just associated to the design?

INITIATION DATE: 08/21/87

INITIATOR: Mechanical Product Definition Committee

STATUS: Not-in-work

RELATED ISSUES: PSCM-24, PSCM-25

DESCRIPTION: The *Product Item Version* within the model aggregates information which is a description for a "design". Many descriptions for many versions may exist simultaneously within the model. However, there is no means within the model to determine what information is contained *within* the actual design of an item as opposed to information which is *related* to the design. For instance, most people would agree that component usage is part of the design for an assembly, but is "where used" or effectivity information part of the design? They may have been designated by the designer as being so, or may have been attached later within the product's life cycle.

This problem manifests itself most clearly when trying to model approvals or securities. An approval usually applies to some subset of information which is known about the design. Delineating the bounds of this subset is not possible within the current model.

ISSUE OPTIONS & EVIDENCE:

OPTION PROPOSED:
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-30	The model does not capture changes in *Approvals* and *Security Classifications* over time.
INITIATION DATE:	10/28/87
INITIATOR:	Roger Gale
STATUS:	Resolved
RELATED ISSUES:	PSCM-25, PSCM-29
DESCRIPTION:	The PSCM Model does not allow for the multiple instances of entities required to record the changes in the business which happen over time. Businesses maintain their history as changes occur. The PDES model should provide for a complete product description including the changes over time found in the business record. An example of this is with the entity *Classified Product Item Version*. Defense Department regulations require that the classification of items change over time. Downgrading is required to take place.

ISSUE OPTIONS & EVIDENCE:

Option 1: The key of *Classified Product Item Version* should include the date of the classification assignment and the cardinality of its relationship as a child should be zero-one-or-many rather than zero-or-one. Similar changes should be made for *Released Product Item Version*.

Con: This is a temporary solution at best (see issue PSCM-29). It would be better to leave the model as it is until a solution to information subsets is found.

OPTION PROPOSED:	Number 1
EXPLANATION:	The solution to information subset identification may elude us for quite some time. This is a "temporary" fix until that solution is found.
DECISION:	
DECISION DATE:	01/13/88

ACTION:

ISSUE # PSCM-31 Categorizations should be modeled as roles.
INITIATION DATE: 10/27/87
INITIATOR: Gail D. Vermilyea
STATUS: In-Work
RELATED ISSUES:
DESCRIPTION: A/C-Usage-Occurrence-Type was created to show a generalization for assembly component. It is suggested that this is not a type of assembly component, but roles that an assembly component can play. The modeling language should be used to show a semantically correct representation.

ISSUE OPTIONS & EVIDENCE:

Option 1: Model as roles.
Pro: The modeling technology should be used consistently throughout PDES.

OPTION PROPOSED:
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:
Some confusion exists within the committee on this issue. Further explanation from the author is being solicited.

ISSUE # PSCM-32 The name of *Planned Physical Unit* should be "Intended Physical Unit" to clarify the meaning of the entity.
INITIATION DATE: 10/28/87
INITIATOR: Thurber Moffett
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION: *Planned Physical Unit* should be Intended Physical Unit. The word "planned" implies a process planning type function. Correct wording should be "intended."

ISSUE OPTIONS & EVIDENCE:

Option 1: Change *Planned Physical Unit* to Intended Physical Unit.
Pro: "Planned" implies a process planning type of function. This entity is meant to describe an "intended" physical unit (e.g., the basis for/result of a decision by a configuration control board.) This option limits incorrect inferences, and more clearly denotes the intent of the entity.
Option 2: Leave the model as it is.
Pro: "Intended" and "Planned" are synonymous. A better approach would be to leave the name as it is, but to insure that the definition of this entity clearly states our intent.

OPTION PROPOSED: Number 2
EXPLANATION: The entity name will be left as it is. The definition will be modified to clearly state that this is an "intended" unit of product.
DECISION:
DECISION DATE: 01/13/88

ACTION:

ISSUE # PSCM-33 Need to be able to capture range ends for physical units
INITIATION DATE: 10/28/87
INITIATOR: PDES Integration Workshop
STATUS: In-Work
RELATED ISSUES:
DESCRIPTION: There is no means for capturing the end of ranges for physical units in the existing model.

ISSUE OPTIONS & EVIDENCE:

Option 1: Each *Physical Unit Range Start* could have an optional range end, as follows:

```
Discrete Physical Unit
      |         |
      |         o z
      |    Range Start
      |       |
      o       o
    Range End
```

Con: This implies that a given *Discrete Physical Unit* may be the start of only one range. Many ranges may start with the same *Discrete Physical Unit*, e.g. 5 through 10, 5 through 15, etc, with each range establishing effectivity of different lower-level components. If ranges are to be uniquely identified, an additional primary key for the 'Range ID' will be required. Effectivity will also be required on ranges.

Option 2: Leave the relationships as they are, but change the name of *Physical Unit Range Start* to *Physical Unit Open Ended Range*.

Pro: The original intent of the model was to be able to identify a *Physical Unit* as being the start of an open ended range, i.e. a range for which no upper bound is known. Conceptually, if one knows the end of the range, one could populate *Physical Unit* with instances for each unit within the range and then establish the correct relationships to each instance.

Con: The name change is good. However, the current model is incorrect in that a *Discrete Physical Unit* may be used as both a single unit and as the start of an open range. For example, effectivity may be established for serial number 5 by itself, as well as effectivity for serial numbers 5 and on.

OPTION PROPOSED:
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

The committee decided that the name of *Physical Unit Range Start* will be changed to *Physical Unit Open Ended Range*. The primary issue, however, remains unresolved. Work on this issue is being pursued by the committee.

ISSUE ≠ PSCM-34 Need ranges for *Physical Unit Lots*.
INITIATION DATE: 10/28/87
INITIATOR: T. Voegeli
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION: The current model allows only for the *Discrete Physical Units* to be the start of a range. The model should allow for the same capability for *Physical Unit Lots* as well.

ISSUE OPTIONS & EVIDENCE:

Option 1: Move *Physical Unit Open Ended Range* to relate directly to *Physical Unit* instead of *Discrete Physical Unit*.
Pro: This will allow any type of physical unit to be a *Physical Unit Open Ended Range*, thereby allowing lots, as well as discrete units, to be designated as the beginning of a range.
Option 2: Leave the model as it is.
Pro: Open-ended ranges for lots make little sense except when talking about time-based effectivity, which is beyond the scope of the current model. Lots, although they may be given specific lot identifiers, are normally controlled by time ranges.

OPTION PROPOSED: Number 2
EXPLANATION: The model will be left as is. The members of the committee present when this decision was made were not well acquainted with production in terms of lots. As such, this solution may require modification in the future.
DECISION:
DECISION DATE: 01/13/88

ACTION:

ISSUE # PSCM-35 The foreign key 'Occurrence ID' in entity *Assembly Component Usage Occurrence Substitute* is unnecessarily duplicated.

INITIATION DATE: 10/28/87

INITIATOR: Yuhwei Yang

STATUS: Resolved

RELATED ISSUES:

DESCRIPTION: 'Occurrence ID' is part of the primary key in entity *Product Item Version Occurrence*, which identifies a unique instance of the usage of a component *Product Item Version* in its context *Product Item Version*. This key is migrated into the entity *Assembly Component Usage Occurrence Substitute*. Since this entity captures the information about the substitution of the component for a specific occurrence in the context, the 'Occurrence ID' in both the reference component and the substitute component are identical; thus there is no need for the duplication of this migrated foreign key.

ISSUE OPTIONS & EVIDENCE:

Option 1: Leave the model as it is.

Option 2: Eliminate one of the 'Occurrence ID's in entity *Assembly Component Usage Occurrence Substitute*.

Pro: Duplication is unnecessary.

Con: The 'Occurrence ID' within *Product Item Version Occurrence* uniquely identifies one instance of usage of a *Product Item Version* within a context. This key is not unique across all usages of all components within the context, only among all usages of a specific component. The substitute *Product Item Version* may have more than one usage within the context, with some usages being substitutes and others being regular component usage. Therefore, both 'Occurrence ID's are necessary, one to identify the particular usage of the component for which the substitution can apply, and the other to identify the particular usage of the component which can be substituted.

OPTION PROPOSED: Number 1

EXPLANATION: As described above, both 'Occurrence ID's are necessary. The documentation for the 'Occurrence ID' is somewhat ambiguous and will be clarified.

DECISION:

DECISION DATE: 01/13/88

ACTION:

ISSUE # PSCM-36 *Product Item Version Structure* should be renamed to *Product Item Version Functional Definition*. The definition and attributes for *Structure* are incorrect.
INITIATION DATE: 02/21/88
INITIATOR: Second PDES Integration Workshop
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION: The name of entity *Product Item Version Structure* does not capture the intent of the entity. A better name would be *Product Item Version Functional Definition*. Also, the definition of *Structure* seems to apply to the intersection entity between it and *Product Item Version*.

ISSUE OPTIONS & EVIDENCE:

Option 1: Change the name of *Product Item Version Structure* to *Product Item Version Functional Definition*. Also, move the definition and attributes of *Structure* into the entity to which they apply, namely *Product Item Version Functional Definition*. *Structure* can therefore become a "shadow" entity.

Con: Although the name change is valid, transferring the attributes (i.e. 'Structure Owner', 'Structure Name', etc.) will violate normalization. These attributes are intended to capture the information about the *Structure* which is independent of the *Product Item Versions* to which the *Structure* is related.

Option 2: Change the name as in Option 1. Retain the attributes in *Structure*, but add similar attributes into *Product Item Version Functional Definition*. For example, 'Definition Owner' and 'Definition Name' are the owner and name of the *Functional Definition* of the *Product Item Version*.

Option 3: Same as Option 2, but eliminate the entity *Structure* altogether.

Pro: The definition of *Structure* is unclear. The original modelers intended this to capture the information which is constant for a given bill of materials, such as an internal "standard" definition for an engineering department's bill of material. Although there may be information about such a "standard" definition which is independent of the *Product Item Versions* to which it is related, the information more correctly falls into the category of "requirements", e.g. minimal design content, required approvals and drawing formats, etc. The relationship between *Structure* and *Product Item Version Functional Definition* is then incorrectly modeled, in that a *Functional Definition* may conform to many such *Structures*. This implies a many-to-many relationship.
Information about *Structure* and the entity which is the resolution of the many-to-many relationship will be extremely difficult to instantiate. Because we do not intend to address either entity in detail, we could make both "shadowed". However, this violates the IDEF1x modeling conventions. Instead, let's remove *Structure* until such time as we can address requirements in detail.

OPTION PROPOSED: Number 3

SECTION 5: PSCM INFORMATION MODEL

EXPLANATION: *Product Item Version Structure* will be renamed to *Product Item Version Functional Definition*. New attributes of 'Definition Name', 'Definition Owner', 'Definition Description', and 'Definition Creation Date' will be added to it. *Structure* will be removed from the model.

DECISION:

DECISION DATE: 04/27/88

ACTION:

ISSUE ≠ PSCM-37	The model should allow for the use of a functional definition of a component within a different functional definition of an assembly.
INITIATION DATE:	02/21/88
INITIATOR:	Second PDES Integration Workshop
STATUS:	Resolved
RELATED ISSUES:	
DESCRIPTION:	The model should allow for the use of a functional definition of a component within a different functional definition of an assembly. For instance, a manufacturing functional definition of an assembly should be able to use the engineering definitions of the components.

ISSUE OPTIONS & EVIDENCE:

Option 1: Remove the restriction that the 'Definition ID' within *Product Item Usage Occurrence* be the same for the context and the component. Migrate in the 'Definition ID' of both the context and component with separate role names to allow the 'Definition ID's to be different.

Pro: This greatly expands the functionality of the model in that it allows for functional definitions of an assembly to call out various functional definitions of the components that they contain. For instance, both a "process planning" and an "assembly planning" definition for the same assembly could specify that the "as designed" definitions of the components are used, although the relationships between the components within each definition of the assembly could vary.

OPTION PROPOSED:	Number 1
EXPLANATION:	The restriction that the 'Definition ID's of the context and component be the same will be removed. Separate role names will be provided for both the context and component 'Definition ID's.
DECISION:	
DECISION DATE:	02/22/88

ACTION:

ISSUE # PSCM-38 Some mechanism for capturing the related Contract number for a *Product Item Version* is required.
INITIATION DATE: 03/31/88
INITIATOR: Elise DeCarlo
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION: The current model has no mechanism for capturing the related Contract number for a *Product Item Version*. This capability is required.

ISSUE OPTIONS & EVIDENCE:

Option 1: Establish a relationship between *Product Item Version* and *Contract* as follows:

```
        Product Item Version       Contract
                  |                    |
                  o                    o
              Product Item Version Contract
```

This allows for a *Contract* to be associated with zero or more *Product Item Versions*, and a *Product Item Version* to be associated with more than on *Contract*.

Con: This is a very simplistic view of *Contract*. Much more detail is needed, as well as associations with *Approvals* for the *Contract* and *Product Item Version Contract*.

OPTION PROPOSED: Number 1
EXPLANATION: *Contract* will be added as in Option 1. Until such time as we can address contracts in more detail, *Contract* will be a "shadow" entity, with 'Contract Number' as the primary key.

DECISION:
DECISION DATE: 03/31/88

ACTION:

ISSUE ≠ PSCM-39 *Higher Assembly Usage Occurrence* should be modeled as a unique assembly tree traversal.

INITIATION DATE: 02/21/88

INITIATOR: Second PDES Integration Workshop

STATUS: Resolved

RELATED ISSUES:

DESCRIPTION: The current model for *Higher Assembly Usage Traversal* allows for much ambiguity and does not identify an occurrence within one tree as the same as that within another. A better approach would be to model it as the unique collection of individual steps (*Next Assembly Usage Occurrences*) between the context and component.

ISSUE OPTIONS & EVIDENCE:

Option 1: Rework the model as follows:

```
Product Item Version Functional Definition
       .    |                    .
      is.   |is context of       ...............
component.  o                                  .
      of. Product Item Usage Occurrence        .is input
        .          -T--------|---------------T- .for
        .           |                        |  o
        .     Assembly Component    Make From Usage Option
        .     Usage Occurrence
        .  -T---------|---------T-
        o   |                   |
   Next Assembly        Higher Assembly
  Usage Occurrence      Usage Occurrence
            |                   |
            o                   o 2:n
     Higher Assembly Usage Occurrence Steps
```

Con: The traversal idea is a good concept, however this model implies independence of the 'component' and 'input' from the context for *Next Assembly Usage Occurrence* and *Make From Usage Option*.

Option2: Use the traversal concept, but model it as follows:

```
Product Item Version Functional Definition
                |          |
is context of   |          |is component of
                o          o
     Product Item Usage Occurrence
              -T--------|---------------T-
               |                        |
     Next Assembly           Product Item Usage Traversal
    Usage Occurrence          ----------|-----------
               |               |                  |
               |      Higher Assembly      Make From Usage
               |     Usage Traversal           Option
               |               :
               o               o 2:n
       Higher Assembly Usage Occurrence Steps
```

Pro: This limits the definition of occurrences to be just *Next Assembly Usage Occurrences*. This is appealing in that it separates this concept from *Make From Usage Option* and *Higher Assembly Usage Traversal*, which are both somewhat different concepts.

Con: This will necessitate considerable re-work on both effectivity and substitutions within the model. Separate substitution entities will be required for *Next Assembly Usage Occurrence* and *Higher Assembly Usage Traversal*, and possibly a third which lies between the two (is it possible to have a substitute which lies at a lower level in the assembly tree than that which it is substituting for?). Effectivity as well will have to be expanded.

Option 3: Use the traversal concept, but impose it upon the current model structure as agreed upon at earlier meetings:

```
      Product Item Version Functional Definition
                     |         |
     is context of   |         |is component of
                     o         o
           Product Item Usage Occurrence
                            |
                 -T---------|--------------T-
                  |                        |
       Assembly Component        Make From Usage
         Usage Occurrence             Option
                    |
      -T-----------|-------------T-
       |                         |
    Next Assembly          Higher Assembly
  Usage Occurrence         Usage Occurrence
           |                     |
           o                     o 2:n
        Higher Assembly Usage Occurrence Steps
```

Pro: This model captures the desired functionality of traversals, as well as keeping intact the surrounding model and business rules. It has added appeal over Option 1 in that both the context and component ID's are available within *Higher Assembly Usage Occurrence*, which facilitates establishing a location relationship between them.

Con: Location references for *Higher Assembly Usage Occurrences* are unnecessary, in that they can be derived from the locations of the steps.

Pro: Location references may still be required. See the arguments in Issue PSCM-6.

Option 4:

```
         Product Item Version Functional Definition
                          |       |
                          o       o
          Product Item Version Definition Usage Occurrence
              |       | |          | |-------------
              |       o o z        o               |
              |     Usage        Usage      Occurrence Type
              o  Substitution   Traversal   -T------|------T-
           Usage      |            |     |   |             |
         Effectivity  o            o z   | Make From     Next
              |     Usage Traversal      |  Usage      Assembly
              |  Substitute Restriction  |              Usage
              o z                        o
            Traversal Effectivity Restriction
```

Con: "Restrictions" should not be in a "conceptual" model. Their purpose here is to state that an entity sometimes can not serve the role for which it is defined. For example, a Usage Substitute states that two components can always be used as substitutes within the next higher assembly, but Usage Traversal Substitute Restriction states that in some cases they cannot be. This is an implementation method for data reduction.

OPTION PROPOSED: Number 3

EXPLANATION: The model will be revised as per Option 3 above. The following name changes were adopted to more clearly denote the intent of the entities:

1. *Product Item Usage Occurrence* will become *Product Item Usage Traversal.*

2. *Assembly Component Usage Occurrence* will become *Assembly Component Usage Traversal.*

3. *Higher Assembly Usage Occurrence* will become *Higher Assembly Usage Traversal.*

4. Higher Assembly Usage Occurrence Steps will be *Higher Assembly Usage Traversal Step.*

5. *Occurrence Effectivity* will become *Traversal Effectivity.*

6. *Assembly Component Usage Occurrence Substitute* will become *Assembly Component Usage Traversal Substitute.*

7. em Geometric Product Item Usage Occurrence will become *Geometric Product Item Usage Traversal.*

Higher Assembly Usage Traversal will be allowed to carry a location reference between the context and component. It was recognized that this may result in conflicting location information, but because some designs locate components relative to higher level assemblies than their immediate parent, it is required.

DECISION:

DECISION DATE: 03/31/88

ACTION:

ISSUE # PSCM-40 *Equivalent Product Item Version* should be dependent upon *Configuration Item* to reflect product requirements.
INITIATION DATE: 03/11/88
INITIATOR: Anne Williams
STATUS: In-Work
RELATED ISSUES:
DESCRIPTION: A given *Product Item Version* may be planned or produced as more than one *Configuration Item*. The *Equivalent Product Item Version* is a substitute independent of usage, i.e., "..such as two 'end' items where one may be sold in place of the other." Therefore, should the *Equivalent Product Item Version* be dependent upon the *Configuration Item* so as to reflect product requirements?

ISSUE OPTIONS & EVIDENCE:

Option 1: Leave the model as it is.
Pro: This model allows equivalence to be captured for nearly any reason, not just because one may be "sold" in place of another.
Option 2: Relate *Equivalent Product Item Version* to the *Configuration Item* rather than the *Product Item Version*.
Option 3: Remove the *Equivalent Product Item Version* entity altogether. Capture equivalence with the *Configuration Item* entity.

OPTION PROPOSED: Number 3
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

The definition of *Equivalent Product Item Version* does imply equivalence between *Configuration Items*, not between *Product Item Versions*. It was decided to remove the entity altogether, and capture equivalence as part of the *Configuration Item* entity. Within the current model, equivalence can be inferred if two separate *Product Item Versions* are used for *Planned Physical Units* of the same *Product Model*. This is insufficient. Full resolution of this issue is being pursued by the committee.

ISSUE # PSCM-41 How does the model identify multiple outputs from the same input for *Make From Usage Option*?

INITIATION DATE: 03/11/88

INITIATOR: Anne Williams

STATUS: Resolved

RELATED ISSUES:

DESCRIPTION: Suppose x is the "make from" and one instance of x may be made into both y_1 and y_2. Suppose further that the occurrence of both y_1 and y_2 is of interest, so that the *Product Item Usage Occurrence* requires two instances. Is the 'Output Quantity' one in each instance of the associated *Make From Usage Option*? Where is the fact that x is made into two y captured? (Which is important or else the 'Output Quantity' is inappropriate as well.) What is the common element in the key? The 'input component' role named attributes are insufficient to trace the commonality. Suppose further that x is the "made from" and one instance of x may be made into z_1, z_2, and z_3, but a single instance of x cannot produce all of the y_i and z_j. The value of the key of x is not changed, since that is independent of its usage. Thus only the 'Structure ID' can vary to capture this information. If this is the case, then the Structure entity needs further explanation.

ISSUE OPTIONS & EVIDENCE:

Option 1: Leave the model as is.

Pro: When designing a part, it is not unusual for the design to specify the parts from which it is allowed to be made, i.e. the various inputs allowed for a given output. This was the original intent of the model. This issue is concerned with a somewhat polar viewpoint of identifying the possible combinations of things into which the part can be made, i.e. the outputs relative to the input. The latter is seldom identified during the design and pre-release timeframe of the scope of this model.

Con: One reason for the 'Occurrence ID' was to be able to establish unique relationships, such as location, between the output and input items. The model as it currently stands cannot support this uniqueness.

Option 2: Allow the available options to be collected into groups:

```
Make From Usage Option       Make From Group
                 |                        |
                 o                        o p
          Make From Group Usage
```

In this model, each Make From Group can contain one or more individual *Make From Usage Options*, while each *Make From Usage Option* can be used in zero or more Make From Groups. Make From Group could be a "shadow" entity.

Con: There is very little, if anything, we wish to say about Make From Group by itself. Even as a "shadow" entity, it would just clutter the model. Having Make From Group stand by itself, i.e. an independent entity, is not very appealing semantically.

Pro: The Make From Groups may have selection ranking and rationale established for them.

Con: Establishing a selection ranking and/or rationale for Make From Groups may cause redundancy and conflicts with those established for individual *Make From Usage Options*.

Option 3: Same as the above, but remove Make From Group:

```
Make From Usage Option
          |
          o
Make From Usage Option Group
```

Make From Usage Option Group has a unique identifier for the group, as well as the identifiers of the *Make From Usage Options* which it contains.

Pro: This solves the cluttering aspect of Option 1.

Con: This hides the fact that what we really have is a many-to- many relationship between the group and the members. Moreover, it will be impossible later to add attributes to the group without violating normalization.

OPTION PROPOSED: Number 3

EXPLANATION: *Make From Usage Option Group* will be added to the model as described in Option 3. This provides for the unique identification and quantity of the instances of the items that a given input can be made into. Further work in this area may be required in the future.

DECISION:

DECISION DATE: 03/29/88

ACTION:

ISSUE # PSCM-42 A given *Physical Unit* can be both "Built" and "Planned".
INITIATION DATE: 03/11/88
INITIATOR: Anne Williams
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION: In the model, *Physical Unit* is categorized as *Built Physical Unit* and *Planned Physical Unit*. Thus a given *Physical Unit* cannot be both "Planned" and "Built". Consequently, once a given *Physical Unit* is built, the "as planned" information, the authorized effectivity, captured in *Occurrence Effectivity*, cannot be retained.

ISSUE OPTIONS & EVIDENCE:

Option 1: Change the key of *Physical Unit* to include the discriminator 'Physical Unit Status Type'.

Option 2: Remove the categorization:

```
Configuration   Product Item
    Item           Version
     |               |
     o               o
    Planned Physical Unit    Product Item
     |               |      Usage Traversal
may be|              |            |
     o z             o            o
 Built Physical   Occurrence Effectivity
      Unit
```

This allows for a *Planned Physical Unit* to also be a *Built Physical Unit*. As details of *Built Physical Unit* are beyond our current scope, it will be a "shadow" entity.

Option 3: Allow a *Physical Unit* to be either:

```
  Physical Unit
   |        |
   o z      o z
 Planned   Built
Physical  Physical
  Unit      Unit
```

Con: This diagram does not capture the constraint that a *Physical Unit* has to be either *Planned* or *Built* or both. Also, it makes little sense to have a *Built Physical Unit* that is not also a *Planned Physical Unit*.

Option 4:

```
            Physical Unit
                  |
                  o p
        Applied Physical Units
          -T-----|-----T-
           |           |
         Built      Planned
       Physical    Physical
         Unit        Unit
```

Con: The arguments against Option 3 still apply here.

OPTION PROPOSED: Number 2

EXPLANATION: This option will allow a physical unit to be both "Built" and "Planned". *Built Physical Unit* will be a "shadow" entity.

DECISION:

DECISION DATE: 03/31/88

ACTION:

ISSUE # PSCM-43 Applicable effectivity, what is actually delivered to the customer, is not yet captured in the model.

INITIATION DATE: 03/11/88

INITIATOR: Anne Williams

STATUS: In-Work

RELATED ISSUES:

DESCRIPTION: Authorized effectivity appears to be against the *Physical Unit* through the *Occurrence Effectivity*, rather than against the *Configuration Item*. Applicable effectivity would be a relationship from *Built Physical Unit* to *Occurrence Effectivity*; the applicable effectivity would not entail an associative entity between *Built Physical Unit* and *Occurrence Effectivity*.

ISSUE OPTIONS & EVIDENCE:

OPTION PROPOSED:

EXPLANATION:

DECISION:

DECISION DATE:

ACTION:

ISSUE # PSCM-44 Identify and define the known intended Structures explicitly rather than defining the form in a text string value for the 'Description' attribute.

INITIATION DATE: 01/19/88

INITIATOR: Roger Gale

STATUS: In-Work

RELATED ISSUES: PSCM-15, PSCM-16, PSCM-36

DESCRIPTION: The entity Structure in the present PSCM model has a non-key attribute 'Description' that appears to be where the definition of the meaning of any particular instance of the entity is found. The figure below illustrates the basic relationships of discussion in that model.

```
                         |
                         o p
Structures     Product Item Version
    |                    |
    |is the              |belongs
    |organization        |to
    |of                  |
    o                    o p
Product Item Version Structure
```

One of the principles of IDEF1X as a semantic model is the meanings of entities are best found in the definitions of the entities and attributes in the model glossary. One test that can be used to determine if a model has been developed to that degree is to ask if a programmer could write a program in a language such as FORTRAN to work on a file complying with the data model if all that is known about the data is the documentation of the model. Using that test, it is unlikely that the programmer would know from the present PDES data model what to expect to receive.

It appears that the modelers of the PSCM have in mind that there is more than one structure (organization of *Product Item Versions*) in the product description but have left it up to the creator of a product description to define each instance of Structure by writing a definition in the form of a text string value for the 'Description' attribute.

I believe that some of the intended structures are known and can be identified and defined. There has been considerable discussion identifying "as designed", "as planned", and "as built" product structures as sets of related *Product Item Versions*.

ISSUE OPTIONS & EVIDENCE:

Option 1: Rename the *Structure* entity "Product Item Structure" and establish it as a generalization with three known sub types in an incomplete set of sub types. As additional structures are identified and defined they may be added to the model.

```
                                     Product Item
                                          |
                                          |
                                         o p
Product Item Structure               Product Item Version
      |                 |                 |
      |               is|                 |is
      |      composed of|                 |assigned to
      |                 o p        o
      |          Product Item Structure Member
      |
Product Structure Type
     -T--------|---T------------T-
      |            |            |
As-Designed  As-Planned  As-Built
Product      Product     Product
Structure    Structure   Structure
```

In this figure, the cardinality of the relationship between the *Product Item Version* and its intersection entity with the Product Item Structure has been changed from one, or many (P) to zero, one, or many. This change would permit knowledge and definition of a *Product Item Version* in an enterprise without its being a member of at least one Product Item Structure. An example of this would be the designation of a *Product Item Version* as preferred for new design. This "status" of a purchased part does not seem to represent a "structure" idea. It is common for manufacturing enterprises to establish purchased parts as preferred parts even when they have not yet been incorporated into a Product Item Structure.

OPTION PROPOSED:
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-45 A *Product Item Version* can have only one shape and size definition. A change in any characteristic of a *Product Item* must result in a new *Product Item Version*.

INITIATION DATE: 01/19/88

INITIATOR: Roger Gale

STATUS: Resolved

RELATED ISSUES:

DESCRIPTION: During the first Integration Workshop, the attendees placed the entity that intersects the PSCM model with the Shape- Size-Element-Group model so that its relationship is the intersection entity that links the *Product Item* with *Structure*. This relationship results in a business rule that declares that the shape and size definition of a *Product Item Version* is dependent on its membership in a particular structure. This also implies that the shape and size of a *Product Item Version* varies from structure to structure. The figure below shows that portion of the model as affirmed in the second integration workshop.

```
                                         |
                                         o p
                 Structures     Product Item Version
                  |                      |
                  |is the                |belongs
                  |organization          |to
                  |of                    |
                  o                      o p
      Shape Size      Product Item Version
     Element Group          Structure
               |               |
             is|             is|
               o               o z
       Product Item Version Shape Size
```

I believe that a *Product Item Version* can have only one shape and size definition. I believe that it is a rule of the manufacturing business that a change in any characteristic of a *Product Item* must result in a new *Product Item Version*. Under these rules, the shape and size is related directly to an instance of *Product Item Version* and not to its membership in a structure.

ISSUE OPTIONS & EVIDENCE:

Option 1: Move the relationship between the SSEG and PSCM models to appear as shown in the figure below.

```
                |establishes            |
                |identification         |
                |for                    |
                o                       o p
          Shape Size        Product Item Version
        Element Group                 |       |
                   |                  |       |
                 is|                  |has    |
                   o                 oz       |
                 Product Item Version         |
                        Shape Size            |
                                              |
                  Structures                  |belongs
                        |is the               |to
                        |organization         |
                        |of                   |
                        o                     o p
               Product Item Version Structure
```

Con: The 'Shape Size Element Group' referred to above is an aggregation of the information describing the shape and size. This aggregation includes not just the goemetric entitites, but also the relationships between them. This aggregation can vary between *Structures*. The most obvious example is in the way the geometric entities are related by location (an "as-designed" *Structure* may relate components to the immediate assembly, which may not be the same immediate assembly as that in a "as-produced" *Structure*). Location, tolerances, and other information may vary.

The goemetric entities themselves may also vary, as witness the situation where the "as-designed" *Structure* specifies numerous optional or substitute components which are allowed. The geometric representation of the assembly can only be an "exact" description if the selection of any substitute or optional component does not alter the geometry in the slightest. In reality, the geometry for such an assembly is usually only an approximation. Other *Structures* may have more precise geometry depending on the intent of their use (a robotic assembly *Structure*, for example, cannot usually deal with even slight ambiguities in geometry).

Option 2: Leave the model as it is.

OPTION PROPOSED: 2
EXPLANATION:
DECISION:
DECISION DATE: 07/12/88

ACTION:

ISSUE # PSCM-46 The entity *Design Change Sequence* contains unnormalized data
INITIATION DATE: 03/31/88
INITIATOR: David Brown
STATUS: In-Work
RELATED ISSUES:
DESCRIPTION: The attribute 'Design Change Reason' is unnormalized in that one reason may apply to many design changes.

ISSUE OPTIONS & EVIDENCE:

Option 1: Pull the reason out into a Change Request entity, and model as follows:

```
         Product Item Version
            |      .
            o z    o
        Design Change Sequence    Change Request
                  |                     |
                  o                     o
            Design Change Sequence Request
```

Con: This model implies that a Change Request can only apply to *Product Item Versions* which are a design change of others. The Change Request may result in the creation of new *Product Item Versions*, which may not be changes to existing *Product Item Versions* (new components may be created to satisfy a change to an assembly).

Option 2: Relate the Change Request directly to *Product Item Version*.

```
              Product Item Version
                 |      .    |
                 o z    o    |
        Design Change Sequence |   Change Request
                             |      |
                             o      o
                     Product Item Version Change Request
```

OPTION PROPOSED:
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-47 Standard part information is lacking from the current model.
INITIATION DATE: 01/19/88
INITIATOR: Roger Gale
STATUS: Not-In-Work
RELATED ISSUES:
DESCRIPTION: Standard part information was originally deemed beyond the scope of the first version of this model. This should be added.

ISSUE OPTIONS & EVIDENCE:

Option 1: Standard items could be modeled as follows:

```
                            Enterprise
           ------------------|   |
          |is the source of      |
          |identification for    |
          o                      |
     Product Item                |
       has|                      |establishes
          o p                    |
     Product Item Version        |
                  is|            |
                    o p          o
                Enterprise Standard Item
              --------|             |   |is superseded
            is|                   is|   |by
              o z                   o   o
      Inactivated Enterprise      Superseding Enterprise
         Standard Item                 Standard Item
```

Enterprise will be a "shadow" entity.

OPTION PROPOSED: 1
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-48 *SHAPE/INT-1* and *FEM/FEM-1* should have a relationship with *Product Item Usage Traversal.*

INITIATION DATE: 06/15/88

INITIATOR: Mike Yinger

STATUS: Not-In-Work

RELATED ISSUES:

DESCRIPTION: *SHAPE/INT-1* and *FEM/FEM-1* should have an integration relationship with *Product Item Usage Traversal.* As it now stands there are many FEM and Shape states that cannot be articulated in the model's present state.

ISSUE OPTIONS & EVIDENCE:

OPTION PROPOSED:

EXPLANATION:

DECISION:

DECISION DATE:

ACTION:

This is a recognized deficiency with the model. The current scope of the model states that deformable conditions and degrees of freedom are not addressed. This issue will be addressed with future model development.

ISSUE # PSCM-49 *Product Model* and *Configuration Item* should be subtypes of *Product Item*.

INITIATION DATE: 06/15/88

INITIATOR: Mike Yinger

STATUS: In-Work

RELATED ISSUES:

DESCRIPTION: *Product Model* and *Configuration Item* are *Product Items* and should not be independent entities. Furthermore, a *Planned Physical Unit* should have a many-to-many relationship with the concept of a *configured product item*.

ISSUE OPTIONS & EVIDENCE:

OPTION PROPOSED:
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-50 Additional enumerations for the allowed security levels are required.
INITIATION DATE: 07/11/88
INITIATOR: Greg Paul
STATUS: In-Work
RELATED ISSUES:
DESCRIPTION: Five additional security classifications should be added to the enumeration of security levels. For the U.S. Department of Defense:

```
Confidential Special Access Requirement
Secret Special Access Requirement
Top_Secret Special Access Requirement
```

and for industry:

```
Proprietary
Competition Sensitive
```

ISSUE OPTIONS & EVIDENCE:

Option 1: Add the above enumerations to the allowed list for attribute 'Level of Security' within entity *Security Classification*.

Con: The current enumerations, plus the first three of those suggested above, are those used by the U.S. Department of Defense and it's contractors. There are myriads of other enumerations; pages worth for the U.S. Department of Energy and other departments, non-U.S. organizations, and company internal security classifications. Instead of targeting the enumerations at one specific enterprise (DoD), the model should be more general in nature.

Option 2: Replace the current enumeration with a STRING, in order to allow any enterprise to use the model.

Con: This option removes the possibility of making the level of security machine interpretable.

Option 3: Sub-type the security levels into the various enterprise-specific enumerations, and have one "catch all" type with a STRING field for everything else:

```
                Security Classification
                          |
          |---------------|-------------|
          |                             |
     DoD Security               Non-DoD Security
    Classification               Classification
+----------------------+  +--------------------------+
|Security-Class-ID (FK)|  |Security-Class-ID (FK)    |
+----------------------+  +--------------------------+
|DoD-Level-of-Security |  |Non-DoD-Level-of-Security |
+----------------------+  +--------------------------+
```

'DoD Level of Security' is an enumeration of the current values allowed within the model and the first three suggested with this issue. 'Non-DoD Level of Security' is a text field (string) to capture everything else.

OPTION PROPOSED: 2
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-51 The model should include a 'design activity code' for organizational units.
INITIATION DATE: 07/11/88
INITIATOR: Greg Paul
STATUS: In-Work
RELATED ISSUES:
DESCRIPTION: The entity **person_or_project** should have an additional attribute of **design activity_code** (an optional STRING). This would correspond to the Federal Supply Code for Manufacturers (FSCM) or Contractor and Government Entities (CAGE) codes.

ISSUE OPTIONS & EVIDENCE:

Option 1: Add the attribute 'Design Activity Code' to the entity **person_or_project**.
Con: The FSCM or CAGE codes are not always assigned to unique organizations. Some situations arise where the buyer will specify that different codes be used for different items produced from the same organizational unit.
Option 2: Add the attribute 'Design Activity Code' as an optional attribute of *Product Item*.
Con: The design activity code may vary between versions of the *Product Item*.
Option 3: Add the attribute 'Design Activity Code' as an optional attribute of *Product Item Version*.

OPTION PROPOSED: 3
EXPLANATION: Within the Express, the attribute 'Design Activity Code' will be added as an optional attribute to *Product Item Version*. Within the IDEF1x model, because of normalization requirements, it will appear as a separate entity titled *Product Item Version Design Activity Code*, with a relationship from *Product Item Version* to it of cardinality zero or one.

DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-52 The cardinality between *Product Item* and security_classes should be zero or one, not zero, one, or many.
INITIATION DATE: 07/11/88
INITIATOR: Greg Paul
STATUS: In-Work
RELATED ISSUES: PSCM-50
DESCRIPTION: In *Product Item*, there can't be [1:#] security_classifications because a document always takes on the highest level of security classification of any information contained within it.

ISSUE OPTIONS & EVIDENCE:

Option 1: Modify the attribute cardinality for security_classifications to be zero or one.
Con: The security classification can vary over time, thereby resulting in the current cardinality of zero, one, or many.
Option 2: Retain the cardinality as it exists within the model, but force the constraint that no two security classifications can exist at the same point in time (no two classification dates can be equal).
Con: The resolution of issue PSCM-50 implies that not just U.S. Department of Defense security levels will be captured. As so, many security classifications may exist at one point in time; one for DoD, others for different enterprise or company specific levels.

Option 3: Retain the cardinality as it exists, add the constraint that no two DoD-related security classifications can have the same classification dates, but don't place this restriction on other types of security classifications.

OPTION PROPOSED:
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-53 The In-Service Bill-Of-Material needs to be captured

INITIATION DATE: 07/05/88

INITIATOR: David Brown

STATUS: Not-In-Work

RELATED ISSUES:

DESCRIPTION: The model should capture the in-service Bill-of- Material. It is often critical to know what version of a given *Product Item* has been most recently installed in a given occurrence within a given end_item_unit.

ISSUE OPTIONS & EVIDENCE:

OPTION PROPOSED:

EXPLANATION:

DECISION:

DECISION DATE:

ACTION:

This issue is recognized as being beyond the current scope of this model. It will be addressed with future model development

ISSUE # PSCM-54 Explicit constraints should be shown for *Higher Assembly Usage Traversal Step*.

INITIATION DATE: 07/05/88
INITIATOR: David Brown
STATUS: In-Work
RELATED ISSUES:
DESCRIPTION: The constraints on *Higher Assembly Usage Traversal Step* should be explicitly shown in the IDEF1x model, not just in the Express.

ISSUE OPTIONS & EVIDENCE:

OPTION PROPOSED:
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-55 Open-Ended Ranges are redundant.
INITIATION DATE: 07/05/88
INITIATOR: David Brown
STATUS: In-Work
RELATED ISSUES:
DESCRIPTION: *Physical Unit Open Ended Range* appears to be redundant with the information in *Traversal Effectivity*. If the range is kept in the model, should it be dependent on *Planned Physical Unit* or some sort of model, since a range really needs a start and finish?

ISSUE OPTIONS & EVIDENCE:

OPTION PROPOSED:
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-56 Location information in the model does not satisfy the needs for multiple representations.
INITIATION DATE: 07/11/88
INITIATOR: Harry Ladd
STATUS: In-Work
RELATED ISSUES:
DESCRIPTION: Within the PSCM model, the *Product Item Version Functional Definition* (PIVFD) is shown as having no more than one primary *Shape*, and occurrences are restricted to have no more than one location. Within the Integration Core model, the primary *Shape* can have multiple representations.
Which geometric representation of the assembly and component does the location within the PSCM model refer to?

ISSUE OPTIONS & EVIDENCE:

Option 1: Revise the PSCM model to state that an occurrence may have more than one location, but each location must refer to a unique geometric model used as the representation of a *Shape Element* contained within the primary *Shape* of the PIVFD of the component (but not necessarily a different geometric model used as the representation of a *Shape Element* contained within the primary *Shape* of the PIVFD of the assembly).

Con: Besides being a nasty constraint to write up in Express, this detracts from the intent of the model, i.e. that in the "real" world, the *Shape* of the instance of the component have no more than one position relative to the *Shape* of the assembly for each occurrence. How this position is represented in the geometric modeling world is of lesser importance.

Option 2: Leave the model as it is. The occurrence can only have one location which applies to one representation of the assembly and one representation of the component.

Con: The information that another representation is intended to be of the same occurrence may not be derivable due to the differences in representation.

Option 3: Request that the notion of instances of component *Shapes* be installed into the *Shape* world. The PSCM model would retain the notion of one location, but instead of referencing the geometric representations, it would reference the appropriate entity in the *Shape* world. The actual location information (matrix, etc.) could still exist only within the geometric representations. Occurrences could then be identified independently of the representations of the assembly or component.

OPTION PROPOSED: 3
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-57 Effective dates of use are required.
INITIATION DATE: 07/11/88
INITIATOR: Mike Whiteman
STATUS: In-Work
RELATED ISSUES:
DESCRIPTION: Within some industries, the engineering/design arena may specify the authorized date for which manufacturing is allowed to start using a certain *Product Item Version*. This may occur concurrently with the "engineering release", or may be granted as a separate action afterwards, when all of the legal paperwork is wrapped up. It may also occur at several instances throughout time (Part A is first effective, then Part B, then, due to problems with B, Part A is effective again.

ISSUE OPTIONS & EVIDENCE:

Option 1: Add the effective date as an optional attribute of *Product Item Version*.
Con: Some overlap exists between this and the effectivity already in the model. Our definition of "release" within the Approvals model seems to include this type of effectivity with it. Also, effective dates are often dependent on a targeted manufacturing facility, and can vary between facilities which produce the same *Product Item*.

OPTION PROPOSED: 1
EXPLANATION:
DECISION:
DECISION DATE:

ACTION:

ISSUE # PSCM-58 The 'Title' for a person or organization is required
INITIATION DATE: 07/11/88
INITIATOR: E. Dean Bray
STATUS: Resolved
RELATED ISSUES:
DESCRIPTION: The model should include an attribute to capture the title of a person or organization.

ISSUE OPTIONS & EVIDENCE:

Option 1: Add the attribute of 'Title' to the entity person_or_project.

Pro: The current scope of the model excludes the modeling of organizational units in detail. This option will supply the desired information without forcing an in-depth investigation into organizations.

OPTION PROPOSED: 1
EXPLANATION:
DECISION:
DECISION DATE: 07/12/88

ACTION:

5.7 Related Documents

This section contains a partial list of documents which the developers of the Product Structure Configuration Management data model considered relevant to the development of the model. This list is not intended to be a source list for the document; it does, however, provide guidance for and exposition of the ideas contained within the model.

1. R. Gale, "PDES Product Data Planning Model (Working Paper)", November 15, 1986 (Revised), 13 pp. + attachment.

2. R. Brooks, R. Goldsmith, "PDES/STEP Product Management Data Model (Proposed)", July 17, 1987 (Revised), 41 pp.

3. PDES Mechanical Product Definition Committee, "Mechanical Products Definition Data Model", March 22, 1986, 96. pp.

4. PDES Mechanical Product Definition Committee, "Mechanical Products Definition Data Model", September 26, 1986, 82 pp. + attachments.

5. PDES Mechanical Product Definition Committee, "Mechanical Products Definition Committee Peoria Effort Documentation", December 19, 1986, 52 pp.

6. "Information Modeling Manual: IDEF1 - Extended (IDEF1X)", ICAM Project Priority 6201, USAF Prime Contract F33615-80-C-5155, December, 1985

SECTION 5: PSCM INFORMATION MODEL

6 GENERAL AEC REFERENCE MODEL

N28

EXTERNAL REPRESENTATION OF PRODUCT DEFINITION DATA

DOCUMENT NUMBER: 3.2.2.1

TITLE: GENERAL AEC REFERENCE MODEL (GARM)

ABSTRACT: This document presents a high level general reference model for AEC product definition data. It is based on previous versions, published as ISO TC184/SC4/WG1 documents N77, N149 and N158. The General AEC Reference Model obtained the Draft status from the AEC sub-committee of WG1 during its meeting in Rotterdam, January 1988. GARM is intended as a general model for all relevant AEC application areas, without extensions for specific product-types. These extensions are not yet available within STEP, but they will be included in future versions. GARM serves as a link with the proposed STEP/PDES Planning model, which contains several fundamental abstraction mechanisms, and more detailed data models. This version of GARM contains also a proposal for the inclusion of a high level topology model, called Meta-topology. During the STEP/PDES meeting in Denver, July 1988, it was decided to include these concepts in the Draft.

KEY WORDS: AEC, Generalisation, Decomposition, Aspects, Classification, Libraries, Meta-topology, Reference-Geometry

DATE: October 12, 1988
OWNER: Wim Gielingh
ISO REPRESENTATIVE: Wim Gielingh
STATUS: Draft

1. INTRODUCTION

This document describes the General AEC Reference Model (GARM) as recommended by the AEC subcommittee for inclusion in STEP/PDES version 1. GARM is an abstract, high level data-model, which is supposed to be general enough to serve the needs of all AEC application areas. The model was originally focussed on the modelling of Buildings, Building systems and Building elements but has been generalised such that it could serve also the needs for other product-types. The original intents of the model are described in chapter 3 Considerations. These considerations may clarify many features of the current model.

GARM was given the Draft status by the AEC sub-committee of WG1 during its meeting in Rotterdam, January 1988. Experiences with implementation, the translation into Express, and comments made in the AEC committees of PDES/STEP and by users in several research projects during the year, did lead to some minor changes.

Although GARM is supposed to cover the needs of all AEC application areas, this has been proven until now for only a limited number of products. Modelling activities for specific product-types and AEC-systems by members of the AEC-subcommittee are not yet finished. These models have to show specific needs for important product-classes in AEC. It is also recognised that a general model such as GARM will never be able to cover all the needs of these application-areas. Therefore it is foreseen that the model has to be extended with data-structures for specific product-types. This can be done by defining sub-types of some GARM-entities for each application area. This document contains some examples for the product-type Buildings; these examples are not part of GARM itself, but show how GARM can be used for specialisation towards a specific product-type.

GARM for STEP/PDES version 1 is described in the language Express. A human interpretable version of the model is described in IDEF1x. This model shows more entities then the ones included in the Express model. These additional entities are formally not part of GARM, but provide some context in order to understand the model better. They indicate also the direction in which the model can be extended in the future. The entities which are not part of GARM are marked by dashed boundaries in the IDEF1x model; see the explanation of the IDEF1x model.

Practically all parts which are now included in version 1 of STEP/PDES are implemented and tested by TNO-IBBC in The Netherlands, by means of the experimental product modeller ProMod. Test implementations of the new concepts for topology, called Meta-topology, are not yet completed.

1.1. Acknowledgement

The basic concepts of the General AEC Reference Model were originally developed by the Department of Informatics of the Institute for Building Materials and Structures (IBBC) of

the Dutch Organisation for Applied Scientific Research (TNO) in the Netherlands. It is improved and changed over the years, due to discussions in the AEC committees of STEP and PDES, and its use in a variety of research projects. The evolution of the model can be made visible by comparison of previously published versions: version 1 (ISO document N77, September 1986), version 2 (N149, May 1987) and version 3 (N158, August 1987). Apart from these, many separate discussion papers and publications related to the GARM were made. It is impossible to summarise them all in this document.

Werner de Bruijn and Anita Eijs of TNO-IBBC helped with the formulation of GARM in Express. Peter Willems of TNO-IBBC proposed Meta-topology, and should be credited for that part of GARM. Peter Kuiper of TNO-IBBC was responsible for the implementation and testing of the concepts; he made many useful comments on GARM and Meta-topology. Frits Tolman of TNO-IBBC made many useful comments on the model and formulated several proposals. Within the AEC committees of PDES and STEP I would like to credit Jeff Wix (Wix McLelland), Robert Aish (currently working at Intergraph), William Danner (NBS/NIST), Pat Harrow (Harrow associates), Michael Gerardi, Rick Lovdahl and Douglas Martin (all NIDDESC), Barbara Warthen (Calma), James Turner (Univ. of Michigan), Trond Heier (CADCOM-Norway), Gail Vermilyea (Vertek assoc.), John Zimmerman (Alied Bendix), and Roger Gale (DACOM) for their contributions and many useful discussions. The GARM is used in various projects, not directly related with STEP, which lead to many other useful comments. Most noteworthy are the Eureka EU130 project for steel structures, projects on product-modelling for Bridges and Roads for Rijkswaterstaat in The Netherlands, and The Innovative Research Programme (IOP) for use of Information Technology in the building industry in the Netherlands. I would like to thank everyone who contributed to this General AEC Reference Model, and hope that we are able to continue this collaboration for the definition of future versions of the GARM.

Remarks concerning this document and the proposed model, preferably marked in red, can be sent to:

Wim Gielingh
TNO-IBBC
P.O.Box 49
2600 AA Delft
The Netherlands

Phone: +31 15 606445
Fax: +31 15 620304

2. CONSIDERATIONS

2.1 Assembly-modelling and parts-modelling

Advanced modelling techniques for CIM which have been published in recent years, are primarily intended for the design and manufacturing of single parts. The modelling of complex systems such as buildings is still an under-developed area. Recent work on assembly-modelling for mechanical products is mainly based on a bottom-up approach: the single parts have to be defined before an assembly model can be made. This conflicts with the working method of a designer, which can be characterised as top-down. It is desired that modelling techniques support the logical process of creation and generation of information. Not only for the sake of the designer, but also for the quality of information-exchange between the representatives of various disciplines involved in product-development.

A scheme design can be considered as a global definition of the building layout and structure, with a global specification of the materials. We see that several modern 3D CAD systems for architectural design represent a building by simplified geometric models. Systems intended for the design of distribution systems (HVAC and plant design) use wireframes as the basis for their geometry.

Even the detailed design does not contain an "exact" geometric definition of components. The geometric part of the detailed design indicates where the components have to be placed, and how they can be connected (detailed drawings describe the principles of the details). The actual "exact" product definitions are provided by the manufacturers of these components.

Summarised: "assemblies" are designed before knowledge about the components exists. Techniques for assembly-modelling should support this.

2.2 Information exchange between various disciplines

AEC-products are characterised by the fact that many disciplines and companies work together in one project. There is a great need for efficient communication between them, but there is usually no principal (dominant) company or discipline that sets the "standard". All parties are about equally important and have there own "standards" and practices.

But even when there would be one company or discipline that could set a standard for communication in a project, we have to consider that these co-operations do not last longer than the project itself. Most companies are even involved in many projects at the same time. This means that they have to communicate not only with various disciplines, but also with various representatives of each discipline.

2.3 Project-oriented and discipline-oriented data

Not all the information that becomes available in the course of a project is "new". In fact, most of it already existed. It existed as knowledge and experience for each of the disciplines involved.

There is a great need to capture this knowledge and experience, not only because it could prevent that certain mistakes are made again, but also because it can make the whole development more efficient. Experiences with earlier projects can help us to have better control on new projects (think of time- and cost-planning).

Much of this information is "hidden" in the form of standard practices, rules, codes, etc. It can also be recorded in the form of previous designs and project-data, standard parts or details, data-bases with cost-information, etc. The advent of expert-systems offers another approach for capturing experience and knowledge.

This re-usable discipline-oriented information is extremely important for the AEC-industry, since buildings are usually not produced in large series. Most of them are unique. What is learned from a project, cannot be applied to that same project again, but only to new projects with other conditions. Re-usable information can also reduce the costs of design and development.

Summarised: information has to be structured such that it can be re-used in other projects. Due to the nature of the AEC-industry, where knowledge and experience is divided over many disciplines and companies, this information will not be stored centrally, but in discipline- and company-oriented libraries.

2.4 Top-down vs. bottom-up creation of the product-model

Architects design roughly top-down. Complex products such as buildings (but also complex systems or parts of a building) are developed by first specifying its functionality as a whole. Then global solutions are created, which contain again (smaller) design-problems. When there are no existing solutions available or applicable, they are solved again by decomposing the problem into smaller problems.

It is desired that a design can be evaluated before it is realised, or even before it is finished. Evaluations are necessary for the support of decisions. Intermediate evaluations of a design are for instance important for the control of costs and functionality. It is a well known fact that the costs of a building are influenced largely by decisions taken in early stages of the design process.

The design-process ends when existing solutions for each of the detail problems are found, or when the problems are specified well enough to start the next stage of the product development process (production planning and manufacturing). Existing solutions are for example solutions found in earlier work, standard solutions (parts, form features), or components offered by suppliers. We see here again the importance of re-usable information.

Suppliers or manufacturers of components tend to work in the opposite direction. Manufacturers have certain skills and tools which form the basis of their productivity. Suppliers try to sell what they have on their shelves. When the market has a demand, they try to react on it by combining skills, use of tools and/or supplies. This is a bottom-up approach.

Summarised: products (buildings, building-systems, etc.) are usually designed in a top-down oriented process. Potential clients and designers specify their needs and requirements, and try to find solutions. Suppliers and manufacturers tend to work bottom-up: they offer solutions for which they try to find potential users.

2.5 Global vs. exact specifications

A technical drawing contains lines and symbols with specific meanings. But what do they mean?

Sometimes lines indicate the *boundary* of a (solid) object. This boundary can be the "exact" boundary of material and air, but it can also be an indication for a global envelope.

Lines may also indicate *simplified geometric descriptions*. For instance the reinforcement of concrete, or pipes and cables. In these cases we have to do with components with a cross-section that is very small (neglectable) compared with their length. Moreover, the drawings on which they appear as lines have an entirely different function as drawings of their cross-section: the lines indicate where they have to be positioned globally, the cross-sections are of use for their production or for local positioning (in the latter case the cross-section can be indicated globally).

Some lines do not even refer to physical objects, but are used for *positioning components*. Examples are center-lines, symmetry-axes, crosses, etc. These symbols can be important for the definition of dimensions and tolerances.

There are several classes of symbols. Some symbols may indicate the location of functional components which are only applied but not designed, such as door-handles and light-fittings. They do not describe their shape. The symbol itself can be a graphical representation of the products function. Other symbols are part of the graphical presentation of information, and have nothing to do with products or components. Examples are arrows, squiggles, alpha-numeric symbols, etc.

The use of 3D geometric modelling systems does not change the need to record different sorts of geometric information. It would be foolish, for instance, to represent the reinforcement of concrete structures as solids in stead of their center-lines.The center-lines would be sufficient for most purposes. Solid modellers are however primarily intended for the description of the exact boundary of the material, in which we are not interested in many cases.

So far we have looked only at geometric data. But other data can also be global, more precise or "exact". For instance, a material can be described as "metal", "steel", "stainless steel" or "AISI 430".

Summarised: Geometric data may have different meanings. And for each meaning the geometry may be specified on different levels of detail. These levels of detail are not only found in geometric data, but also in other data for product-specification.

2.6 The design team problem

If projects are complex there are usually several designers involved in the whole process. One designer is responsible for the "high level" solution, and delegates specific sub-problems to specialists. When several designers work on the same project at the same time, they have to be able to do this independently. It is very well possible that their results conflict. In that case it is up to the designer who is responsible for the higher design-level to re-specify the requirements and constraints for each of the sub-tasks.

This division of design-tasks is very common in architecture. The architect is traditionally the one who is responsible for the design on a global level. Details and sub-systems are worked out by other specialists. When their results are conflicting, it is the architect who has to solve the problem by (re)specifying constraints for each of the sub-tasks.

The interaction of several people working on one design, the administration of changes, and the responsibility for decisions are aspects which have to be addressed. This problem is vital for STEP, because the division of work leads to an important information-exchange between disciplines, such as specialists and an architect, where the latter has to integrate all the information into one consistent design.

Related to this problem is the proposition of alternatives. Each of the specialists may offer several alternative solutions for a specific design-problem. They have to be integrated in the global design and evaluated, so that the best one can be selected.

2.7 Product information as a function of time

What we know about a product is dependent of its stage of development. It is evident that the amount of data increases during the products development and life-cycle. But not only the quantity of facts is a function of time: also the meaning and structure of data changes.

Roger Gale describes in Appendix I of the PDES Initiation Effort Report (ref. 6) data configurations for three stages: "as designed", "as planned" and "as built". Since we may be interested also in the product "as required" and "as used", I propose to develop models also for these two stages. In order to let the AEC model correspond with other models, I prefer to use these five characteristic steps and to develop models which describe product-information:

as required

as designed

as planned

as built

as used

Models should reflect the type of information which becomes available during each of these stages. We should realise that this division is not an attempt to model the process of product-development: it is an aid to separate data-models with different characteristics.This global division can be used until further differentiations appear to be necessary.

2.8 Integration of representations, ownerships and versions

A product is represented in different ways for different applications. A building is represented traditionally by drawings, but for certain applications scale-models or numerical models for calculations are used. In the computer-age we will tend to use more and more numerical models. But there are many ways to represent a building or a building-component: each application uses its own representation.

How are these representations kept consistent? And how will systems with different internal representations communicate with each other?This problem has not yet been addressed by the PDES and STEP communities. The logical layer of STEP 1.0 will contain several methods to represent the geometry of a product:

- Drawings
- Wireframes
- Surfaces
- Boundary-representation
- CSG-representation
- FEM-representation

Even within one method, such as Surface-representations and Boundary-representations of solids, we see different and incompatible representations (for instance the exact and facetted B-reps).

We recognise at least four levels of integration:

- Integration of *different product-representations*;
- Integration of identical product-representations kept by *different owners* (f.i. disciplines); Integration of identical product-representations with different levels of detail. This includes the time-dependent factor of modelling.
- Integration of *different versions* of identical product-representations with the same level of detail, kept by the same owner.

3. GENERAL AEC REFERENCE MODEL

3.1 Basic principles

GARM is based on the idea that product-information is clustered around so-called Product Definition Units (PDU's). A PDU can be the whole product, but also sub-systems, elements, components, parts or features of a product. In fact, a PDU can be any part of the product interesting enough to record information about.

This information is given as a collection of *characteristics* of the product. Each characteristic is related to an *aspect*. Examples of aspects are strength, costs, durability, safety, etc.

The meaning of a PDU and a characteristic is determined by a set of abstraction mechanisms which can be considered as general and user-independent. The major ones are:

1. Specialisation. This gives more meaning to a PDU or characteristic, depending of the application area which uses GARM. The following application areas are currently identified within AEC: Buildings, Civil Engineering, Process Plants, Ship building, Terrain Mapping. Each area can be specialised further. Specialisation is modelled usually via "is-a" relations or sub-typing in IDEF1x, NIAM or Express.

2. Decomposition. The decomposition abstraction describes how a product can be decomposed into smaller units. Or, in reverse, be composed to larger units. Decomposition is usually modelled via "contains" or "part-of" relations. Composition (also called aggregation) via "is contained by" relations.

3. Life-cycle. This abstraction addresses the various stages of the life-cycle of a product. The following stages are identified in GARM:

- as required
- as designed
- as planned
- as built (or: as produced, as manufactured)
- as used
- as altered
- as demolished

4. Classification[1]. The classification-abstraction allows us to give information for each instance of a PDU, for a group of identical PDU's, and for PDU's which may be different but have much in common. The latter group can be described by means of parametric models. Classification reduces redundant information. It is incorporated by means of the level-discriminator in GARM.

1The term Classification is used for this principle by Potter and Trueblood [Computer, June 1988, p.53-63]. The principle is clearly related with Generalisation/Specialisation, and could be regarded as a user-defineable type of specialisation. The major difference between Classification and Generalisation/Specialisation is the fact that the latter is semantical, the other not.

3.2 Characteristics

Each characteristic of a Product Definition Unit is related to an *aspect*; see also diagram A.0. Aspects are for example strength, costs, durability and safety. They can be defined more explicitly for a specific application area. A range of aspects for buildings is given in diagram B.1 and B.1.1 of this document, based on an existing ISO standard. These diagrams and the corresponding ISO standard are not considered as a part of GARM; they form an addition.

A product needs certain characteristics due to its function and external influences. For instance, strength characteristics are required because the product has to carry external loads. These external influences are included in the model as *agents*. A classification of agents for buildings is given in diagrams B.2, B.2.1 and B.2.2, again based on an existing ISO standard.

Shape and material are not regarded as aspects in GARM. Both shape and material cannot be handled in the same way as the aspects mentioned before, since each aspect has a "shape-view" and a "material-view" of the product. Shape and Material are represented in Aspect-oriented Product Models. See also chapter 3.10.

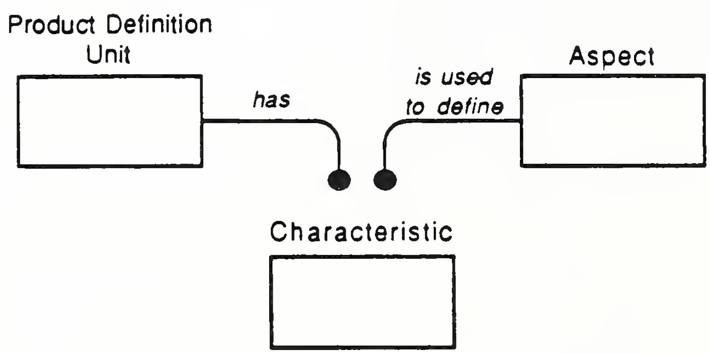


Fig.1. A Product Definition Units has Characteristics. Each Characteristic relates to an Aspect

3.3 Life- cycle

The life-cycle concept is important since information about the product can be time-dependent. Or more precisely: dependent of the stage of the product in it's life-cycle. The stages which are embodied in GARM are considered as general and independent of the life-cycle process itself. The distinction between the stages is based on clear differences in the type of information about the product, not on the order of processes which produce this information.

There are now seven stages identified in GARM, going from requirements definition to demolition. The Product Definition Unit can be specified for each stage, by applying the stage-discriminator; the resulting subtypes of the PDU are given in the following table:

Stage	PDU subtype
as required	Functional Unit
as designed	Technical Solution
as planned	Planned Unit
as built	Physical Unit
as used	Operational Unit
as altered	Alteration Unit
as demolished	Demolition Unit

The Functional Unit describes the product "as required". The name Functional Unit is chosen since it describes the required functionality of the PDU.

A Technical Solution is a concept that may meet the requirements formulated by the Functional Unit. Many Technical Solutions are "standard", such as standard components, connections, features, etc. They can be stored in Libraries.

A Planned Unit is a concept that describes how a Technical Solution can be produced. Planned Units can also be "standardised" and stored in Libraries. A Planned Unit collects information about production planning.

A Physical Unit describes the product "as built" (or "as produced", "as manufactured"). In contrast to the three previous stages, this stage is the first one where the product physically exists.

An Operational Unit describes the PDU in use. It is clear that the characteristics of the PDU may change in time, so Operational Units may also differ in time.

An Alteration Unit is used to describe maintenance, renovations, modifications and/or upgrading of the PDU.

The Demolition Unit collects information about the product during and after demolition.

The stage-discriminator can be used also for some other high level entities, such as characteristic. This leads to three major types of characteristics:

- Required Characteristics (as required)
- Expected Characteristic (as designed, as planned)
- Measured Characteristic (as built, as used, as altered, as demolished)

The required characteristics can again be given for each stage: design, production planning, production, use, alteration (including maintenance) and demolition. It was stated before that the distinction between the units is not based on the order of processes which lead to them: it is for this reason why all these different types of requirements all belong to the Functional Unit, even though certain requirements could be specified later, for instance during production planning or building.

The expected characteristics are identical to analysis results. They are based on simulations with (numerical) models of the product, not the real product itself. The measured characteristics can be the result of testing of physically existing samples (measurements which are always done for occurrences of the PDU), or measurement of the product in use, during and after alteration, or during and after demolition.

3.4 Specialisation

The GARM contains generic entities, not dependent of a specific type of product. In many cases it is however of importance to indicate what the product is (a wall, a column, a pipe, etc.), or to be more specific on aspects and characteristics for certain product-types. This can be done by introducing the specialisation - discriminator to these entities. Within STEP, three layers are defined which correspond to major levels of specialisation:

- General STEP
- Industry-type
- Product-type

An additional fourth non-STEP layer is added to incorporate specific entities which are defined outside STEP. The latter layer can be used for additional national standards, company-standards, etc. The General STEP layer contains entities which are general for all types of Industry. The Industry-layer contains currently three broad classes of industry: AEC, Mechanical and Electrical. GARM is can be used to serve the needs of the whole AEC-industry. On the product-type layer we can divide each industry again into more specific product-types. AEC can be divided further into Buildings, Civil works, Ships and Plants.

Generalisation/Specification

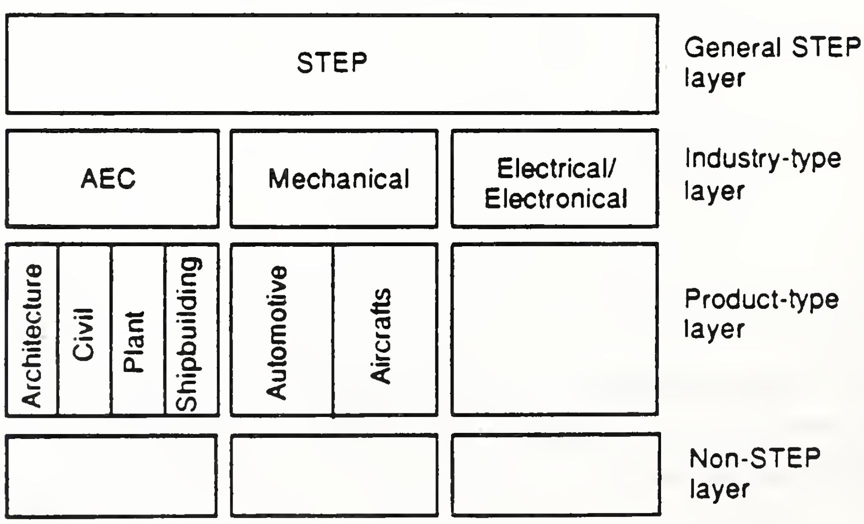


Fig.2. Four layers to specify product definition data. The fourth one is not part of STEP; it is an additional layer which allows the incorporation of other standards

Entities which belong to the product-type layer can be defined as subtypes of entities on the Industry-type layer; in these cases they inherit properties from their super-types. It is also possible that entities exist only on the Product-type layer and do not have a super-type.

The idea of specialisation is applied to three entities in GARM, and included in this document. This is done for the entities Functional Unit, Aspect and Agent, specialised to the product-type Buildings. See the B-diagrams of the IDEF1x model. Since these specialisations are based on existing ISO-standards or existing classifications, all the specific entities are considered as being part of the non-STEP layer.

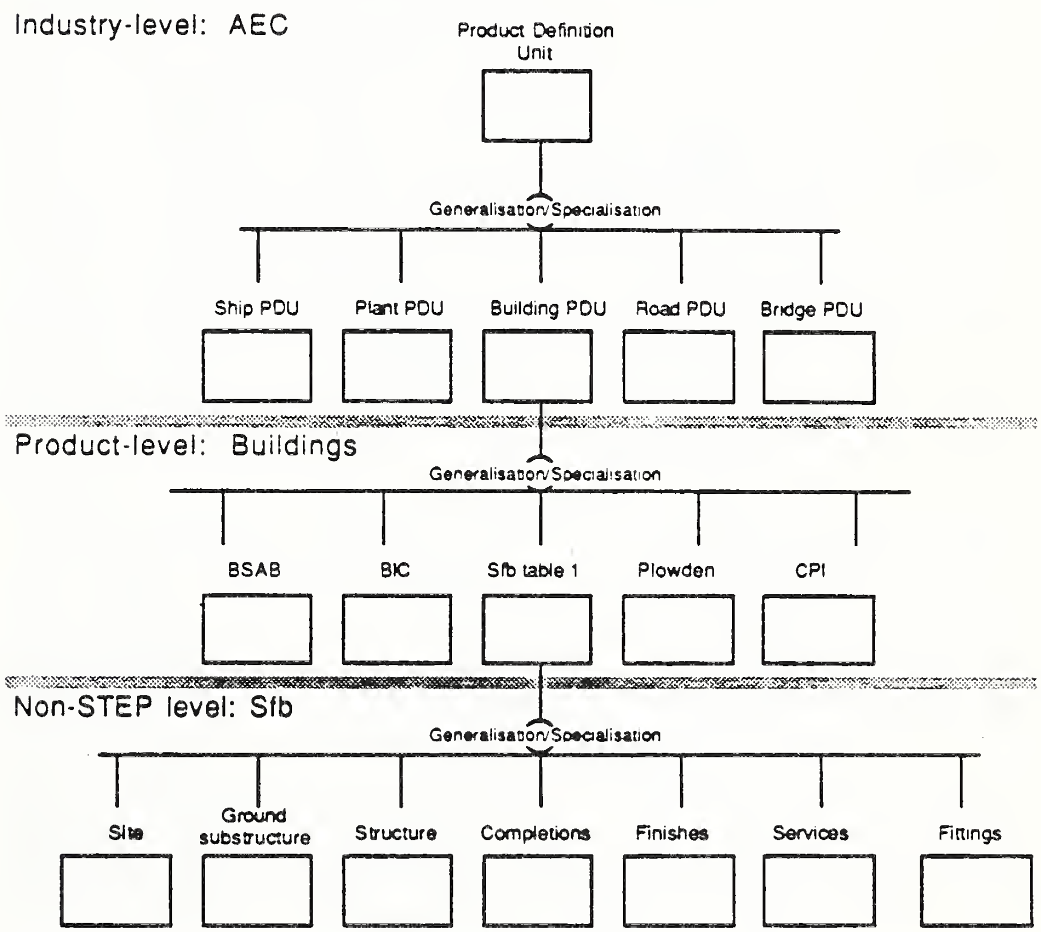


Fig. 3. An example of specialisation within AEC. The General AEC reference model identifies on the Industry level five classes of products: Ships, Plants, Buildings, Roads and Bridges. Several classifications are in use for building products (systems, elements), such as the Swedish BSAB, the British BIC, Sfb (adopted by CIB for international use), Plowden and CPI. The CI/Sfb code identifies on its turn seven classes of building-elements. Each of them can be decomposed further (not shown here). The Sfb code does not belong to the STEP standard, and is therefore put on the Non-STEP layer.

3.5 The Level discriminator

It is possible to define Product Definition Units on three levels. The "lowest" level is based on a description of a PDU for each occurrence; it is therefore called the *occurrence* level. Since this would lead to an enormous amount of redundant data, a second level is introduced, called *specific*. PDU's which are identical and occur many times can be described on this level. It is only necessary to give location and orientation for each occurrence, and all the other information can be given on the specific level. On the third level it is possible to define parametric descriptions of the PDU. This level is called *generic*. An example is given in fig. 4 for a window-frame.

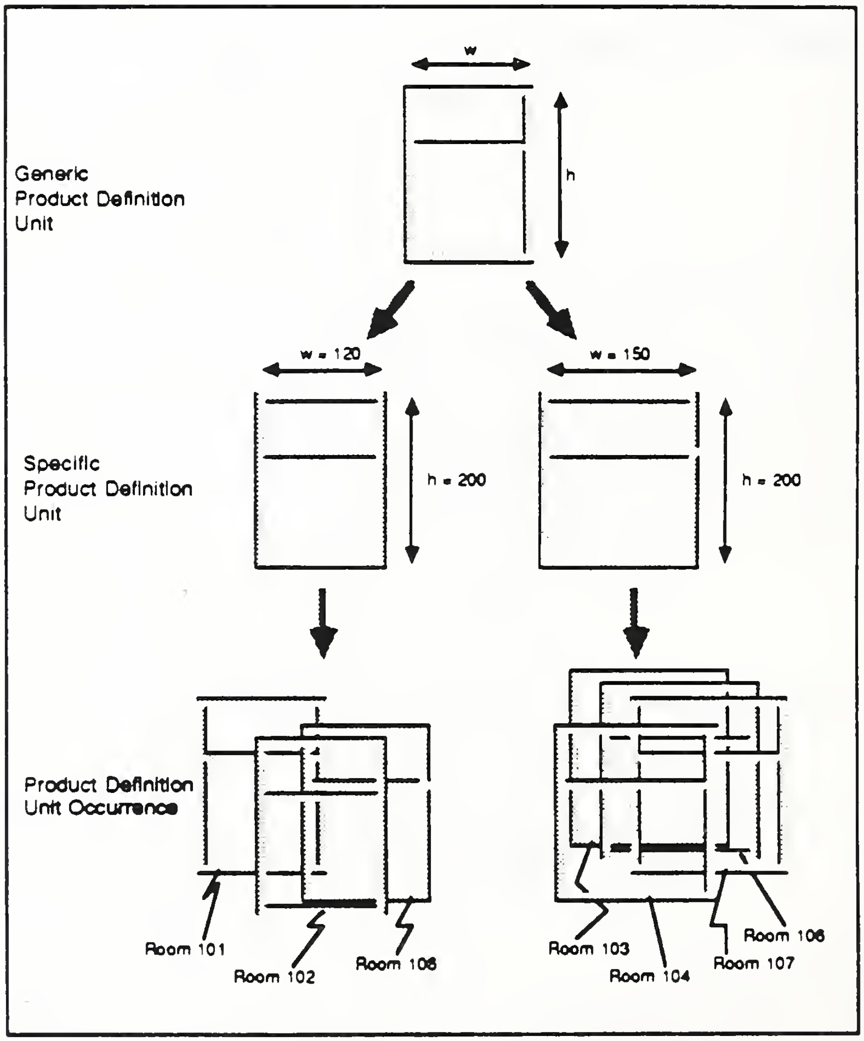


Fig. 4. Classification by means of the level-discriminator in GARM. A Window-frame can described parametrically (with parameters w and h) on the generic level. Once the parameter values are known we have specific versions of the window-frame. A specific frame can occur however many times in a Building. Each instance is described on the Occurrence level (this will usually be only its place).

Occurrence PDU's refer always to Specific PDU's, and inherit all their properties. The same dependency exists between Specific and Generic PDU's.

Generic PDU's may form a hierarchy. Suppose that we start with a generic PDU with six parameters. If two of the six parameters are given, the PDU is less generic then the original one, but it is still generic. Only if all parameters are known, the PDU has become specific. The hierarchical structure that can be formed by generic PDU's is included in the GARM by means of the entity Generic_PDU_structure.

Generic PDU's can be placed in a PDU-library. If we apply the stage-discriminator, we can distinguish between Functional Unit libraries (being for instance standard reference specifications, standards, codes of practice), Technical Solution libraries (standard part catalogues, standard details, feature libraries), Planned Unit libraries (production units such as process features and rough materials), and Physical Unit libraries (stored components, spares, etc.).

For practical reasons it is allowed to have only one Occurrence of a Specific PDU. So also when a Specific PDU occurs only once, it will be defined on the specific level too. This has the advantage that all relevant information of the PDU can always be given on the Generic level, except for location and orientation. It is also allowed to have Generic PDU's without parameters, making them identical to Specific PDU's. This means that non-parametric definitions of PDU's can also be stored in libraries. For reasons of reduction of data it is however recommended to parameterise PDU's as far as possible.

3.6 Decomposition

Several modelling exercises showed that a generalised concept for decomposition, independent of the other abstraction mechanisms, cannot be defined. In most cases the decomposition depends on specialisation (the type product to be modelled) and life-cycle (decomposition in the design stage is not the same as decomposition in the production-stage). GARM includes a decomposition model (Diagram A.3) which is fairly general, and at least is able to bridge the gap between most AEC oriented models and Mechanical Product oriented models. But this model is not complete and ambiguous, it is intended primarily for explanation. Therefore it is not included in the Express version. GARM offers another, more flexible alternative to define decomposition: by means of the decomposition of Technical Solutions in Functional Units. This is described in the next chapter. Let us focus first on some decomposition issues.

Diagram A.3. shows the composition/decomposition of PDU's in general terms. It contains three major classes of PDU's: systems, parts and features. A system can be decomposed into sub-systems and/or into parts. Any system can play the role of sub-system. A part can be decomposed further into features. Three classes of features are identified: Pattern features, Compound features and Primitive features. The primitive features form the most elementary features; they can be used to build more complex features, so-called compound features. Pattern features are sets of compound features which are placed in a pattern. The pattern can be defined through a procedure. Please note that the model allows the definition of compound features created from only one primitive feature, and that a pattern feature may contain only one compound feature. It is therefore possible to have for instance a pattern, which contains only one cylindric hole-feature. Note also that a compound feature may contain one or more pattern features.

Systems can be divided further into Arrangements and Assemblies; the difference between both types is the connectivity between the sub-systems and/or parts: assemblies contain physically connected sub-systems, arrangements contain sub-systems which are not physically connected. Here we have the first problem with making such distinctions: connectivity is dependent of aspect. For example, two separate buildings could be regarded as an arrangement from, for instance, the constructors point of view. However, for the electric systems engineer they will be physically connected (by wires).

Related to this is the meaning of words such as system and assembly to life-cycle stage. System has a functional meaning and is related to functional design. Assembly however has more meaning for manufacturing and construction. The distinction between part and assembly may differ in the design and production planning stages: what is regarded as a part by the designer, could be regarded as an assembly by the manufacturer. Related to this issue are questions such as: is an (electronic) chip a part or an assembly?

These questions are considered as academic for the time being; GARM offers a more general composition/decomposition mechanism embedded in the design and manufacturing stages. Only design has been worked out yet.

Concerning the dependence of aspect: each PDU is considered to have one primary function and zero, one or more secondary functions. The aspect-view of the primary function is regarded as decisive for the decomposition in the design stage of GARM. We may call this the primary decomposition of a PDU. The secondary aspect-views may lead to secondary decompositions, which are *derived* decompositions.

3.7 Decomposition in the design stage

When a design-problem is very complex, a designer can solve it by decomposing it into a set of smaller problems, which are easier to solve. A "Technical Solution" consists thus of a set of new "Functional Units". Each Functional Unit will be specified again (usually not in a formal sense), so that Technical Solutions can be found for them. This decomposition of complex problems into structured sets of smaller problems continues until available solutions are found.

This "divide-and-conquer" strategy is well-known, and leads to a user-defined decomposition of the product. Other approaches, such as a bottom-up strategy, or operations research approaches, are also possible with the construct of Functional Units and Technical Solutions. The fact that the stage-oriented entities Functional Unit and Technical Solution are used for composition/decomposition may seem confusing at a first glance, but it reflects simply the design approach which is applied: requirements given for detailed components, parts or features of the product, are dependent of more global technical solutions chosen in an earlier design stage.

This approach has also advantages for the re-use of information: intermediate design-solutions can be stored in Technical Solution Libraries so that they can be re-used in other projects, shared with colleagues, etc.

The user-defined composition/decomposition of the product makes GARM flexible in practical use. Without any predefined constraints on decomposition, it is of course possible that strange decompositions are specified, such as "a part containing an assembly". We may assume however that a designer will not create such a structure, unless his view of the product changes (remember the example of a chip mentioned in the previous chapter). Predefined constraints for certain product-types and/or aspects can be added. They provide additional rules for decomposition and may help making the product model more reliable and meaningful. We have to realise however that this will only be true if these constraints are reliable themselves. See also the remarks made in the previous chapter.

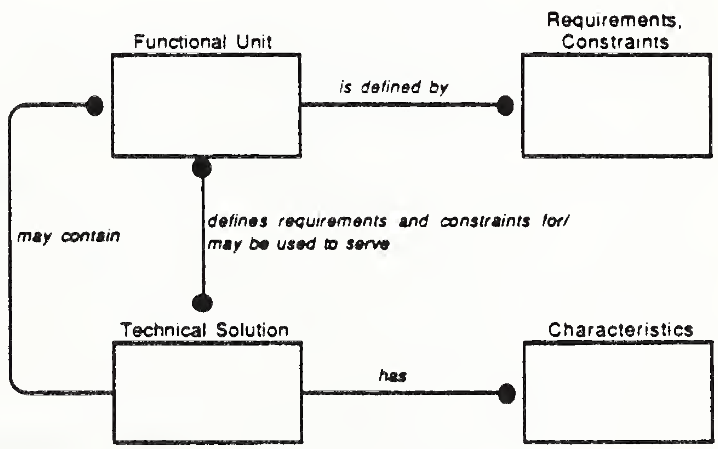
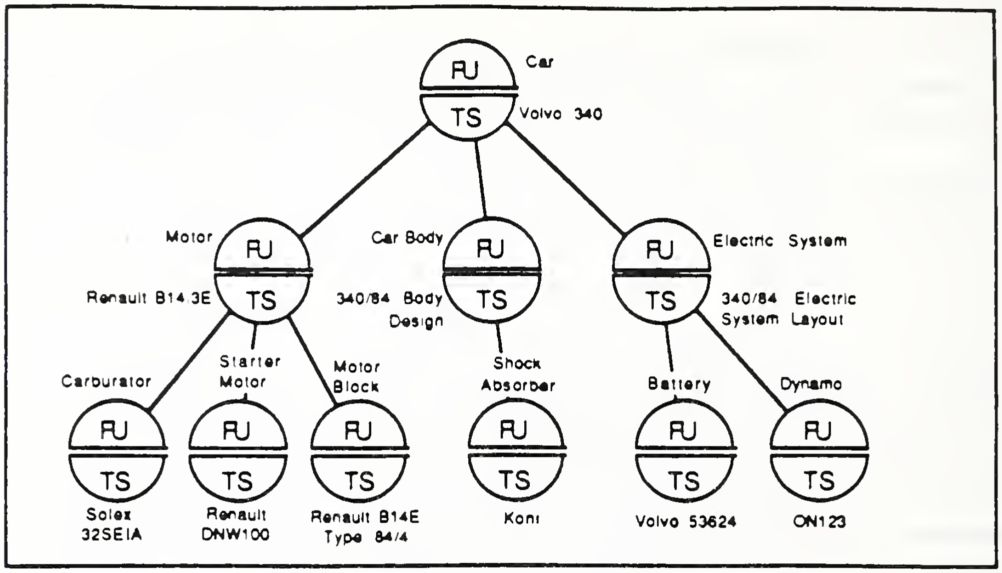


Fig. 5. A Technical Solution can be decomposed into a (structured) set of Functional Units. This modelled by means of the "may contain" relation between Functional Unit and Technical Solution.



Hierarchical Structure

Fig.6. Example of the decomposition of a product (a car) by means of Functional Units and Technical Solutions. A Functional Unit (FU) and a Technical Solution (TS) are represented here as semi-circles. A car is a Functional Unit (it has a function and it can be defined by means of a set of requirements). A Volvo 340 is a possible Technical Solution which may satisfy the requirements. The Volvo itself contains again certain Functional Units, which may have been produced by other manufactures. A complete product-model is therefore regarded as an aggregation (composition) of components with certain functions, which can be fulfilled by various (interchangeable) technical solutions.

3.8 Decomposition and the design-team problem

Complex products are not designed by one person, but by a team of designers which work fairly independent on different parts or systems of the product. In fact, certain parts or systems can be designed and produced by other companies. This problem has been addressed in chapter 3 Considerations as the design-team problem. GARM reflects the distribution of responsibilities by the distinction between Functional Units and Technical Solutions. Every designer in the design-team works on one or more Technical Solutions, which can be used to fulfil the needs of one or more Functional Units. Every designer may play two different roles: the role of the *user* of a Technical Solution, or the role of the *creator* of a Technical Solution. This is reflected in the model by the different ownerships of Technical Solutions and Functional Units: the owner of the Functional Unit is a (potential) user of a Technical Solution.

3.9 Evaluation of Technical Solutions and Decisions

This part of the model is described graphically in Diagram A.6. Technical Solutions which can be used to fulfil the needs of a Functional Unit, may have the status "alternative", "selected" or "rejected". As long as no decision has been taken, the status remains alternative. In order to compare alternatives and to select the best one, each Technical Solution will be evaluated. Each evaluation takes one or more aspects into account. For this purpose we make aspect oriented models in which the Technical Solution is represented. These models are used by experts for analysis. Examples of such aspect oriented models are FEM-models, solid models including aspect oriented material- and surface-property definitions (for visual inspection, mass-properties, collision-checks, etc.), and models for energy-, light-, safety-, strength- and cost-calculations.

An analysis leads to one or more analysis results, which are are included in GARM as Expected Characteristics. A Decision is based on Arguments in which a comparison is made between Required and Expected Characteristics. It is also possible to base a Decision on formal Approvals by Experts.

Please note that GARM makes use of the Allowed Parameter Domain entity for Required Characteristics. This entity is also used in another context for the description of Generic Technical Solutions. Allowed Parameter Domain helps us to define minimal or maximal values of Required Characteristics, but also value domains or discrete values. The Express function "In_Domain" checks if Expected Characteristics are within the Allowed Domains.

3.10 Shape and Material

Shape and Material are not regarded as aspects of a product in GARM. Any aspect such as strength, durability, cost, energy and safety, may have its own shape and material definition. Shape and Material are therefore *dependent* of aspect. Moreover, they are also dependent of stage (the products life-cycle): a designed shape will not always be identical to the required shape, a planned shape (based on production technology) will not always be the same as designed shape, and the produced (manufactured) shape will not be identical to the planned shape. During use and maintenance, shape may change again. The same is true for material-properties.

The distinction between shape and material is made on a general level for Procedural and Explicit Product Definitions. This is only worked out for Technical Solutions so far, so we see them in diagram A.5. as Procedural and Explicit *Technical Solution* Definitions. A Technical Solution may have several procedural or explicit definitions, depending on representation and detail level. A distinction is made between shape and material *representations*, and this migrates through the representation-key to the procedural and explicit T.S. definition.

An overview of shape-representations and their possible dependencies is given in diagram A.8.

GARM has two major classes of shape-definition, each with a different purpose: *reference-geometry* and *analysis-geometry*. Reference geometry is used by a designer to locate components, define global shape, etc. It is usually independent of aspect; a dependency of aspect may occur however when one aspect is dominant over other aspects in a specific (sub)system of the product. For instance, the shape-definition of a building-structure will be very much oriented towards the strength-aspect. Analysis-geometry is always a *derived* geometry, intended to study the behaviour and characteristics of the product for a specific aspect. This makes Analysis-geometry always dependent of aspect.

The reference geometry can be used to define the location of and the relations between Functional Units. This will discussed in chapter 3.12. GARM supports the use of non-manifold Reference Geometry/Topology .

[Material definitions are not yet studied in detail. It is therefore unclear if a similar division into reference- and analysis- material-definitions can be made. Most material specifications are aspect dependent, but the fact that most products have a primary aspect and several secondary aspects may help in finding reference material-definitions]

3.11 Functional networks

A technical solution may contain a structured set of Functional Units. Structure is essential in a product-model: a product is not just a collection of components: the way these components are combined and connected determines how the whole product behaves. This (functional) structure can be modelled in a general way with a network: each Functional Unit refers to a Node in the network. Each node has zero, one or more Ends, and two Ends can be connected via an Interface. This allows us to define any kind of network relationship between Functional Units.

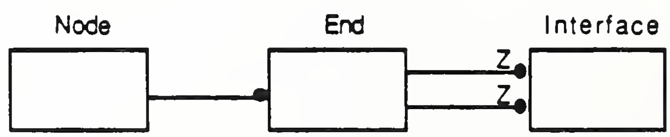


Fig. 7.Generalised network, modelled as Nodes with Ends, which are connected via Interfaces. Each Interface refers to two Ends. But Ends are not always connected directly via Interfaces; they can also be linked with higher level networks via Ports. These Ends are called Free within their own network.

In the real world we see that anything may be related to anything else. The real world can be described by a huge network of relations between PDU's. Modelling the real world by such a "Universal" network would be an impossible task however. In chapter 3.7 we discussed the possibility to "divide and conquer" complex problems and products by delegation of responsibilities. Each *creator* of a Technical Solution is responsible for the structure of his Functional Units, not those of others. A *user* of a Technical Solution doesn't want to see all the details of the solution he uses; the only thing he needs to know is how to use and apply it, by connecting it to other Technical Solutions. For instance: if someone buys a car-radio, he is not interested in the various components in the radio, he should only know how it can be connected with the electric circuit, the speakers and the antenna of his car. The plugs which are provided to connect the radio with the electric circuit, speakers and antenna are considered here as *ports* of the Technical Solution. The designer of the car-radio should make clear how the components of his device are connected with these ports; it is however not his responsibility to model the connection with the systems in the car. These connections are modelled by the owner (or creator) of the car.

We see this principle sketched in fig.8. This diagram shows how the principle can be used to describe functional networks *within* products (systems), but also *between* products (systems). Therefore it allows us to model functional networks of very complex products (systems) by dividing it into sub-products (sub-systems), without destroying the functional relations between this sub-products (sub-systems). The diagram shows not only Functional Units and Technical Solutions, but also ports of Technical Solutions and Ends of Functional Units as Nodes in the network. Ends which are connected by interfaces within the network of one Technical Solution are called *Mated Ends*. Ends which are not connected in such a network are *Free Ends*; they refer always to a Port. A Port can be matched on its turn with a Mated End on a higher level in the hierarchical tree, thus describing how the ports of two products (or systems) are connected with each other. By following the grey arrows in the diagram we can trace how a component of the radio is connected with components of the Electric System of the car.

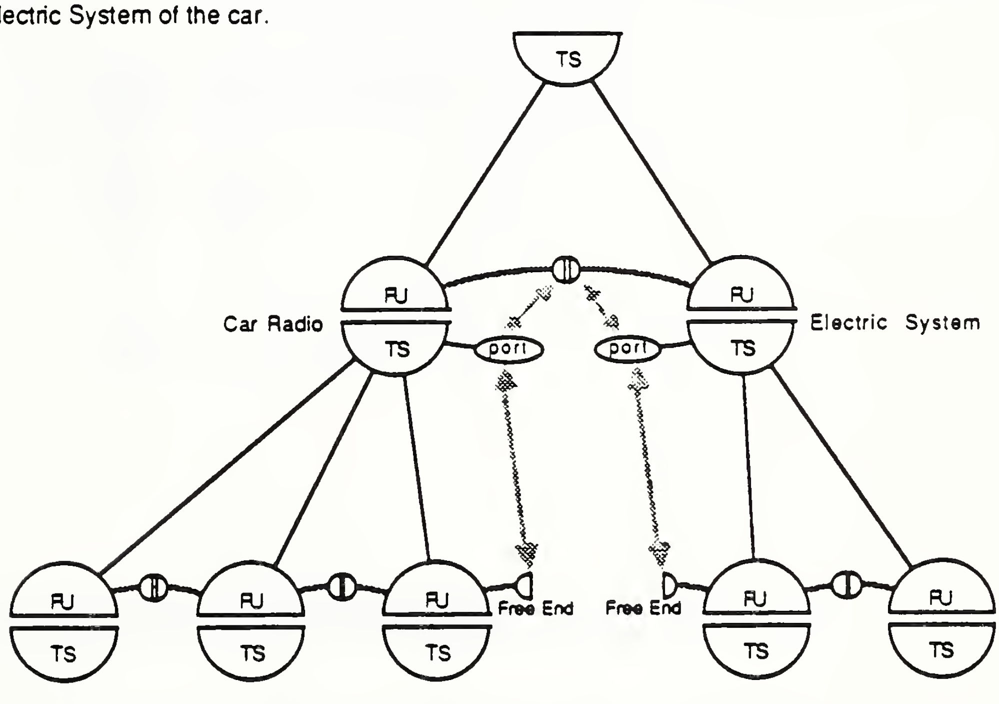


Fig.8 Functional Units which belong to different Technical Solutions can be connected via Free Ends, ports of Technical Solutions, and connected Ends on a higher level. For example, a car radio can be connected with the electric system of the car on the higher level. How the ports of the radio and the electric system are connected with internal components is defined by the decomposition of a Port in Free Ends.

3.12 Reference geometry, Topology and Networks

Several applications use a geometry/topology model to define the interrelations between the Functional Units. We will call this *reference geometry* from now on. Several types of geometry/topology can be used for it: for Plant design we see often the use of wireframe models, for ship design a combination of wireframes and surface models, for part design solid models, and for building design compound Boundary representations and non-manifold topology. Non-manifolds are able to combine all the representations mentioned before, and can therefore be regarded as more general.

Several studies done by TNO-IBBC showed however that non-manifolds are not sufficient to cover all the required topological constructs. For instance, the window-frame sketched in fig.9 cannot be modelled with conventional topology. The frame has one middle-stanchion of which the end-vertices are located on the horizontal sills. These vertices do not form a boundary of the sills; they are boundaries of the middle-stanchion and represent the connections of stanchion and sills. A next step is the definition of loops in this topological model. We can distinguish three loops: the outer one for the whole window, and two inner ones for the glass panels. The inner ones share the edge representing the stanchion, and each inner loop coincides partially with the outer loop.

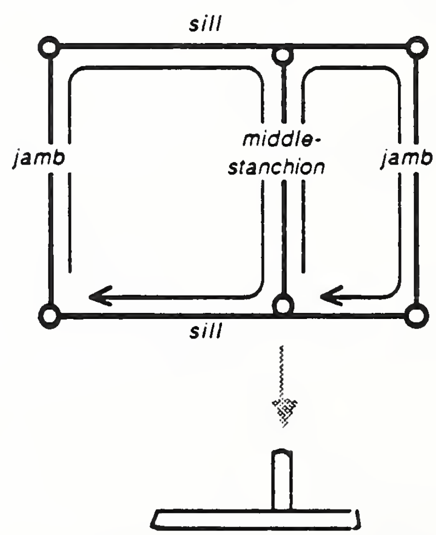


Fig.9. Example of a topology which allows the positioning of vertices on edges where the vertices do not bound or split the edges.

This example is based on the idea that we should be able to define vertices on edges (without splitting the edge), vertices on faces and in volumes, edges on faces and in volumes, and faces in volumes. Placing vertices on edges or faces is required in *idealised* models: models with a simplified geometric shape definition. The vertex may represent the location of an implicitly defined object or feature. For instance a light-switch on a wall or a hole in a floor by means of a vertex on a face. Similarly, a beam supporting a floor could be idealised as an edge on a face.

There is another drawback of conventional topology: we cannot describe the connectivity of topological entities which belong to different levels in a hierarchical structure, nor the connectivity of topological entities which belong to different branches of a hierarchical tree. Conventional topology cannot be combined with tree-like structures.

Thirdly, it would be nice if we could combine the idea of networks with topology, by interpreting topology as a network with nodes, ends and interfaces.

These requirements can be met by a data-structure which is put on top of conventional topology, which we call Meta-topology. The selected name suggest that Meta-topology can be used to define and interpret conventional topology, and this is really what it does.

To understand the basic concepts, we will look first at conventional topology and create a generalised concept. Conventional topology has entities which correlate to the dimensionality of the geometric entities they combine. For instance, a vertex is of dimensionality zero, edge of dimensionality one, face of dimensionality two and volume of dimensionality three. We will call them domain-entities from now on. These entities have many-to-many relationships in a non-manifold topology, which can be normalised by the introduction of so-called boundary-entities. See fig.10. In conventional topology we find also a loop-concept to define outer and inner boundaries of a face (the inner ones are representing holes), or a shell-concept to define the outer and inner shells of a volumetric body (where the inner ones are voids). We will leave them out for the time being, and bring them in again via another principle.

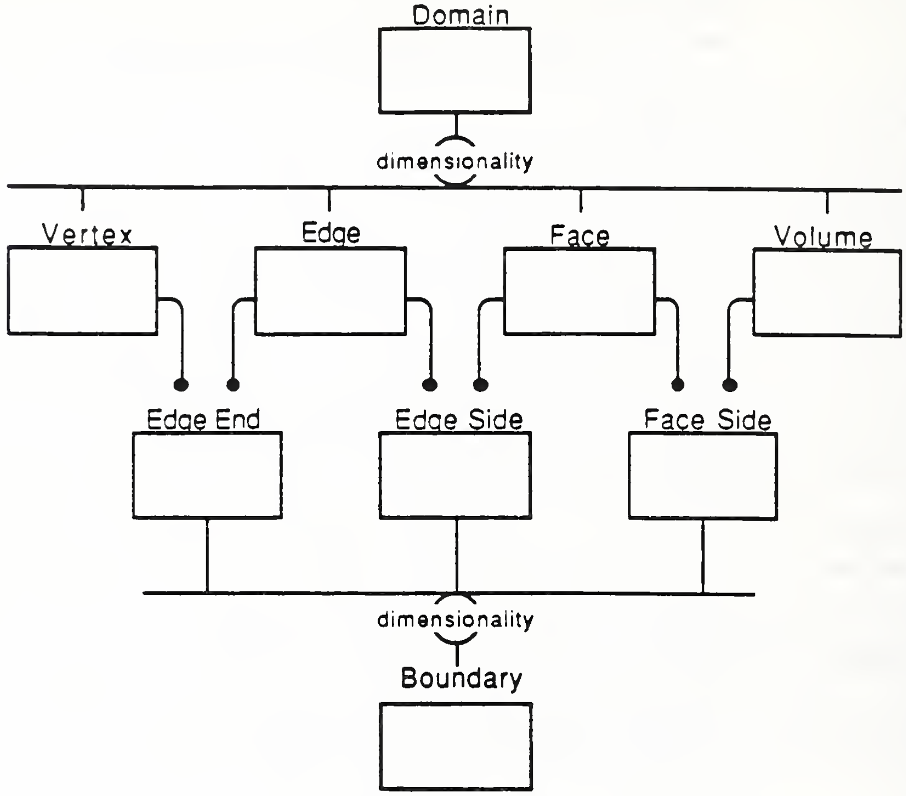


Fig. 10. Conventional topology can be generalised with two entities, called Domain and Boundary. Domain replaces the entities vertex, edge, face and volume. Boundary resolves the many to many relationships between the Domain entities.

The resulting data-structure looks very symmetrical and simple. It can be generalised easily by two entities, called Domain and Boundary. Domain has a dimensionality attribute, and for Boundary a simple rule applies that it will refer always to two Domain entities which differ in dimensionality. Their difference is always one: if the first domain is of dimensionality zero (a vertex), the other is of dimensionality one (an edge).

We will now put the idea of holes and voids in again. This can be done by defining another entity, called Void. This entity connects also two Domain entities, but this time they can be of the same dimensional order. Void adds an inside and outside notion to Domains: for instance, if the Domains are of dimensionality two (meaning faces), one face can be located inside the other. This is identical to the idea of an inner loop located inside an outer loop. But Void can also place a vertex on an edge, on a face, or in a volume; an edge on a face or in a volume; or a face inside a volume.

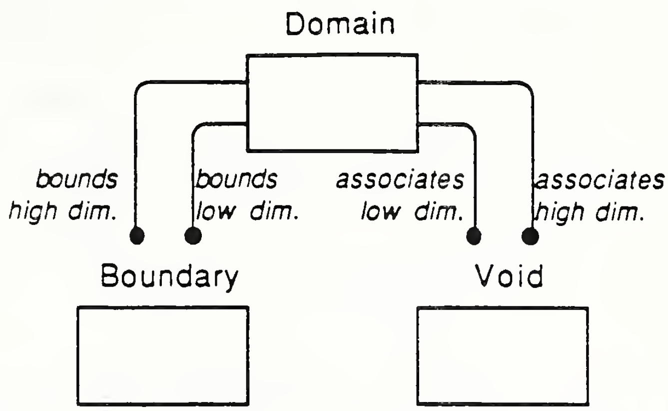


Fig. 11. Generalised Topology: Domain replaces the entities vertex, edge, face and volume. Boundary connects the Domain entities by conventional topological rules. Void is used to associate Domain entities which are located within other Domain entities. They can be of the same or different dimensionality. If they are the same, the can be used to define holes in faces or voids in volumes. If they are different. they can be used to place for instance a vertex on an edge, a face or in a volume; an edge on a face or in a volume; or a face in a volume.

Let us now relate the three new topological concepts to the network discussed before. We see that Domain (representing the conventional topological entities vertex, edge, face and volume) can serve as a node in a network. Both Boundary and Void play the role of interfaces.

We mentioned the problem of hierarchical structures in topology earlier. On the network level this was solved by defining so-called Free Ends of nodes in the network. These free ends refer to ports of the Technical Solution to which the Functional Units belong, and they can be connected again via interfaces on the higher functional network level (see fig. 8.). For this purpose, we have to introduce the concept of Ends in the generalised topology. Ends which are connected via Voids will be called Regions, and Ends which are connected via Boundaries will be called Sides; see table 2. Under certain conditions it is possible to define Free Ends in Topology, being either Regions or Sides. These conditions are embedded as rules in the Express definition of Meta-topology. The Free Ends will always refer to a Port of the Technical Solution to which they belong. If these Ports are connected via a similar functional network or topology on a higher level, the Free Ends will be connected via that route in the hierarchy.

The IDEF1x diagram shows three levels of generalisation/specialisation: on the highest (most general) level we have the general network with nodes, ends and interfaces. One level lower we have Meta-topology, which is a specialisation of the general network in order to make a an interpretation of topology possible. On Meta-topology level we have five entities, subtypes of the network entities, which are the counterparts of the five entities in generalised topology:

Meta-topology level	Generalised Topology level
Node	Domain
Boundary End	Side
Boundary Interface	Boundary
Domain End	Region
Domain Interface	Void

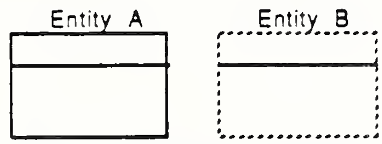
table 2

Below Meta-topology we have generalised topology. The Domain entity has an attribute dimensionality which can be used to specialise into the conventional topological entities Vertex, Edge, Face and Volume.

Various sub-types of the entities mentioned here are included in diagram C.1., in order to make the rules for the interpretation of topology as a network visible. These embedded rules are not defined by means of subtypes in Express, but by means of attributes and WHERE-statements of the super-types.

4. IDEF1x MODEL

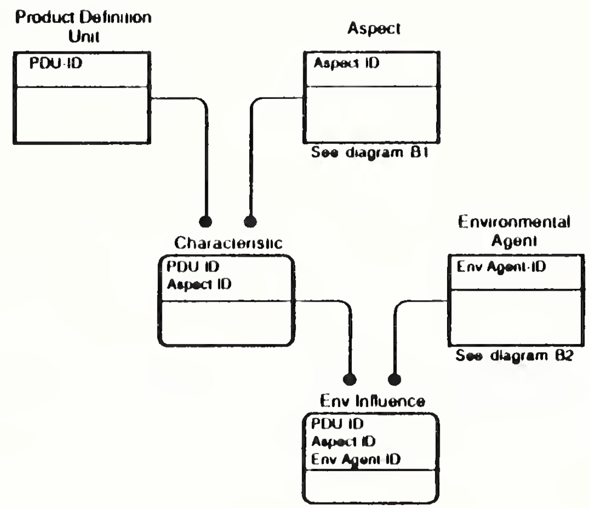
The IDEF1x model is given on the following pages. Entities which are included in the IDEF1x model for explanation, but do not belong to version 1 of the standard, are indicated as dotted boxes. For example Entity B in the diagram below does not belong to the standard.



You will notice also grey horizontal bands in the model, which separate levels of specialisation in AEC. This mainly done for explanation purposes. On the product-type layer, only the product-type Buildings has been indicated. Also some existing ISO standards and classification codes are translated into IDEF1x and added to the model on the non-STEP layer.

Contents:

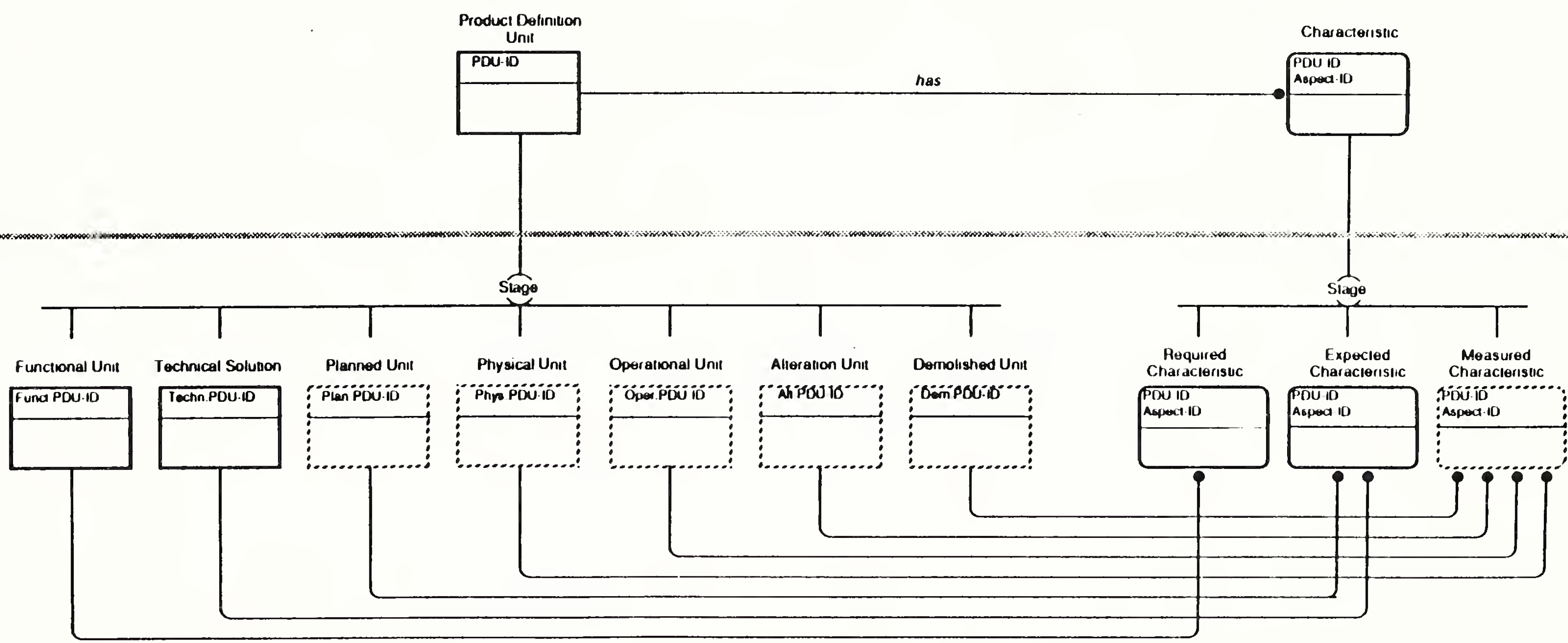
A.0 PDU Characteristics



Author: Wim Gielingh Status: Working Date 14 September 1988 PDU-discrimination:	ISO TC184/SC4/WG1 Reference Model for AEC Product Data A.0 Product Definition Unit Characteristics	Reader: Date:

A.1 PDU Stages

Stage

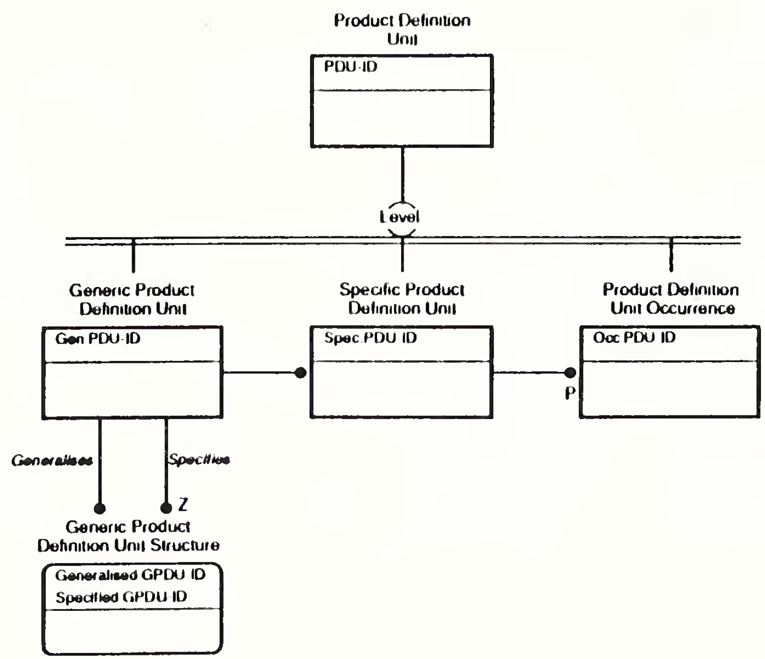


Author: Wim Gielingh Status: Working Date:14 September 1988 PDU-discrimination: Stage	ISO TC184/SC4/WG1 Reference Model for AEC Product Data A.1. Product Definition Stages	Reader: Date:

N28

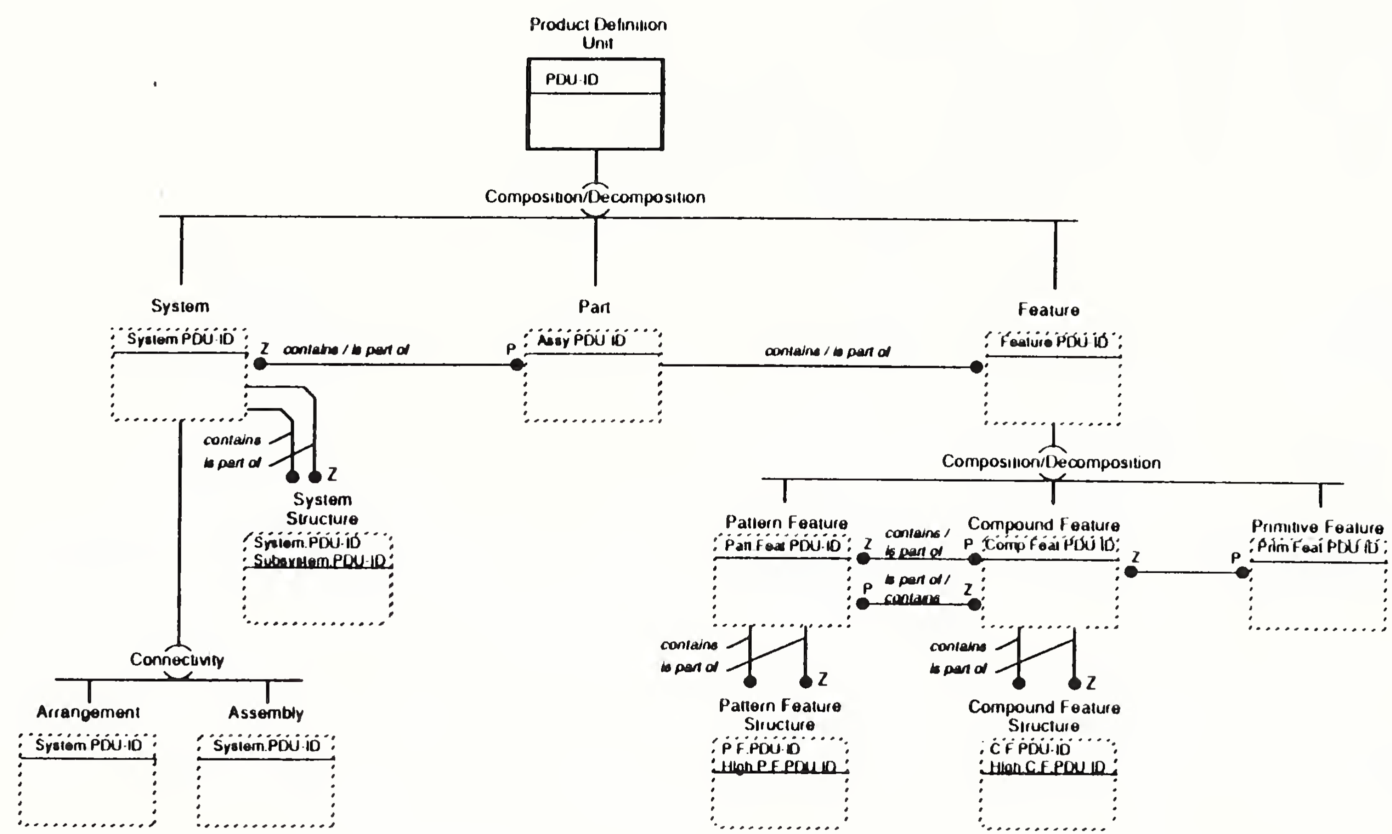
A.2 PDU Levels

Level



Author: Wim Gielingh Status: Working Date: July 1, 1988 PDU-discrimination: Level	ISO TC184/SC4/WG1 Reference Model for AEC Product Data A.2. Product Definition Levels	Reader: Date:

A.3 PDU Decomposition



Author: Wim Gielingh
Status: Working
Date: 23 September, 1988
PDU-discrimination: Aggregation/Decomposition

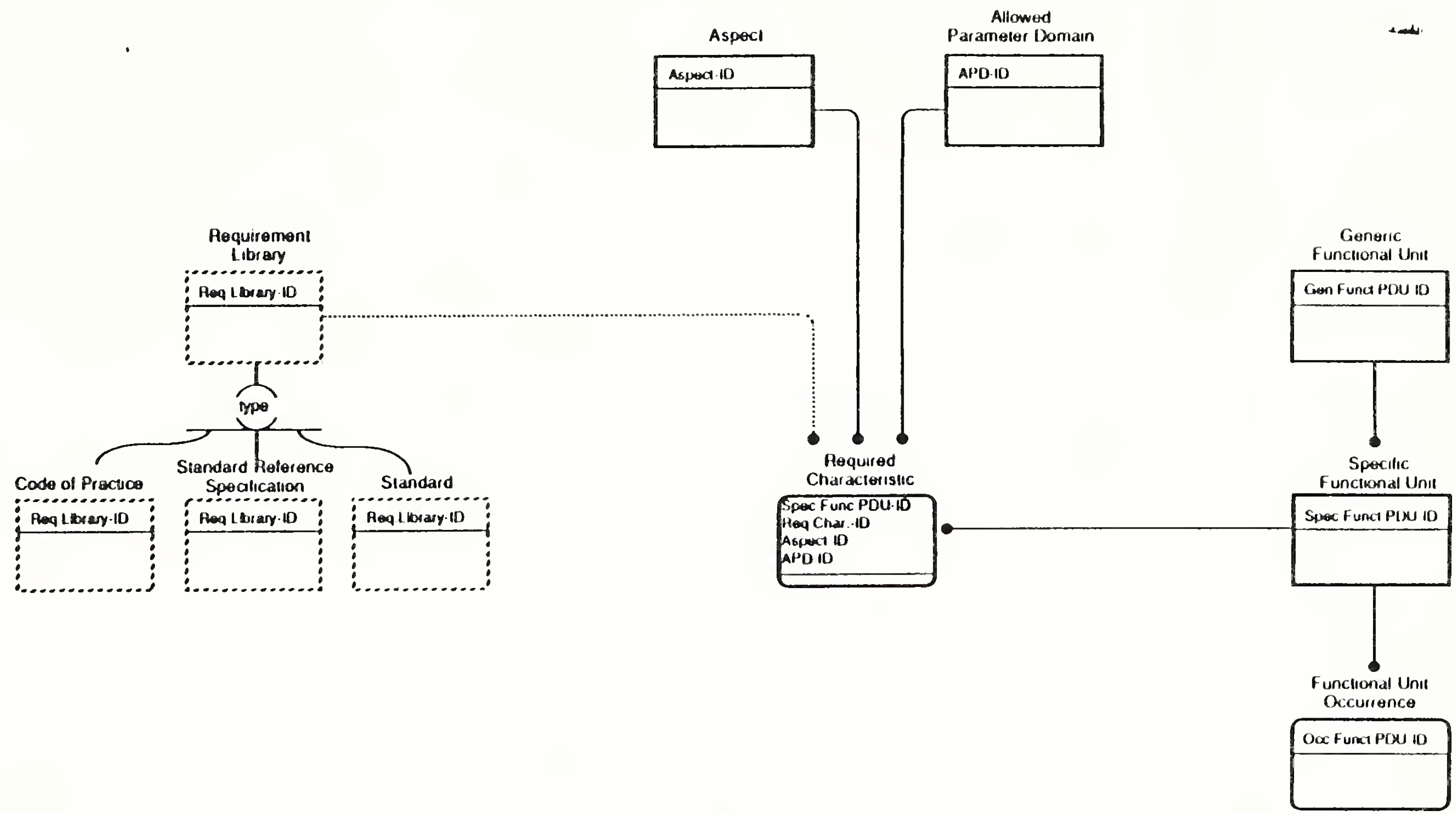
ISO TC184/SC4/WG1
Reference Model for AEC Product Data

A.3. Composition/Decomposition

Reader:
Date:

N288

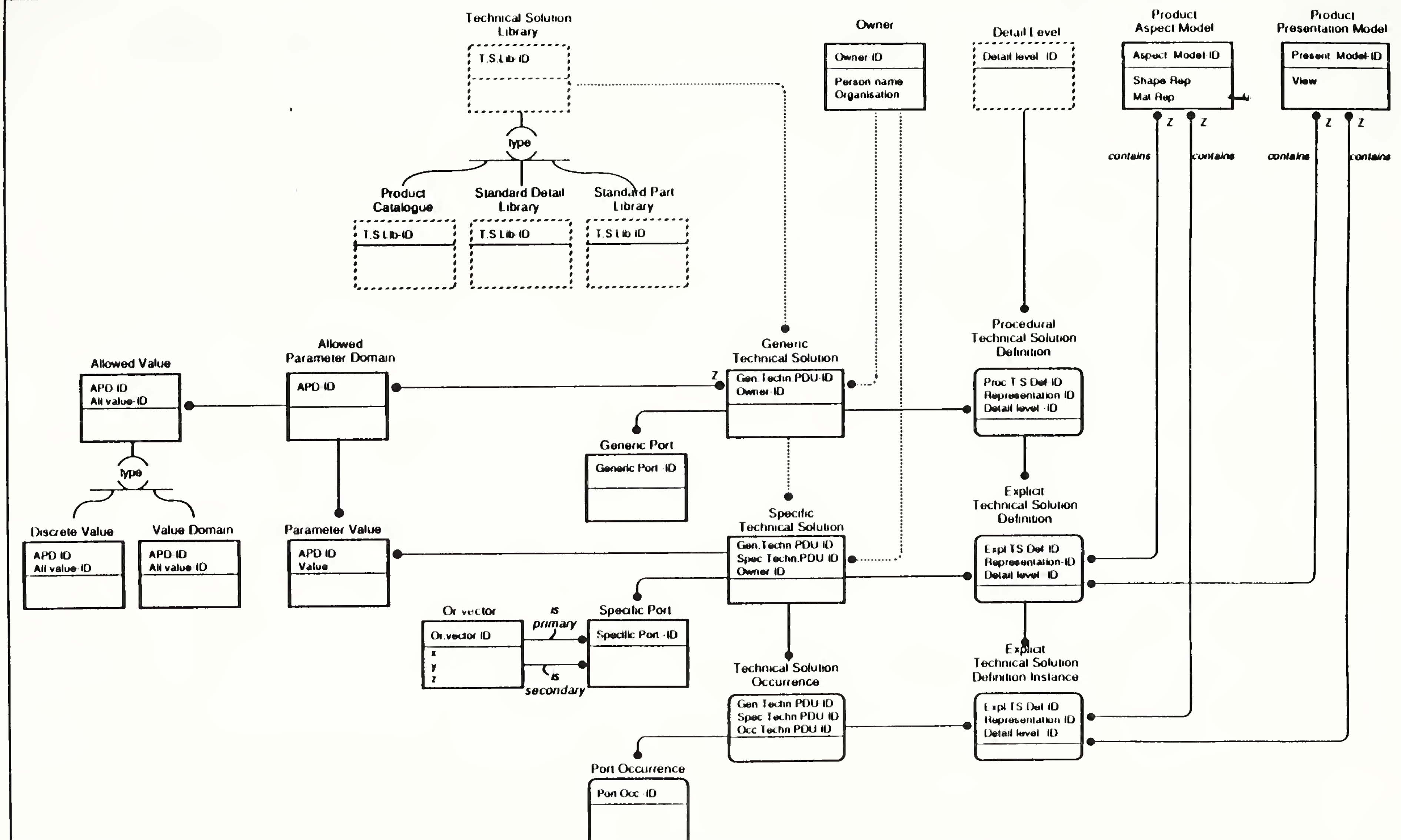
A.4 Requirement definition



Author: Wim Gielingh Status: Working Date: September 16, 1988 PDU-discrimination: Level/Stage	ISO TC184/SC4/WG1 Reference Model for AEC Product Data A.4 Requirement Specification	Reader: Date:

N28

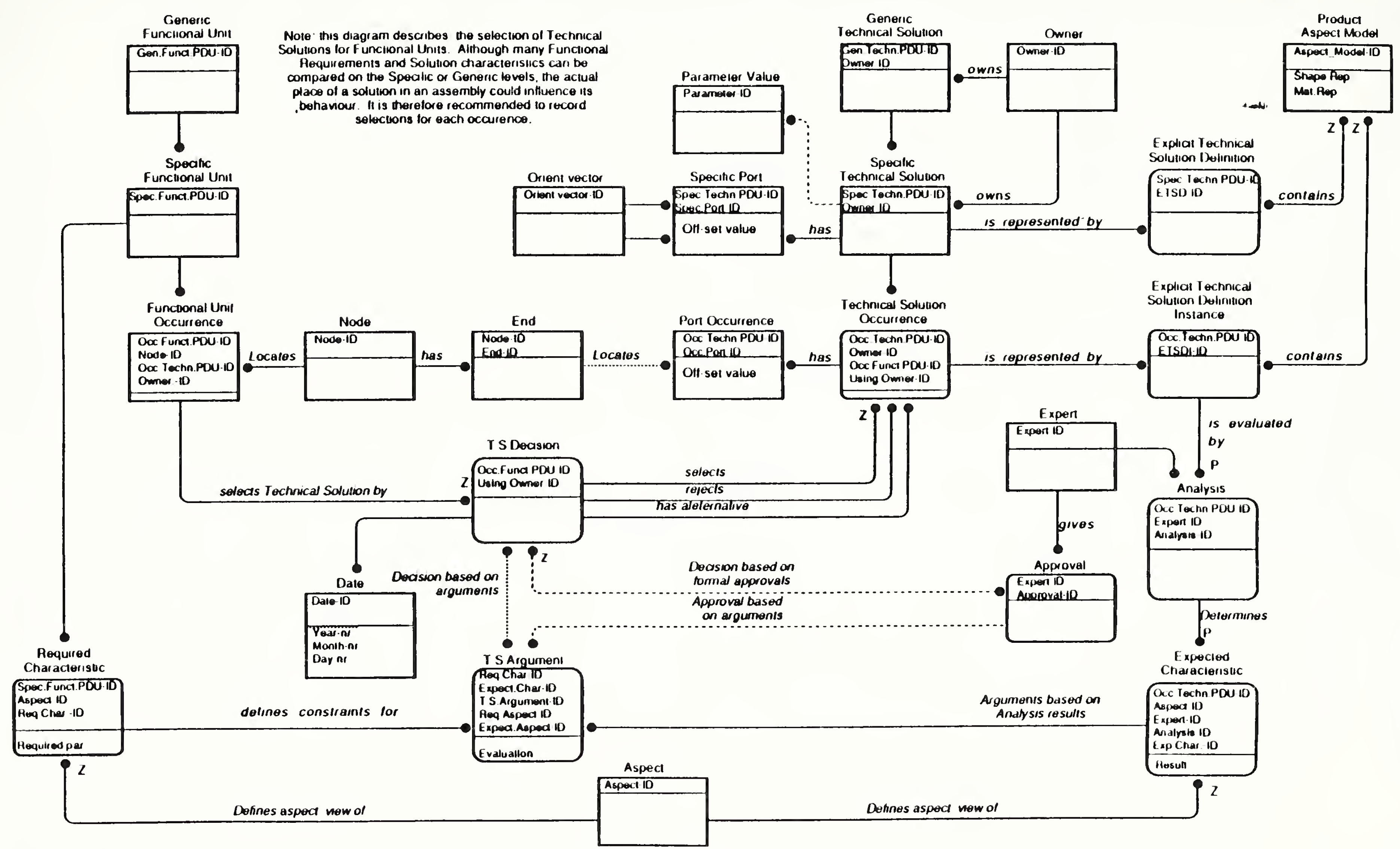
A.5 Design specifications



Author: Wim Gielingh Status: Working Date: September 20, 1988 PDU-discrimination: Level/Stage	ISO TC184/SC4/WG1 Reference Model for AEC Product Data A.5 Design Specifications	Reader: Date:

N28

A.6 Evaluation and Selection of Technical Solutions



Author: Wim Gielingh
Status: Working
Date: 4 October 1988
PDU-discrimination: Level/Stage/Status

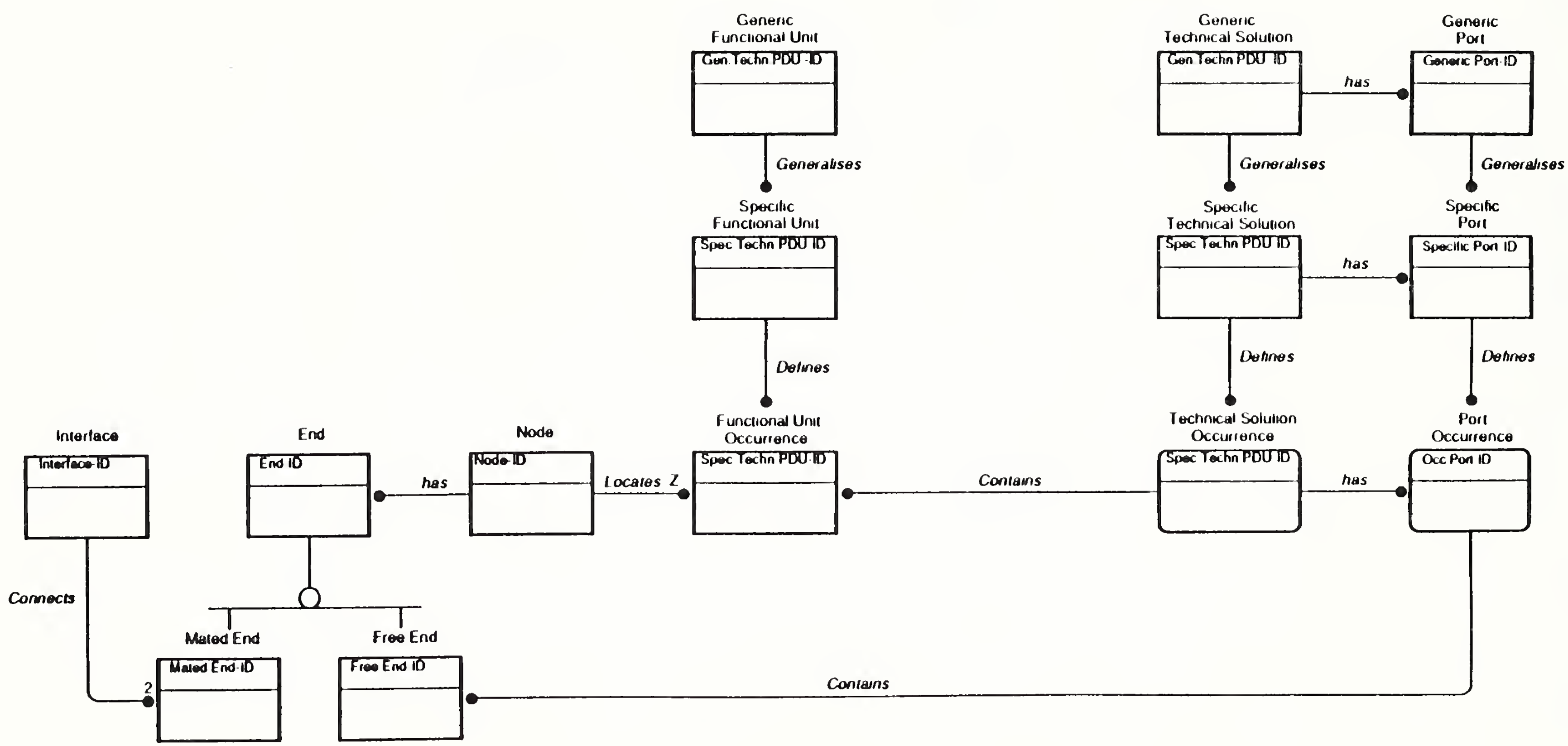
ISO TC184/SC4/WG1
Reference Model for AEC Product Data

A.6 Evaluation and Selection of Technical Solutions

Reader:
Date:

N288

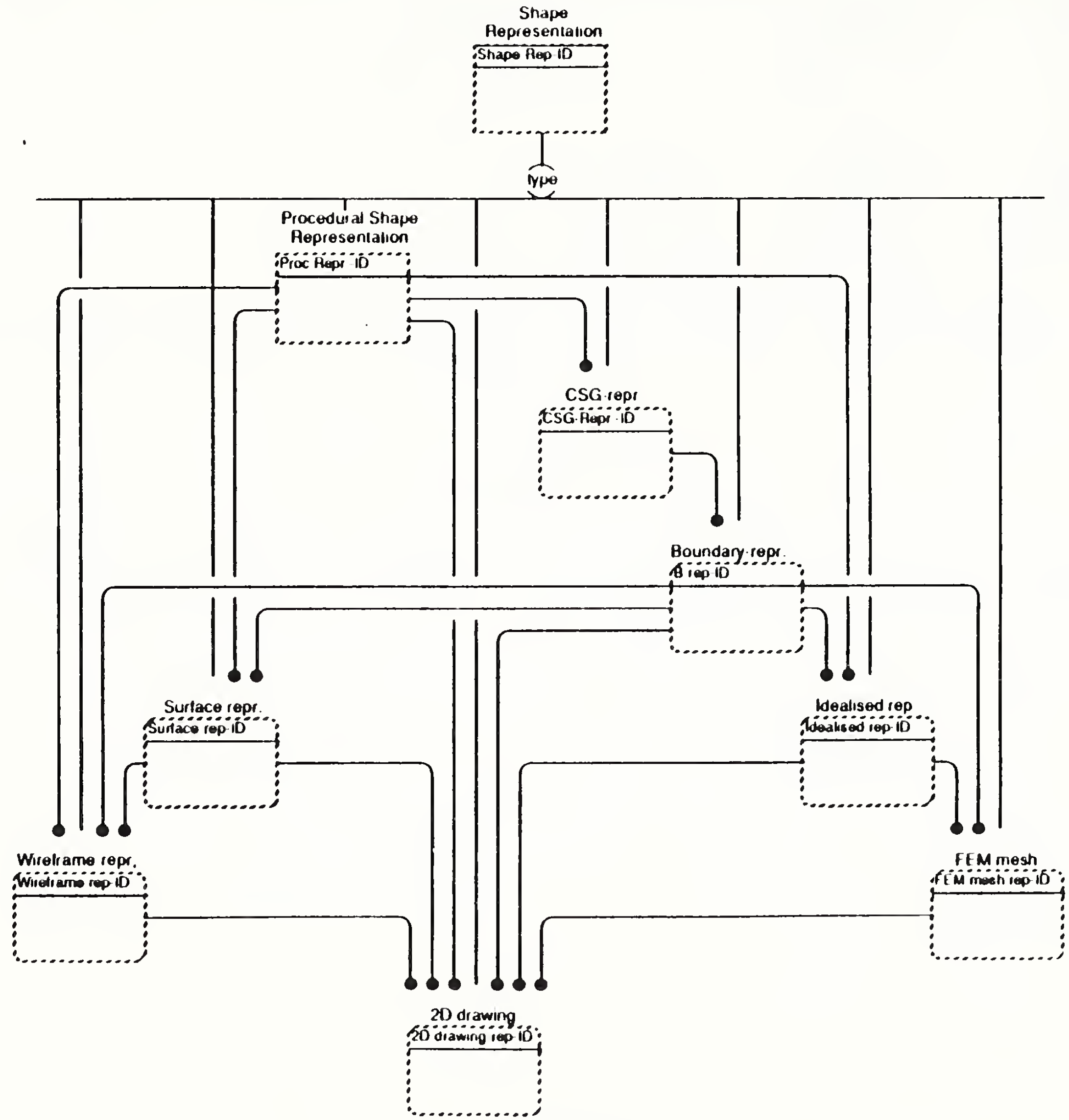
A.7 Decomposition of Technical Solutions



Author: Wim Gielingh Status: Working Date: 27 September 1988 PDU-discrimination: Level/Stage	ISO TC184/SC4/WG1 Reference Model for AEC Product Data A.7 Decomposition of Technical Solutions	Reader: Date:

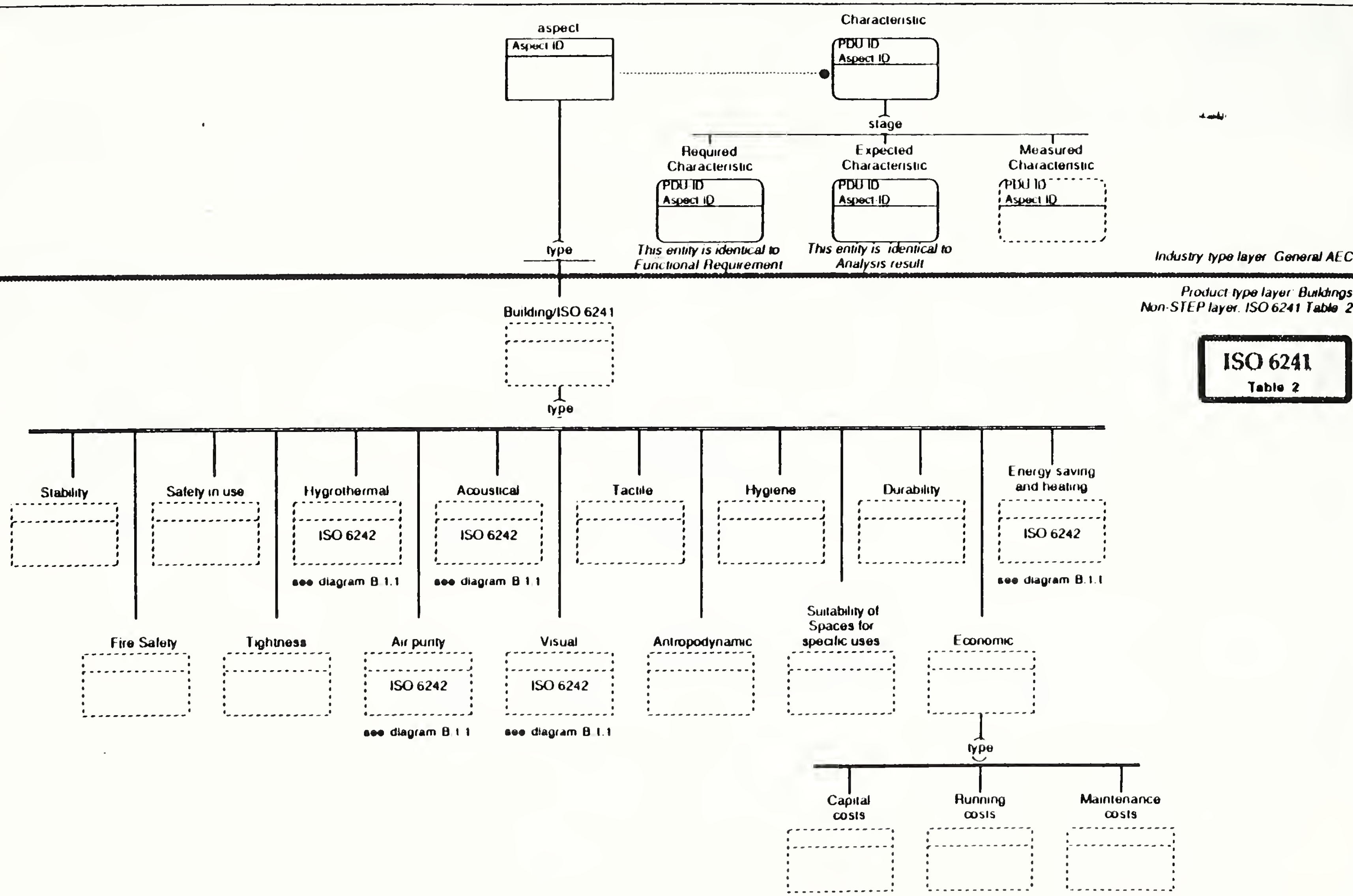
N288

A.8 Shape representations



Author: Wim Gielingh Status: Working Date: September 21, 1988 PDU-discrimination:	ISO TC184/SC4/WG1 Reference Model for AEC Product Data A.8 Representation/Presentation	Reader: Date:

B.1 Aspects



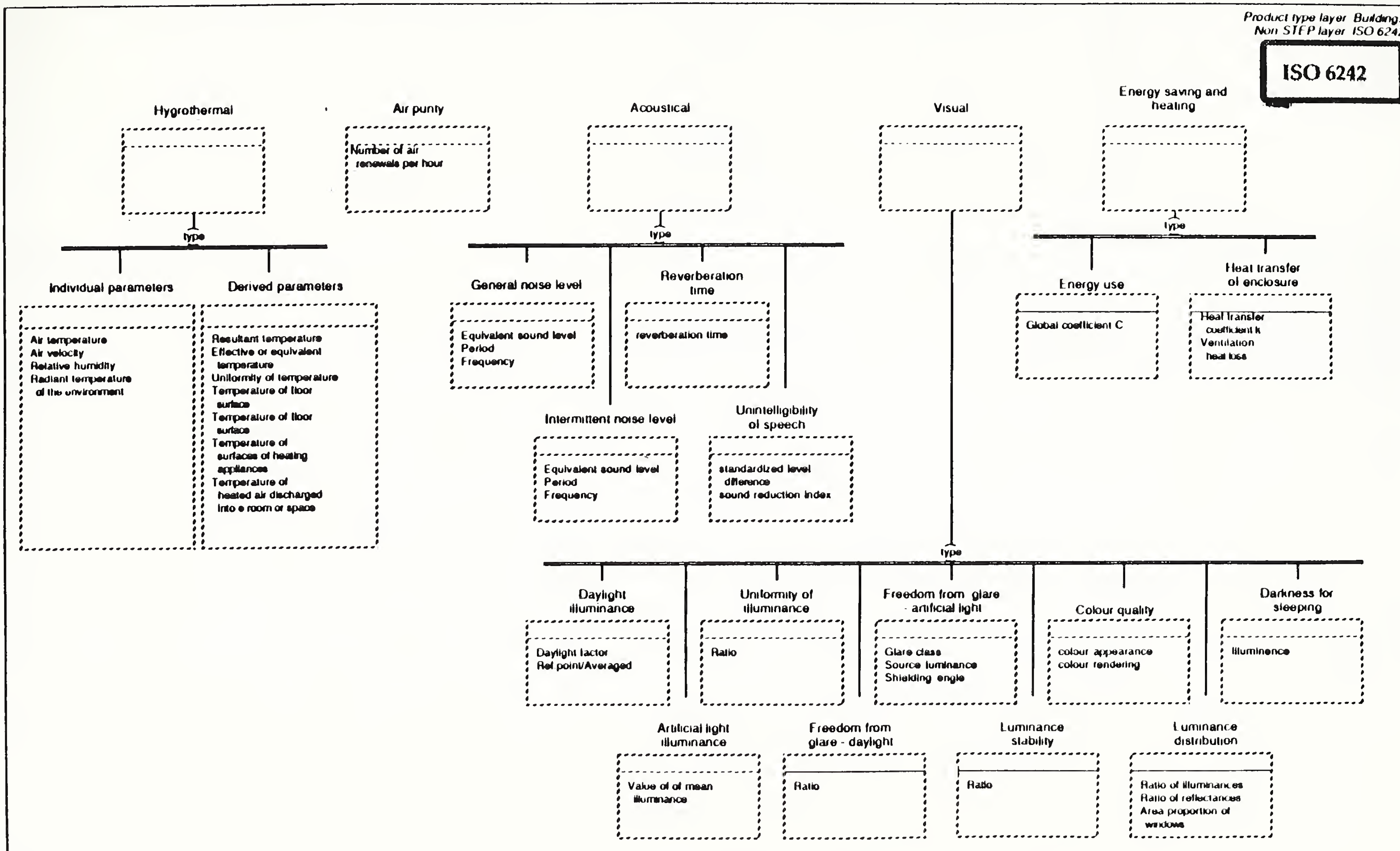
Author: Wim Gielingh
Status: Working
Date: July 4, 1988
PDU-discrimination:

ISO TC184/SC4/WG1
Reference Model for AEC Product Data
ISO 6241
B.1 Aspect

Reader:
Date:

N288

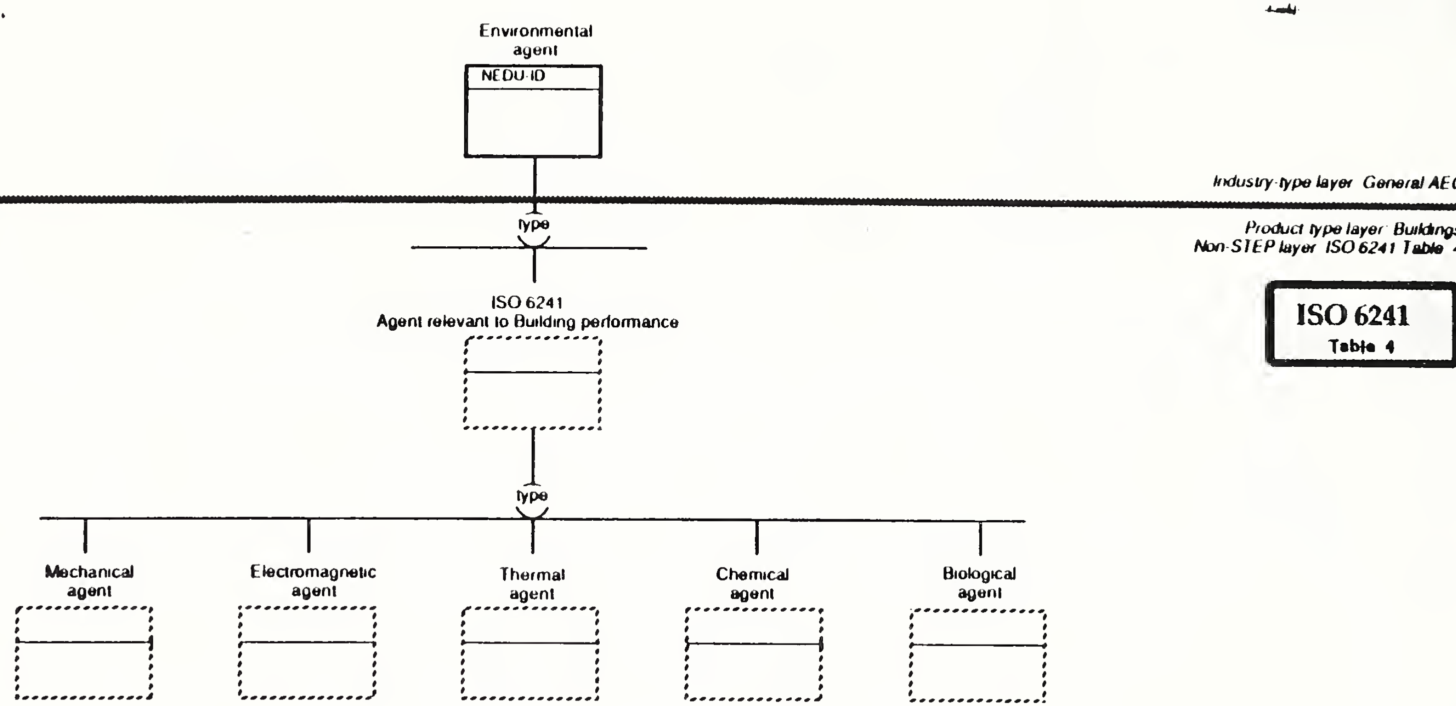
B.1.1 Aspects



Author: Wim Gielingh Status: Working Date: July 4, 1988 PDU-discrimination:	ISO TC184/SC4/WG1 Reference Model for AEC Product Data ISO 6242 B.1.1 Aspect	Reader: Date:

B.2 Agents

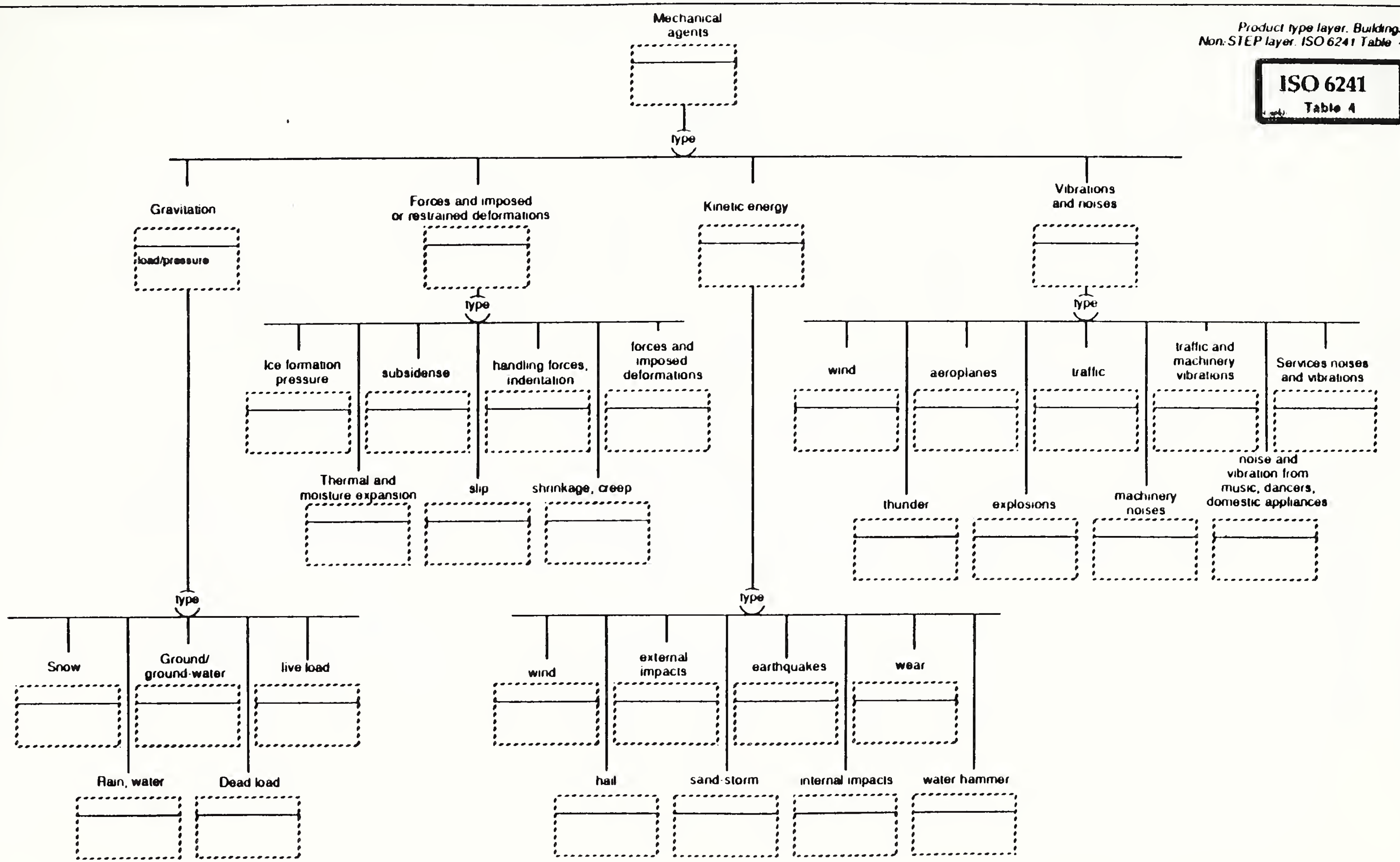
ISO 6241



Author: Wim Gielingh Status: Working Date: September 15, 1988 PDU-discrimination:	ISO TC184/SC4/WG1 Reference Model for AEC Product Data B.2 Agents	Reader: Date:

N288

B.2.1 Mechanical Agents



Author: Wim Gielingh
Status: Working
Date: July 4, 1988
PDU-discrimination:

ISO TC184/SC4/WG1
Reference Model for AEC Product Data
ISO 6241-1984
B.2.1. Mechanical Agents

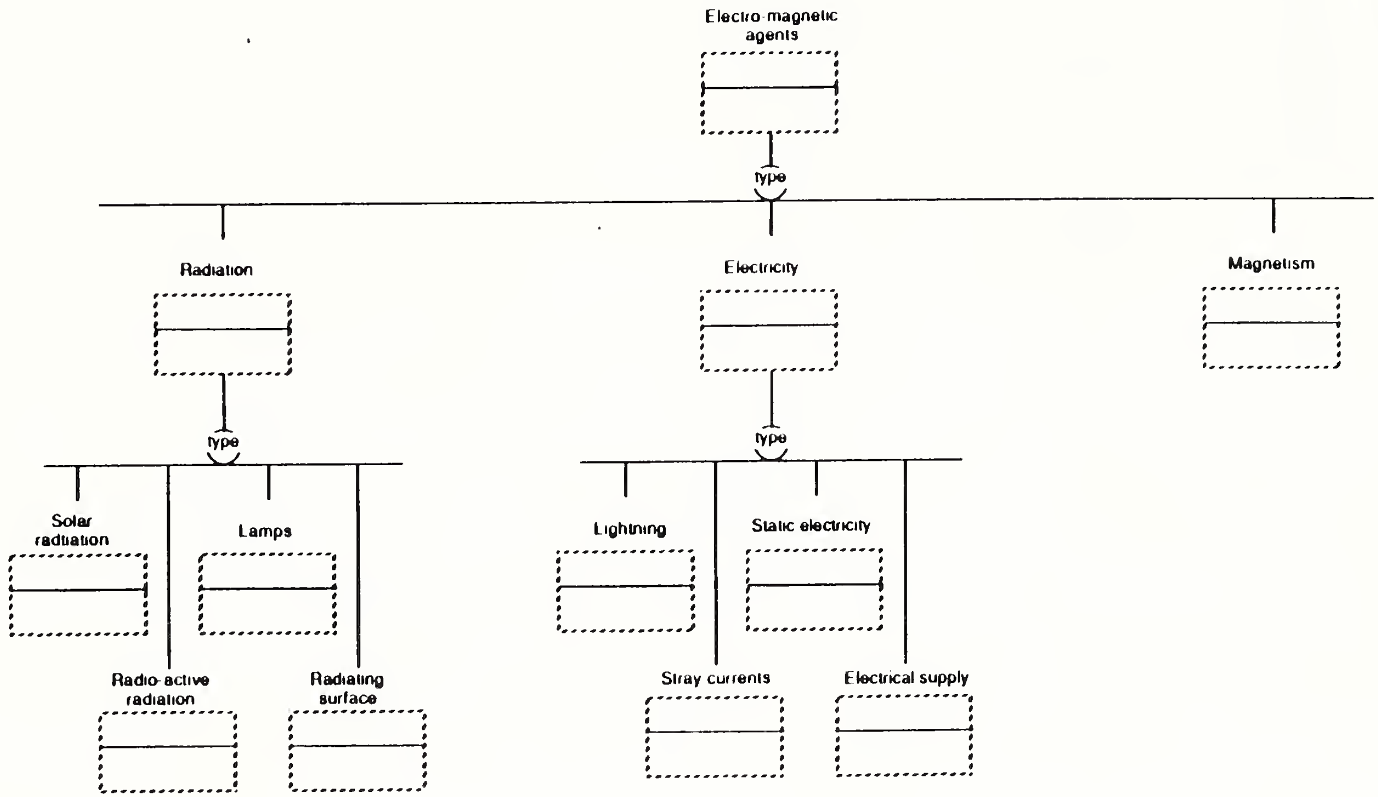
Reader:
Date:

N28

B.2.2 Electro-magnetic agents

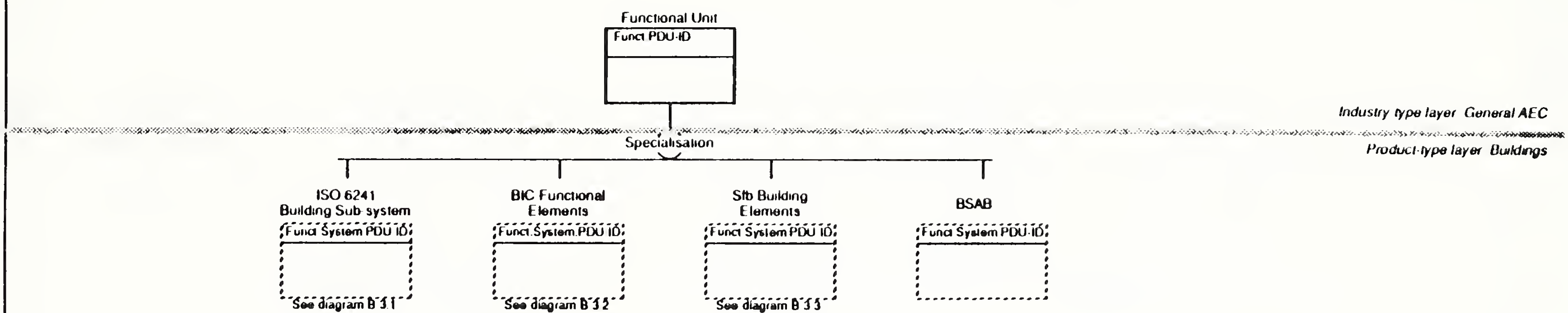
Product type layer: Buildings
Non STEP layer: ISO 6241 Table 4

ISO 6241
Table 4



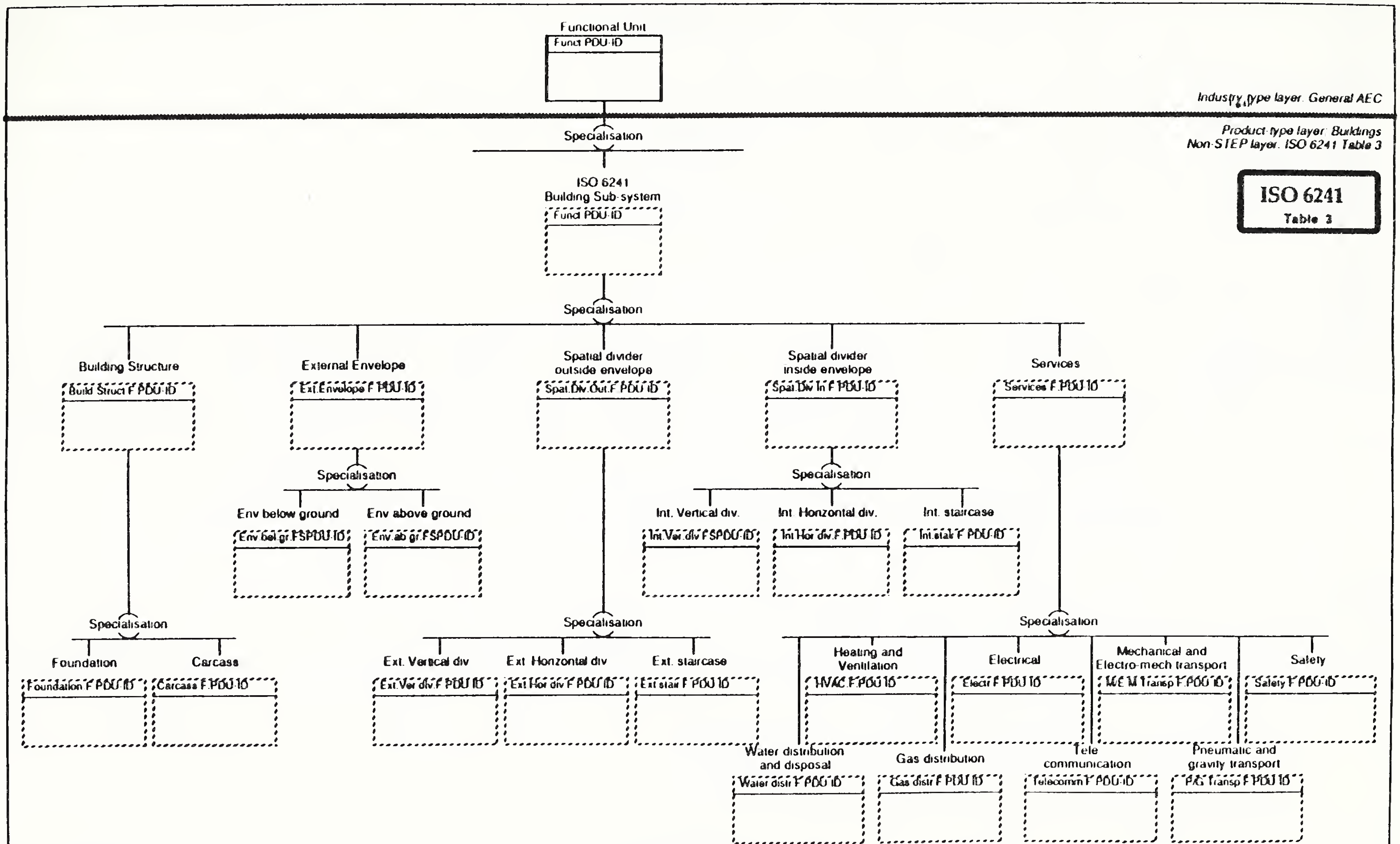
Author: Wim Gielingh Status: Working Date: July 4, 1988 PDU-discrimination:	ISO TC184/SC4/WG1 Reference Model for AEC Product Data ISO 6241-1984 B.2.2. Electro-magnetic agents	Reader: Date:

B.3. Functional units



Author: Wim Gielingh Status: Working Date: 14 September, 1988 PDU-discrimination: Specialisation/Decomposition	ISO TC184/SC4/WG1 Reference Model for AEC Product Data B.3.0. Functional systems	Reader: Date:

B.3.1 Functional units - ISO 6241



Author: Wim Gielingh
Status: Working
Date: 14 September, 1988
PDU-discrimination: Specialisation/Decomposition

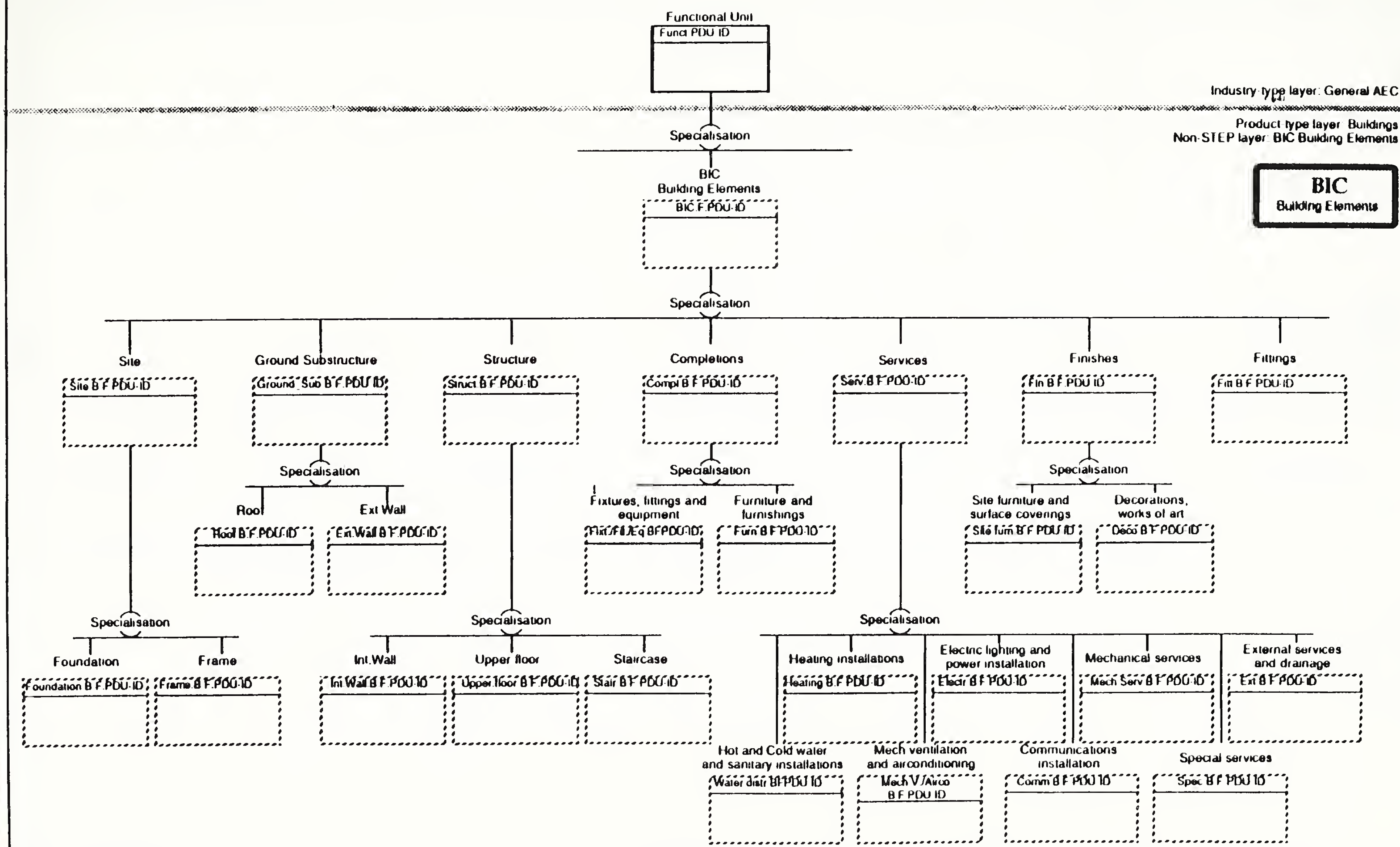
ISO TC184/SC4/WG1
Reference Model for AEC Product Data

B.3.1. Functional Units - ISO 6241 table 3

Reader:
Date:

N288

B.3.2 Functional units - BIC



Author: Wim Gielingh
Status: Working
Date: 14 September, 1988
PDU-discrimination: Specialisation/Decomposition

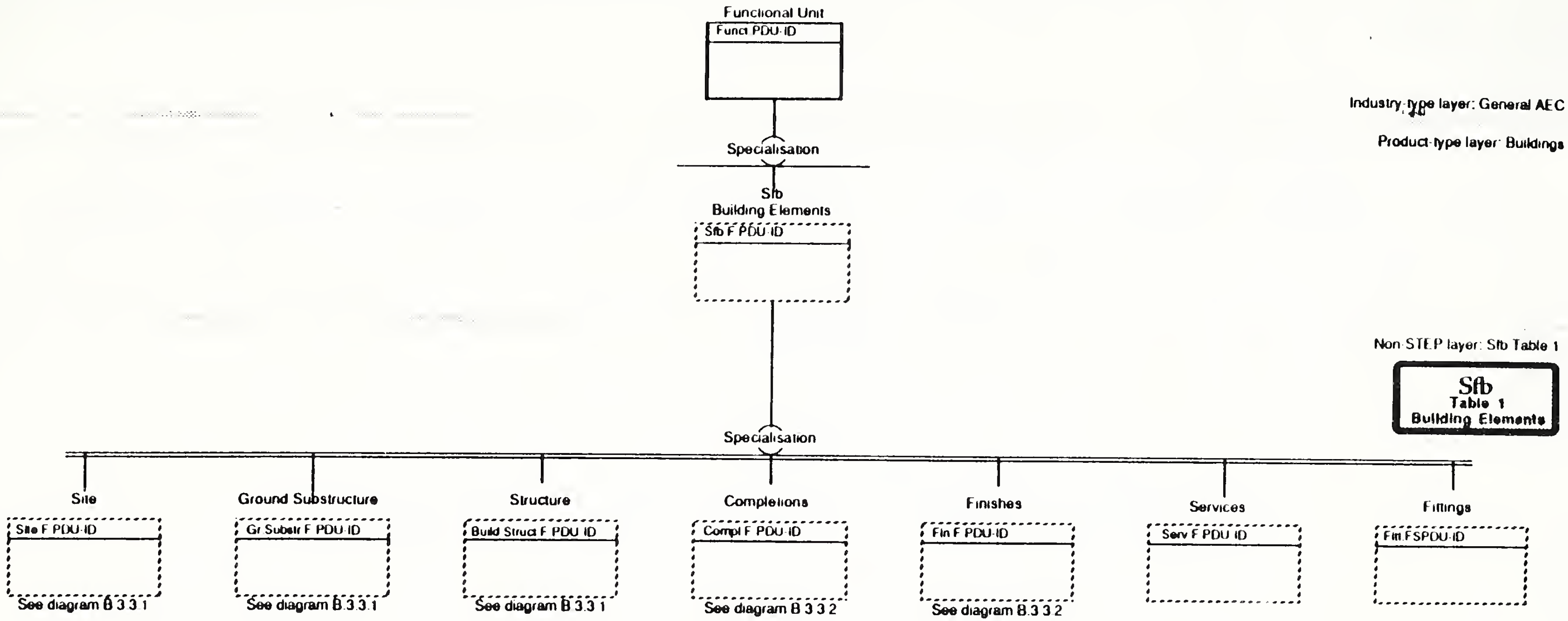
ISO TC184/SC4/WG1
Reference Model for AEC Product Data

B.3.2. BIC Functional Building Systems

Reader:
Date:

N28

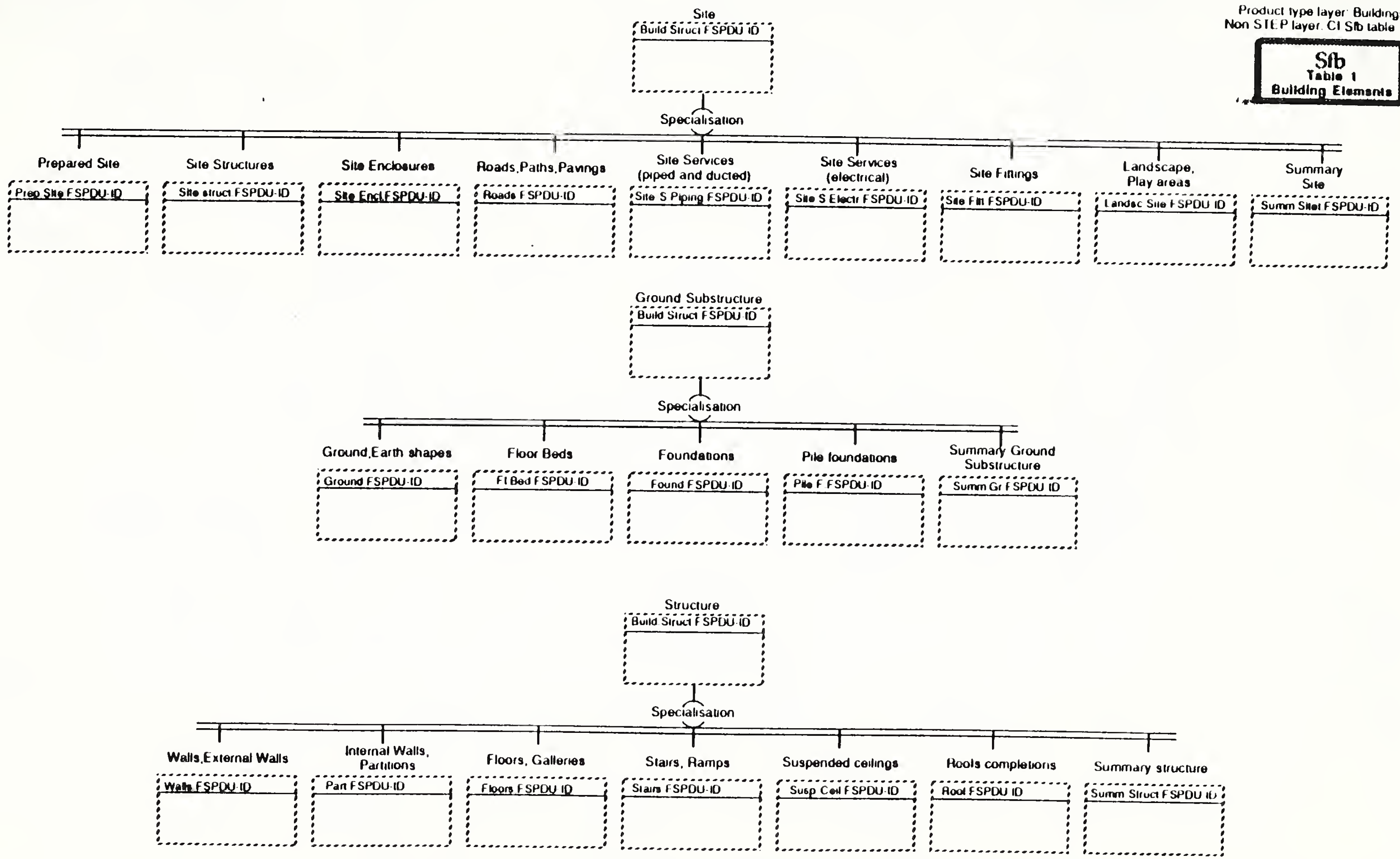
B.3.3 Functional units - Sfb



Author: Wim Gielingh Status: Working Date: 14 September, 1988 PDU-discrimination: Specialisation/Generalisation	ISO TC184/SC4/WG1 Reference Model for AEC Product Data B.3.3. Sfb Functional Elements	Reader: Date:

N288

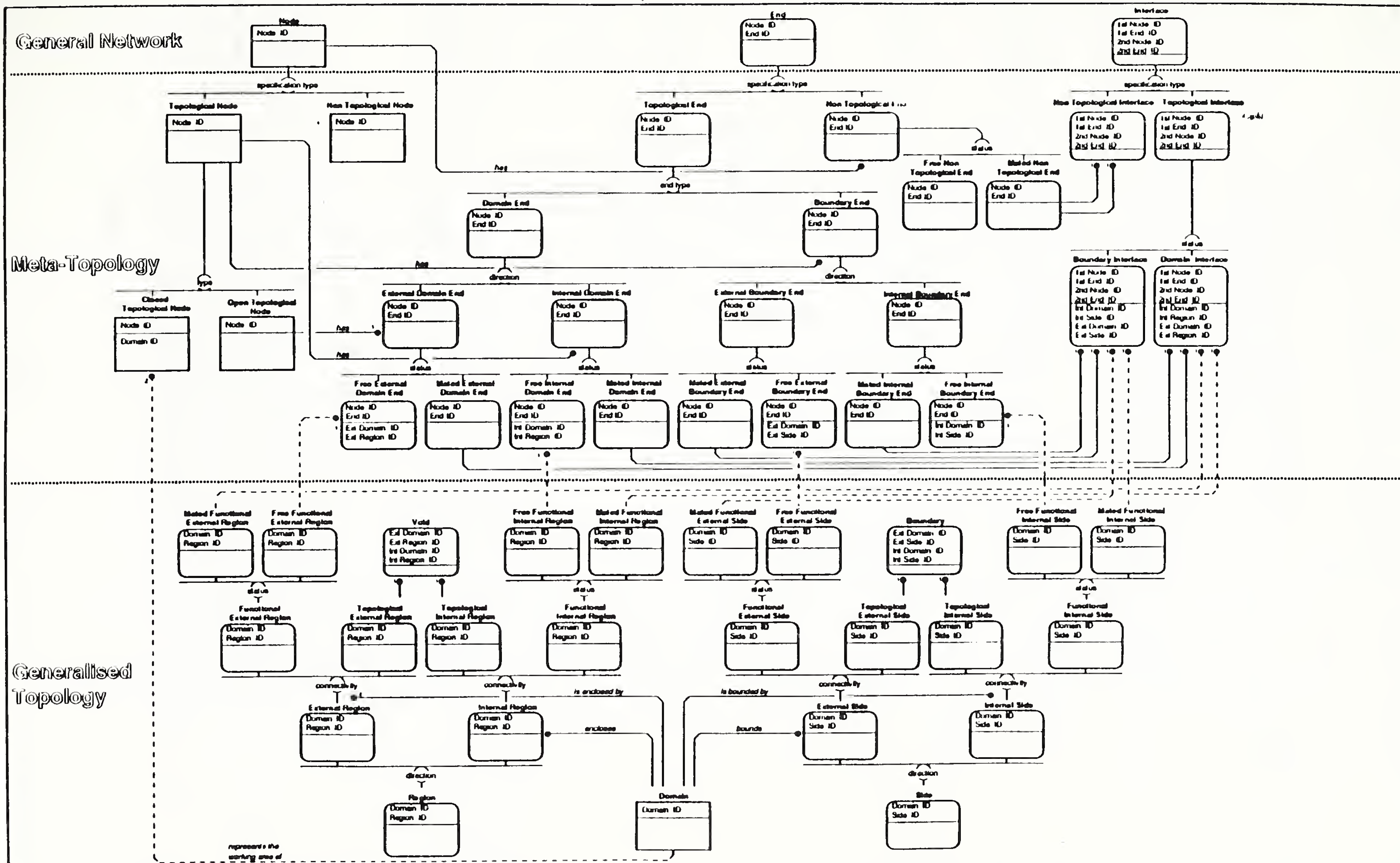
B.3.3.1 Functional units - Sfb



Author: Wim Gielingh Status: Working Date: 14 September, 1988 PDU-discrimination: Specialisation/Generalisation	ISO TC184/SC4/WG1 Reference Model for AEC Product Data B.3.3.1. Sfb Functional Elements	Reader: Date:

C.1 Meta-topology

N288



Author: Peter Willems Date: October 3, 1988 Scope: Status: Working	ISO TC184/SC4/WG1 General Reference Model for AEC Product Data C.1. Meta-Topology	Reader: Date:

5. ENTITY POOL

5.1 High level entities

Product Definition Unit

Product or any subsystem, part or feature of a product, interesting enough to record information about.

Characteristic

Characteristic of a Product Definition Unit. A characteristic may apply to any aspect, and can be required, expected or measured.

Aspect

View of a product definition unit, used to define its characteristics. Examples of aspects are strength, safety, durability and costs.

Environmental Influence

Influence of the environment on the characteristics of a Product Definition Unit

Environmental Agent

External factor which influences the characteristics of a product. Examples of Environmental Agents are wind, rain, loads, earthquakes, insects, acids, etc.

5.2. PDU stages

Functional_Unit

Describes function, requirements and constraints of a product, or any physical or non-physical part of a product. Functional Units may be specified by classification-codes such as Sfb and CSI.

Technical_Solution

Describes a technical solution for a functional unit. A Technical Solution can be an Assembly, a Part, a Feature, a connection, a joint, etc. Generic Technical Solutions can be stored in a library. Each Technical Solution may contain a cluster of Functional Units which specify its components and their functional structure. This decomposition is made explicit on the Specific level, and unique on the Occurrence level.

Planned Unit

Describes how a technical solution can be realised, using available production techniques. Or, in other words, the Planned Unit describes the Product Definition Unit as it is planned for production.

Physical Unit

Describes the Product Definition Unit as it is realised (manufactured or built).

Operational Unit

Describes the product Definition Unit in operation.

Alteration Unit

Describes the Product Definition Unit as it is changed (modified, renovated, maintained, upgraded). A change can be required if:

a. The characteristics of the Operational PDU are changed and do not meet the required characteristics (due to wear, aging, etc.).

b. The required characteristics are changed (new function, new requirements, etc.)

Demolition Unit

Describes the Product Definition Unit as it is demolished.

5.3 PDU levels

Generic PDU

A Generic Product Definition Unit is a parametrically defined PDU. It is logically stored in a library. This can be a public library of standard requirements, parts and products, but also a library owned by the company or the designer. Parameters may have constraints on their values; the Generic PDU refers therefore for each parameter to the entity Allowed Parameter Domain. This entitity defines ranges of allowed values, being either discrete values or value domains with an upper and a lower boundary. Generic PDU's may form a tree-like structure, if some but not all parameters are defined. For instance, a PDU which has originally six parameters may have two of them defined, leaving four unknown parameters. It is still a Generic PDU, though less generic then the original one. Not all Generic PDU's are necessarily parameterised: they can be without parameters. In such a case it is similar to a Specific Product Definition Unit. This possibility allows users to store all their product information in (local) libraries, so that they can be re-used. It allows users also to generalise and parameterise them in a later stage.

Specific PDU

A fully defined Product Definition Unit. A Specific PDU refers to one Generic PDU, but has all its parameters defined. The parameter values should be within the allowed parameter domains. The Express-version of GARM has a function In_Domain which checks this constraint. The owner of a Specific PDU is the one who has defined the parameter values. This may be a different one then the owner of the Generic PDU.

Occurrence PDU

Describes each occurence (or instance) of a PDU. To avoid redundancy in the data, almost all relevant information about the PDU is available on the Generic and/or Specific level. The occurrence level contains only place and orientation of the PDU. However, the Occurrence PDU plays also an important role for evaluation, testing and measurement: each instance of a specific PDU may have different behaviour.

5.4 PDU aggregation and decomposition

System

A system is an aggregation of parts, generally with a specific function. Systems can be divided into two main groups: assemblies and arrangements, depending of their connectivity. Assemblies are regarded as systems which contain physically connected parts or subsystems, Arrangements are systems which have physically non-connected parts or subsystems. The difference between both sub-types of systems is not yet relevant in GARM; they are currently included for explanation only. The idea "system" is usually very much related to a function. In contrast, the idea "assembly" is more related to the production process. Systems can be described geometrically by means of idealised models (wireframes, surface models, non-manifolds); if they are described as a solid with one outer shell, the outer shell represents the envelope of the system.

Part

A part is a component of a product which is manufactured from one (piece of) original material. The idea of a part is very much related to the production process. Moulds, sheet metal products, profiles or milled components can all be regarded as parts, in contrast with assemblies and/or arrangements. It is therefore possible that what is considered as a part in the design process, can be considered as an assembly in the planning and manufacturing stage. Parts can be described by means of idealised models (wireframes, surface models) but also as solids with one outer shell, where the outer shell represents the boundary of the material.

Feature

A feature is considered as a region of interest on the surface of a part. Features may form a hierarchy with, on the highest level, pattern features. Pattern Features may be placed in a hierarchy of sub-patterns, but end with compound features. Compound features may also be placed in a hierarchy of sub-compound features, which end with so-called primitive features. The latter type of features cannot be decomposed further.

Pattern feature

See feature

Compound feature

See feature

Primitive feature

See feature

5.5 PDU levels and stages combined

This section addresses now only Functional Units and Technical Solutions. The model will be extended with other stages in future versions.

Generic.Functional_Unit

Functional Unit described in general terms, such as used in regulations, standards, etc. They can be regarded as parametrically defined Units. If all parameters are known, it becomes a Specific Functional Unit.

Specific.Functional_Unit

Completely defined Functional Unit. The Specific Functional Unit is used to remove redundant information from the model: one Specific Functional Unit can be referred to by several Occurrences.

Occurrence.Functional_Unit

Instance of a Specific Functional Unit.

Generic.Technical_Solution

A Generic Technical Solution is a parametrically defined Technical Solution. It is logically stored in a library. This can be a public library of standard parts and products, but also a library owned by the company or the designer. Each Generic Technical Solution has an

Owner: the person and/or organisation who created it. Parameters may have constraints on their values; the Generic Technical Solution refers therefore for each parameter to the entity Allowed Parameter Domain. This entitity defines ranges of allowed values, being either discrete values or value domains with an upper and a lower boundary. Generic Technical Solutions may form a tree-like structure, if some but not all parameters are defined. For instance, a Technical Solution which has originally six parameters may have two of them defined, leaving four unknown parameters. It is still a Generic Technical Solution, though less generic then the original one. Not all Generic Technical Solutions are necessarily parameterised: they can be without parameters. In such a case it is similar to a Specific Technical Solution. This possibility allows designers to store all their solutions in (local) libraries, so that they can be re-used. It allows designers also to generalise and parameterise them in a later stage.

Specific.Technical_Solution

A fully defined Technical Solution. A Specific Technical Solution refers to one Generic Technical Solution, but has all its parameters defined. The parameter values should be within the allowed parameter domains. The Express-version of GARM has a function In_Domain which checks this constraint. The owner of a Specific Technical Solution is the one who has defined the parameter values. This may be a different one then the owner of the Generic Technical Solution.

Occurrence.Technical.Solution

This a Technical Solution which is used to fullfill the needs of an Occurrence Functional Unit. It represents each instance of a Specific Technical Solution. To avoid redundancy in the data, all relevant information about the Technical Solution is available on the Generic and/or Specific level. The occurrence level contains only place and orientation of the Technical Solution, through reference to its using Occurrence Functional Unit. However, the Occurrence Technical Solution plays also an important role for the evaluation of the Technical Solution: its place in an environment (system, assembly) may influence its behaviour. A Specific Technical Solution may be useful on one place, but useless on another place. The analysis and analysis results (expected characteristics) are therefore recorded on the Occurrence level. An Occurrence Technical Solution has two owners: the one who created it (inherited from the Specific Technical Solution), and the one who uses it (inherited from the Occurrence Functional Unit). This double ownership is important for the evaluation of solutions and the recording of decisions: it is the using owner who decides which status an Occurrence Technical Solution has. This status (being Rejected or Selected) is recorded by the entity Technical Solution Decision. If an Occurrence Technical Solution does not refer to a T.S.Decision, its status is regarded as "alternative".

Library

Collection of product definition data, centered at Product Definition Units, which is project-independent. Libraries can be owned by individual companies, groups or associations of companies, governmental bodies, departments in a company, experts, etc. Libraries can be publically accessible or protected.

Functional_Unit.Library

Subtype of Library. A Library containing Generic Functional Units. Examples are (building) regulations, standard reference specifications, etc.

Technical_Solution.Library

Subtype of Library. A Library containing Generic Technical Solutions. Examples are product catalogues, standard part libraries, standard detail libraries, etc.

Allowed_Parameter_Domain

Defines the allowed value of a Parameter. The domain may consist of a range of discrete values or value domains with an upper and lower boundary.

Discrete_Value

Discrete allowed value of a parameter. See also Allowed_Parameter_Domain.

Value_Domain

Allowed value-domain of a parameter, defined by an upper and a lower boundary. See also Allowed_Parameter_Domain.

5.6 Hierarchical Product Composition (diagram A.7)

Port

A port allows a Technical Solution to be connected with other Technical Solutions, which are not yet known. The actual connection of Technical Solution Occurrences is defined by the functional network of Functional Unit Occurences. This is done via the entities Node, End and Interface. It is therefore essential that each Port Occurence coincides with an End of the using Functional Unit Occurence.

A port on a high level may decompose in a set of Ends on a lower level in the hierarchical tree. This allows us to describe relations between functional units which belong to different branches in the hierarchical tree.

Generic Port

Port defined on a generic level. The number of ports may be parameterised.

Specific Port

Port defined on a specific level. The number of ports, their position and orientation are known. Ech port has two orientation-vectors and an attribute Off-set value, which defines the location of the origin of the port. This origin is positioned on the primary orientation-vector of the port.

Port Occurrence

The Port Occurence refers to a specific port and through that all the relevant information contained by the specific port, such as the orientation vectors and the off-set value. In addition, a Port Occurrence refers to an End of the Functional Unit Occerrence which uses the Technical Solution. A Port Occurrence may copy the decomposition into Free Ends described on the Generic and Specific levels, so that every unique interface between components of the Technical Solution with components of other Technical Solutions can be described.

Node

Functional Units are placed in a network which describes the functional relations between them. The refer to a Node in this network. Each Node has Ends and Ends are connected through Interfaces.

End

An End of a Node, used to describe Interfaces between Nodes in a network. There are two types of Ends: mated and free ends.

Mated End

A Mated End is connected with another Mated End through an Interface on the same level of hierarchical decomposition of the product.

Free End

A Free End is not connected with another End on the same level of hierarchical decomposition of the product. They are used to define interfaces on a higher level of the decomposition. A Free End refers always to a Port of the Technical Solution which contains the functional network.

Interface

An interface connects two Mated Ends of Nodes in the functional network.

Group

A Group is collection of Functional Units which are connected through one functional network. Functional Units which refer to Nodes in different networks do not belong to the same group. The Group entity is usefull for implementation purposes and will have more significant meaning if Kinematic Constraints are included in the model. A proposal for Kinimatic Constraints is included in version 2 and 3 of GARM, but is left out for version 1.0 of STEP.

Procedural Technical Solution Definition

Parametric definition of a Generic Technical Solution. The parametric definition is assumed to be of a procedural form. Each Generic Technical Solution may have more than one procedural definition, depending of the type of representation and the level of detail given. STEP version 1 does not yet handle parametric product definitions, but this entity must be regarded as a logical placeholder for such definitions. Representations are split into shape and material representations. From these, only shape-representation has been worked out (see Express definition). Detail level is used to distinguish between global and detailed product definitions.

Explicit Technical Solution Definition

Explicit (non-parametric) definition of a Specific Technical Solution, usually obtained by evaluation of a parametric/procedural definition.

Explicit Technical Solution Definition Instance

For certain applications it is essential to have each occurrence of a Technical Solution represented. The explicit definition of each occurrence or instance is indicated by this entity.

Representation

Two classes of representations are currently defined in GARM: shape-representations and material-representations. Only the former one is specified.

Shape representation

The shape of a product can be represented by various computerised models. The following subtypes of this entity are recognised: procedural shape definition, CSG-solid rep., Exact Boundary-rep., Facetted Boundary-rep., FEM-rep, Idealised-rep, Wireframe-rep., and Surface-rep.

Detail level

This entity is included to indicate the level of detail in a product description. In this version of GARM only the detail levels of shape and size can be indicated. The detail level is indicated by giving the treshold which is used for the shape definition. This treshold should be given in the unit used for the geometric definition, and should be a multiple of 10 or 0.1 of this unit. For instance, values such as 0.01 mm, 0.1 mm, 1 mm, 10 mm, 100 mm and 1m are examples of valid tresholds. If the treshold is for instance 10 mm, no details smaller than 10 mm are supposed to be recorded. The entity detail level replaces the idea of scale in a technical drawing; the scale is used there to reveil or hide detail information.

Owner

Person and/or organisation who "owns" a Technical Solution. Ownership reflects responsibility. The owner of a Specific Technical Solution may differ from the owner of the corresponding Generic Technical Solution. The latter one can be regarded as the creator of the (parametric) Generic Technical Solution, the first one as the person or organisation who uses it and has defined the values of the parameters. Functional Unit Occurences and the corresponding Specific Functional Units are owned by the creator of the Technical Solution Occurence which contains them. This owner is also responsible for the status of a Technical Solution Occurrence which can be used for each Functional Unit Occurrence. The Technical Solution Occurrence has therefore two owners: the using owner (responsible for selection or rejection) and the creating owner.

Decision

Decision concerning the selection or rejection of a Product Definition Unit. This entity defines the status of a PDU. Two discriminations can be applied to this entity: stage and level. The stage discriminator is used to distinguish between decisions regarding Functional Units, Technical Solutions, Planned Units, etc. The level discriminator distinguishes between decisions on a generic, specific and/or occurrence level. The generic entity Decision has only one subtype in version 4 of GARM, the Technical Solution Occurrence Decision.

Generic Technical Solution Decision

Decision concerning the selection or rejection of a Generic Technical Solution. It can be overruled by a decision on the Specific level. The decision determines what the status of Generic Technical Solution for a Generic Functional Unit is: either rejected or selected. If no decision is recorded, the Technical Solution is regarded as an alternative.

Specific Technical Solution Decision

Decision concerning the selection or rejection of a Specific Technical Solution. It can be overruled by a decision on the Occurrence level.The decision determines what the status of a Specific Technical Solution for a Specific Functional Unit is: either rejected or selected. If no decision is recorded, the Technical Solution is regarded as an alternative.

Technical Solution Occurrence Decision

Decision concerning the selection or rejection of a Technical Solution Occurrence. The decision determines what the status of Technical Solution Occurrence for a Functional Unit Occurrence is: either rejected or selected. If no decision is recorded, the Technical Solution is regarded as an alternative. Decisions refer to either Technical Solution Arguments on Aspects of the PDU, or Technical Solution Approvals by Experts.

Technical Solution Occurrence Argument

Argument for the decision about a Technical Solution. An argument is related to an aspect, and compares the required characteristics of a Functional Unit with the expected characteristics of a Technical Solution. Currently only Arguments on the Occurrence level are supported by GARM version 4. The logical attribute evaluation makes use of the function "In-Domain" which is included in the Express version of GARM.

Expert

Person and/or Organisation who is regarded as an authority for the approval or disapproval of a Technical Solution. This approval or disapproval can be used for a decision concerning the status of a Technical Solution.

Approval

Logical approval or disapproval of a Technical Solution for a Functional Unit. It is given by an Expert.

Analysis

This entity represents an analysis (by calculation, simulation or any other method) done by an Expert to judge the quality of a Technical Solution. An analysis to one or more results, named Expected Characteristics.

5.7. Functional Network and Meta-topology

Node

A Functional Unit is represented by a Node in the network. Nodes may have a limited number of Ends. Topological Nodes can be distinguished in Open Topological Nodes and Closed Topological Nodes. A Closed Topological Node refers directly to a Domain, while an Open Topological Node refers indirectly via an External Domain End or even a Domain Interface.

End

Each End can be connected to another End by means of an Interface. The constituting Ends of an Interface are considered as Mated Ends. The counterpart of the mated state is the (local) free state. In that case the End refers to a Port (Occurrence) of its aggregated Technical Solution Occurrence.

A Boundary End declares one side of a possible Boundary Interface between two Nodes. A further refinement defines the direction of the dimensional hop: an Internal Boundary End refers to the boundaries of the working area of the Node it belongs to, the dimensional order will decrease by one in this direction. an External Boundary End offers the working area of the Node it belongs to, to act as a boundary itself, the dimensional order will increase by one in this direction.

A Domain End declares one side of a possible Domain Interface between two Nodes. A further refinement defines the direction of the dimensional hop: an Internal Domain End refers to the inner regions of the working area of the Node it belongs to, the dimensional order will stay the same or decrease in this direction. an External Domain End offers the working area of the Node it belongs to, to act as an inner region itself, the dimensional order will stay the same or increase in this direction.

Interface

An Interface establishes a relationship between two Ends of two different Nodes within the same functional network. A Boundary Interface establishes a relationship between two Nodes, in which the working area of one Node acts as the boundary of the working area of the other Node.

A Domain Interface establishes a relationship between two Nodes, in which the working area of one Node resides completely within the working area of the other Node.

Domain

A Domain defines an area of a certain dimensional order. To complete the definition it may refer, indirectly,to Domains of lesser or equal dimensional order. E.g. a second order Domain (Face) refers indirectly (by means of Sides and Boundaries or Boundary Interfaces) to first order Domains (Edges) which represent its outer boundaries. To specify inner areas a Domain may refer indirectly (by means of Regions and Voids or Domain Interfaces) to Domains of lower or equal order. E.g. a second order Domain (Face) may specify internal areas of again 2nd order (inner loop), 1st order (Edge shaped) and 0th order (Vertex shaped).

Side

A Side declares one side of a functional or topological relationship of type "Domain A is bounded by/bounds Domain B". Functional relationships are handled by Boundary Ends, which may be free or mated by means of a Boundary Interface. Pure topological relationships are established by a Boundary.

Region

A Region declares one side of a functional or topological relationship of type "Domain A encloses/is enclosed by Domain B". Functional relationships are handled by Domain Ends, which may be free, or mated by means of a Domain Interface. Pure topological relationships are established by a Void.

Void

A Void establishes a topological relationship between two Domains, in which one Domain resides completely within another Domain. The dimensional order of the enclosed Domain should be less or equal to the dimensional order of the enclosing Domain.

Boundary

A Boundary establishes a topological relationship between two Domains, in which one Domain is bounded by/bounds another Domain. The Domains in question should differ one in dimensional order.

5.8. Aspects and Characteristics

Required Characteristic

A required characteristic is given for an Aspect and defines allowed values for this Aspect through the entity Allowed_Paramter_Domain. For example, the Requirement can be the maximal Heat transfer of an enclosure. Entity Required_Characteristic refers to the Aspect-subtype Heat_transfer_of_enclosure and to entity Allowed_Parameter_Domain, where the maximal value is given.

Expected Characteristic

A n expected characteristic is calculated for an Aspect by means of an analysis, and gives the calculated value for this Aspect through the entity Parameter_Value. If we take the same example given for the previous entity, Entity Expected_Characteristic refers to the Aspect-subtype Heat_transfer_of_enclosure and to entity Parameter_Value, where the expected value is given.

Measured Characteristic

A measured characteristic is measured from the physical (existing) product, for an Aspect . If we take the same example given for the previous entities, Entity Measured_Characteristic refers to the Aspect-subtype Heat_transfer_of_enclosure and to entity Parameter_Value, where the measured value is given.

5.9 Producttype level - Buildings

Functional Unit Specialisation

Several product classifications are currently in use in the Building Industry. Most of them are based on the function of Building Systems or Building Elements. They are therefore regarded as classifications of Functional Units. Most classifications cover a wide range of Functional Units, but none of them will be complete enough to serve the goals of STEP. Since there are so many different classifications in use, they can be selected on the Non-STEP layer for AEC as specialisations of GARM Functional Units. Four important classifications are included: ISO standard 6241 table 1, the British Building Industry Code (BIC), the CI-Sfb table 1 code, and the BSAB code of functional elements. They can be extended however with other classifications by user-groups, nations, etc.

ISO 6241 Table 1

Table 1 of this standard gives an overview of Building systems, and relates them to the corresponding Sfb codes. The purpose of ISO 6241 is provide a framework for the definition of performance standards of buildings.

Building Industry Code

The Building Industry Code is a British classification code. It is included here as an example and as an alternative for Sfb.

CI-Sfb table 1

Table 1 of the Sfb classification code is primarily a functional classification of building elements. It is therefore included as an optional classification of Functional Units.

Agent specialisation

ISO 6241 Table 4

Table 4 of ISO 6241 gives an overview of agents relevant to building performance. This table is reworked as a classification of Agents. No attributes are given in the standard.

Aspect specialisation

ISO 6241 Table 2

Table 2 of ISO 6241 gives a list of 14 broad classes of requirement definitions for buildings. A 15th class, Energy saving and heating, is added by ISO 6242. The table gives some examples of requirements for each class. Of these, the class Economic requirements is included in the model as being a supertype of Capital costs, Running costs and Maintenance costs. The reuirements are regarded here as aspects, for which requirements can be given. This allows to include analysis results and measurments for the same aspects in the product model.

ISO 6242

Four requirement classes in ISO 6241 table 2 are further refined in this standard. The requirements are given to a very detailed level, allowing the definition of attributes. A special case are the Hygrothermal paramaters: ISO 6242 sees four fundamental individual parameters, from which a list of other parameters can be derived. Both lists are included here, but it is clear that only the Individual parameters should belong to the domain of the product model.

7 SHIP STRUCTURAL SYSTEM INFORMATION MODEL

The purpose of this document is to provide, in an information model, an adequate framework for the formulation of an International standard for ship's structure. In particular, the goal has been to identify sufficient information, such that data which represents the majority of a ship's structural system can be communicated digitally via this standard between two CAD/CAM modeling systems without manual intervention or interpretation. This structural information model is but one of many such models which are required for definition and integration prior to the formulation of an International exchange standard.

7.1 Purpose & Scope

7.1.1 Purpose

The purpose of this document is to provide, in an information model, an adequate framework for the formulation of an international data exchange standard for ship's structure. In particular, the goal has been to identify sufficient information, such that data which represents the majority of a ship's structural system can be communicated digitally via this standard between two CAD/CAM modeling systems without manual intervention or interpretation. This structural information model is but one of many such models which are required for definition and integration prior to the formulation of an international exchange standard.

While the information contained in this document is not complete, it received sufficient review by both the Navy/Industry Digital Data Exchange Standards Committee (NIDDESC) and the Architecture, Engineering, and Construction (AEC) Committee to warrant submission to the ISO TC184/SC4 organization at this time. It is expected that subsequent revisions to this document will be made to incorporate refinements, and information not now included. For these reasons, it is not the intent that the information contained herein, in its present form, be used as a shipbuilding industry standard. As with other information models, it should be noted that the information contained herein is only one of many possible ways of representing a ship's structural system.

In the context of this standrad's purview, the product of an AEC industry activity is sites, factories, plants, ships (floating plants), and/or buildings. These are in turn composed of various systems.

Engineering systems contain equipment interconnected by distribution systems. The reference model for distribution system is described in ISO TC184/SC4/WG1 document number 3.2.2.2.

A structural system provides supporting structure, integrity, shelter, and habitability. The scope of this document is a steel structural system, and in particular a ship's structural system. It should be noted that although this model was developed specifically for a ship's structural system. many of the concepts and entities presented in this model can readily be applied to other type of steel structures.

7.1.2 Model Integration

This model is an application model. As such it incorporates concepts and in some cases specific entities from other models in the PDES organization. These models include Geometry, Topology, Form Features, Solids, Materials, and Product Structure Configuration Management (PSCM).

Although much work remains to be done, integration of this model with the above mentioned models has begun within the Integration Committee.

Integration points have been identified in this model. These points are boxed in and labeled on the NIAM diagrams and also identified in the entity desciptions. Subsequent revision to the model will carry the integration effort further.

7.1.3 Scope

The scope of this document includes the level of definition of a structural product model resulting from the completion of detailed design and lofting. Specifically included are all items listed below. Nesting data for plates and shapes is excluded because typically it is developed in an organizationally specific manner or format. The intent is to include product definition data (e.g. the product model) since this data is logically transferable between different organizations responsible for building and/or maintaining the ship.

This document includes a definition of geometry, topology and property data for the following items:

- Lines
- Stiffened surfaces (shell, bulkheads, decks, web frames, etc.)
- Cutouts, lightening holes and penetrations
- Weld data and bevels
- Stiffener data (scantlings/traces/orientation/end cuts
- Material definition (thickness, type, material, etc.)
- Brackets, collar plates
- Stanchions
- Units/Assemblies
- Foundations
- Rudder

This version does not define the following:

- Hangers for distributive systems
- Non-structural tanks
- Struts and bossings
- Castings and forgings
- False decks and gratings

These five items will be reviewed for inclusion in subsequent versions of this document Of the above mentioned items, castings and forgings stand out as an integration area with the Solids Modeling group.

It is not the intent of this document to cover configuration management of structural items in any great detail. While some attempt has been made with the inclusion of a Date/Time concept (refer to Figure 65) this is intended to be a place holder for future work. This has been identified as an area of integration with the Product Structure Configuration Management (PSCM) model.

This document examines the multiple relationships between the above items from several different points of view. In this manner a complete model is developed, portion by portion, without the necessity to focus on the complexity of the entire model.

7.1.4 Modeling Methodology

The Information Model contained herein has been developed using the Nijssen Information Analysis Method (NIAM). Model verification has been accomplished by the generation of a Neutral Data Model (NDM) using Control Data's PRECISE PC-IAST NIAM tool. The Express version of the model (Section 7.6) has been generated directly from the original NIAM binary model. It was then manually modified to incoroporate various improvements as well as comments received by Peter Wilson of General Electric. It is the intent of both NIDDESC and the AEC committee that NIAM be the original source or baseline modeling methodology.

The NIAM diagrams in this document collectively represent this model. Entities surrounded with boxes of a solid line type are also referenced in other figures within this document as noted. Entities or groups of entities surrounded by boxes of a dashed line type are appropriately noted to indicate model integration points and/or place holders for future work. The description of entities within this report have the entities capitalized. This indicates that any entity capitalized appears elsewhere in this report or on the NIAM diagrams.

Section 7.4 contains an alphabetical listing of all the entities that appear in the ship structural systems information model. Section 1.3.3 contains a guide to reading Nijssen Information Analysis Method (NIAM) diagrams. Section 7.5 is an alphabetical listing of definitions of ship structural terms for those unfamiliar with the shipbuilding industry. Section 7.6 is the Express version of the model.

Figure 62, extracted from Document Number 3.2.2.2, illustrates the relationship of structural systems to an overall model of AEC projects. The following relationships may be stated with NIAM constructs:

A Project has sites and/or building and/or plants/ships.

A site is part of one or many projects.

A building is part of one and only one project.

Every building is on one and only one site.

A site may have one or more buildings.

A plant/ship is part of one and only one project.

SECTION 7: SHIP STRUCTURAL SYSTEM INFORMATION MODEL

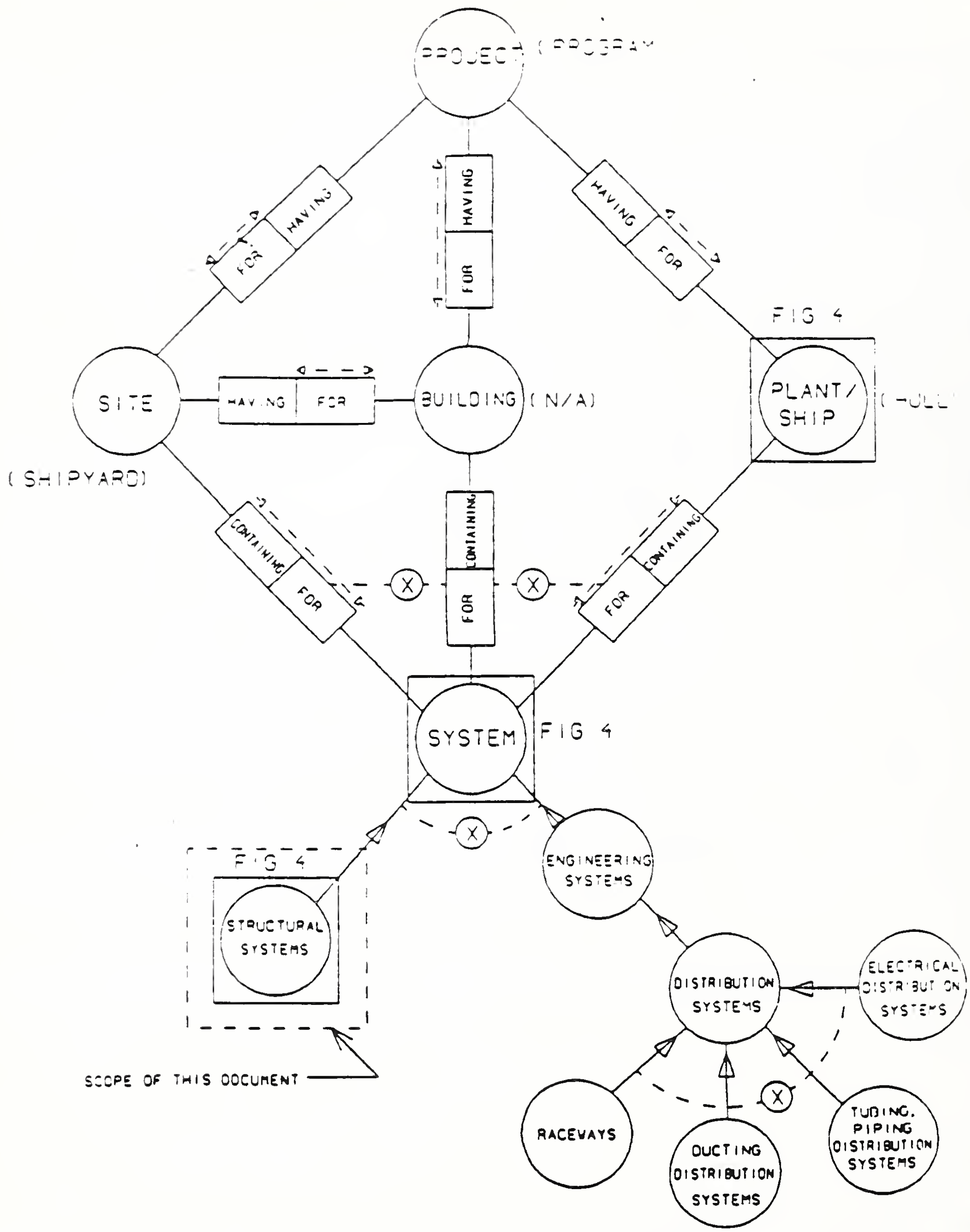


Figure D-62: Global AEC Reference Model

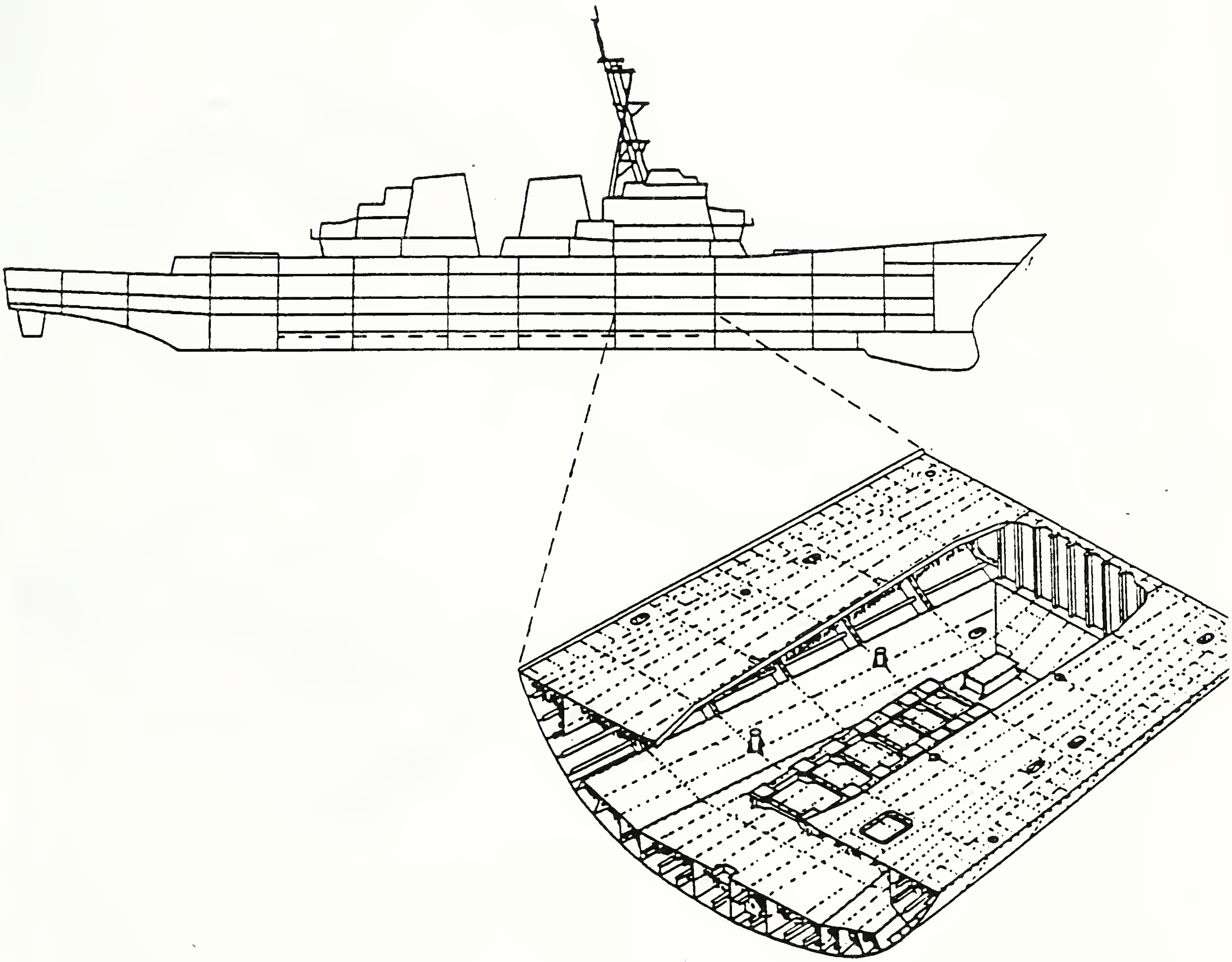


Figure D-63: A Hull with its Unit Assemblies

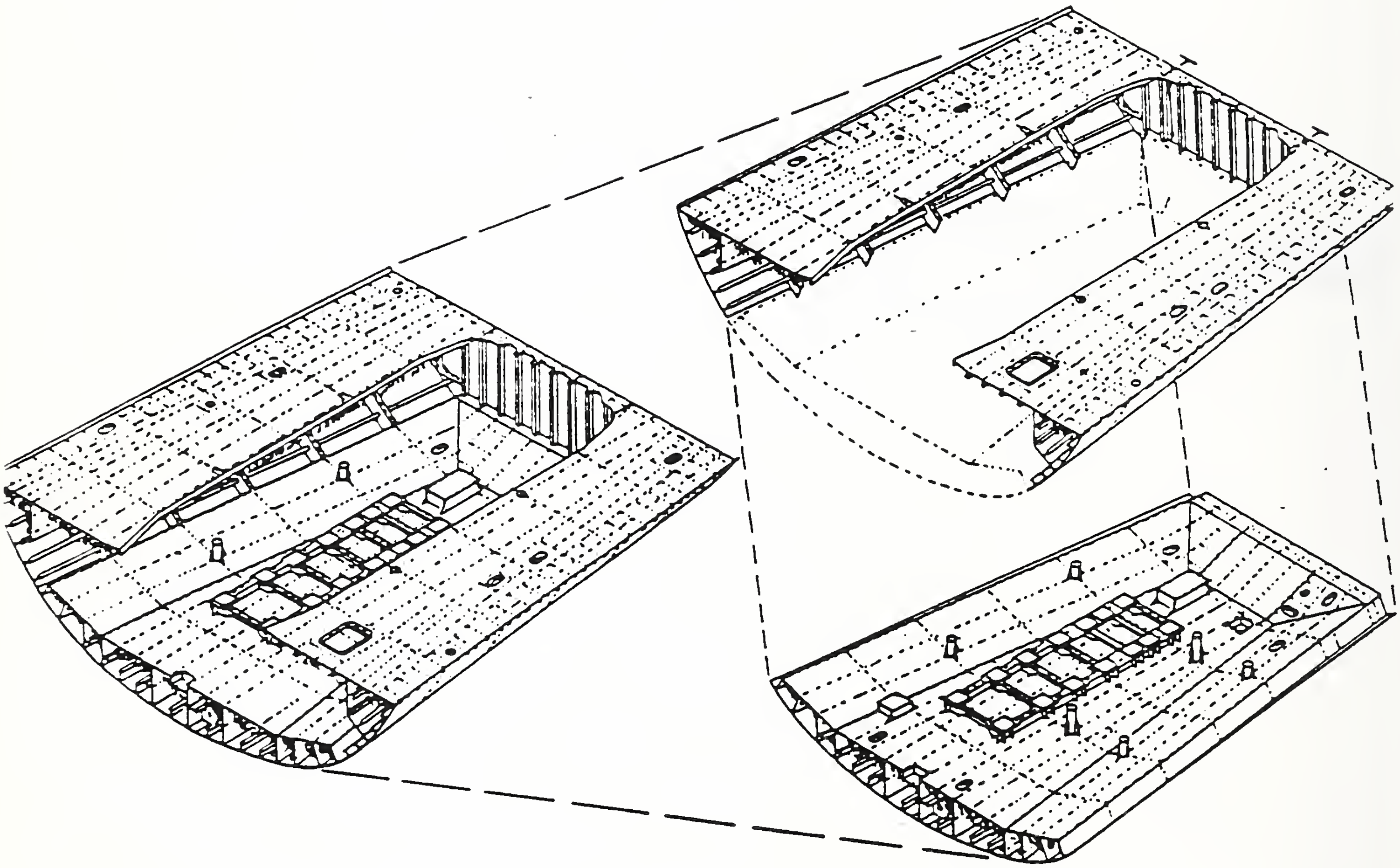


Figure D-64: A Unit Assembly is made up of Subassemblies

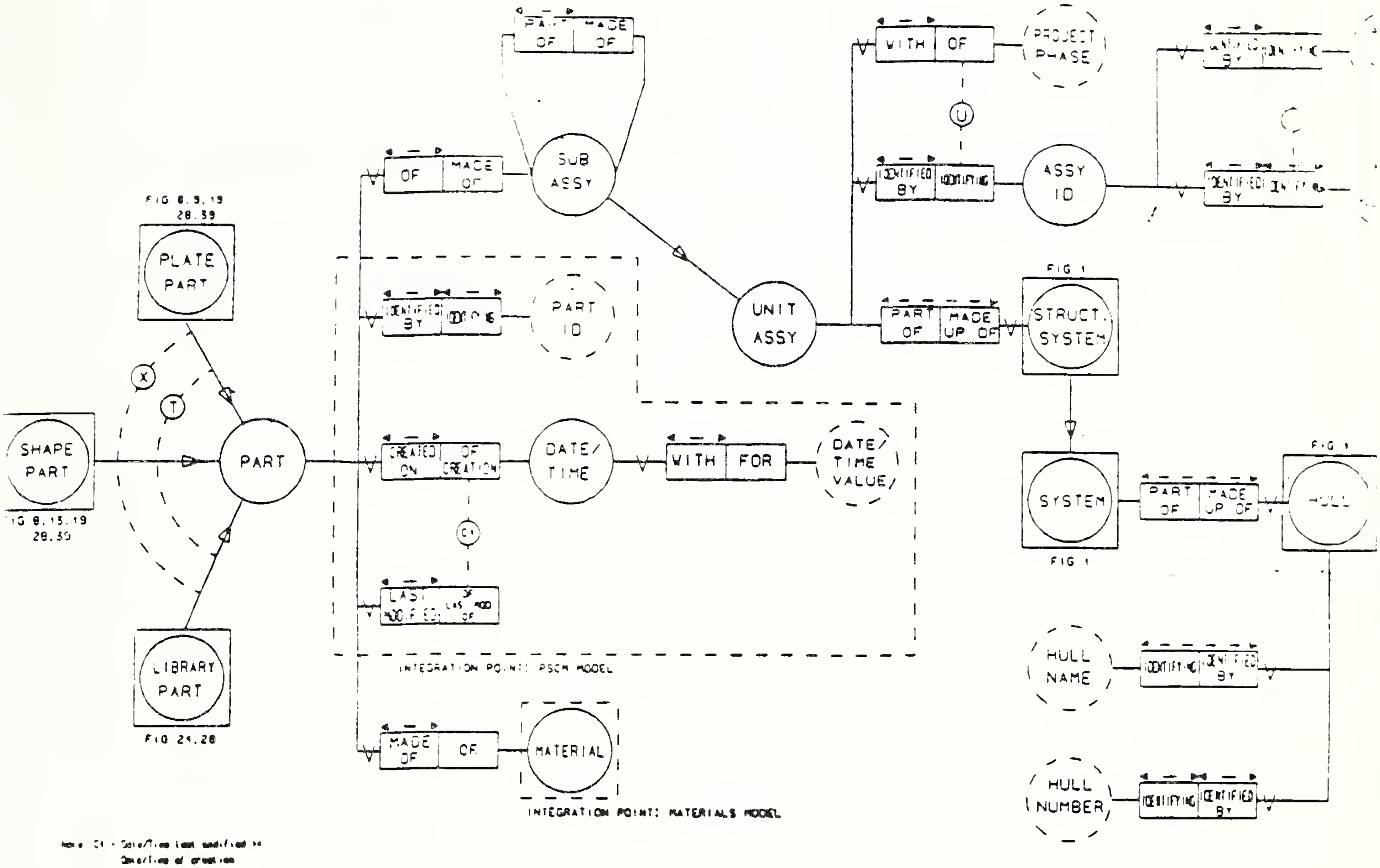


Figure D-65: NIAM Diagram showing Hull/Assembly/Part Relationships

Sites may have systems.

Every building has one or more systems.

Every plant/ship has one or more systems.

A system is either of a site, a building, or a plant/ship.

A system is either a structural system or an engineering system.

Engineering systems are a subset of systems.

Distribution systems are a subset of engineering systems.

Electrical distribution systems, piping/tubing distribution systems, ductwork distribution systems, and raceways are subsets of distribution systems.

7.2 Ship Structure Reference Model

7.2.1 Fundamental Concepts of a Ship Structural System

Basic modeling objects for a ship's structural system are chosen to be surfaces, shapes, holes, welds, bevels and material properties. This is not meant to be an exhaustive list, but rather an indication of a fundamental approach to object role modeling. For example, flat decks, transverse bulkheads, and longitudinal bulkheads etc., with the exception of spatial orientation, all exhibit similar characteristics. These characteristics are that they all lie on a defined surface, they all contain one or more plates, they may contain one or more holes, they may be stiffened by one or more shapes, etc. To object role model these items separately (as well as a host of other functional structural items) would be to apply an improper concentration on function, rather than a simultaneous consideration of function, physical shape, and geometric and topological definition.

The NIAM diagram in Figure 65 illustrates the concepts of hull, assembly and part. Also included are material characteristics and configuration management information.

The relationships expressed in Figure 65 are:

Entity Name: **HULL** **NOLOT**

Entity Number: **SS-1** **(Figure 63)**

A HULL is a collection of systems which comprise a ship (product model).

Integration Point:

Product Structure Model; Product Model Entity.

Business Rules:

- A HULL must be identified by one or more HULL NAME's.
- A HULL must be identified by exactly one HULL NUMBER.
- A HULL must be made up of one or more SYSTEM's.

Entity Name: **SYSTEM** **NOLOT**

Entity Number: **SS-2**

A SYSTEM is a functionally related group of elements (potentially recursive). Some examples of functional groupings are Structure, HVAC or Electrical Systems.

Business Rules:

- A SYSTEM may be part of any number of HULL's.
- A SYSTEM must be with exactly one SYSTEM ID.

Entity Name: STRUCTURAL SYSTEM NOLOT

Entity Number: SS-3

A STRUCTURAL SYSTEM is a collection of structural parts used, in general, to compartment and support all other systems.

Business Rules:

- A STRUCTURAL SYSTEM is a kind of SYSTEM.
- A STRUCTURAL SYSTEM must be made up of one or more UNIT ASSEMBLY's.

Entity Name: UNIT ASSEMBLY NOLOT

Entity Number: SS-4 (Figure 63 & 64)

A UNIT ASSEMBLY gathers together parts and/or sub-assemblies. The UNIT ASSEMBLY can represent a logical grouping (HFO tank, space 42, etc.) or a physical grouping associated with actual construction phases of the hull.

Business Rules:

- A UNIT ASSEMBLY must have one ASSEMBLY ID.
- A UNIT ASSEMBLY may be part of any number of STRUCTURAL SYSTEM's.
- A UNIT ASSEMBLY must be with exactly one PROJECT PHASE.
- Every UNIT ASSEMBLY is associated uniquely with one combination of

  a PROJECT PHASE of the UNIT ASSEMBLY and
  an ASSEMBLY ID identifying the UNIT ASSEMBLY.

Entity Name: ASSEMBLY-ID NOLOT

Entity Number: SS-5

An ASSEMBLY-ID is the unique identifier of a unit-assembly. The ASSEMBLY-ID consists of an assembly-id-name and an assembly-id-number.

Business Rules:

- An ASSEMBLY-ID must have one ASSEMBLY-ID-NAME.
- An ASSEMBLY-ID must have one ASSEMBLY-ID-NUMBER.
- An ASSEMBLY-ID must identify a UNIT-ASSEMBLY.

Entity Name: SUB-ASSEMBLY NOLOT

Entity Number: SS-6 (Figure 64)

A SUB-ASSEMBLY is a collection of parts and/or other SUB-ASSEMBLIES. SUB-ASSEMBLIES can gather parts and/ or SUB-ASSEMBLIES into logical groupings (decks, bulk-heads, etc.) or into physical groupings representing actual construction phases of he hull.

Business Rules:

- A SUB-ASSEMBLY is a kind of UNIT ASSEMBLY.
- A SUB-ASSEMBLY may be made of any number of PART's.
- A SUB-ASSEMBLY may be made of any number of SUB-ASSEMBLY's.
- A SUB-ASSEMBLY may be part of at most one SUB-ASSEMBLY.

Entity Name: PART NOLOT

Entity Number: SS-7

A PART is a unique structural element or component consumed during the production process.

Integration Point:

Product Structure Model; Product Item Entity.

Business Rules:

- All PARTs must be LIBRARY PARTs, or PLATE PARTs, or SHAPE PARTs.
- A PART must be created on exactly one DATE TIME.
- A PART must beid entified by exactly one PART ID.
- A PART must be last modified on exactly one DATE TIME.
- A PART must be made of exactly one MATERIAL.
- A PART must be of exactly one SUB-ASSEMBLY.

Entity Name: DATE/TIME NOLOT

Entity Number: SS-8

A DATE-TIME is expressed in the form YYMMDD.HHMMSS.

Integration Point:

Miscellaneous Resources Model; Unit and Time-Unit entities.

Business Rules:

- A DATE/TIME may be creation date of any number of PART's.
- A DATE/TIME may be last modified date of any number of PART's.
- A DATE/TIME must be with exactly one DATE/TIME VALUE.

Entity Name: MATERIAL **NOLOT**

Entity Number: SS-9

A MATERIAL is the substance making up a part. This entity includes a description of the MATERIAL and its properties.

Integration Point:

Materials Model; Material entity.

Business Rules:

- A MATERIAL may be used for any number of PARTs.

7.2.2 Ship Geometry/Topology

The product model for a ship's structural system must contain both the topological relationships, the geometry, material properties and joining information (welds and bevels). Topology is expressed in terms of surfaces which bound other surfaces or shapes, surfaces which are bounded by other surfaces, and shapes which lie on surfaces and are bounded by surfaces and/or other shapes. Figure 66 illustrates the most general case of molded surface definition relative to plate thickness and the added complexity of two joining plates/surfaces. Figure 67 illustrates the geometric and topological relationships for surfaces.

The NIAM diagrams which illustrate these concepts are shown in Figures 68, 69 & 70.

The relationships expressed in Figures 68 and 69 are:

Entity Name: BOUNDED SURFACE **NOLOT**

Entity Number: SS-10

A BOUNDED SURFACE is a parameterized space, representing an orientable locus of points, bounded by a set of molded curves which forms a closed contour. A BOUNDED SURFACE may be planar (deck) or sculptured (shell).

Integration Point:

Geometry Model; Bounded Surface

Business Rules:

- A BOUNDED SURFACE must be bounded by one or more SURFACE EDGE's.
- A BOUNDED SURFACE may be bounding any number of COMPARTMENT's.
- A BOUNDED SURFACE may define any number of MOLDED CURVE's.
- A BOUNDED SURFACE may define any number of NODE's.
- A BOUNDED SURFACE may define any number of STRUCTURAL OPENING's.
- A BOUNDED SURFACE must be identified by exactly one SURFACE ID.
- A BOUNDED SURFACE must have exactly one SURFACE TYPE.

Entity Name: **CURVED SURFACE** **NOLOT**

Entity Number: **SS-11**

A CURVED SURFACE is a non-planar surface representing such areas as the shell of a ship.

Business Rules:

- A CURVED SURFACE is always a kind of BOUNDED SURFACE.
- A CURVED SURFACE cannot also be an ELEMENTARY SURFACE.

Entity Name: **ELEMENTARY SURFACE** **NOLOT**

Entity Number: **SS-12**

An ELEMENTARY SURFACE is a simple surface such as planar, conical or cylinderical surface.

Integration Point:

Geometry Model; Elementary Surface entity.

Business Rules:

- An ELEMENTARY SURFACE is a kind of BOUNDED SURFACE.
- An ELEMENTARY SURFACE cannot also be a CURVED SURFACE.

Entity Name: **BSPLINE SURFACE** **NOLOT**

Entity Number: **SS-13**

A BSPLINE SURFACE identifies a particular mathematical representation for a curved surface.

Integration Point:

Geometry Model; Bspline Surface entity.

Business Rules:

- A BSPLINE SURFACE is a kind of CURVED SURFACE.
- A BSPLINE SURFACE cannot also be a BEZIER SURFACE.

Entity Name: **BEZIER SURFACE** **NOLOT**

Entity Number: **SS-14**

A BEZIER SURFACE identifies a particular mathematical representation for a curved surface.

Integration Point:

Geometry Model; Bezier Surface entity.

Business Rules:

- A BEZIER SURFACE is a kind of CURVED SURFACE.
- A BEZIER SURFACE cannot also be a BSPLINE SURFACE.

Entity Name: **PLANAR SURFACE** **NOLOT**

Entity Number: **SS-15**

A PLANAR SURFACE is a surface with no curvature anywhere within its boundaries.

Integration Point:

Geometry Model; Plane entity.

Business Rules:

- A PLANAR SURFACE is a kind of ELEMENTARY SURFACE.
- A PLANAR SURFACE must be located by exactly one POSITION POINT.
- A PLANAR SURFACE must be oriented by exactly one UNIT VECTOR.
- Every PLANAR SURFACE is associated uniquely with one combination of a POSITION POINT locating the PLANAR SURFACE and an UNIT VECTOR orienting the PLANAR SURFACE.

Entity Name: SURFACE EDGE NOLOT

Entity Number: SS-16

A SURFACE EDGE is one of a sequence of molded curves bounding a surface.

Business Rules:

- A SURFACE EDGE must be bounding exactly one BOUNDED SURFACE.
- A SURFACE EDGE must be defined by exactly one MOLDED CURVE.
- A SURFACE EDGE must be identified by exactly one EDGE SEQUENCE NUMBER.

Entity Name: MOLDED CURVE NOLOT

Entity Number: SS-17

A MOLDED CURVE is a curve on a surface, a curve bounding a surface or a curve formed by the intersection of two surfaces.

Integration Point:

Geometry Model; Curve, Curve on a Surface, Intersection Curve entities.

Business Rules:

- A MOLDED CURVE must be defined by exactly one CURVE GEOMETRY.
- A MOLDED CURVE may be defined on any number of BOUNDED SURFACE's.
- A MOLDED CURVE may define any number of PART FLANGE's.
- A MOLDED CURVE may define any number of PATH SEGMENT's.
- A MOLDED CURVE may define any number of SURFACE EDGE's.
- A MOLDED CURVE must be identified by exactly one CURVE ID.
- A MOLDED CURVE may be locating any number of NODE's.

Entity Name: COMPARTMENT NOLOT

Entity Number: SS-18

A COMPARTMENT is an enclosed space within a ship. An example of a COMPARTMENT is Auxiliary Machinery Room 2.

Business Rules:

- A COMPARTMENT must be bounded by one or more BOUNDED SURFACE's.
- A COMPARTMENT must be identified by exactly one COMPARTMENT ID.
- A COMPARTMENT may be identified by any number of COMPARTMENT NAMES.

Entity Name: SURFACE ID **NOLOT**

Entity Number: SS-19

A SURFACE ID is a unique identifier for a surface. It consists of a surface id name and a surface id number.

Business Rules:

- A SURFACE ID must have exactly one SURFACE ID NAME.
- A SURFACE ID must have exactly one SURFACE ID NUMBER.
- A SURFACE ID must identify exactly one BOUNDED SURFACE.
- Every SURFACE ID is associated uniquely with one combination of

  a SURFACE ID NAME for the SURFACE ID and
  a SURFACE ID NUMBER for the SURFACE ID.

Entity Name: POSITION POINT **NOLOT**

Entity Number: SS-20

POSITION POINT provides the X, Y, and Z coordinates for positioning a planar surface.

Integration Point:

Geometry Model; Point entity.

Business Rules:

- A POSITION POINT must have exactly one X POSITION.
- A POSITION POINT must have exactly one Y POSITION.
- A POSITION POINT must have exactly one Z POSITION.
- A POSITION POINT may be locating any number of PLANAR SURFACE's.
- Every POSITION POINT is associated uniquely with one combination of

a X POSITION for the POSITION POINT and
a Y POSITION for the POSITION POINT and
a Z POSITION for the POSITION POINT.

Entity Name: **UNIT VECTOR** **NOLOT**

Entity Number: **SS-21**

A UNIT VECTOR is a vector with a magnitude of one, indicating direction in space.

Integration Point:

Geometry Model; Vector, Direction entities.

Business Rules:

- An UNIT VECTOR may define any number of SHAPE ORIENTATION's.
- An UNIT VECTOR must have exactly one X VALUE.
- An UNIT VECTOR must have exactly one Y VALUE.
- An UNIT VECTOR must have exactly one Z VALUE.
- An UNIT VECTOR may be orienting any number of NC TEXT's.
- An UNIT VECTOR may be orienting any number of PARAMETRIC OPENING's.
- An UNIT VECTOR may be orienting any number of PLANAR SURFACE's.
- Every UNIT VECTOR is associated uniquely with one combination of

  a X VALUE for the UNIT VECTOR and
  a Y VALUE for the UNIT VECTOR and
  a Z VALUE for the UNIT VECTOR.

Entity Name: **CURVE GEOMETRY** **NOLOT**

Entity Number: **SS-22**

The CURVE GEOMETRY provides the mathematical representation for a curve in space.

Business Rules:

- A CURVE GEOMETRY must define one or more MOLDED CURVE's.
- A CURVE GEOMETRY may define any number of NON-PARAMETRIC ENDCUT's

  for a LOWER FLANGE
  for an UPPER FLANGE
  for a WEB

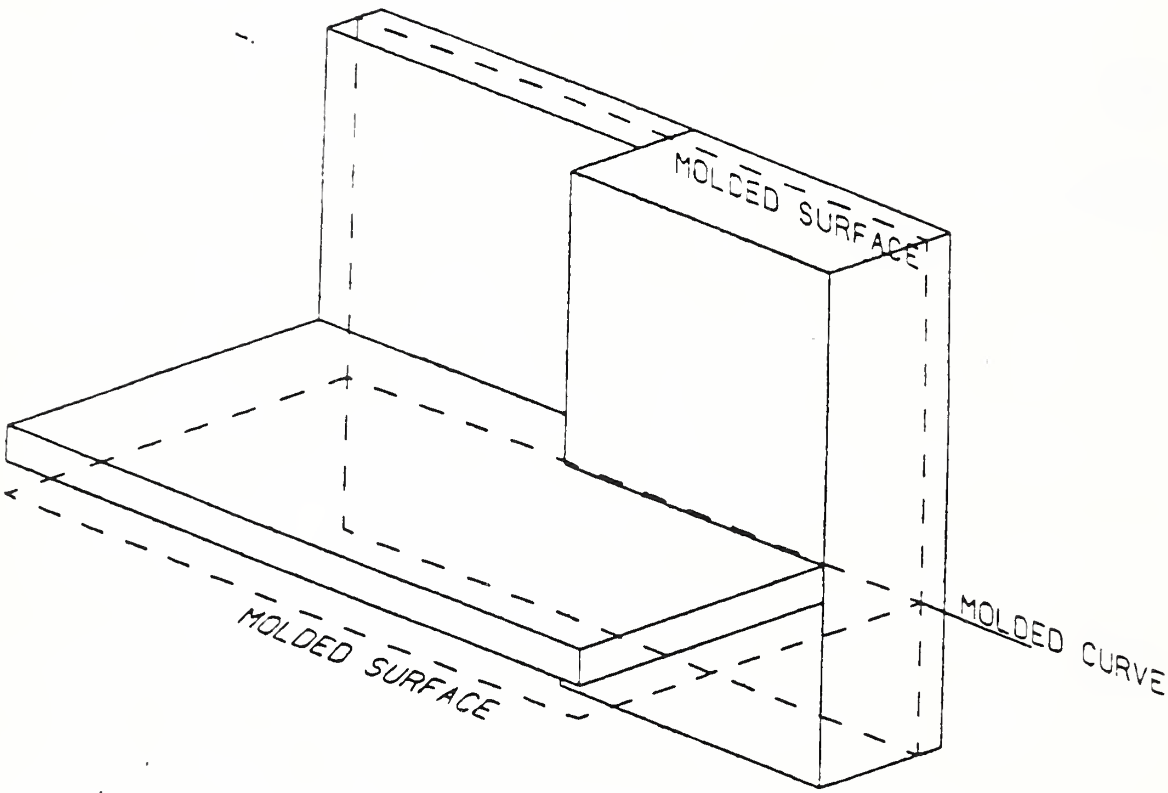


Figure D-66: Molded Surface having Plate Parts with Thickness

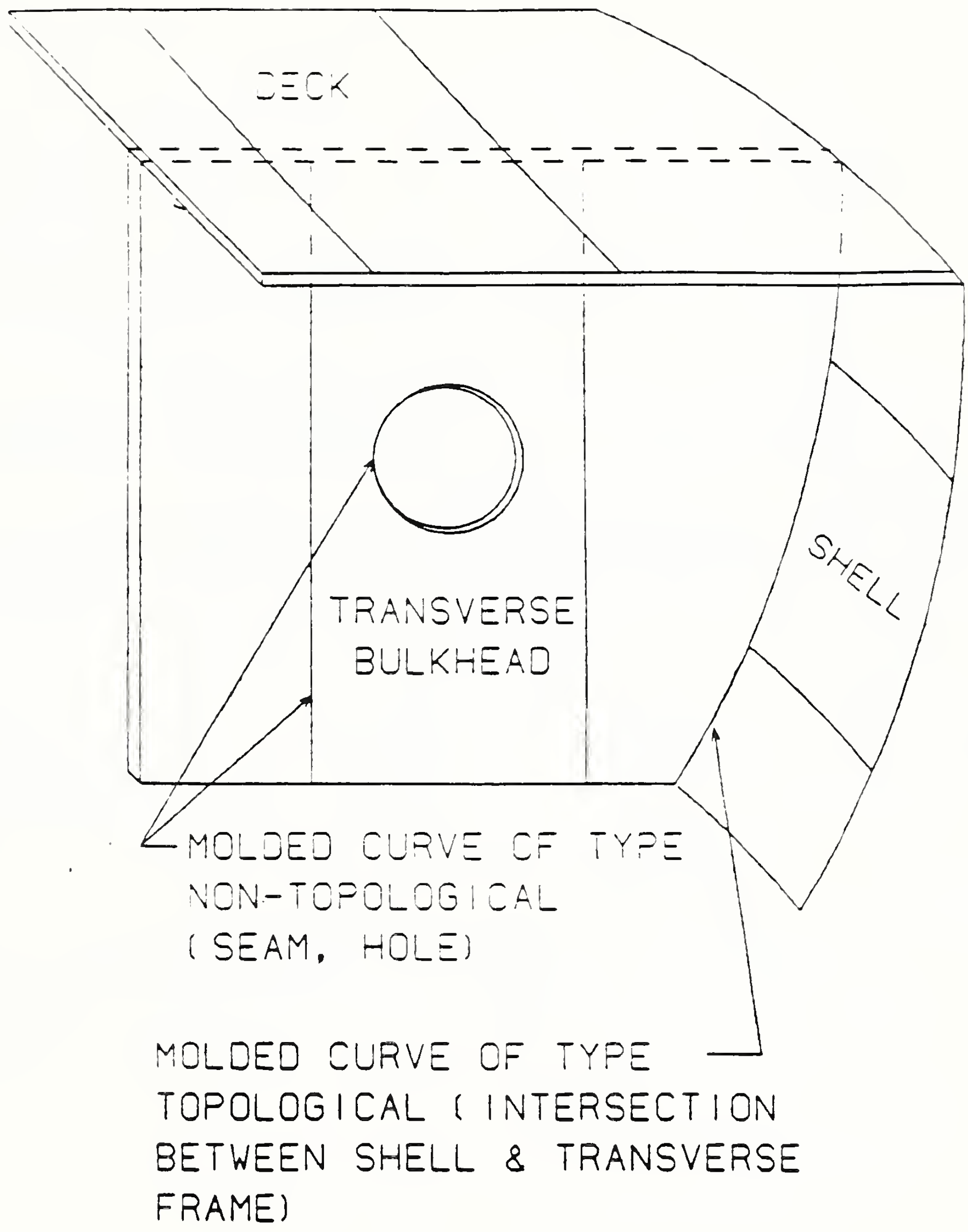


Figure D-67: Molded Curve Types

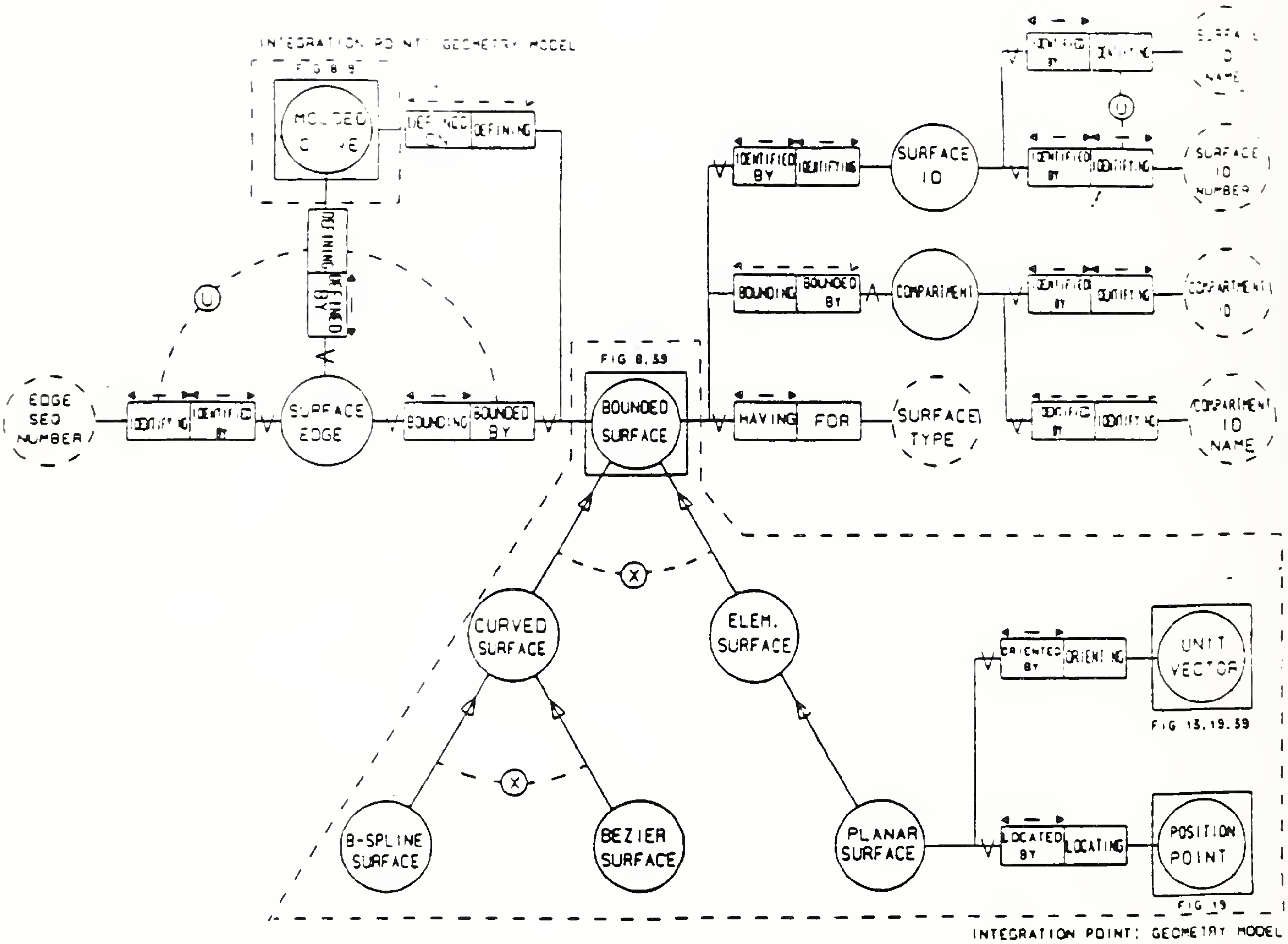


Figure D-68: NIAM Diagram showing Surface and Curve Geometry Relationships

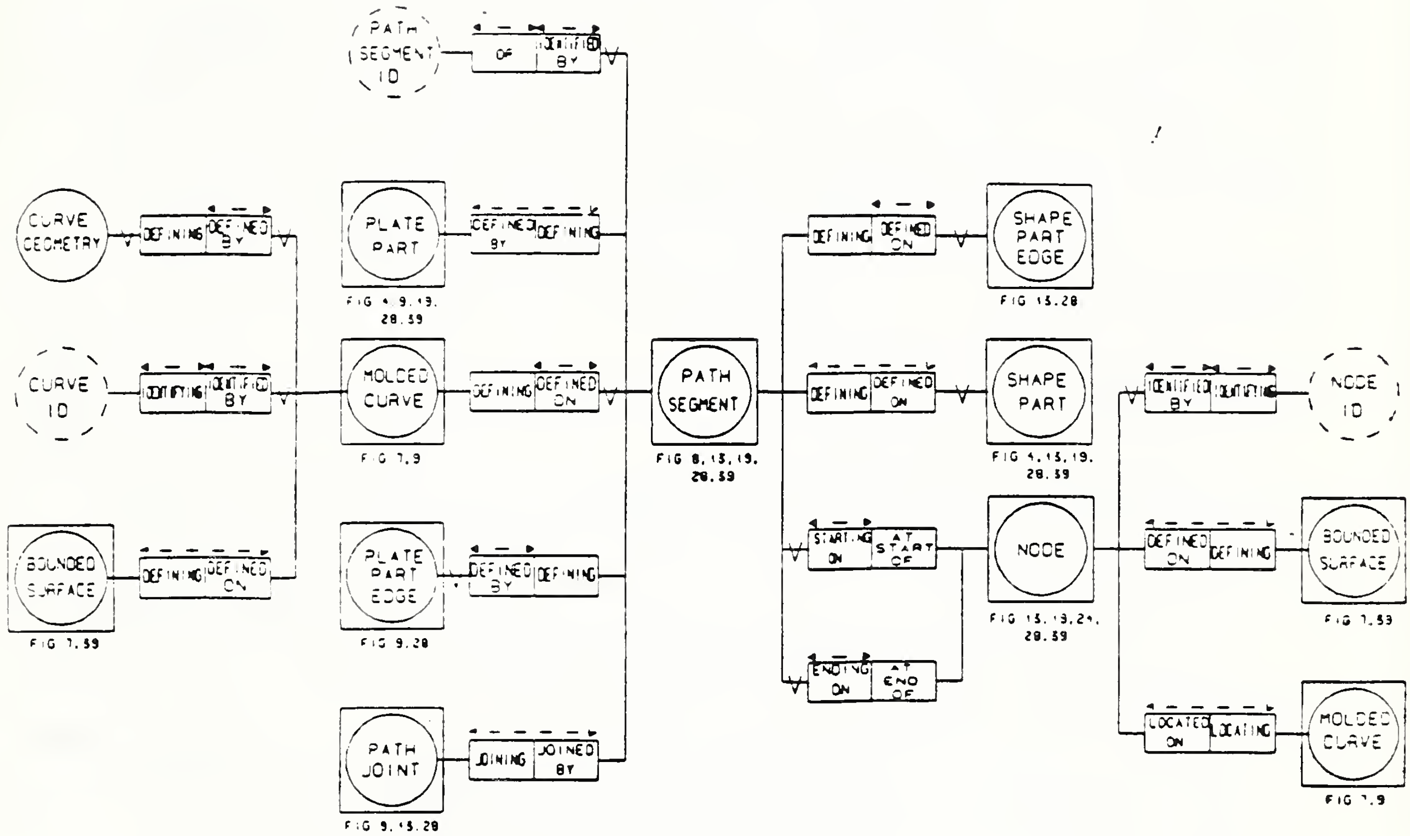


Figure D-69: NIAM Diagram showing Part and Surface Relationships

7.2.3 Parts

Plate Parts

A plate part is a part cut from flat material stock. The part may be used as is or subsequently bent to form a flange (flange part). Plate parts are defined on surfaces and may be offset from that surface some distance.

The relationships expressed in Figure 70 are:

Entity Name: **PLATE PART** **NOLOT**

Entity Number: **SS-23**

A PLATE PART is a part cut from flat material stock. The part may be used as is or subsequently bent to form a flange (flange part) or rolled.

Integration Point:

Product Structure Model; Product Item entity.

Business Rules:

- A PLATE PART is always kind of PART.
- A PLATE PART cannot also be a SHAPE PART.
- A PLATE PART may be cut by any number of STRUCTURAL OPENING's.
- A PLATE PART may be defined by any number of PATH SEGMENT's.
- A PLATE PART may have any number of PART FLANGE's.
- A PLATE PART must identify one or more PLATE PART EDGE's.
- A PLATE PART may be joined by any number of NODAL JOINT's.
- A PLATE PART may be marked with any number of NC MARK's.
- A PLATE PART must be with exactly one PART THICKNESS.
- A PLATE PART must be with exactly one PLATE SURFACE OFFSET.

Entity Name: **PLATE PART EDGE** **NOLOT**

Entity Number: **SS-24**

A PLATE PART EDGE is one of an ordered sequence of path segments defining the outer contour of a PLATE PART.

Business Rules:

- A PLATE PART EDGE must be defined by exactly one PATH SEGMENT.
- A PLATE PART EDGE must be identified by exactly one EDGE SEQUENCE NUMBER.
- A PLATE PART EDGE must be identified by exactly one PLATE PART.
- A PLATE PART EDGE may be joined by any number of PATH JOINT's.
- A PLATE PART EDGE may have at most one EDGE PREPARATION. (Figure 71)
- Every PLATE PART EDGE is associated uniquely with one combination of

  a PLATE PART identifying the PLATE PART EDGE and
  an EDGE SEQUENCE NUMBER identifying the PLATE PART EDGE and
  a PATH SEGMENT defining the PLATE PART EDGE.

Entity Name: NODE NOLOT

Entity Number: SS-25

A NODE is the logical equivalent of a geometric point. It is a unique (topological) point R3 with dimensionality 0 and extent 0.

Integration Point:

Topology Model; Vertex entity and Geometry Model: Point entity.

Business Rules:

- A NODE may be at end of any number of PATH SEGMENT's.
- A NODE may be at start of any number of PATH SEGMENT's.
- A NODE may be defined on any number of BOUNDED SURFACE's.
- A NODE must be identified by exactly one NODE ID.
- A NODE may be located on any number of MOLDED CURVE's.
- A NODE may be locating any number of LIBRARY PART's.
- A NODE may be locating any number of NC TEXT's.
- A NODE may be locating any number of NODAL JOINT's.
- A NODE may be locating any number of PARAMETRIC OPENING's.
- A NODE may be locating any number of SHAPE ORIENTATION's.

Entity Name: **PATH SEGMENT** **NOLOT**

Entity Number: **SS-26**

A PATH SEGMENT is a bounded portion of a MOLDED CURVE defined by two nodes with the positive direction of the path from the first node to the second node.

Integration Point:

Geometry Model; Curve, Bounded Curve, Trimmed Curve entities.

Business Rules:

- A PATH SEGMENT must be defined on exactly one MOLDED CURVE.
- A PATH SEGMENT may define any number of NON-PARAMETRIC OPENING's.
- A PATH SEGMENT may define any number of PLATE PART's.
- A PATH SEGMENT may define any number of PLATE PART EDGE's.
- A PATH SEGMENT may define any number of SHAPE PART's.
- A PATH SEGMENT may define any number of SHAPE PART EDGE's.
- A PATH SEGMENT may define any number of TRACE MARK's.
- A PATH SEGMENT must be ending on exactly one NODE.
- A PATH SEGMENT must be identified by exactly one PATH SEGMENT ID.
- A PATH SEGMENT may be joined by any number of PATH JOINT's.
- A PATH SEGMENT must be starting on exactly one NODE.
- Every PATH SEGMENT is associated uniquely with one combination of

  a NODE at start of the PATH SEGMENT and
  a NODE at end of the PATH SEGMENT.

Entity Name: **PART FLANGE** **NOLOT**

Entity Number: **SS-27** **(Figure 72)**

A PART FLANGE is a plate part that has a roll or knuckle along a path segment to form a flange.

Business Rules:

- A PART FLANGE must be defined on exactly one MOLDED CURVE.
- A PART FLANGE may be for any number of PLATE PART's.
- A PART FLANGE may have at most one ENDCUT.
- A PART FLANGE must have exactly one FLANGE ANGLE.
- A PART FLANGE must have exactly one FLANGE RADIUS.

Entity Name: **EDGE PREPARATION** **NOLOT**

Entity Number: **SS-28** **(Figure 71)**

The EDGE PREPARATION is the physical description of a plate part edge or shape part edge. It is typically a bevel or chamfer as required by the welding process.

Business Rules:

- An EDGE PREPARATION must be described by exactly one EDGE PREPARATION DESCRIPTION.
- An EDGE PREPARATION may be for any number of PLATE PART EDGE's.
- An EDGE PREPARATION may be for any number of SHAPE PART EDGE's.
- An EDGE PREPARATION may be for any number of STRUCTURAL OPENING's.

Shape Parts

Some of the complex relationships which exist for structural shapes are illustrated by Figure 73. The shapes which are shown attach to a surface/plate on a straight or curved line. They have a standard cross section. They may be twisted. They are intercostal or continuous. They are bounded by surface/plate and/or other shapes.

Shapes also penetrate other shapes. They may have non-standard cross sections. Stanchions are shapes which are not attached to surface/plate but are bounded by surface/plate via a nodal joint. Shapes have end cuts which can take on a wide variety of configurations. These may be modeled by parameters against a given end cut type to determine geometry.

Shape-part transition pieces, as seen in Figure 84 can be handled as shape-parts. The transition section is a type of endcut.

The NIAM diagrams shown in Figures 74 and 75 express the following relationships:

Entity Name: **SHAPE PART** **NOLOT**

Entity Number: **SS-29**

A SHAPE PART is a rolled, extruded, or built up structural shape.

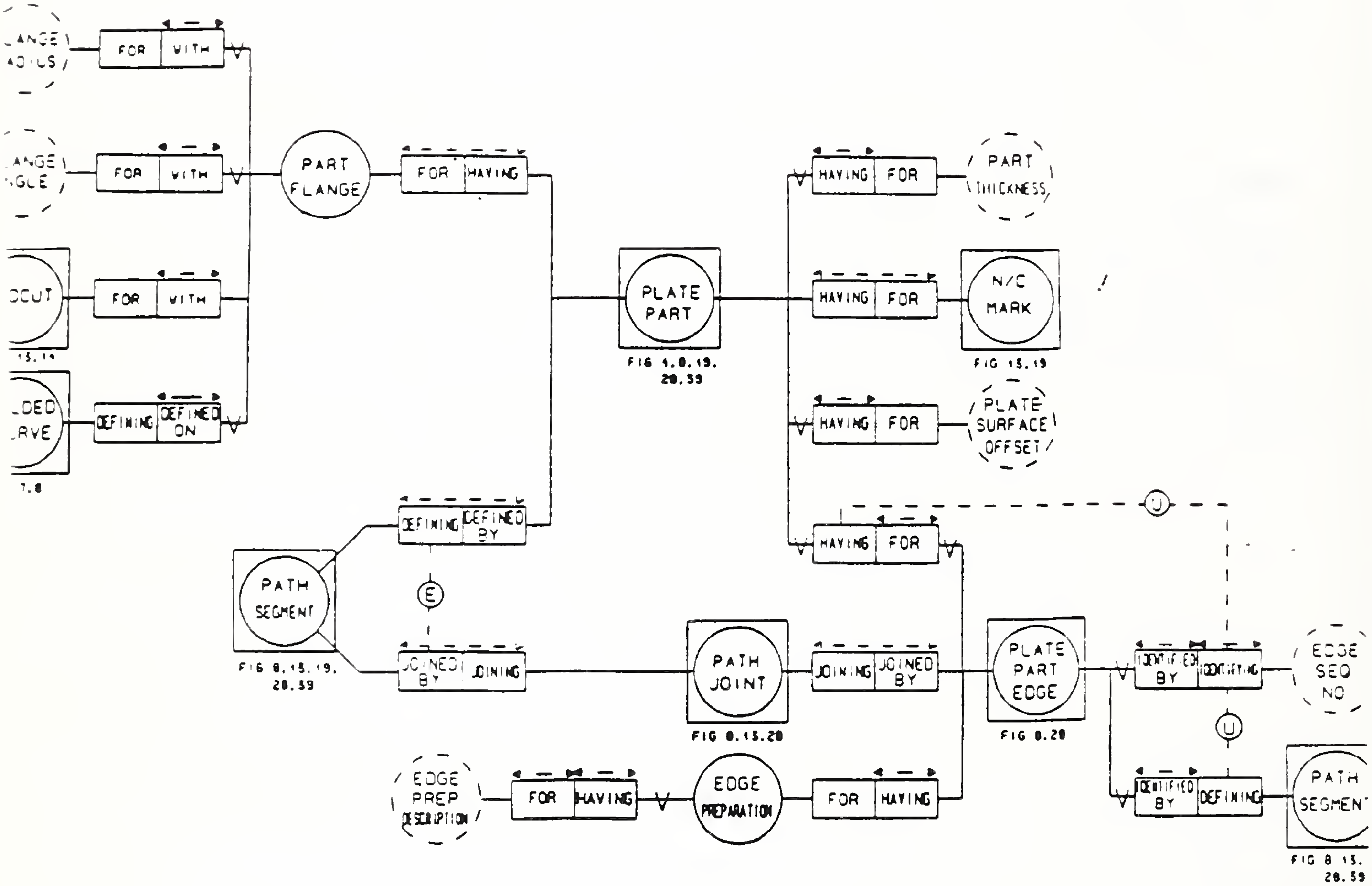
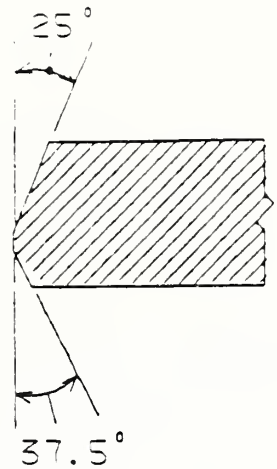
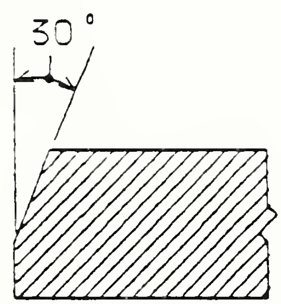


Figure D-70: NIAM Diagram showing Plate Part and Surface Relationships



EXAMPLE OF A PLATE PART WITH EDGE PREPARTION(DOUBLE BEVEL)



EXAMPLE OF A PLATE PART WITH EDGE PREPARTION(SINGLE BEVEL)

Figure D-71: Edge Preparation - Beveling

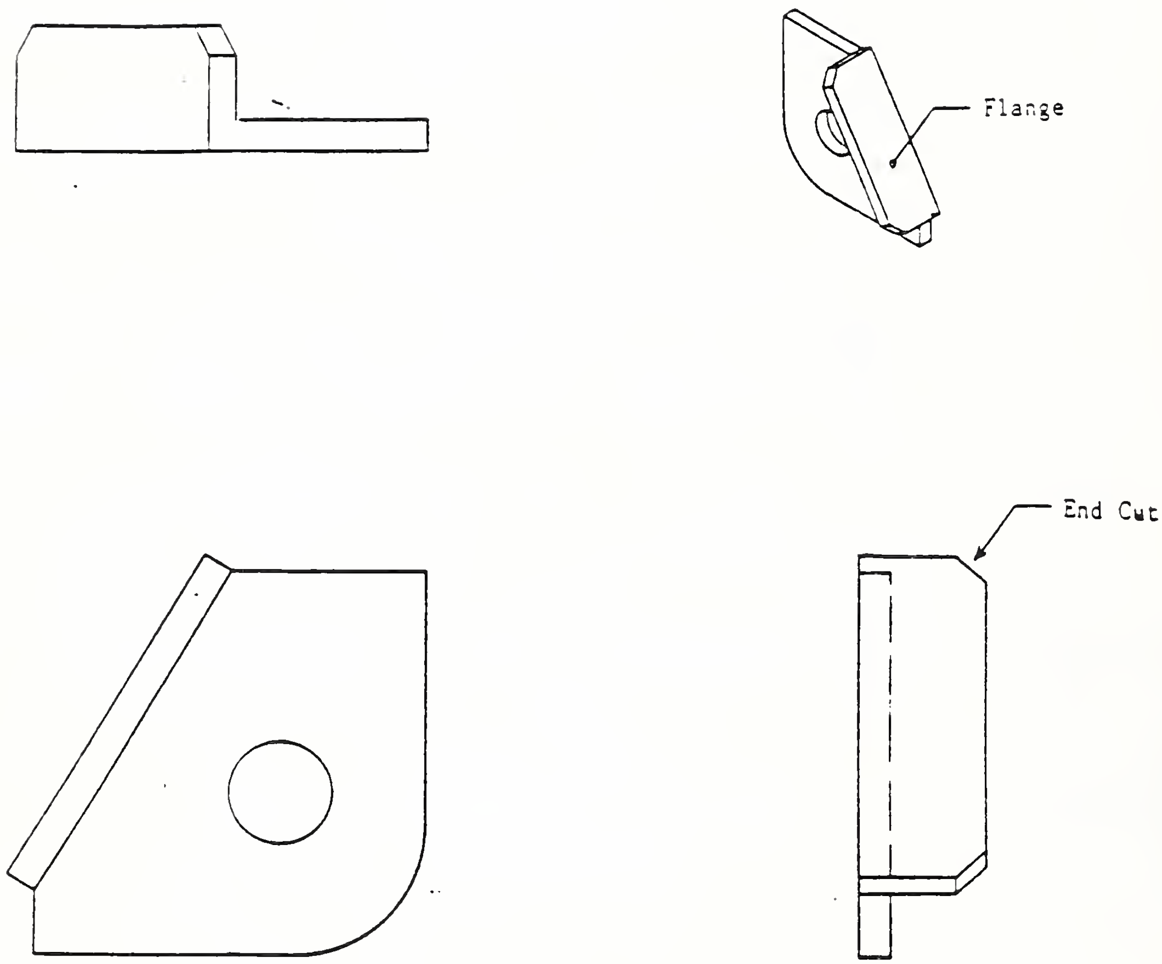


Figure D-72: Example of a Flanged Plate Part

Integration Point:

Product Structure Model; Product Item entity.

Business Rules:

- A SHAPE PART is a kind of PART.
- A SHAPE PART cannot also be a PLATE PART.
- A SHAPE PART may be cut by any number of STRUCTURAL OPENING's.
- A SHAPE PART must be defined on one or more PATH SEGMENT's.
- A SHAPE PART must be ending with exactly one ENDCUT.
- A SHAPE PART must be ending with exactly one SHAPE CLEARANCE.
- A SHAPE PART must be identified with exactly one CROSS SECTION.
- A SHAPE PART must be identified with exactly one SHAPE PART TYPE.
- A SHAPE PART must have one or more SHAPE PART EDGEs.
- A SHAPE PART may be joined by any number of NODAL JOINT's.
- A SHAPE PART must be located by exactly one SHAPE REFERENCE POINT. (Figure 82)
- A SHAPE PART may be marked by any number of NC MARK's.
- A SHAPE PART must be offset by exactly one SHAPE SURFACE OFFSET. (Figure 83)
- A SHAPE PART must be oriented by one or more SHAPE ORIENTATION's.
- A SHAPE PART may be penetrating any number of CUTOUT HOLE's.
- A SHAPE PART must be starting with exactly one ENDCUT.
- A SHAPE PART must be starting with exactly one SHAPE CLEARANCE.

Entity Name: **SHAPE CLEARANCE** **NOLOT**

Entity Number: **SS-30** **(Figure 76)**

A SHAPE PART CLEARANCE is the distance between the end nodes defining a path segment, and the actual extreme starting and ending points of the shape part defined on that path segment. In effect, the actual length of a shape part may be $\leq$ the length of the path segment it is defined on.

Business Rules:

- A SHAPE CLEARANCE may be at end of any number of SHAPE PART's.
- A SHAPE CLEARANCE may be at start of any number of SHAPE PART's.
- A SHAPE CLEARANCE must have exactly one CLEARANCE LENGTH.

Entity Name: **CROSS SECTION** **NOLOT**

Entity Number: **SS-31** **(Figure 77)**

A CROSS SECTION provides a description of the cross section geometry (a cut perpendicular to the linear axis) of the shape part. CROSS SECTION descriptions may be parametric or non-parametric.

Business Rules:

- All CROSS SECTION's must be NON-STANDARD CROSS SECTION's, or STANDARD CROSS SECTION's.
- A CROSS SECTION may be for any number of SHAPE PART's.
- A CROSS SECTION must be identified by exactly one CROSS SECTION TYPE.

Entity Name: **SHAPE PART EDGE** **NOLOT**

Entity Number: **SS-32**

A SHAPE PART EDGE is an ordered sequence of path segments defining the edge of a SHAPE PART.

Business Rules:

- A SHAPE PART EDGE must be defined by exactly one PATH SEGMENT.
- A SHAPE PART EDGE must be identified by exactly one EDGE SEQUENCE NUMBER.
- A SHAPE PART EDGE must be identified by exactly one SHAPE PART.
- A SHAPE PART EDGE may be joined by any number of PATH JOINT's.
- A SHAPE PART EDGE may have at most one EDGE PREPARATION.
- Every SHAPE PART EDGE is associated uniquely with one combination of

  a SHAPE PART identifying the SHAPE PART EDGE and
  an EDGE SEQUENCE NUMBER identifying the SHAPE PART EDGE and
  a PATH SEGMENT defining the SHAPE PART EDGE.

Entity Name: **SHAPE ORIENTATION** NOLOT

Entity Number: **SS-33** **(Figure 78)**

A SHAPE ORIENTATION defines the orientation of a shape part at a node. Multiple SHAPE ORIENTATIONs can be used to represent a twisted shape part.

Business Rules:

- A SHAPE ORIENTATION must be defined at one or more NODE's.
- A SHAPE ORIENTATION must be defined by exactly one UNIT VECTOR.
- A SHAPE ORIENTATION may be orienting any number of SHAPE PART's.
- Every SHAPE ORIENTATION is associated uniquely with one combination of

  a UNIT VECTOR defining the SHAPE ORIENTATION and
  a NODE locating the SHAPE ORIENTATION.

Entity Name: **STANDARD CROSS SECTION** NOLOT

Entity Number: **SS-34** **(Figure 79)**

A STANDARD CROSS SECTION is a parametric description whose parameters are provided through the unified numbering system established in accordance with ASTM and SAE.

Business Rules:

- A STANDARD CROSS SECTION is a kind of CROSS SECTION.
- A STANDARD CROSS SECTION cannot also be a NONSTANDARD CROSS SECTION.
- A STANDARD CROSS SECTION must have one or more CROSS SECTION PARAMETERs.
- A STANDARD CROSS SECTION must be with exactly one STANDARD SHAPE ID.

Entity Name: **NON-STANDARD CROSS SECTION** NOLOT

Entity Number: **SS-35**

A non-STANDARD CROSS SECTION is a non-parametric description. Specific parameters associated with web and flanges, for a specific shape part type, are provided by this description.

Business Rules:

- A non-STANDARD CROSS SECTION is a kind of CROSS SECTION.
- A non-STANDARD CROSS SECTION cannot also be a STANDARD CROSS SECTION.
- A non-STANDARD CROSS SECTION may be with at most one LOWER FLANGE THICKNESS.
- A non-STANDARD CROSS SECTION may be with at most one OWER FLANGE WIDTH.
- A non-STANDARD CROSS SECTION may be with at most one UPPER FLANGE THICKNESS.
- A non-STANDARD CROSS SECTION may be with at most one UPPER FLANGE WIDTH
- A non-STANDARD CROSS SECTION must be with exactly one WEB HEIGHT.
- A non-STANDARD CROSS SECTION must be with exactly one WEB THICKNESS.

Entity Name: CROSS SECTION PARAMETER NOLOT

Entity Number: SS-36

A CROSS SECTION PARAMETER describes an attribute of a shape part cross section. These include web and flange dimensions, fillet radius, etc.

Business Rules:

- A CROSS SECTION PARAMETER may be for any number of STANDARD CROSS SECTION's.
- A CROSS SECTION PARAMETER must have exactly one CROSS SECTION CODE.
- A CROSS SECTION PARAMETER must have exactly one CROSS SECTION VALUE.
- Every CROSS SECTION PARAMETER is associated uniquely with one combinati

  a CROSS SECTION CODE for the CROSS SECTION PARAMETER and

  a CROSS SECTION VALUE for the CROSS SECTI ON PARAMETER.

Entity Name: NC MARK NOLOT

Entity Number: SS-37 (Figure 80)

An NC MARK is a piece of text or contour marked on a plate part or shape part by a CAM device such as a burning machine or robot. Such marking is typically performed by a punch marker, zinc marker or an ink-jet marker.

Business Rules:

- All NC MARK's must be NC TEXT's, or TRACE MARK's.
- A NC MARK must be identified by exactly one NC MARK ID.
- A NC MARK must be identified by exactly one NC MARK TYPE.
- A NC MARK may be marking any number of PLATE PART's.
- A NC MARK may be marking any number of SHAPE PART's.

Entity Name: **NC TEXT** **NOLOT**

Entity Number: **SS-38**

A NC TEXT is a type of NC mark consisting of a character string.

Business Rules:

- A NC TEXT is a kind of NC MARK.
- A NC TEXT cannot also be a TRACE MARK.
- A NC TEXT must be defined with exactly one NC TEXT STRING.
- A NC TEXT must be identified by at most one NC TEXT PARAMETER.
- A NC TEXT must be located at exactly one NODE.
- A NC TEXT must be oriented by exactly one UNIT VECTOR.
- Every NC TEXT is associated uniquely with one combination of
  an UNIT VECTOR orienting the NC TEXT and
  a NODE locating the NC TEXT.

Entity Name: **NC TEXT PARAMETER** **NOLOT**

Entity Number: **SS-39**

A NC TEXT PARAMETER describes an attribute of an NC text mark, ie. size, font, etc.

Business Rules:

- A NC TEXT PARAMETER may be for one or more NC TEXTs.
- A NC TEXT PARAMETER must have exactly one NC TEXT PARAMETER CODE.
- A NC TEXT PARAMETER must have exactly one NC TEXT PARAMETER VALUE.
- Every NC TEXT PARAMETER is associated uniquely with one combination of

  a NC TEXT PARAMETER CODE for the NC TEXT PARAMETER and
  a NC TEXT PARAMETER VALUE for the NC TEXT PARAMETER.

Entity Name: **ENDCUT** **NOLOT**

Entity Number: **SS-40** **(Figure 81)**

The ENDCUT provides the flange/web cut details at the ends of a shape part.

Integration Point:

Form Features Model.

Business Rules:

- All ENDCUT's must be non-PARAMETRIC ENDCUT's, or PARAMETRIC ENDCUT's.
- An ENDCUT may be at end of any number of SHAPE PART's.
- An ENDCUT may be at start of any number of SHAPE PART's.
- An ENDCUT must be defined with exactly one ENDCUT TYPE.
- An ENDCUT may be for any number of PART FLANGE's.

Entity Name: **ENDCUT PARAMETER** **NOLOT**

Entity Number: **SS-41**

An ENDCUT PARAMETER describes an attribute of a parametric endcut. For example, snipe and radius are two attributes of an endcut.

Business Rules:

- An ENDCUT PARAMETER may be for any number of PARAMETRIC ENDCUT's.
- An ENDCUT PARAMETER must have exactly one ENDCUT PARAMETER CODE.
- An ENDCUT PARAMETER must have exactly one ENDCUT PARAMETER VALUE.
- Every ENDCUT PARAMETER is associated uniquely with one combination of

  an ENDCUT PARAMETER CODE for the ENDCUT PARAMETER and
  an ENDCUT PARAMETER VALUE for the ENDCUT PARAMETER.

Entity Name: **PARAMETRIC ENDCUT** NOLOT

Entity Number: **SS-42**

A PARAMETRIC ENDCUT is an endcut with the cut geometry specified by reference to a standard endcut and its associated endcut parameters.

Business Rules:

- A PARAMETRIC ENDCUT is a kind of ENDCUT.
- A PARAMETRIC ENDCUT cannot also be a non-PARAMETRIC ENDCUT.
- A PARAMETRIC ENDCUT must have one or more ENDCUT PARAMETERs.
- A PARAMETRIC ENDCUT must be with exactly one PARAMETRIC ENDCUT ID.

Entity Name: **NON-PARAMETRIC ENDCUT** NOLOT

Entity Number: **SS-43**

A NON-PARAMETRIC ENDCUT is an endcut with non-standard cut geometry.

Business Rules:

- A NON-PARAMETRIC ENDCUT is a kind of ENDCUT.
- A NON-PARAMETRIC ENDCUT cannot also be a PARAMETRIC ENDCUT.
- A NON-PARAMETRIC ENDCUT may have at most one CURVE GEOMETRY describing the endcut of

  a LOWER FLANGE
  an UPPER FLANGE.

Entity Name: **TRACE MARK** NOLOT

Entity Number: **SS-44**

A TRACE MARK is a numerically controlled (N/C) marking contour

Business Rules:

- A TRACE MARK is a kind of NC MARK.
- A TRACE MARK cannot also be a NC TEXT.
- A TRACE MARK must be defined on exactly one PATH SEGMENT.
- A TRACE MARK must be with exactly one MARK INTERVAL.
- A TRACE MARK must be with exactly one MARK LENGTH.

Library Parts

Parts which are used throughout the ship with geometrically similar shape and size may be modeled once in a part library and used again and again wherever needed. Chocks, brackets and gussets are examples of possible library parts. These may also be referred to as relocatable parts.

There are two types of library parts; parametric and non-parametric. Non-parametric library parts are defined with a specific geometry and always look the same wherever used. Parametric library parts are defined through the use of standard geometric shapes with a corresponding set of parameters. Values for these parameters are input at the time of usage, therefore, two different occurrences of the same parametric library part may appear physically different.

Library parts are specified by their library identification, by their occurrence (as a part), by their location (at a node), by their translation and orientation from the node via a transformation entity, and parameters if necessary. The actual geometry of the library part is not a part of this model. It is intended that the geometry be generated by the host CAD system from the above mentioned attributes. Library parts, being a subtype of part, all carry part attributes such as material, date-time modified/created, and association with a unit-assembly or sub-assembly.

Various issues concerning library part geometry need further attention. These issues involve the details of part geometry representation within a library, cross referencing between libraries of different systems, and methods for the exchange or merging of different libraries. Such issues are currently being addressed by other TC184/SC4/WG1 participants and within the Marine industry. It is intended that enhancements will be added to this model in the future.

The NIAM diagram in Figure 85 illustrates the concepts of Library Part.

Examples of library parts can be found in Figures 86, 87, and 88. Figures 86 and 87. show tight and non-tight collar pieces. Such collars are used to enhance the structural effectiveness of a shape-part penetrating plate-part. In the case of tight collars, fluid (typically fuel or water) is isolated from one side of the penetration to the other. Figure 88 illustrates another type of library part, a tripping bracket.

The relationships shown in Figure 85 are:

Entity Name: **PART LIBRARY** **NOLOT**

Entity Number: **SS-45**

A PART LIBRARY is a collection of standard parts, also refered to as a standard parts catalog, which provides a pre-defined part for use in the Ship Product Model.

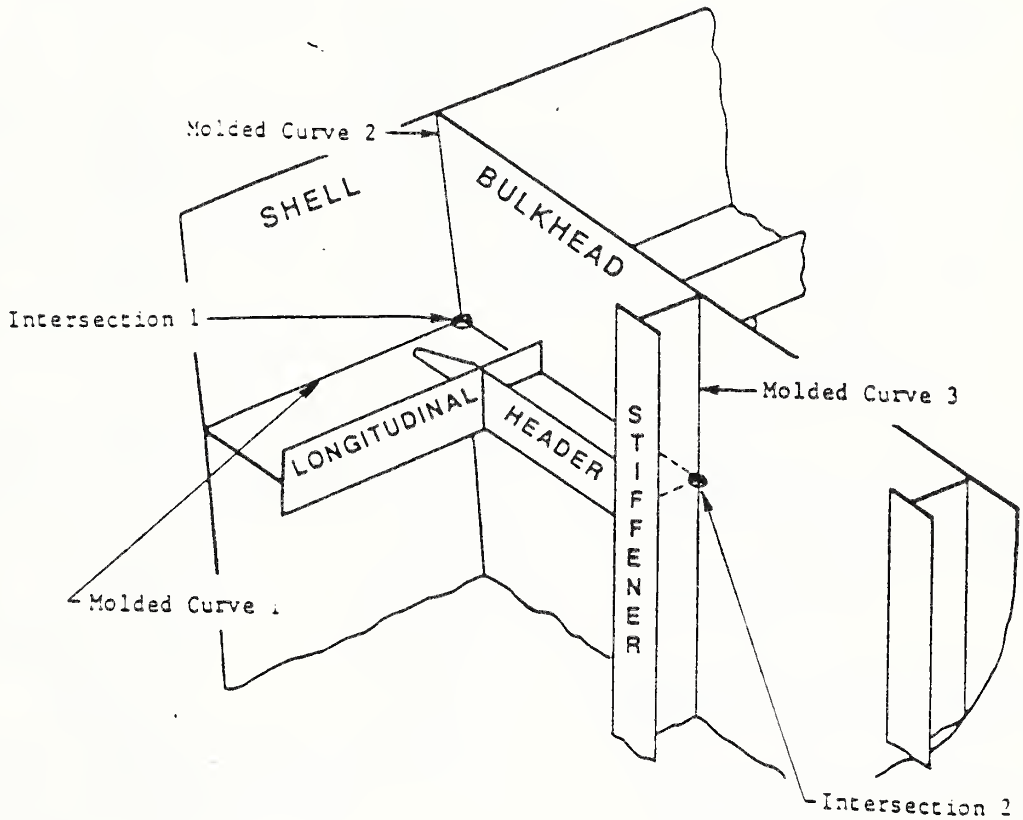


Figure D-73: Example of Structural Stiffeners

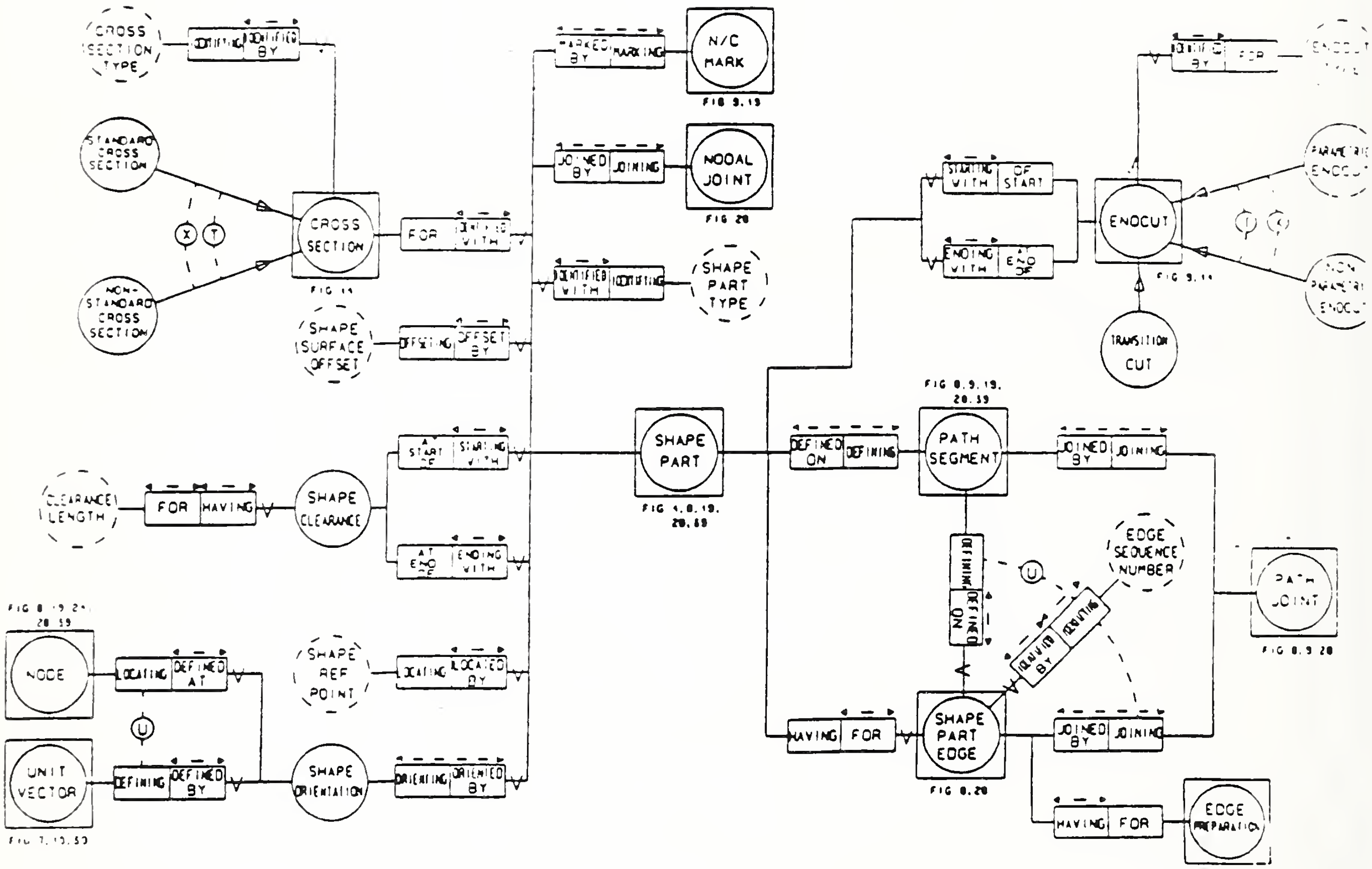


Figure D-74: NIAM Diagram showing Structural Shape Relationships

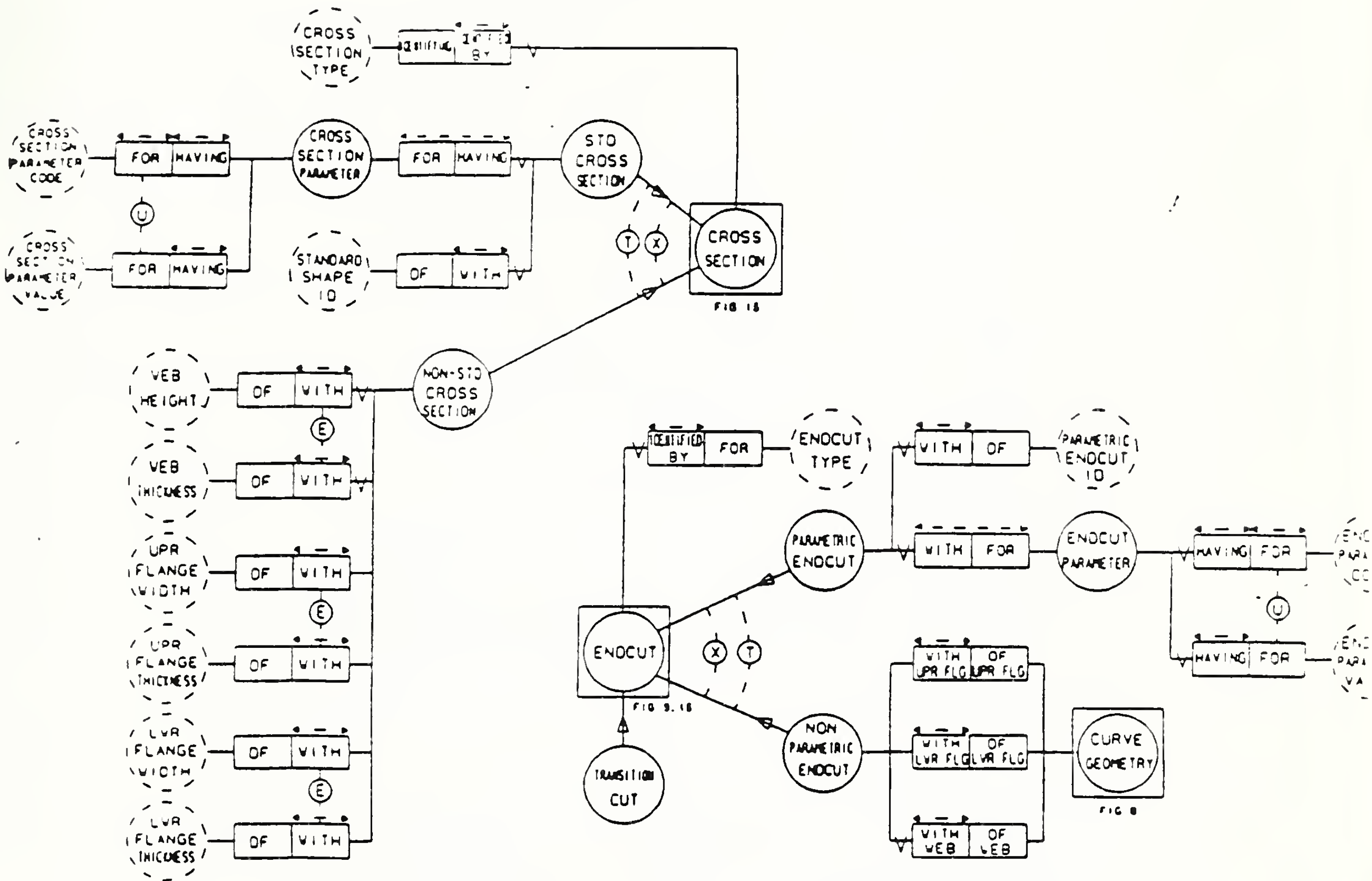


Figure D-75: NIAM Diagram showing Structural Shape Relationships

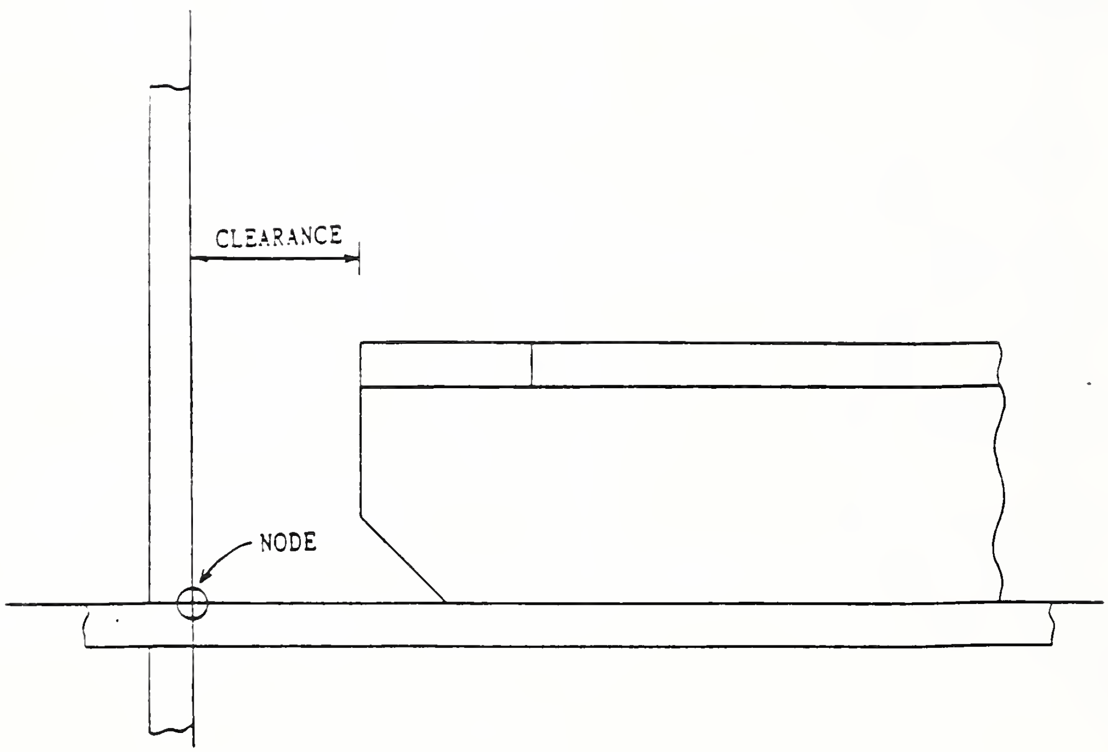


Figure D-76: Shape Part End Clearance

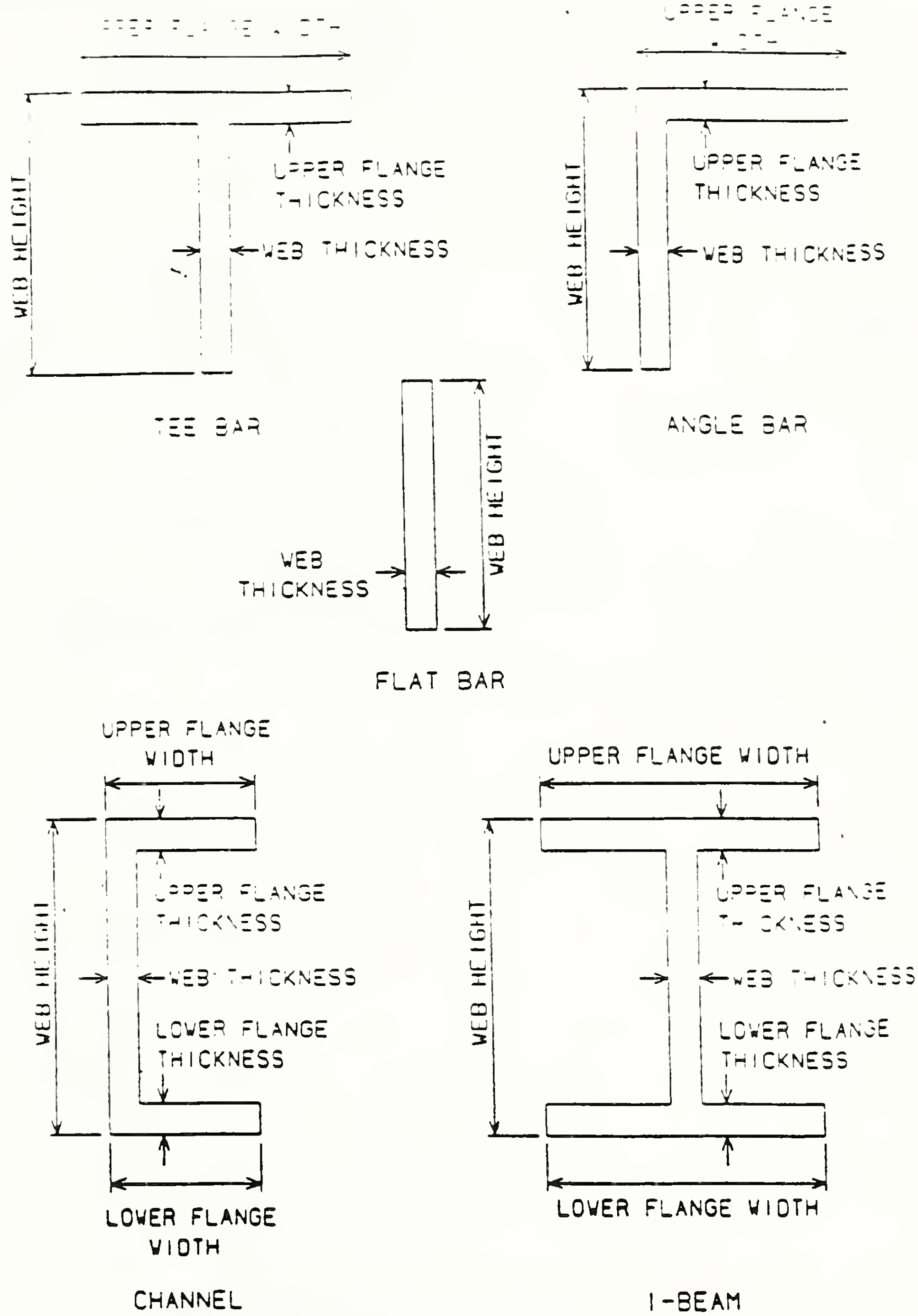
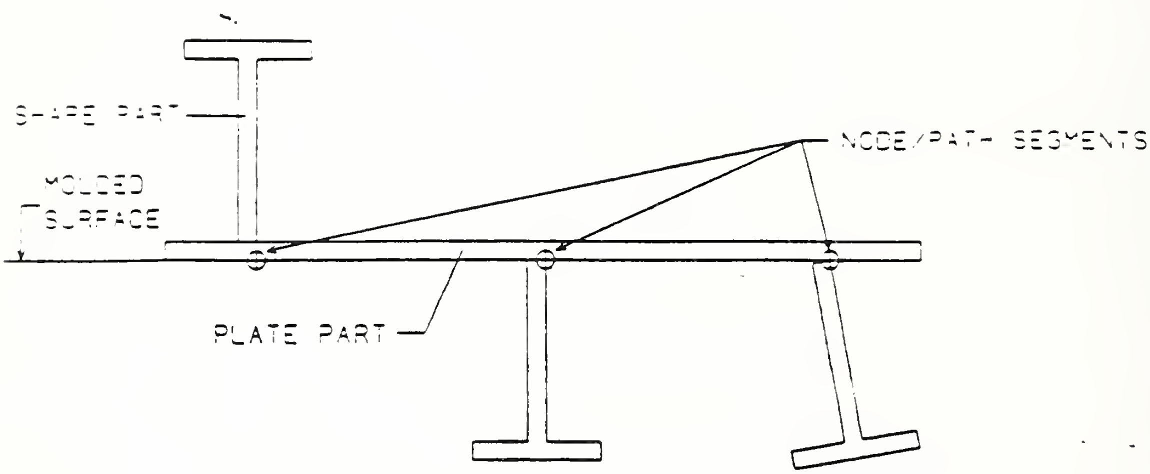


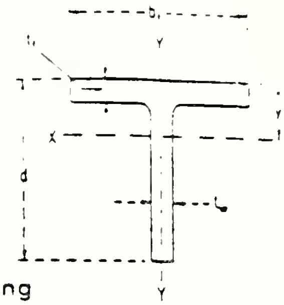
Figure D-77: Shape Part Cross Sections



EXAMPLE OF VARIOUS SHAPE ORIENTATIONS

Figure D–78: Shape Part Orientation

STRUCTURAL TEES (Cut from W Shapes)



Theoretical Dimensions and Properties for Designing

Section Number	Weight per Foot	Area of Section A	Depth of Section d	Flange Width b_f	Flange Thickness t_f	Stem Thickness t_w	Axis X-X I_x	Axis X-X S_x	Axis X-X r_x	Axis X-X y	Axis Y-Y I_y	Axis Y-Y S_y	Axis Y-Y r_y
	lb	in.^{2}	in.	in.	in.	in.	in.^{4}	in.^{3}	in.	in.	in.^{4}	in.^{3}	in.
WT18 x	150	44.1	18.370	16.655	1.680	0.945	1230	86.1	5.27	4.13	648	77.8	3.83
	140	41.2	18.260	16.595	1.570	0.885	1140	80.0	5.25	4.07	599	72.2	3.81
	130	38.2	18.130	16.550	1.440	0.840	1060	75.1	5.26	4.05	545	65.9	3.78
	122.5	36.0	18.040	16.510	1.350	0.800	995	71.0	5.26	4.03	507	61.4	3.75
	115	33.8	17.950	16.470	1.260	0.760	934	67.0	5.25	4.01	470	57.1	3.73
WT18 x	105	30.9	18.345	12.180	1.360	0.830	985	73.1	5.65	4.87	206	33.8	2.58
	97	28.5	18.245	12.115	1.260	0.765	901	67.0	5.62	4.80	187	30.9	2.56
	91	26.8	18.165	12.075	1.180	0.725	845	63.1	5.62	4.77	174	28.8	2.55
	85	25.0	18.085	12.030	1.100	0.680	786	58.9	5.61	4.73	160	25.6	2.53
	80	23.5	18.005	12.000	1.020	0.650	740	55.8	5.61	4.74	147	24.6	2.50
	75	22.1	17.925	11.975	0.940	0.625	698	53.1	5.62	4.78	135	22.5	2.47
	67.5	19.9	17.775	11.950	0.790	0.600	636	49.7	5.66	4.96	113	18.9	2.38
WT16.5 x	120.5	35.4	17.090	15.860	1.400	0.830	871	65.8	4.96	3.85	466	58.8	3.63
	110.5	32.5	16.965	15.805	1.275	0.775	799	60.8	4.96	3.81	420	53.2	3.59
	100.5	29.5	16.840	15.745	1.150	0.715	725	55.5	4.95	3.78	375	47.6	3.56
WT16.5 x	76	22.4	16.745	11.565	1.055	0.635	592	47.4	5.14	4.26	136	23.6	2.47
	70.5	20.8	16.650	11.535	0.960	0.605	552	44.7	5.15	4.29	123	21.3	2.43
	65	19.2	16.545	11.510	0.855	0.580	513	42.1	5.18	4.36	109	18.9	2.39
	59	17.3	16.430	11.480	0.740	0.550	469	39.2	5.20	4.47	93.6	16.3	2.32
WT15 x	105.5	31.0	15.470	15.105	1.315	0.775	610	50.5	4.43	3.40	378	50.1	3.49
	95.5	28.1	15.340	15.040	1.185	0.710	549	45.7	4.42	3.35	336	44.7	3.46
	86.5	25.4	15.220	14.985	1.065	0.655	497	41.7	4.42	3.31	299	39.9	3.43
WT15 x	66	19.4	15.155	10.545	1.000	0.615	421	37.4	4.66	3.90	98.0	18.6	2.25
	62	18.2	15.085	10.515	0.930	0.585	396	35.3	4.66	3.90	90.4	17.2	2.23
	58	17.1	15.005	10.495	0.850	0.565	373	33.7	4.67	3.94	82.1	15.7	2.19
	54	15.9	14.915	10.475	0.760	0.545	349	32.0	4.69	4.01	73.0	13.9	2.15
	49.5	14.5	14.825	10.450	0.670	0.520	322	30.0	4.71	4.09	63.9	12.2	2.10

Note: Taken from Bethlehem Steel Catalog of Structural Shapes (Catalog 3277C)

Figure D-79: Standard Cross Section Parameters (Codes and Values) for a Typical Structural Shape

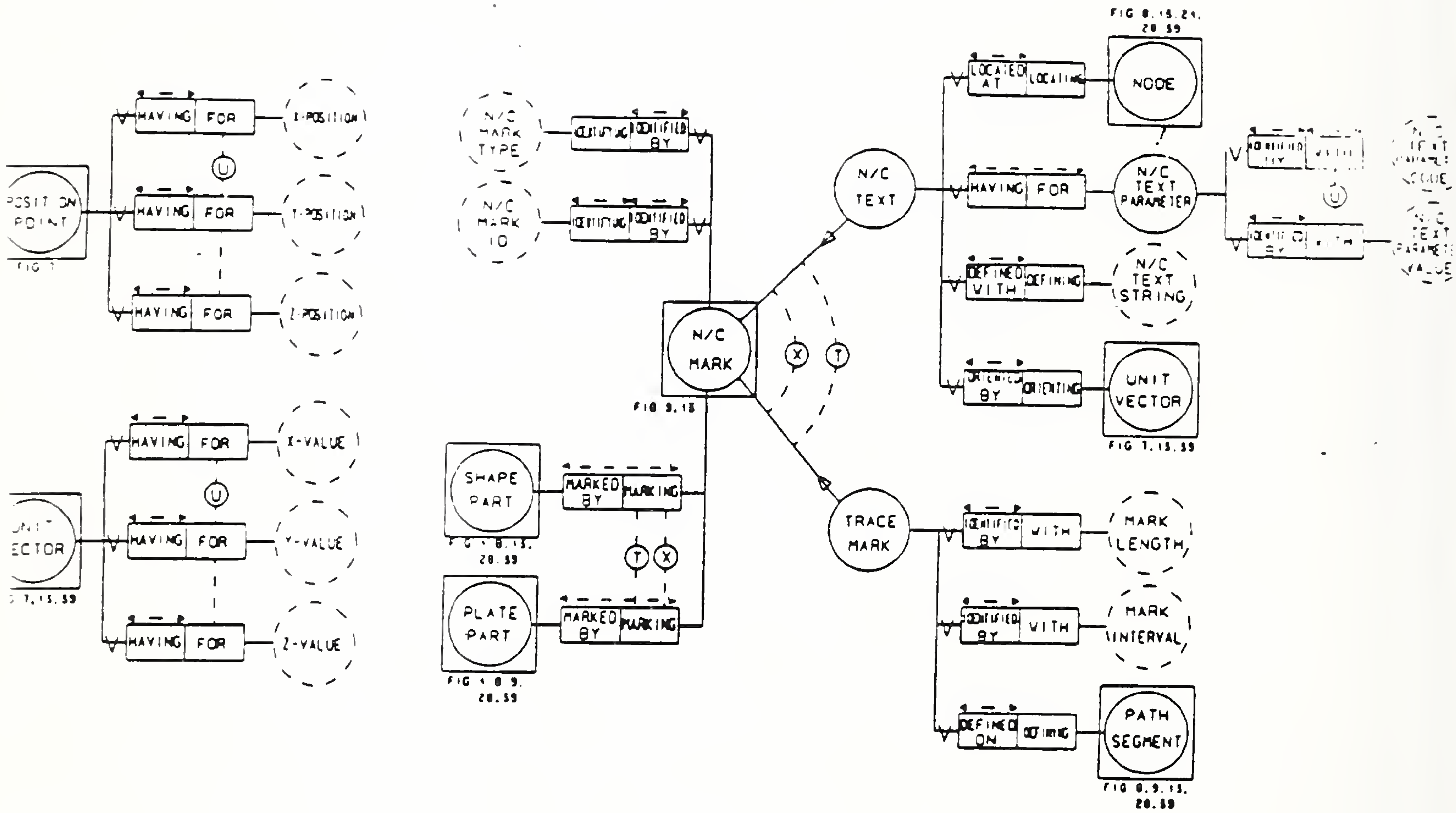


Figure D-80: NIAM Diagram showing N/C Data Relationships

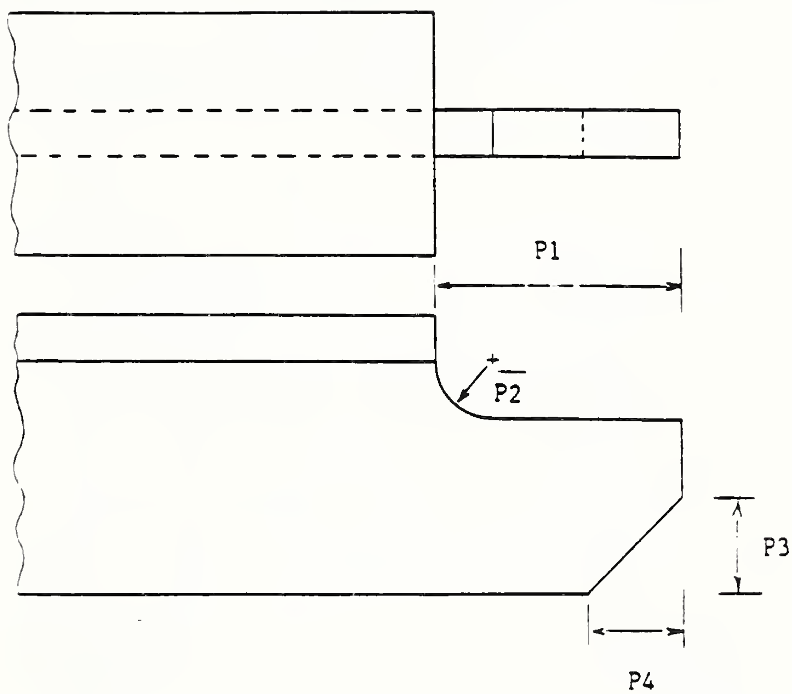


Figure D-81: Shape Part End Cut (Parametric)

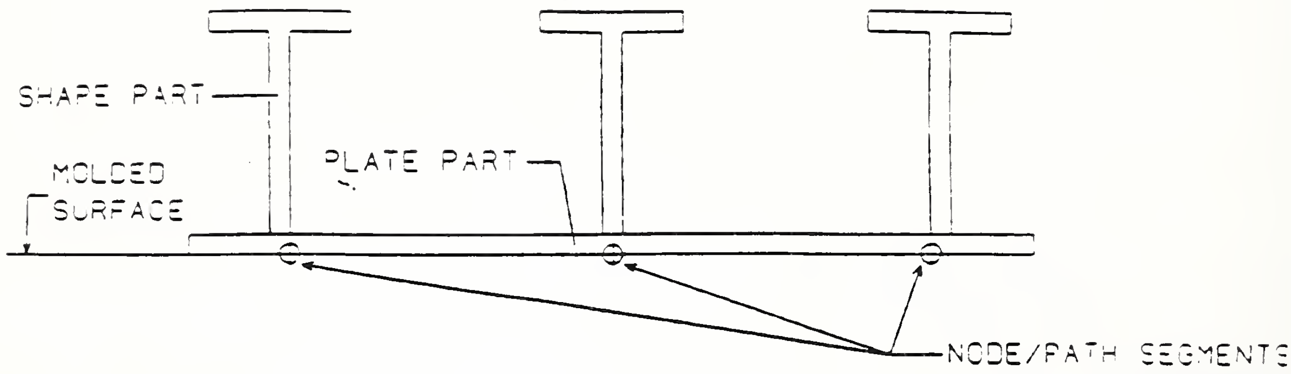


Figure D-82: Shape Part Reference Point

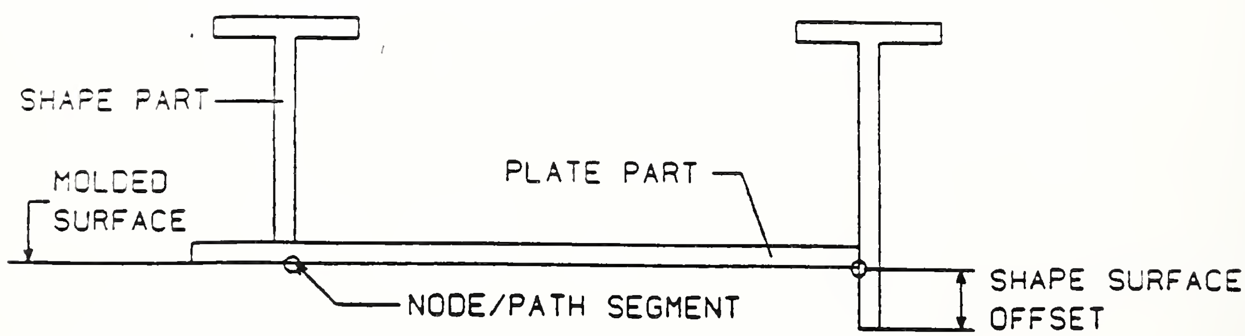


Figure D-83: Shape Part Surface Offset

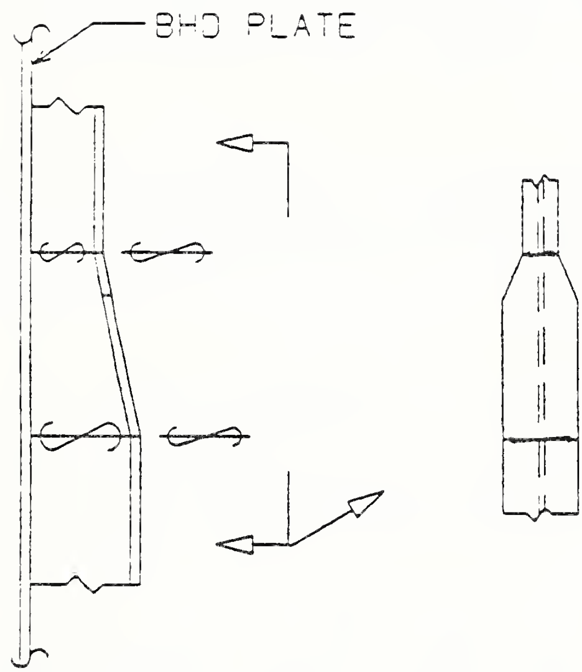


Figure D-84: Example of a Shape Part Transition Piece

Business Rules:

- A PART LIBRARY must contain one or more LIBRARY PART's.
- A PART LIBRARY must be identified by exactly one LIBRARY ID.
- A PART LIBRARY must be identified by exactly one LIBRARY VERSION.
- Every PART LIBRARY is associated uniquely with one combination of

  a LIBRARY ID identifying the PART LIBRARY and
  a LIBRARY VERSION identifying the PART LIBRARY.

Entity Name: LIBRARY PART NOLOT

Entity Number: SS-46

A LIBRARY PART is a pre-defined part contained in a library. There are two types of LIBRARY PARTS, parametric and non-parametric. Chocks, brackets and gussets are examples of LIBRARY PARTS.

Business Rules:

- A LIBRARY PART is always a kind of PART.
- A LIBRARY PART cannot also be a PLATE PART or SHAPE PART.
- All LIBRARY PART's must be NON-PARAMETRIC LIBRARY PART's, or PARAMETRIC LIBRARY PART's.
- A LIBRARY PART must be defined in one or more PART LIBRARY's.
- A LIBRARY PART must be identified by exactly one LIBRARY PART ID.
- A LIBRARY PART must be identified by exactly one LIBRARY PART TYPE.
- A LIBRARY PART may be joined by any number of NODAL JOINT's.
- A LIBRARY PART must be located by exactly one NODE.
- A LIBRARY PART must be oriented by exactly one TRANSFORMATION MATRIX.

Entity Name: NON-PARAMETRIC LIBRARY PART NOLOT

Entity Number: SS-47

A non-PARAMETRIC LIBRARY PART is a type of library part whose geometry (shape, size) is the same for every occurence.

Business Rules:

- A NON-PARAMETRIC LIBRARY PART is a kind of LIBRARY PART.
- A NON-PARAMETRIC LIBRARY PART cannot also be a PARAMETRIC LIBRARY PART.

Entity Name: **PARAMETRIC LIBRARY PART** **NOLOT**

Entity Number: **SS-48**

A PARAMETRIC LIBRARY PART is a type of library part whose geometry is necessarily defined by both the library part ID and parameter values associated with a particular library part occurence. Two PARAMETRIC LIBRARY PARTS with the same library part ID may or may not be physically identical.

Business Rules:

- A PARAMETRIC LIBRARY PART is a kind of LIBRARY PART.
- A PARAMETRIC LIBRARY PART cannot also be a NON-PARAMETRIC LIBRARY PART.
- A PARAMETRIC LIBRARY PART must have one or more LIBRARY PART PARAMETERs

Entity Name: **PART PARAMETER** **NOLOT**

Entity Number: **SS-49**

A PART PARAMETER describes the attributes associated with a PARAMETRIC LIBRARY PART. These include shape and size.

Business Rules:

- A PART PARAMETER may be for any number of PARAMETRIC LIBRARY PART's.
- A PART PARAMETER must have exactly one PART PARAMETER CODE.
- A PART PARAMETER must have exactly one PART PARAMETER VALUE.
- Every PART PARAMETER is associated uniquely with one combination of

  a PART PARAMETER CODE for the PART PARAMETER and
  a PART PARAMETER VALUE for the PART PARAMETER.

Entity Name: **TRANSFORMATION MATRIX** **NOLOT**

Entity Number: **SS-50**

A TRANSFORMATION MATRIX is a 4 x 4 matrix specifying the translation and orientation (rotation) of a library PART.

Integration Point:

Geometry Model; Transformation entity.

Business Rules:

- A TRANSFORMATION MATRIX may be orienting any number of LIBRARY PART's.

7.2.4 Structural Joint

A joint is a connection between plate parts and/or shape parts. Usually the connection is made by welding. Examples of typical joints can be seen in Figures 90 through 95.

The relationships expressed by the NIAM diagram in Figure 89 are:

Entity Name: **JOINT** **NOLOT**

Entity Number: **SS-51**

A JOINT is used to define the physical connection between shape parts and plate parts. JOINTs occur at path segments and nodes. The type of joint is identified as either path joint or nodal joint.

Business Rules:

- All JOINT's must be NODAL JOINT's, or PATH JOINT's.
- A JOINT must be identified by exactly one JOINT ID.
- A JOINT must have exactly one JOINT TYPE.
- A JOINT must be joined by exactly one JOINING PROCEDURE.
- A JOINT may be penetrating any number of CUTOUT HOLE's.
- A JOINT must refer to exactly one STANDARD DETAILS REFERENCE.
- A JOINT may be with at most one INSPECTION PROCEDURE.

Entity Name: **BOLT JOINT** **NOLOT**

Entity Number: **SS-52**

A BOLT JOINT is a type of joint using threaded fastenings to secure two or more parts together.

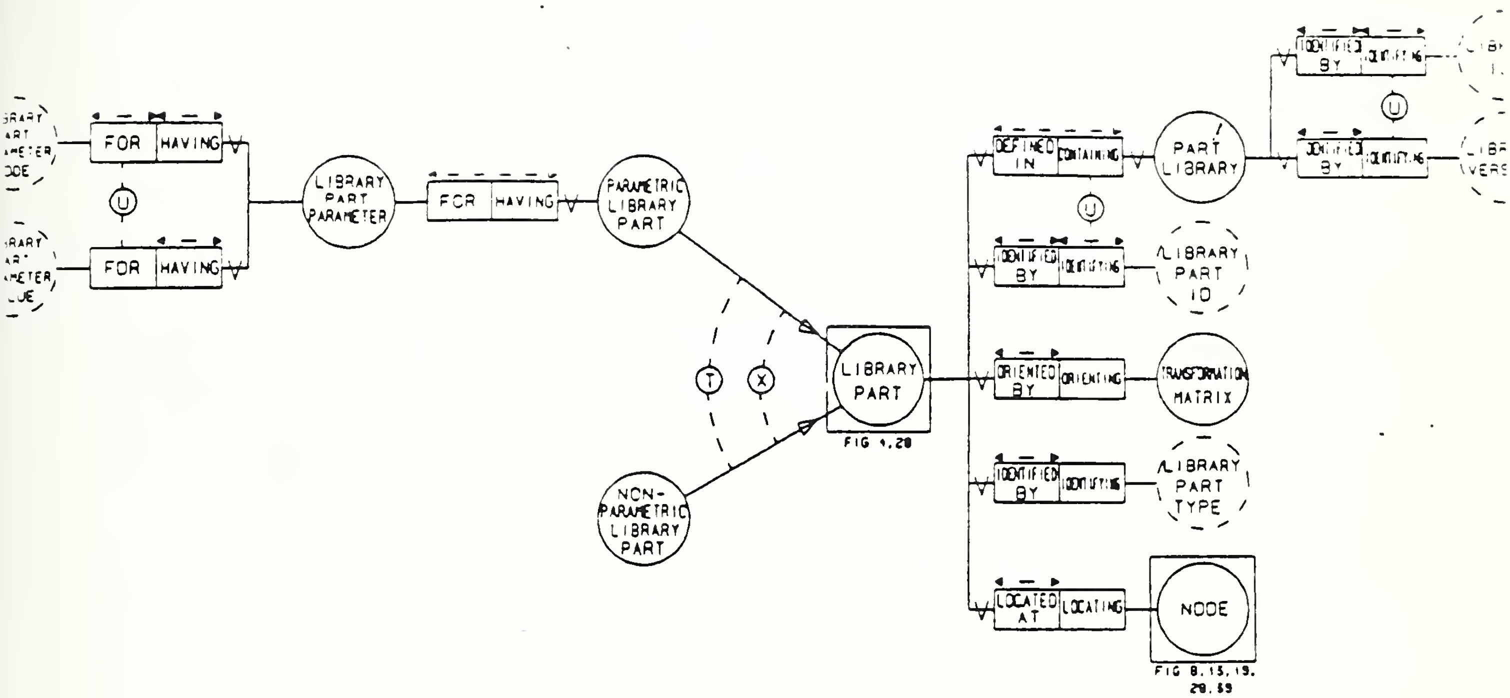


Figure D-85: NIAM Diagram showing Library Part Relationships

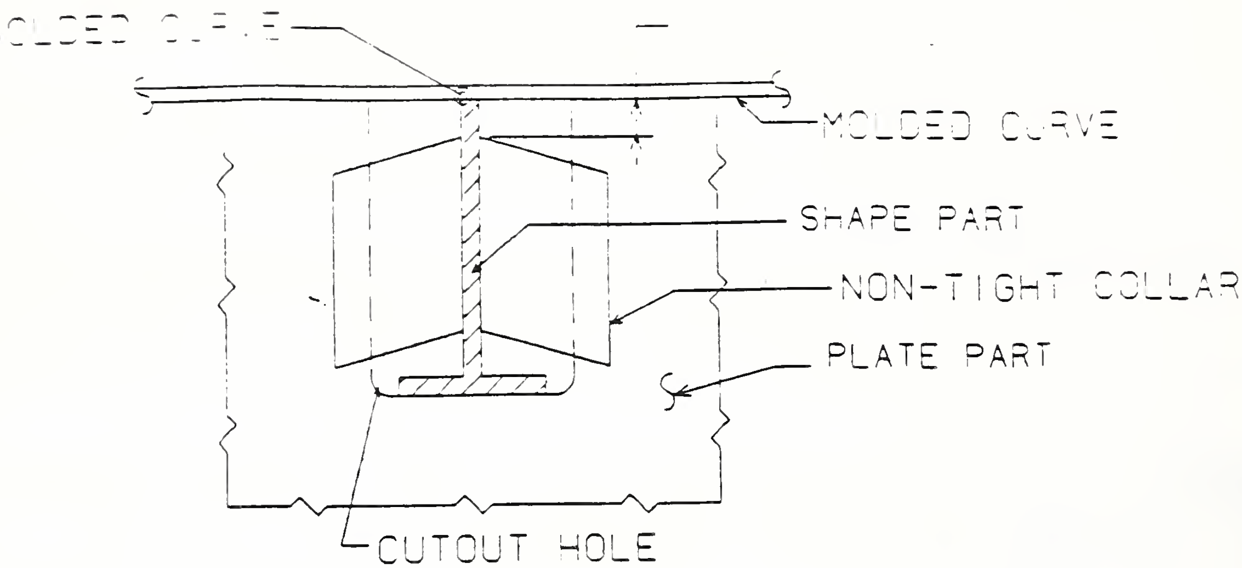


Figure D-86: Example of a Cutout Hole and a Non-Tight Collar//(Library Part)

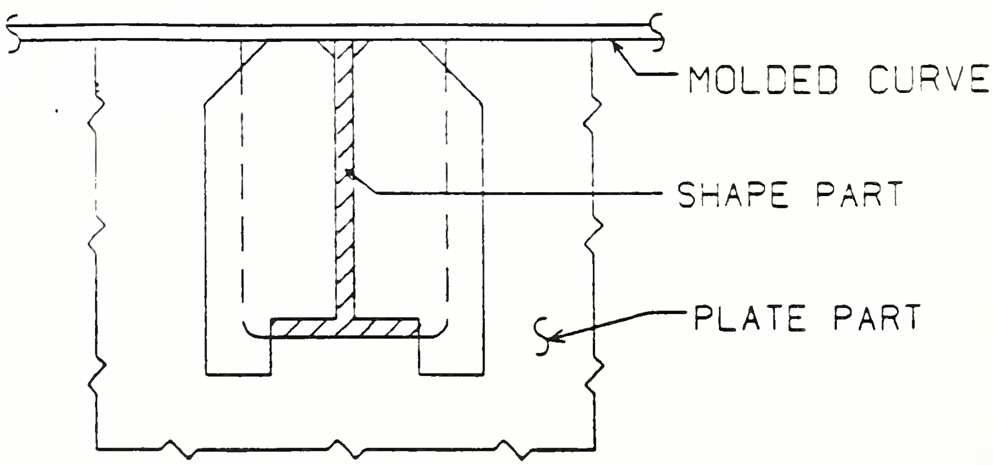


Figure D-87: Example of a Cutout Hole and a Tight Collar//(Library Part)

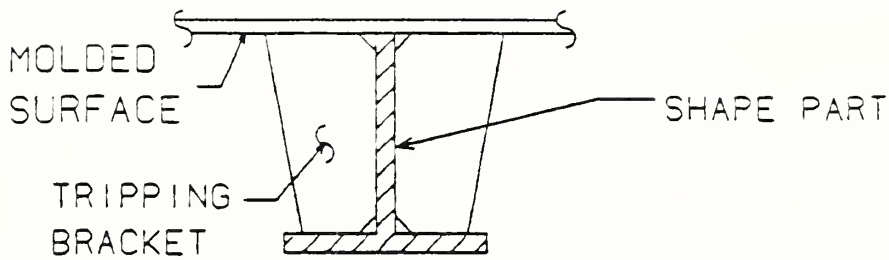


Figure D-88: Example of a Tripping Bracket (Library Part)

Business Rules:

- A BOLT JOINT is a kind of joint.
- A BOLT JOINT cannot also be a RIVET JOINT, or a WELD JOINT.
- A BOLT JOINT must be described by one or more BOLT PARAMETER's.
- A BOLT JOINT must be described by exactly one BOLT PROCESS.

Entity Name: **BOLT PARAMETER** **NOLOT**

Entity Number: **SS-53**

A BOLT PARAMETER describes an attribute of a bolt used in a bolt joint, ie. bolt diameter, bolt length, threads/inch etc.

Business Rules:

- A BOLT PARAMETER may be for any number of BOLT JOINT's.
- A BOLT PARAMETER must have exactly one BOLT PARAMETER CODE.
- A BOLT PARAMETER must have exactly one BOLT PARAMETER VALUE.
- Every BOLT PARAMETER is associated uniquely with one combination of

  a BOLT PARAMETER CODE for the BOLT PARAMETER and
  a BOLT PARAMETER VALUE for the BOLT PARAMETER.

Entity Name: **NODAL JOINT** **NOLOT**

Entity Number: **SS-54**

A NODAL JOINT is a type of joint. A NODAL-JOINT is a connection between at least two parts occuring at a node.

Business Rules:

- A NODAL JOINT is a kind of JOINT.
- A NODAL JOINT cannot also be a PATH JOINT.
- A NODAL JOINT may be joining any number of LIBRARY PART's.
- A NODAL JOINT may be joining any number of PLATE PART's.
- A NODAL JOINT may be joining any number of SHAPE PART's.
- A NODAL JOINT must be located at exactly one NODE.

Entity Name: **PATH JOINT** NOLOT

Entity Number: **SS-55**

A PATH JOINT is a type of joint. A PATH JOINT is a connection between at least two part (shape or plate) edges.

Business Rules:

- A PATH JOINT is a kind of JOINT.
- A PATH JOINT cannot also be a NODAL JOINT.
- A PATH JOINT may be joining any number of PATH SEGMENT's.
- A PATH JOINT may be joining any number of PLATE PART EDGE's.
- A PATH JOINT may be joining any number of SHAPE PART EDGE's.

Entity Name: **RIVET JOINT** NOLOT

Entity Number: **SS-56**

A RIVET JOINT is a type of joint using rivets to securly fasten two or more parts together.

Business Rules:

- A RIVET JOINT is a kind of JOINT.
- A RIVET JOINT cannot also be a BOLT JOINT, or a WELD JOINT.
- A RIVET JOINT must have one or more RIVET PARAMETERs.
- A RIVET JOINT must be with exactly one RIVET PROCESS.

Entity Name: **RIVET PARAMETER** NOLOT

Entity Number: **SS-57**

A RIVET PARAMETER contains an attribute of a rivet used in a riveted joint, ie. rivet diameter, rivet length, etc.

Business Rules:

- A RIVET PARAMETER may be for any nu mberof RIVET JOINT's.
- A RIVET PARAMETER must have exactly one RIVET PARAMETER CODE.
- A RIVET PARAMETER must have exactly one RIVET PARAMETER VALUE.
- Every RIVET PARAMETER is associated uniquely with one combination of

  a RIVET PARAMETER CODE for the RIVET PARAMETER and
  a RIVET PARAMETER VALUE for the RIVET PARAMETER.

Entity Name: **WELD JOINT** **NOLOT**

Entity Number: **SS-58**

A WELD JOINT is a type of joint using material fusion to join parts together along a seam. This is the most common type of joint used in shipbuilding today.

Business Rules:

- A WELD JOINT is a kind of JOINT.
- A WELD JOINT cannot also be a BOLT JOINT, or a RIVET JOINT.
- A WELD JOINT must be of exactly one WELD SIZE.
- A WELD JOINT must be with exactly one WELD PROCESS.
- A WELD JOINT must be with exactly one WELD TYPE.

7.2.5 Structural Openings

Structural openings consist of a variety of different types, namely; lightening or access holes, distribution system penetrations, and cutouts through which stiffeners pass. These basic types are illustrated in Figures 96 through 100. All structural openings have been classified into these three basic types for the following reasons. Access/lightening holes are either voids or have doors/closures attached to them; distribution system penetrations are openings through which either vent duct, electrical cable, or pipe penetrate; and cutouts are holes through which other structures penetrate.

A structural opening can be defined as a parametric opening or a non-parametric opening. Non-parametric openings have their location and geometry defined by a path segment. Parametric openings are identified by a parametric opening type, a reference to a standard library opening or a macro. Their locations are specified by nodes and they are oriented by vectors.

Again for a variety of reasons, it is necessary to capture both geometry and topology of holes.

The NIAM diagram shown in Figure 101 expresses the following relationships:

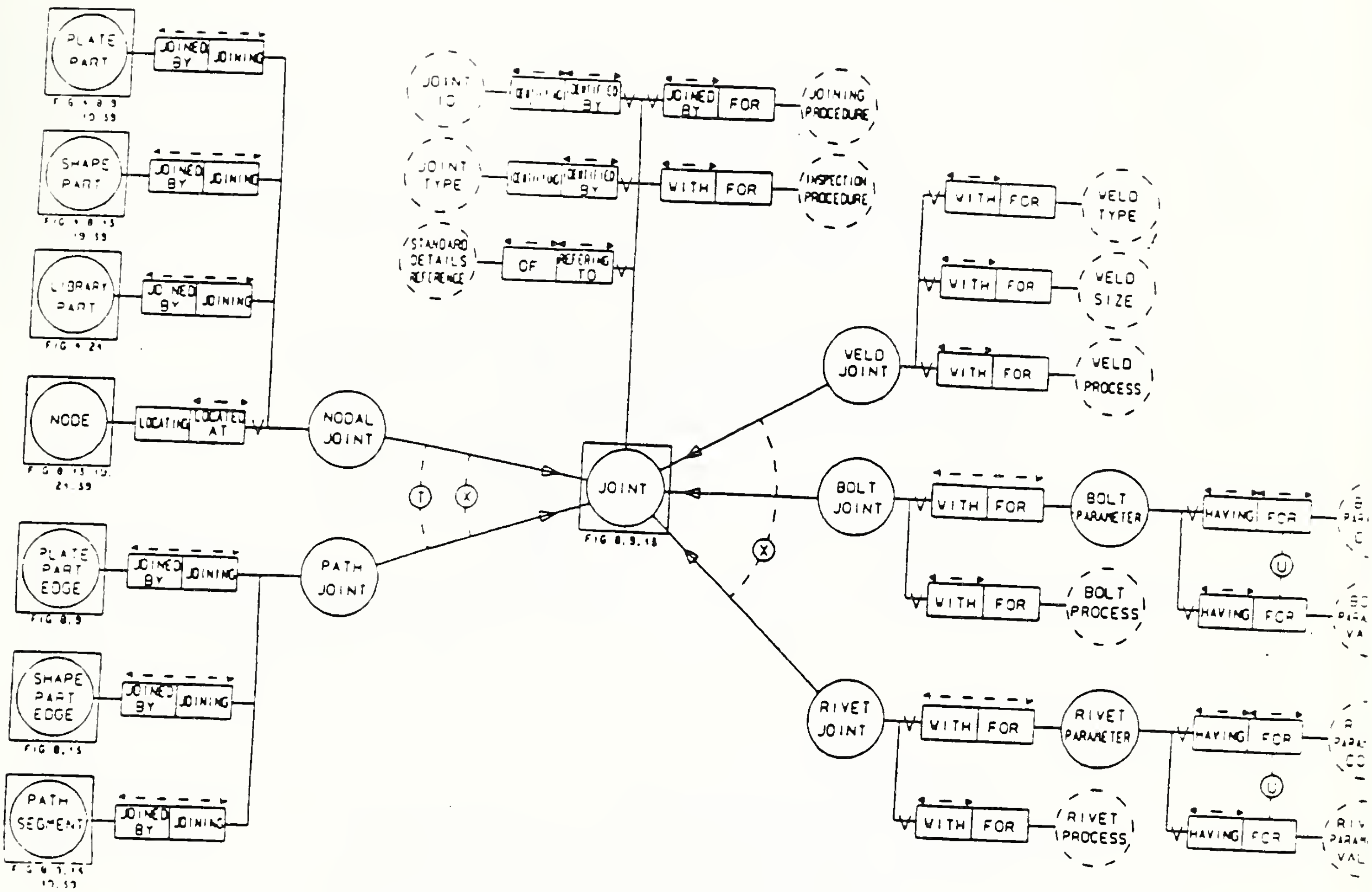
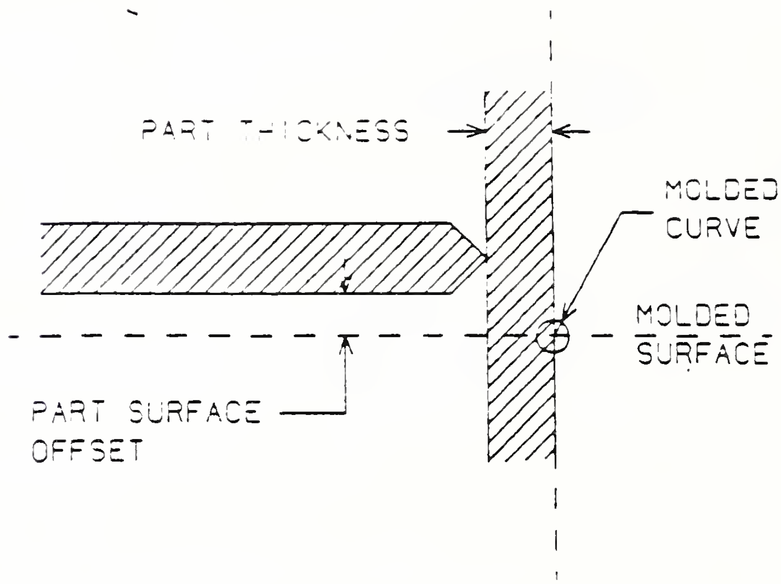


Figure D-89: NIAM Diagram showing Joint Relationships



JOINT BETWEEN TWO PARTS ILLUSTRATING DERIVATION OF ACTUAL PART GEOMETRY FROM THE INTERSECTION OF TWO SURFACES (MOLDED CURVE) USING PART THICKNESS AND PART SURFACE OFFSET.

Figure D-90: Plate Thickness and Surface Offset

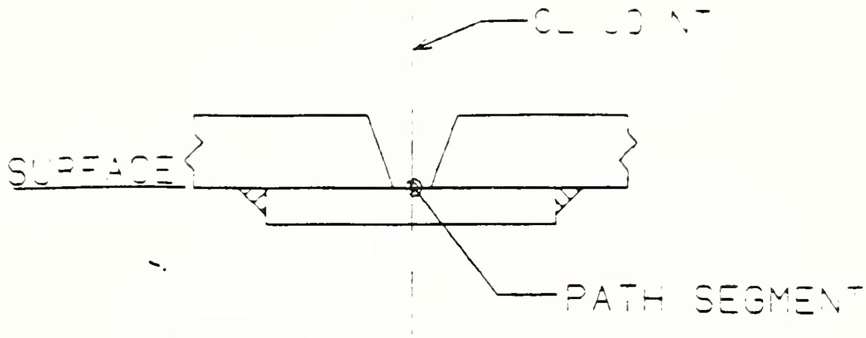


Figure D-91: Example of a Joint between two Plates Parts with Edge Preparation and a Welding Process using a Backing Strip

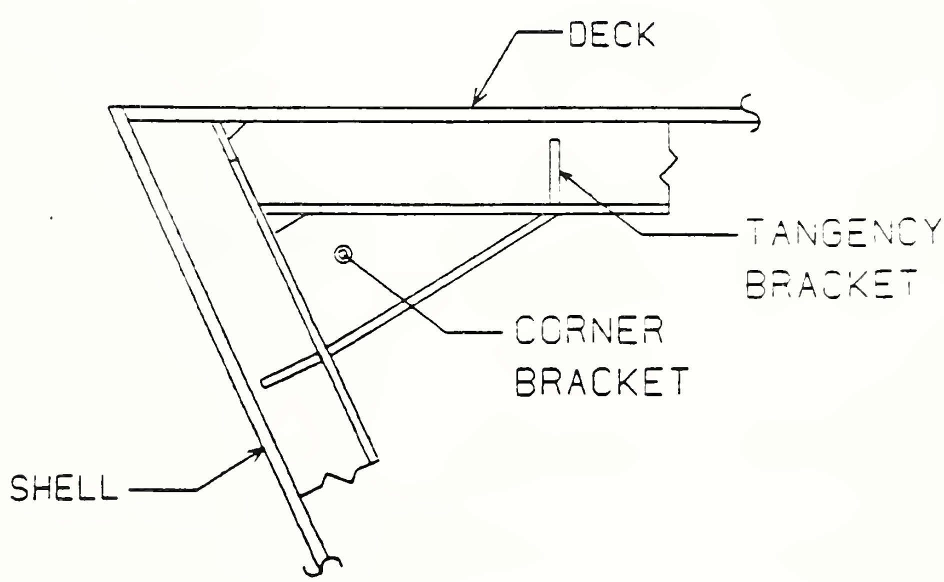


Figure D-92: Example of a Deck/Shell Intersection with Plate/Shape Parts

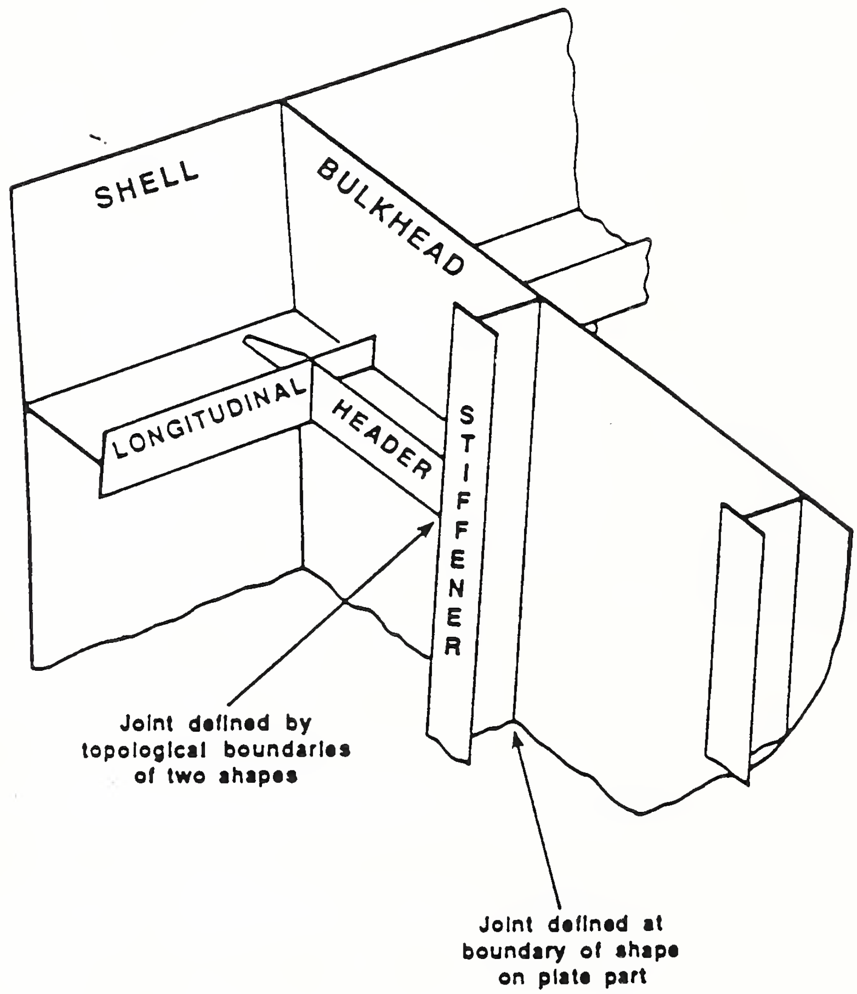


Figure D–93: Joints between Plate Parts and Shape Parts

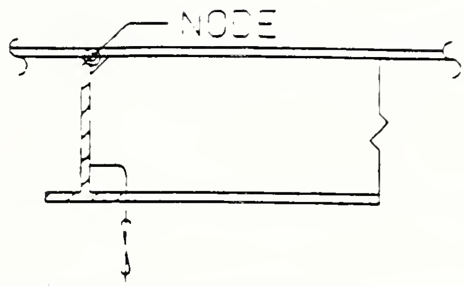


Figure D-94: Example of two intersecting Shape Parts

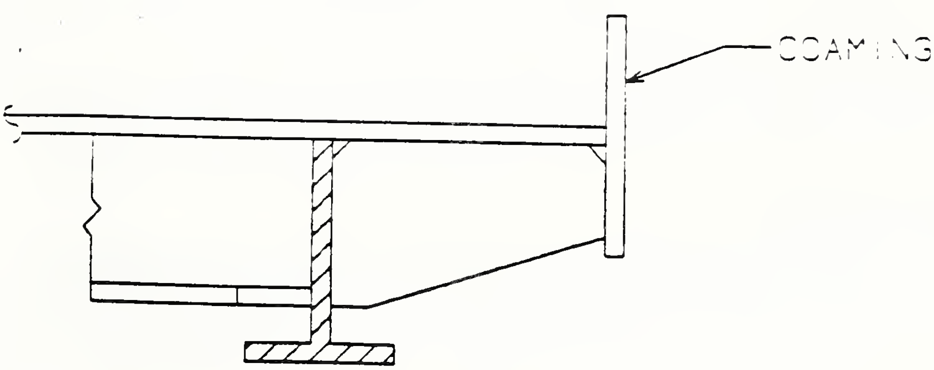


Figure D-95: Detail of a Typical Opening Showing a Coaming as a Plate Part

Entity Name: **STRUCTURAL OPENING** **NOLOT**

Entity Number: **SS-59**

A STRUCTURAL OPENING is an opening in a part to allow penetration of another part, penetration of a distribution system part, passage of a fluid or to lighten the part.

Business Rules:

- All STRUCTURAL OPENING's must be NON-PARAMETRIC OPENING's, or PARAMETRIC OPENING's.
- All STRUCTURAL OPENING's must be ACCESS/LIGHTENING HOLE's, or CUTOUT HOLE's, or SYSTEM PENETRATION HOLE's.
- A STRUCTURAL OPENING may cut any number of PLATE PART's.
- A STRUCTURAL OPENING may cut any number of SHAPE PART's.
- A STRUCTURAL OPENING must be defined by exactly one HOLE TYPE.
- A STRUCTURAL OPENING may be defined on at most one BOUNDED SURFACE.
- A STRUCTURAL OPENING must be identified by exactly one HOLE ID.
- A STRUCTURAL OPENING may have at most one EDGE PREPARATION.

Entity Name: **PARAMETRIC OPENING** **NOLOT**

Entity Number: **SS-60**

A PARAMETRIC OPENING is a type of structural opening whose opening is defined by one or more parameter codes and their corresponding opening parameter values.

Business Rules:

- A PARAMETRIC OPENING is a kind of STRUCTURAL OPENING.
- A PARAMETRIC OPENING cannot also be a non-PARAMETRIC OPENING.
- A PARAMETRIC OPENING must have one or more OPENING PARAMETER's.
- A PARAMETRIC OPENING must be identified by exactly one PARAMETRIC OPENING ID.
- A PARAMETRIC OPENING must be located at exactly one NODE.
- A PARAMETRIC OPENING must be oriented by exactly one UNIT VECTOR.
- Every PARAMETRIC OPENING is associated uniquely with one combination of

an UNIT VECTOR orienting the PARAMETRIC OPENING and
a NODE locating the PARAMETRIC OPENING.

Entity Name: **NON-PARAMETRIC OPENING** **NOLOT**

Entity Number: **SS-61**

A NON-PARAMETRIC OPENING is a type of structural opening whose opening is defined by a path segment representing the hole contour.

Business Rules:

- A NON-PARAMETRIC OPENING is a kind of STRUCTURAL OPENING.
- A NON-PARAMETRIC OPENING cannot also be a PARAMETRIC OPENING.
- A NON-PARAMETRIC OPENING must be defined by exactly one PATH SEGMENT.

Entity Name: **AIR ESCAPE** **NOLOT**

Entity Number: **SS-62**

An AIR ESCAPE is a type of cutout hole. Its function is to prevent air pockets from forming when fluids are being transfered into a tank.

Business Rules:

- An AIR ESCAPE is a kind of CUTOUT HOLE.
- An AIR ESCAPE cannot also be an OIL STOP, or a RATHOLE.

Entity Name: **ACCESS LIGHTENING HOLE** **NOLOT**

Entity Number: **SS-63**

An ACCESS/LIGHTENING HOLE is a type of structural opening used primarily for accessing areas of a hull or for lightening structure. Access/Lightening holes may be penetrated by distributive system parts.

Business Rules:

- An ACCESS LIGHTENING HOLE is a kind of STRUCTURAL OPENING.
- An ACCESS LIGHTENING HOLE cannot also be a SYSTEM PENETRATION HOLE.
- An ACCESS LIGHTENING HOLE may be penetrated by any number of DISTRIBUTION SYSTEM PART's.

Entity Name: **CUTOUT HOLE** **NOLOT**

Entity Number: **SS-64**

A CUTOUT HOLE is a type of structural opening. It is generally a hole in a plate part or a shape part to allow the penetration by a shape part. Other uses are rathole, air escape and oil stop. These holes are not penetrated by shape parts.

Business Rules:

- A CUTOUT HOLE is a kind of STRUCTURAL OPENING.
- A CUTOUT HOLE may be penetrated by at most one JOINT.
- A CUTOUT HOLE may be penetrated by at most one SHAPE PART.

Entity Name: **SYSTEM PENETRATION HOLE** **NOLOT**

Entity Number: **SS-65**

A SYSTEM PENETRATION HOLE is a type of structural opening that is penetrated by a distribution system part or a penetration part.

Business Rules:

- A SYSTEM PENETRATION HOLE is a kind of STRUCTURAL OPENING.
- A SYSTEM PENETRATION HOLE cannot also be an ACCESS LIGHTENING HOLE.
- A SYSTEM PENETRATION HOLE must be penetrated by one or more DISTRIBUTION SYSTEM PART's.
- A SYSTEM PENETRATION HOLE must be penetrated by exactly one PENETRATION PART.

Entity Name: **OIL STOP** **NOLOT**

Entity Number: **SS-66**

An OIL STOP is a type of cutout hole used in combination with a welded joint to form a fluid barrier between welded parts.

Business Rules:

- An OIL STOP is a kind of CUTOUT HOLE.
- An OIL STOP cannot also be an AIR ESCAPE, or a RATHOLE.

Entity Name: **RATHOLE** NOLOT

Entity Number: **SS-67**

A RATHOLE is a type of cutout hole.

Business Rules:

- A RATHOLE is a kind of CUTOUT HOLE.
- A RATHOLE cannot also be an AIR ESCAPE, or an OIL STOP.

Entity Name: **OPENING PARAMETER** NOLOT

Entity Number: **SS-68**

An OPENING PARAMETER is a set of one or more attributes defining the size and shape of a parametric (standard) opening. An opening parameter, for example, defines the radius of a circular opening.

Business Rules:

- An OPENING PARAMETER may be for any number of PARAMETRIC OPENING's.
- An OPENING PARAMETER must have exactly one OPENING PARAMETER CODE.
- An OPENING PARAMETER must have exactly one OPENING PARAMETER VALUE.
- Every OPENING PARAMETER is associated uniquely with one combination of

  an OPENING PARAMETER CODE for the OPENING PARAMETER and
  an OPENING PARAMETER VALUE for the OPENING PARAMETER.

Entity Name: **DISTRIBUTION SYSTEM PART** NOLOT

Entity Number: **SS-69**

A DISTRIBUTION SYSTEM PART is a part that distributes fluid, electricity, air, etc. A DISTRIBUTION SYSTEM PART may be enclosed by a penetration part (a sleeve or insert) when it penetrates a structural part.

Integration Point:

NIDDESC Outfit Model; Distribution System Part.

Business Rules:

- A DISTRIBUTION SYSTEM PART must be identified by exactly one DISTRIBUTIVE SYSTEM PART ID.
- A DISTRIBUTION SYSTEM PART may be penetrating any number of ACCESS LIGHT HOLE's.
- A DISTRIBUTION SYSTEM PART must be penetrating exactly one PENETRATION PART.
- A DISTRIBUTION SYSTEM PART may be penetrating any number of SYSTEM PENETRATION HOLE's.

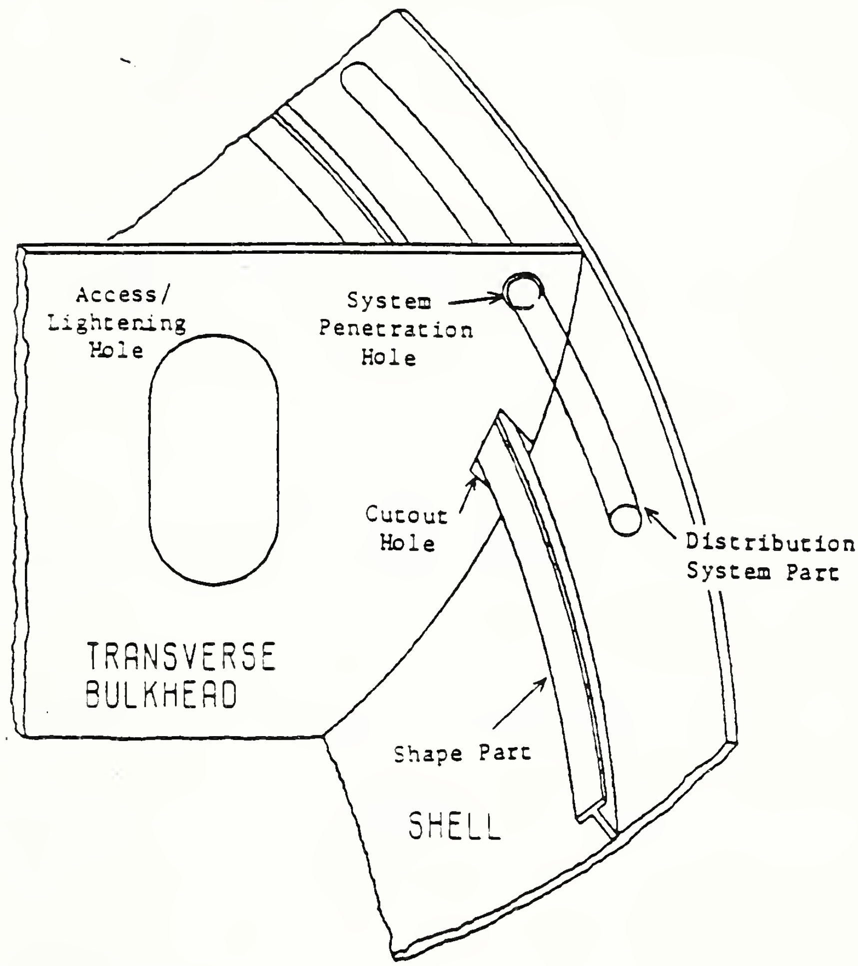


Figure D-96: Example of Structural Opening

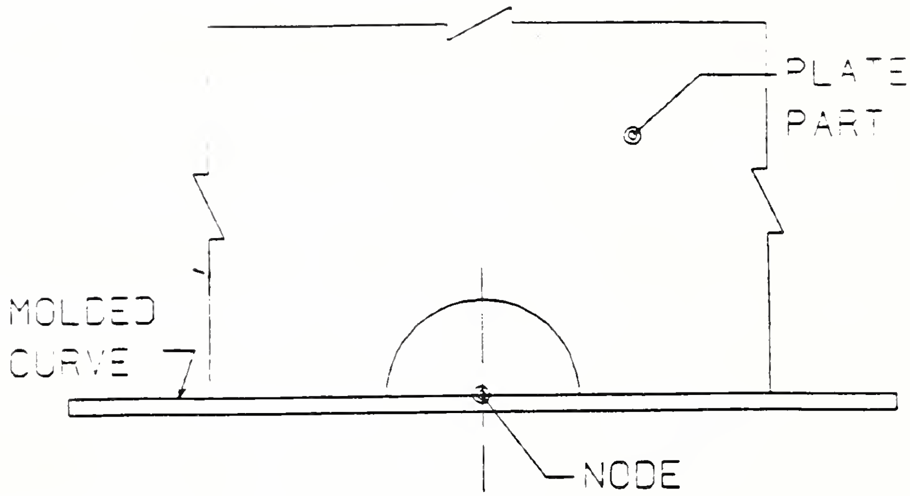


Figure D-97: Example of a Drain Hole on a Plate Part located at a Node on a Molded Curve

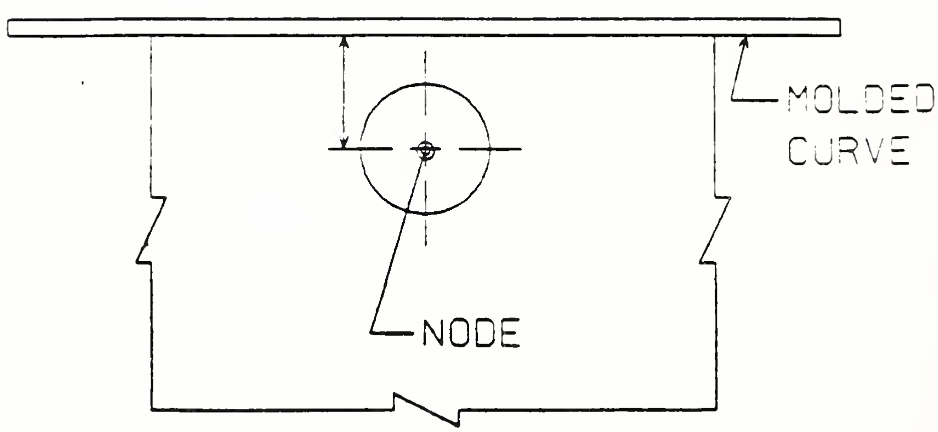


Figure D-98: Example of an Air Escape located at a node on a Bounded Surface

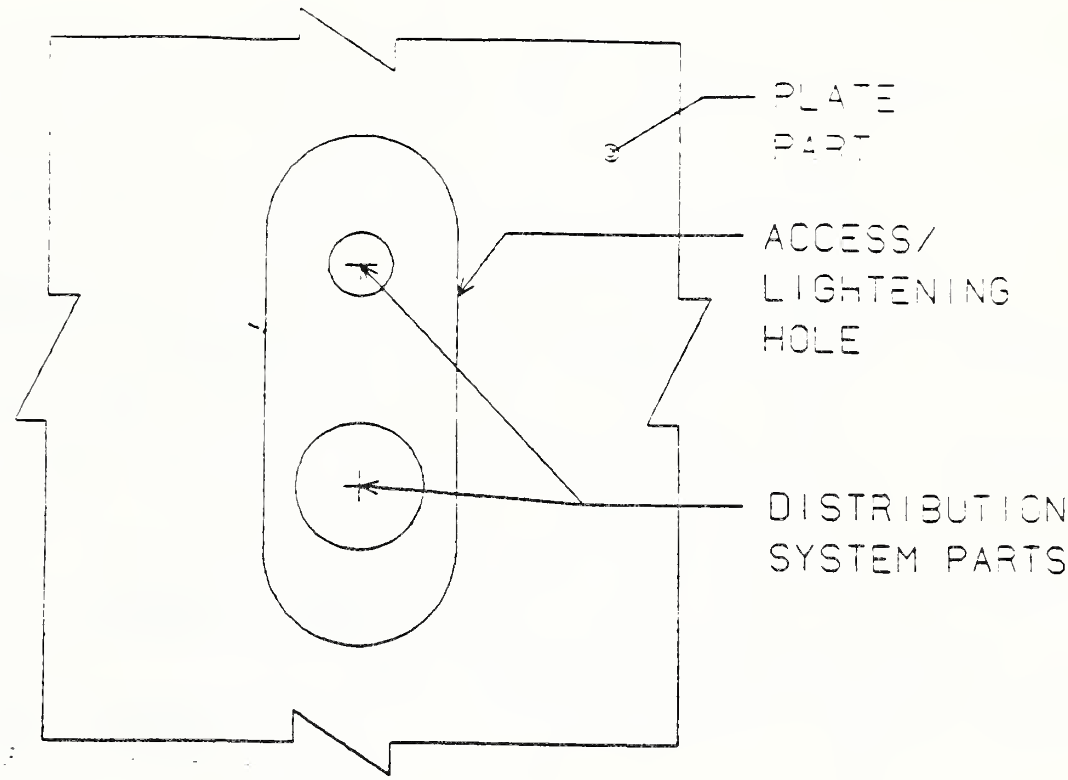


Figure D-99: Example of an Access/Lightening Hole being penetrated by two Distribution System parts

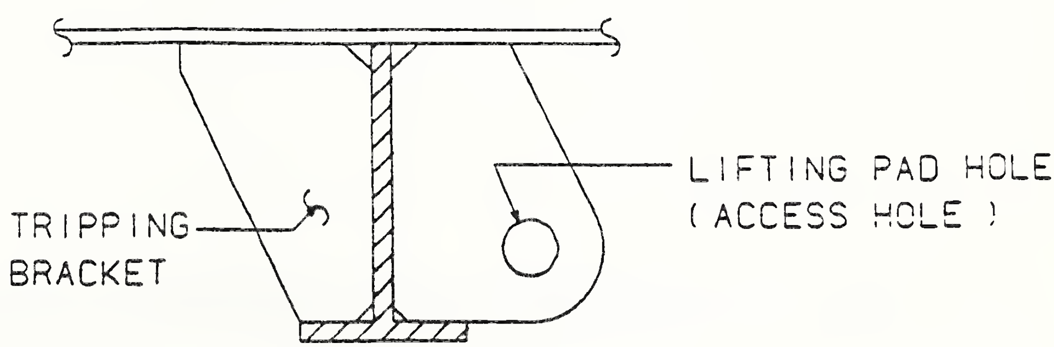


Figure D-100: Example of an Access/Lightening Hole - Lifting Pad Hole

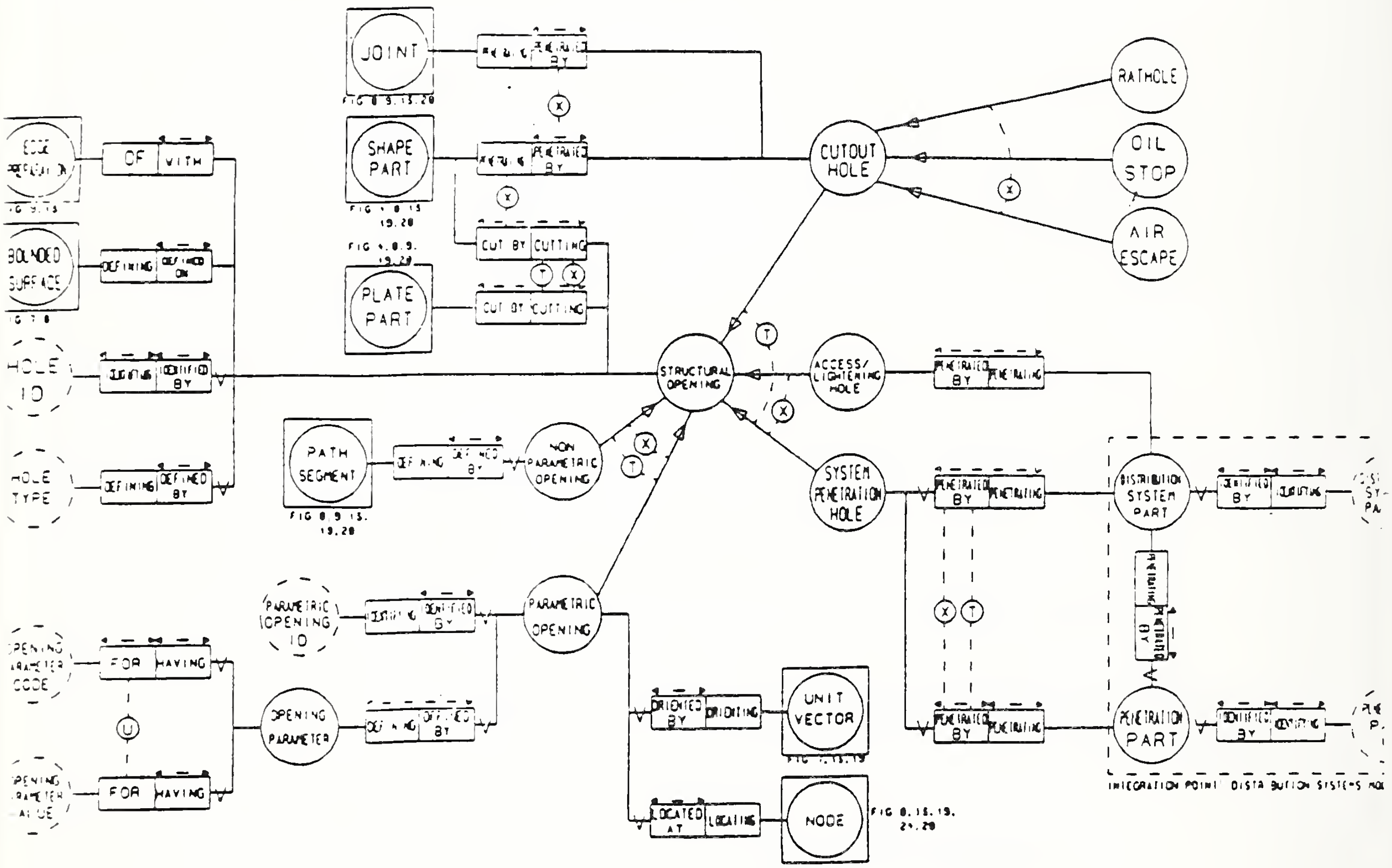


Figure D-101: NIAM Diagram showing Structural Openung Relationships

7.3 References

- Dunn, Mark; -Form Features Information Model-, Version 4, Draft, 17 August 1988, Document 3.1.1.3
- Gielingh, Wim; -Global Reference Model for AEC Product Definition Data-, Version 3, Draft (ISO Document N158), 3 September 1987
- Calkins, Bruce; Lovdahl, Rick; Martin, Doug; Matthews, Jon; -NIDDESC Outfit Information Transfer Model and Methodology-, Working Draft, Version 1.0, 14 July 1988
- Rourke, Pat; -Reference Model for Distribution Systems-, Working Paper, Version 1.1, 30 October 1986.
- Voegeli, Tom; -Product Structure and Configuration Management (PSCM)-, Draft, Version 2.1, 18 August 1988, Document 3.1.1.4
- Schenck, Douglas; -Information Modeling Language Express-, Draft, August 1988, N268
- Yang, Yuhwei; -Integration Core Model-, PDES Version 1, Working Version, 8 July 1988
- Heier, Trond; -Proposal for a Reference Model in Plant Design-, Working Document, 5 July 1988, Document 3.2.2.6

7.4 Alphabetical Listing of Entities

Access/Lightening-Hole	An ACCESS/LIGHTENING hole is a type of structural opening used primarily for accessing areas of a hull or for lightening structure. Access/Lightening holes may be penetrated by distributive system parts.
Air-Escape	An AIR-ESCAPE is a type of cutout hole. Its function is to prevent air pockets from forming when fluids are being transferred into a tank.
Assembly-ID	An ASSEMBLY-ID is the unique identification of a UNIT-ASSEMBLY. The ASSEMBLY-ID consists of an assembly ID name and an assembly ID number.
Assembly-ID-Name	An ASSEMBLY-ID-NAME is used to identify a type of unit assembly. Examples include unit, sub-assembly, and section.
Assembly-ID-Number	An ASSEMBLY-ID-NUMBER is a numeric identifier of the UNIT/ASSEMBLY, i.e. 3200-0045.
BSpline-Surface	A B-SPLINE SURFACE identifies a particular mathematical representation for a curved surface.
Bezier-Surface	A BEZIER-SURFACE identifies a particular mathematical representation for a curved surface.
Bolt-Parameter	A BOLT-PARAMETER describes an attribute of a bolt used in a bolted joint. These include bolt diameter, bolt length, threads/inch, etc.
Bolt-Parameter-Code	A BOLT-PARAMETER-CODE identifies a variable which is used to define an attribute of a bolt, i.e. D = diameter, L = length, T = threads/inch.
Bolt-Parameter-Value	A BOLT-PARAMETER-VALUE provides a value for a specific bolt parameter code, i.e. (diameter =) 1,25 inch.
Bolt-Process	The BOLTING-PROCESS describes the process for a bolted joint. This includes references to standards or detailed drawings providing bolting pattern, tightening torque, etc.
Bounded-Surface	A BOUNDED-SURFACE is a parameterized space, representing an orientable locus of points, bounded by a set of MOLDED CURVES which form a closed contour. A bounded surface may be planar (deck) or sculptured (shell).
Clearance-Length	A CLEARANCE-LENGTH is the measurement for a specific shape-clearance.
Compartment	A COMPARTMENT is an enclosed space within a ship. An example of a compartment is auxiliary machinery room 2.
Compartment-ID	A COMPARTMENT-ID is a unique identifier for a compartment, for example, "1-346-0-L".
Compartment-Name	A COMPARTMENT-NAME is an identifier for a compartment, for example, "CREW LIVING SPACE NO. 4".

Cross-Section	A Shape-Part CROSS-SECTION provides a description of the Cross Section geometry (a cut perpendicular to the linear axis) of the shape-part. Cross-section descriptions may be parametric or non-parametric.
Cross-Section-Code	The CROSS-SECTION-CODE identifies the variables defining the cross-section of a shape-part. For example: TF = average flange thickness, R = fillet radius, T = clear web height between fillets, etc.
Cross-Section-Parameter	A CROSS-SECTION-PARAMETER describes an attribute of a shape-part-cross-section. These include web and flange dimensions, fillet radius, etc.
Cross-Section-Type	A CROSS-SECTION-TYPE indicates the type of cross section; S = standard cross section, N = non-standard cross section.
Cross-Section-Value	A CROSS-SECTION-VALUE provides the value for an associated cross-section-code, ie. (TF =) 0,75 inch.
Curve-Geometry	The CURVE-GEOMETRY provides the mathematical representation for a curve in space.
Curve-ID	A CURVE-ID is a unique identifier for a molded curve.
Curved-Surface	A CURVED-SURFACE is a non-planar surface representing such areas as the shell of a ship.
Cutout Hole	A CUTOUT-HOLE is a type of structural-opening. It is generally a hole in a plate-part or a shape-part to allow the penetration by a shape-part. Other uses are rathole, air escape and oil stop. These holes are not penetrated by shape parts.
Date-Time	A DATE-TIME is expressed in the form YYMMDD.HHMMSS. Integration Point: Miscellaneous Resources Model, units, time-unit entities.
Dist-Sys-Part-ID	A DISTRIBUTION-SYSTEM-PART-ID is the unique identifier of a distribution part.
Dist-System-Part	A DISTRIBUTIVE-SYSTEM-PART is a part that distributes fluid, electricity, air, etc. A distribution system part may be enclosed by a penetration part (a sleeve or insert) when it penetrates a structural part.
Edge-Prep-Descript	An EDGE-PREPARATION-DESCRIPTION is the description of the physical geometry or condition at a plate or shape edge.
Edge-Preparation	The EDGE-PREPARATION is the physical description of a plate-part edge or shape-part edge. It is typically a bevel or chamfer as required by the welding process.
Elementary-Surface	An ELEMENTARY-SURFACE is a simple surface such as a planar, conical or cylindrical surface.
Endcut	The ENDCUT provides the flange/web cut details at the ends of a shape part.
Endcut Parameter Code	An ENDCUT-PARAMETER-CODE identifies a variable which defines an attribute of an ENDCUT, i.e. P1 = WEB-SNIPE-LENGTH.

Endcut-Parameter	An ENDCUT-PARAMETER describes an attribute of a parametric-endcut. For example, snipe and radius are two attributes of an endcut.
Endcut-Parameter-Value	An ENDCUT-PARAMETER-VALUE contains the data for an associated endcut-parameter-code, ie. P1 = WEB-SNIPE-LENGTH.
Endcut-Type	An ENDCUT-TYPE indicates the type of endcut; P = Parametric Endcut, N = Non- Parametric Endcut.
Flange-Angle	A FLANGE-ANGLE is the angle of the bend of a flange on a flanged part.
Flange-Radius	A FLANGE-RADIUS is the radius of the bend of a flanged part.
Hole-ID	A HOLE-ID is the unique identifier for a structural-opening.
Hole-Type	A HOLE-TYPE identifies a type of structural-opening, i.e. P = parametric and N = non-parametric.
Hull	A HULL is a collection of systems which comprise a ship (product model).
Hull-Name	A HULL-NAME is the character identification of a hull, i.e. USS Thomas S. Gates.
Hull-Number	A HULL-NUMBER is the numeric identifier of a hull. The HULL-NUMBER typically takes the form of a Navy hull number or a manufacturers hull number, i.e. H420.
Inspection-Procedure	An INSPECTION-PROCEDURE provides the description for the inspection of a joint. The INSPECTION-PROCEDURE may include references to standard inspection procedures, i.e. MAGNAFLUX, Visual, X-ray, etc.
Joining-Procedure	The JOINING-PROCEDURE describes the joining of parts at a joint. Joining procedures may include references to standard inspection procedures.
Joint	A joint is used to define the physical connection between SHAPE-PARTS and PLATE-PARTS. Joints occur at path segments and nodes. The type of joint is identified as PATH-JOINT and NODE-JOINT.
Joint ID	A JOINT-ID is the unique identification of a joint.
Joint-Type	A JOINT-TYPE identifies the type of a joint, i.e. Bolted = W, R = Riveted, etc.
Library-ID	A LIBRARY-ID is the unique identifier of a library containing library parts.
Library-Part	A LIBRARY-PART is a pre-defined part contained in a library. There are two types of library parts, parametric and non-parametric. Chocks, brackets and gussets are examples of library parts.
Library-Part-ID	A LIBRARY-PART -ID is the unique identifier of a library part within a library.
Library-Version	The LIBRARY-VERSION identifies the version of the library being used, i.e. REV. A, REV. B, etc.

Lower-Flange-Thick	A LOWER-FLANGE-THICKNESS is the thickness of the lower flange of a shape part.
Lower-Flange-Width	A LOWER-FLANGE-WIDTH is the width of the lower flange of a shape part.
Mark Length	The MARK-LENGTH provides the length of the incremental marks of a trace mark.
Mark-Interval	The MARK-INTERVAL provides the spacing between incremental marks of a trace mark.
Material	A MATERIAL is the substance making up a part. This entity includes a description of the material and its properties.
Molded Curve	A MOLDED-CURVE is a curve on a surface, a curve bounding a surface or a curve formed by the intersection of two surfaces.
NC-Mark-Type	An NC-MARK TYPE indicates the type of NC-MARK, i.e. S = String (TEXT), T = TRACE.
NC-Mark	An NC-MARK is a piece of text or contour marked on a plate-part or shape-part by a cam device such as a burning machine or robot. Such marking is typically performed by a punch, zinc marker or an ink-jet marker.
NC-Mark-ID	An NC-MARK-ID is the unique identifier of an NC mark.
NC-Text	An NC-TEXT is a type of NC-mark consisting of a character string.
NC-Text-Par	An NC-TEXT-PARAMETER describes an attribute of an NC-TEXT-MARK, i.e. size, font, etc.
NC-Text-Parm-Code	The NC-TEXT-PARAMETER-CODE provides the variables associated with NC-TEXT, i.e. C = CHARACTER HEIGHT.
NC-Text-Parm-Value	The NC-TEXT-PARM-VALUE provides the value for an associated NC-TEXT-PARAMETER-CODE, i.e. C (CHARACTER SIZE) = 1.0 INCH.
NC-Text-String	The NC-TEXT-STRING provides the actual character data to be marked by the NC marking machine on the part.
Nodal-Joint	A NODAL-JOINT is a type of joint. A nodal-joint is a connection between at least two parts occurring at a node.
Node	A NODE is the logical equivalent of a geometric point. It is a unique (topological) point R3 with dimensionality 0 and extent 0.
Node-ID	A NODE-ID is a unique identifier for a node.
Nonpar-Lib-Part	A NON-PARAMETRIC-LIBRARY-PART is a type of library-part whose geometry (shape, size) is the same for every occurrence.
Nonpar-Opening	A NON-PARAMETRIC-OPENING is a type of structural opening whose opening is defined by a path-segment representing the hole contour.
NSTD-Cross-Section	A NON-STANDARD-CROSS-SECTION is a non-parametric description. Specific parameters associated with web and flanges, for a specific shape part type, are provided by this description.

Oil-Stop	An OIL-STOP is a type of cutout-hole used in combination with a welded-joint to form a fluid barrier between welded parts.
Opening-Parameter	An OPENING-PARAMETER is a set of one or more attributes defining the size and shape of a parametric (standard) opening. An opening parameter, for example, defines the radius of a circular opening.
Par-Opening-ID	A PARAMETRIC-OPENING-ID is the unique identifier of a parametric (standard) opening. It is the name of a standard library opening or the name of the macro creating the opening.
Parametric-Opening	A PARAMETRIC-OPENING is a type of structural opening whose opening is defined by one or more parameter codes and their corresponding opening parameter values.
Parm-Endcut	A PARAMETRIC-ENDCUT is an endcut with the cut geometry specified by reference to a standard endcut and its associated endcut parameters.
Parm-Endcut-ID	A PARAMETRIC-ENDCUT-ID is the unique identifier of a parametric (standard) endcut. For example T86S.
Part	A PART is a unique structural element or component consumed during the production process. Integration point: Product structure model; product item entity.
Part-Flange	A PART-FLANGE is a plate-part that has a roll or knuckle along a path-segment to form a flange.
Part-ID	A PART-ID is the unique identifier of a part. It may take the form of a manufacturing stock number. Integration Point: PSCM model; Product Item ID Entity.
Part-Library	A PART-LIBRARY is a collection of standard parts, also referred to as a standard parts catalog, which provides a pre-defined part for use in the ship product model.
Par-Library-Part	A PARAMETRIC-LIBRARY-PART is a type of library-part whose geometry is necessarily defined by both the library-part-ID and parameter values associated with a particular library-part occurrence. Two parametric library parts with the same library-part-ID may or may not be physically identical.
Part-Parameter	A LIBRARY-PART-PARAMETER describes the attributes associated with a parametric-Library-part. These include shape and size.
Part-Parm-Code	The parametric library PART-PARAMETER-CODE identifies the variables which define a parametric-library-part, i.e. T = thickness.
Part-Parm-Value	A parametric library PART-PARAMETER-VALUE gives the values associated with a specific parametric library part-parameter-code, i.e. (thickness =) 0,375 inch.
Part-Thickness	The PART-THICKNESS is the thickness of a plate-part.
Path-Joint	A PATH-JOINT is a type of joint, a path-joint is a connection between at least two part (shape or plate) edges.

Path-Segment	A PATH-SEGMENT is a bounded portion of a molded-curve defined by two nodes with the positive directiion of the path from the first node to the second node. Integration point: Geometry Model, curve, bounded-curve, trimmed-curve entities.
Path-Segment-ID	A PATH-SEGMENT-ID is the unique identifier of a path-segment.
Pen-Part-ID	A PENETRATION-PART-ID is the unique identifier of a penetration part.
Penetration-Part	A PENETRATION-PART is a sleeve or insert enclosing a distribution system part in locations where the distribution system part penetrates a structural part.
Planar-Surface	A PLANAR-SURFACE is a surface with no curvature anywhere within its boundaries. Integration point: Geometry model, plane entity.
Plate-Part	A PLATE-PART is a part cut from flat material stock. The part may be used as is or subsequently bent to form a flange (Flange Part) or rolled.
Plate-Part-Edge	A PLATE PART EDGE is one of a ordered sequence of path segments defining the outer contours of a plate part.
Position-Point	A POSITION-POINT provides the X, Y, and Z coordinates for positioning a planar surface. Integration point: Geometry model; point entity.
Project-Phase	The PROJECT-PHASE is a design stage, i.e., functional design, detail design, etc.
Rathole	A RATHOLE is a type of cutout hole. Its main function is to allow fluid to drain from pockets formed by joined parts where it might otherwise be retained.
Rivet-Joint	A RIVET-JOINT is a type of joint using rivets to securely fasten two or more parts together.
Rivet-Parameter	A RIVET-PARAMETER contains an attribute of a rivet used in a riveted joint, i.e. rivet diameter, rivet length, etc.
Rivet-Parm-Code	A RIVET-PARMAMETER-CODE identifies the variables defining the attributes of a rivet. For example, D = Diameter, L = Length, etc.
Rivet-Process	The RIVET-PROCESS describes the process for a riveted joint. This includes references to standards or detailed drawings providing riveting pattern, etc.
Std-Cross-Section	A STANDARD-CROSS-SECTION is a parametric description whose parameters are provided through the unified numbering system established in accordance with ASTM and SAE.
Shape-Clearance	A SHAPE-PART CLEARANCE is the distance between the end nodes defining a path-segment, and the actual extreme starting and ending points of the shape-part defined on that path-segment. In effect, the actual length of a shape part may be equal to length of the path segment it is defined on.

Shape-Orientation	A SHAPE-ORIENTATION defines the orientation of a shape-part at a node. Multiple shape orientations can be used to represent a twisted shape-part.
Shape-Part	A SHAPE-PART is a rolled, extruded, or built up structural shape. Integration point: Product structure model, product-item entity.
Shape-Part-Edge	A SHAPE-PART-EDGE is one of a set of one or more path segments defining the edge of a shape part.
Shape-Part-Type	A SHAPE-PART-TYPE identifies the type of shape part: I = I-Beam, T = T-Beam, L = Angle Bar, F = Flat Bar and C = Channel.
Shape-Reference-Pt	A SHAPE-REFERENCE-POINT identifies the position of a shape part relative to a path segment, i.e. centered on or to either side of the path segment.
Shape-Surf-Offset	A SHAPE-SURF-OFFSET provides the offset value of a shape part from a molded surface.
Std-Details-Ref	The STANDARD DETAILS REFERENCE provides references to standard details, drawings, and procedures identifying information for creating a joint.
Std-Shape-ID	A STANDARD-SHAPE-ID is the unique identifier of a shape-part with standard-cross-section. The identifier uses the unified numbering system in accordance with ASTM and SAE. For example, WT6-26.5.
Structural-Opening	A STRUCTURAL-OPENING is an opening in a part to allow penetration of another part, penetration of a distribution system part, passage of a fluid or to lighten the part.
Structural-System	A STRUCTURAL-SYSTEM is a collection of structural parts used, in general, to compartment and support all other systems.
Sub-Assembly	A SUB-ASSEMBLY is a collection of parts and/or other sub-assemblies. Sub-assemblies can gather parts and/or sub-assemblies into logical grouping (decks, bulkheads, etc.) or into physical groupings representing actual construction phases of the hull.
Surface-Edge	A SURFACE-EDGE is one of a sequence of molded curves bounding a surface.
Surface-ID	A SURFACE-ID is a unique identifier for a surface. It consists of a surface-ID-name and a surface-ID-number.
Surface-ID-Number	A SURFACE-ID-NUMBER is the unique part of the surface-ID which defines the surface, i.e., 320-0045.
Surface-Type	A SURFACE-TYPE identifies what type of surface, elementary or curved, is being described.
System	A SYSTEM is a functionally related group of elements (potentially recursive). Some examples of functional groupings are structure, HVAC or electrical systems.
System-ID	A SYSTEM-ID is the unique identifier of a system.

System-Pen-Hole	A SYSTEM-PENETRATION-HOLE is a type of structural opening that is penetrated by a distribution system part or a penetration part
Trace-Mark	A numerically controlled (N/C) marking contour.
Transform-Matrix	A TRANSFORMATION-MATRIX is a 4 X 4 matrix specifying the translation and orientation (rotation) of a library-part. Integration point: geometry model, transformation entity.
Transition Cut	A TRANSITION CUT defines the cuts to produce a transition between two different size shape parts.
Unit Assembly	A UNIT-ASSEMBLY gathers together parts and/or sub-assemblies. The unit-assembly can represent a logical grouping (HFO, TANK, SPACE 42, etc.) or a physical grouping associated with actual construction phases of the hull.
Unit-Vector	A UNIT-VECTOR is a vector with a magnitude of one, indicating direction in space. Integration point: geometry model; vector, direction entities.
Upper-Flange-Thick	An UPPER-FLANGE-THICKNESS is the thickness of an upper flange of a shape part.
Upper-Flange-Width	An UPPER-FLANGE-WIDTH is the width of the upper flange of a shape part.
Web-Height	The WEB-HEIGHT is the height of the web of a shape-part.
Web-Thickness	The WEB-THICKNESS is the thickness of the web of a shape-part.
Weld-Joint	A WELD-JOINT is a type of joint using material fusion to join parts together along a seam. This is the most common type of joint used in shipbuilding today.
Weld-Process	The WELD-PROCESS describes the process for a welded joint. This includes references to standards and detailed drawings.
Weld-Size	The WELD-SIZE is the size of a welded joint.
Weld-Type	The WELD-TYPE identifies the type of weld; fillet, full penetration, etc.
X-Position	X-POSITION provides the X coordinate for positioning a planar surface in space.
X-Value	X-VALUE provides the X value for the unit vector orienting a planar surface in space.
Y-Position	Y-POSITION provides the Y coordinate for positioning a planar surface in space.
Y-Value	Y-VALUE provides the Y value for the unit vector orienting a planar surface in space.
Z-Position	Z-POSITION provides the Z coordinate for positioning a planar surface in space.
Z-Value	Z-VALUE provides the Z value for the unit vector for orienting a planar surface in space.

7.5 Definition of Structural Terms

(Extracted from Ship Design and Construction, SNAME, 1980 with revisions for computer terminology)

Appendages	The portions of a vessel extending beyond the main hull outline including such items as rudder, shafting, struts, bossings, and bilge keels.
Beam, cant	Deck beams aft radially disposed and extending from the transom beam to the cant frame heads at the deck edge.
Beam, deck	An athwartship horizontal structural member, usually a rolled shape, supporting a deck or flat.
Beam, molded	The maximum breadth of the hull measured between the inboard surfaces of the side shell plating of flush-plated ships, or between the inboard surfaces of the inside strakes of lap seam-plated vessels.
Bevel	The angle between the flanges of a frame or other member. (When greater than a right angle, open bevel, when less, closed or shut; also, to chamfer).
Bilge	Intersection of bottom and side. May be rounded or angular as in a chine form hull. The lower parts of holds, tanks and machinery spaces where bilge water may accumulate.
Bilge bracket	A vertical transverse flat plate welded to the tank top or margin plate and to the frame in the area of the bilge.
Bilge keel	A long longitudinal fin fitted at the tuen of the bilge to reduce rolling. Commonly it consists of plating attached to the shell plating by welding or by angles.
Bilge strake	Course of shell plates at the bilge.
Bitt, mooring	Short metal column (usually two) extending up from a base plate attached to the deck for the purpose of securing and belaying wire ropes, hawsers, etc. Used to secure a ship to a pier or tugboat. Also called a bollard.
Body plan	A drawing consisting of two half transverse elevations or end views of a ship, both having a common vertical centerline, so that the right-hand side represents the ship as seen from ahead, and the left-hand side as seen from astern. On the body plan appear the forms of the various cross sections, the curvature of the deck lines at the side, and the projections, as straight lines of the waterlines, the buttock lines, and the diagonal lines.
Bossing or boss	The curved swelling outboard portion on ship's shell plating that surrounds and supports the propeller shaft.
Bossing plate	Steel plate covering the bulged portion of hull where the propeller shaft passes outboard.

Bracket	A plate used to connect rigidly two or more structural parts, such as deck beam to frame, or bulkhead stiffener to the deck or tank top (usually triangular in shape).
Breasthook	A triangular plate bracket joining port and starboard side stringers at the stem.
Bulkhead	A term applied to the vertical partition walls which subdivide the interior of a ship into compartments or rooms. The various types of bulkheads are distinguished by their location, use, kind of material or method of fabrication, such as forepeak, longitudinal, transverse, watertight, wire mesh, pilaster, etc. Bulkheads which contribute to the strength of a vessel are called strength bulkheads, those which are essential to the watertight subdivision are watertight or oiltight bulkheads, and gastight bulkheads serve to prevent the passage of gas or fumes.
Bulkhead,	A term applied to the first main transverse bulkhead forward of the sterpost. This bulkhead forms the forward boundary of the after peak tank.
Bulkhead,	A term applied to the first main transverse bulkhead forward from the bottom of the hold to the freeboard deck and it is designed to keep water out of the forward hold in case of bow collision damage.
Bulkhead deck	The bulkhead deck is the uppermost deck up to which the transverse watertight bulkheads and shell are carried.
Bulkhead, screen	A term applied to a light nonwatertight transverse bulkhead fitted in some Great Lakes ore carriers. Its greater flexibility allows it to survive the effects of the unloading machinery.
Bulwark	Fore-and-aft vertical plating immediately above the upper edge of the sheer strake.
Butt	The end joint between two plates or other members which meet end to end.
Buttock	The intersection of the molded surface with any vertical longitudinal plane not on the centerline.
Camber	The rise or crown of a deck, athwartship; also called round of beam.
Cant Frame	A frame not square to the centerline at the counter of the ship and connected at the upper end to the cant beams.
Center girder	A vertical plate on the ship's centerline between the flat keel and inner bottom or rider plate, extending the length of the ship. Also called center vertical keel, CVK, or center keelson.
Centerline	The middle line of the ship, extending from stem to stern at any level.

Chock	A heavy smooth-surfaced fitting usually located near the edge of the weather deck through which wire ropes or fiber hawsers may be led, usually to piers. One of several pieces of metal precisely fitted between machinery units and their foundations to assure alignment, also made by pouring plastic material in place. A small piece of plate fitted to one side of a plated structure opposite the landing of a structural member on the other side.
Coaming, hatch	The vertical plating bounding a hatch for the purpose of stiffening the edges of the opening and resisting entry of water below.
Cofferdam	Narrow void space between two bulkheads or floors that prevents leakage between the adjoining compartments.
Compartmentation	The subdividing of the hull by transverse watertight bulkheads so that the ship may remain afloat under certain assumed conditions of flooding.
Date-Time	Date and time in the form YYMMDD.HMMSS. Integration point: Miscellaneous Resources model, units, time-unit entities.
Deadrise	Athwartship rise of the bottom from the keel to the bilge.
Deck	A platform in a ship corresponding to a floor in a building. It is the plating, planking, or covering of any tier of beams either in the hull or superstructure of a ship.
Deck, freeboard	Deck to which freeboard is measured; the uppermost continuous deck having permanent means of closing all weather openings.
Deck height	The vertical distance between the molded lines of two adjacent decks.
Deckhouse	An enclosed erection on or above the weather deck that does not extend from side to side of the ship.
Deck, platform	A lower deck, usually in the cargo space, which does not contribute to the longitudinal strength of the ship.
Deck, stringer	The strake of deck plating that runs along the outboard edge of a deck.
Double bottom	Compartments at the bottom of a ship between inner bottom and the shell plating, used for ballast water, fresh water, fuel oil, etc.
Doubling (doubler) plate	A plate fitted outside or inside of and faying (touching) against another to give extra local strength or stiffness.
Draft	The depth of the ship below the waterline measured vertically to the lowest part of the hull, propellers, or other reference point. When measured to the lowest projecting portion of the vessel, it is called the extreme draft, and when measured at the stern, the after draft, the average of the forward draft and the after draft is the mean draft, and the mean draft when in full load condition is the load draft. Also, in cargo handling, the unit of cargo being hoisted on or off the ship by the cargo gear at one particular hoist.
Face plate	Generally a narrow stiffening plate fitted along the inner edge of web frames, stringers, etc. to form the flange of the member.

Fair	To smooth or fair up a ship's lines, eliminating irregularities; to assemble the parts of a ship so that they will be fair, i.e. without kinks, bumps, or waves.
Flange	The part of a plate or shape bent at right angles to the main part; to bend over to form an angle.
Floor	Vertical transverse plate immediately above the bottom shell plating, often located at every frame, extending from bilge to bil ge.
Foundation	The structural supports for the boilers, main engines or turbines and reduction gears are called main foundations. Supports for machinery space auxiliary machinery are called auxiliary foundations. Deck machinery supports are called, for example, steering engine foundation, winch foundation, etc.
Frame	A term used to designate one of the transverse members that make up the riblike part of the skeleton of a ship. The frames act as stiffeners, holding the outside plating in shape and maintaining the transverse form of the ship. (See also longitudinal.)
Frame spacing	The fore-and-aft distance, heel to heel, of adjacent transverse frames.
Girder	A continuous member running fore-and-aft under a deck for the purpose of supporting the deck beams and deck. The girder is generally supported by widely spaced pillars. Also, the vertical fore-and-aft plate members on the bottom of single or double bottom ships.
Gusset plate	A bracket plate lying in a horizontal, or nearly horizontal plane.
Hatch (hatchway)	An opening in a deck through which cargo and stores are loaded unloaded.
Hawespipe	Tube through which anchor chain is led overboard from the windlass wildcat on deck through the ship's side. Bolsters form rounded endings at the deck and shell to avoid sharp edges. Stockless anchors are usually stowed in the hawsepipe.
Hull	The structural body of a ship, including shell plating, framing, decks, bulkheads, etc.
Hull girder	That part of the hull structural material effective in the longitudinal strength of the ship as a whole, which may be treated as analagous to a girder.
Inboard	Inside the ship; toward the centerline.
Inner bottom	Plating forming the top of the double bottom; also called tank top.
Intercostal	Made in separate parts: between floors, frames or beams, etc. The opposite of continuous.
Keel	The principal fore-and-aft component of a ship's framing, located along the centerline of the bottom and connected to the stem and stern frames. Floors or bottom transverses are attached to the keel.
Keel, bilge	(See bilge keel)
Keel, center vertical	The vertical, centerline web of the keel.

Keel, flat plate	The horizontal, centerline, bottom shell strake constituting the lower flange of the keel.
Keel blocks	Heavy wood or concrete blocks on which ship rests during construction.
Keelson, side	Fore-and-aft vertical plate memeber located above the bottom shell on each side of the center vertical keel and some distance therefrom.
Kerf	The material removed by the cutting device, such as a burning torch, in preparing a structural member; width of the kerf must be known in Numerical Control (N/C) cutting.
Kingpost	A strong vertical post used instead of a mast to support a boom and rigging to form a derrick; also called samson post.
Knee, beam	Bracket connecting a deck beam and frame.
Knuckle	An abrupt change in direction of the plating, frames, keel, deck, or other structure of a ship.
Lap	A joint in which one part overlaps the other.
Length, overall	The extreme length of a ship measured from the foremost point of the stem to the aftermost part of the stern.
Length between perpendiculars	The length of a ship between the forward and after perpendiculars. The forward perpendicular is a vertical line at the intersection of the foreside of the stern and the summer load waterline. The after perpendicular is a vertical line at the intersection of the summer load line and the line at the intersection of the summer load line and the after side of the rudder post or sternpost, or the centerline of the rudder stock if there is no rudder post or sternpost.
Lightening hole	A hole cut in a structural member to reduce its weight.
Limber hole	A small hole or slot in a frame or plate for the purpose of preventing water or oil from collecting; a drain hole.
Liner	A flat or tapered strip placed under a plate or shape to bring it in line with another part that it overlaps; a filler.
Lines (plan)	The plans or drawings that show the shape or form of the ship. A lines plan includes station and/or frame lines, water lines, buttocks and diagonals.
Longitudinals	Fore-and-aft structural shape or plate members attached to the underside of decks, flats, or to the inner bottom, or on the inboard side of the shell plating, in association with widely spaced transverses, in the longitudinal framing system.
Manhole	A round or oval hole cut in decks, tanks, etc. for the purpose of providing access.
Margin angle	Angle connecting margin plate to shell.
Margin bracket	A bracket connecting a side frame to the margin plate at the bilge, sometimes called bilge bracket.

Margin line	A line, not less than 3 in. below the top of the bulkhead deck at side, defining the highest permissible location on the side of the ship of any damage waterplane in the final condition of sinkage, trim and heel.
Margin plate	The outboard strake of the inner bottom. When the margin plate is turned down at the bilge it forms the outboard boundary of the double bottom, connecting the inner bottom to the shell plating at the bilge.
Molded lines	Lines defining the geometry of a hull as a surface without thickness; structural members are related to molded lines according to standard practice (unless otherwise shown on drawings) e.g., the inside surface of flush shell plating is on the molded line, also the underside of deck plating.
Pillar	Vertical member or column giving support to a deck girder, flat or similar structure: also called stanchion.
Rabbet	A groove, depression, or offset in a member into which the end or edge of another member is fitted, generally so that the two surfaces are flush. A rabbet in the stem or stern frame would take the ends or edges of the shell plating, resulting in a flush surface.
Scantlings	The dimensions of a ship's frames, girders, plating, etc.
Sea chest	An enclosure, attached to the inside of the underwater shell and open to the sea, fitted with a portable strainer plate. A sea valve and piping connected to the sea chest passes sea water into the ship for cooling, fire or sanitary purposes. Compressed air or steam connections may be provided to remove ice or other obstructions.
Seam	Fore-and-aft joint of shell plating, deck and tank top plating.
Sheer	The longitudinal curve a a vessel's decks in a vertical plane, the usual reference being to the ship's side; in the case of a deck having a camber, its centerline sheer may also be given in offsets. Due to sheer, a vessel's deck height above the baseline is higher at the ends than amidships.
Shell plating	The plates forming the outer side and bottom skin of the hull.
Stanchion	Vertical column supporting decks, flats, girders, etc. Also called a pillar. Rail stanchions are vertical metal columns on which fence-like rails are mounted.
Stiffener	An angle, T-bar, channel, built-up section, etc. used to stiffen plating of a bulkhead, etc.
Stringer	A term applied to a fore-and-aft girder running along the side of a ship at the shell and also to the outboard strake of plating on any deck.
Tripping bracket	Flat bars or plates fitted at various points on deck girders, stiffeners, or beams as reinforcements to prevent their free flanges from tripping.

Trunk	A vertical or inclined space or passage formed by bulkheads or casings, extending one or more deck heights, around openings in the decks, through which access can be obtained and cargo, stores, etc. handled, or ventilation provided without disturbing or interfering with the contents or arrangements of the adjoining spaces.
Waterline	The line of the water's edge when the ship is afloat; technically, the intersection of any horizontal plane with the molded form.
Web frame	A built-up frame to provide extra strength, usually consisting of a web plate flanged or otherwise stiffened on its edge, spaced several frame spaces apart, with the smaller, regular frames in between.

7.6 EXPRESS Model

1.0 SHIP STRUCTURAL MODEL

1.1 Hull Unit Assembly

1.1.1 Hull Unit Assembly Types
- 1.1.1.1 HULL NAME
- 1.1.1.2 HULL NUMBER
- 1.1.1.3 PROJECT PHASE
- 1.1.1.4 ASSEMBLY ID NAME
- 1.1.1.5 ASSEMBLY ID NUMBER
- 1.1.1.6 PART ID
- 1.1.1.7 SYSTEM ID

1.1.2 Hull Unit Assembly Entities
- 1.1.2.1 HULL
- 1.1.2.2 SYSTEM
- 1.1.2.3 STRUCTURAL SYSTEM
- 1.1.2.4 UNIT-ASSEMBLY
- 1.1.2.5 ASSEMBLY ID
- 1.1.2.6 SUB ASSEMBLY
- 1.1.2.7 PART
- 1.1.2.8 DATE-TIME
- 1.1.2.9 MATERIAL

1.2 Ship Geometry/Topology

1.2.1 Ship Geometry/Topology Types
- 1.2.1.1 COMPARTMENT ID
- 1.2.1.2 COMPARTMENT NAME
- 1.2.1.3 SURFACE ID NAME
- 1.2.1.4 SURFACE ID NUMBER
- 1.2.1.5 SURFACE TYPE
- 1.2.1.6 CURVE ID

1.2.2 Ship Geometry/Topology Entities
- 1.2.2.1 BOUNDED SURFACE
- 1.2.2.2 SURFACE ID
- 1.2.2.3 CURVED SURFACE
- 1.2.2.4 ELEMENTARY SURFACE
- 1.2.2.5 B-SPLINE SURFACE
- 1.2.2.6 BEZIER SURFACE
- 1.2.2.7 PLANAR SURFACE
- 1.2.2.8 SURFACE EDGE
- 1.2.2.9 MOLDED CURVE

1.3.2.2.8 CROSS SECTION PARAMETER
1.3.2.2.9 NC MARK
1.3.2.2.10 NC TEXT
1.3.2.2.11 NC TEXT PARAMETER
1.3.2.2.12 TRACE MARK
1.3.2.2.13 ENDCUT
1.3.2.2.14 ENDCUT PARAMETER
1.3.2.2.15 PARAMETRIC ENDCUT
1.3.2.2.16 NON-PARAMETRIC ENDCUT
1.3.2.2.17 TRANSITION CUT

1.3.3 Library Parts
1.3.3.1 Library Part Types
1.3.3.1.1 LIBRARY ID
1.3.3.1.2 LIBRARY PART ID
1.3.3.1.3 LIBRARY PART TYPE
1.3.3.1.4 LIBRARY VERSION
1.3.3.1.5 PART PARAMETER CODE
1.3.3.2 Library Part Entities
1.3.3.2.1 PART LIBRARY
1.3.3.2.2 LIBRARY PART
1.3.3.2.3 NON-PARAMETRIC LIBRARY PART
1.3.3.2.4 PARAMETRIC LIBRARY PART
1.3.3.2.5 PART PARAMETER

1.4 Structural Joints

1.4.1 Structural Joint Types
1.4.1.1 BOLT PARAMETER CODE
1.4.1.2 BOLT PROCESS
1.4.1.3 INSPECTION PROCEDURE
1.4.1.4 JOINT ID
1.4.1.5 JOINT TYPE
1.4.1.6 JOINING PROCEDURE
1.4.1.7 RIVET PARAMETER CODE
1.4.1.8 RIVET PROCESS
1.4.1.9 STANDARD DETAILS REFERENCE
1.4.1.10 WELD TYPE
1.4.1.11 WELD PROCESS
1.4.2 Joint Entities
1.4.2.1 JOINT
1.4.2.2 BOLT JOINT
1.4.2.3 BOLT PARAMETER
1.4.2.4 NODAL JOINT

```
*)
SCHEMA ipim_ship_structure_schema;
    EXPORT EVERYTHING;
(*
```

1.1.1 HULL-UNIT-ASSEMBLY TYPES

1.1.1.1 HULL NAME

A HULL NAME is the character identification of a hull, ie. USS Thomas S. Gates.

```
*)
TYPE hull_name = string(32);
END_TYPE;
(*
```

1.1.1.2 HULL NUMBER

A HULL NUMBER is the numeric identifier of a hull. The HULL NUMBER typically takes the form of a Navy hull number or a manufacturers hull number, ie. H420.

```
*)
TYPE hull_number = string(6);
END_TYPE;
(*
```

1.1.1.3 PROJECT PHASE

The PROJECT PHASE is a design stage, ie. functional design, detail design, etc.

```
*)
TYPE project_phase = string(10);
END_TYPE;
(*
```

1.1.1.4 ASSEMBLY ID NAME

An ASSEMBLY ID NAME is used to identify a type of unit assembly. Examples include Unit, Sub-assembly, and Section.

```
*)
TYPE assembly_id_name = string(32);
END_TYPE;
(*
```

1.1.1.5 ASSEMBLY ID NUMBER

An ASSEMBLY ID NUMBER is the numeric identifier of a unit assembly, ie. 3200-0045.

```
*)
TYPE assembly_id_number = string(6);
END_TYPE;
(*
```

1.1.1.6 PART ID

A PART ID is the unique identifier of a part. It may take the form of a manufacturers stock number. Integration point: PSCM model: Product Item ID entity.

```
*)
TYPE part_id = string(6);
END_TYPE;
(*
```

1.1.1.7 SYSTEM ID

A SYSTEM ID is the unique identifier of a system.

```
*)
TYPE system_id = string(32);
END_TYPE;
(*
```

1.1.2 HULL-UNIT-ASSEMBLY ENTITIES

1.1.2.1 HULL

A HULL is a collection of systems which comprise a ship (product model). Integration point: Product Structure Model; Product Model entity.

```
*)
ENTITY hull;
  identified_by_hull_name    : SET [1:#] OF hull_name;
  identified_by_hull_number  : UNIQUE hull_number;
  made_up_of_system          : SET [1:#] OF system;
END_ENTITY;
(*
```

1.1.2.2 SYSTEM

A SYSTEM is a functionally related group of elements (potentially recursive. Some examples of functional groupings are structure, HVAC or electrical systems.

```
*)
ENTITY system
  SUPERTYPE OF (structural_system);
  with_system_id            : UNIQUE system_id;
  part_of_hull              : OPTIONAL SET [1:#] OF hull;
END_ENTITY;
(*
```

1.1.2.3 STRUCTURAL SYSTEM

A STRUCTURAL SYSTEM is a collection of structural parts used, in general, to contain and support all other systems.

```
*)
ENTITY structural_system
  SUBTYPE OF (system);
  made_up_of_unit_assembly  : SET [1:#] OF unit_assembly;
END_ENTITY;
(*
```

1.1.2.4 UNIT-ASSEMBLY

A UNIT-ASSEMBLY gathers together parts and/or sub-assemblies. The UNIT-ASSEMBLY can represent a logical grouping (HFO tank, Space 42, etc.) or a physical grouping associated with actual construction phases of the hull.

```
*)
ENTITY unit_assembly
  SUPERTYPE OF (sub_assembly);
  with_project_phase           : project_phase;
  identified_by_assembly_id    : UNIQUE assembly_id;
  part_of_structural_system    : OPTIONAL SET [1:#] OF structural_system;
WHERE
  is_uniquely_determined_by (with_project_phase,
                             identified_by_assembly_id);
END_ENTITY;
(*
```

1.1.2.5 ASSEMBLY ID

An ASSEMBLY ID is the unique identifier of a unit assembly. The ASSEMBLY ID consists of an assembly ID name and an assembly ID number.

```
*)
ENTITY assembly_id;
  having_assembly_id_number  : assembly_id_number;
  having_assembly_id_name    : assembly_id_name;
  identifying_unit_assembly  : UNIQUE unit_assembly;
WHERE
  is_uniquely_determined_by (having_assembly_id_name,
                             having_assembly_id_number);
END_ENTITY;
(*
```

1.1.2.6 SUB-ASSEMBLY

A SUB-ASSEMBLY is a collection of parts and/or other sub-assemblies. Sub-assemblies can gather parts and/or sub-assemblies into logical groupings (decks, bulkheads, etc.) or into physical groupings representing actual construction phases of the hull.

```
*)
ENTITY sub_assembly
  SUBTYPE OF (unit_assembly);
  made_of_sub_assembly      : OPTIONAL SET [1:#] OF sub_assembly;
  part_of_sub_assembly      : OPTIONAL sub_assembly;
  made_of_part              : OPTIONAL SET [1:#] OF part;
END_ENTITY;
(*
```

1.1.2.7 PART

A PART is a unique structural element or component consumed during the production process.
Integration point: Product Structure Model, Product Item entity.

```
*)
ENTITY part
  SUPERTYPE OF (library_part XOR
                plate_part   XOR
                shape_part);
  identified_by_part_id       : UNIQUE part_id;
  of_sub_assembly             : sub_assembly;
  last_modified_on_date_time  : date_time;
  created_on_date_time        : date_time;
  made_of_material            : material;
END_ENTITY;
(*
```

1.1.2.8 DATE/TIME

A DATE/TIME is expressed in the form yymmdd.hhmmss.
Integration point: Miscellaneous Resources Model; Unit and Time-Unit entities.

```
*)
ENTITY date_time;
  with_date_time_value             : UNIQUE date_time_value;
  last_modification_date_of_part   : OPTIONAL SET [1:#] OF part;
  creation_date_of_part            : OPTIONAL SET [1:#] OF part;
END_ENTITY;
(*
```

1.1.2.9 MATERIAL

A MATERIAL is the substance making up a part. This entity includes a description of the material and its properties.
Integration point: Materials Model; Material entity.

```
*)
ENTITY material;
  of_part                       : OPTIONAL SET [1:#] OF part;
END_ENTITY;
(*
```

1.2 SHIP GEOMETRY/TOPOLOGY

1.2.1 SHIP GEOMETRY/TOPOLOGY TYPES

1.2.1.1 COMPARTMENT ID

A COMPARTMENT ID is the unique identifier for a compartment. An example of a COMPARTMENT ID is "1-346-0-L".

```
*)
TYPE compartment_id = string(32);
END_TYPE;
(*
```

1.2.1.2 COMPARTMENT NAME

A COMPARTMENT NAME is an identifier for a compartment. An example of a COMPARTMENT NAME is "CREW LIVING SPACE NO. 4".

```
*)
TYPE compartment_name = string(50);
END_TYPE;
(*
```

1.2.1.3 SURFACE ID NAME

The SURFACE ID NAME is a part of the Surface ID and may be non-unique. Deck and Bulkhead are both examples of a surface-id-name.

```
*)
TYPE surface_id_name = string(8);
END_TYPE;
(*
```

1.2.1.4 SURFACE ID NUMBER

A SURFACE ID NUMBER is the unique part of the Surface ID which defines the surface, ie. 320-0045.

```
*)
TYPE surface_id_number = string(6);
END_TYPE;
(*
```

1.2.1.5 SURFACE TYPE

A SURFACE TYPE indicates the type of surface, that is, whether the surface is a B-SPLINE SURFACE, a BEZIER SURFACE, or a PLANAR surface.

```
*)
TYPE surface_type = ENUMERATION OF
     (b-spline_surface,
      bezier_surface  ,
      planar_surface  );
END_TYPE;
(*
```

1.2.1.6 CURVE ID

A CURVE ID is the unique identifier for a molded curve.

```
*)
TYPE curve_id = string(6);
END_TYPE;
(*
```

1.2.2 SHIP GEOMETRY/TOPOLOGY ENTITIES

1.2.2.1 BOUNDED SURFACE

A BOUNDED SURFACE is a parameterized space, representing an orientable locus of points, bounded by a set of molded curves which forms a closed contour. A BOUNDED SURFACE may be planar (deck) or sculptered (shell).
Integration point: Geometry Model; Bounded Surface entity.

```
*)
ENTITY bounded_surface
  SUPERTYPE OF (curved_surface XOR
                elementary_surface);
  identified_by_surface_id    : UNIQUE surface_id;
  having_surface_type         : surface_type;
  defining_node               : OPTIONAL SET [1:#] OF node;
  defining_molded_curve       : OPTIONAL SET [1:#] OF molded_curve;
  bounded_by_surface_edge     : SET [1:#] OF surface_edge;
```

```
    defining_structural_opening : OPTIONAL SET [1:#] OF
                                          structural_opening;
    bounding_compartment        : OPTIONAL SET [1:#] OF compartment;
  END_ENTITY;
  (*
```

1.2.2.2 SURFACE ID

A SURFACE ID is a unique identifier for a surface. It consists of a Surface ID Name and a Surface ID Number.

```
  *)
  ENTITY surface_id;
    having_surface_id_number    : UNIQUE surface_id_number;
    having_surface_id_name      : surface_id_name;
    identifying_bounded_surface : OPTIONAL UNIQUE bounded_surface;
  WHERE
    is_uniquely_determined_by (having_surface_id_number,
                               having_surface_id_name);
  END_ENTITY;
  (*
```

1.2.2.3 CURVED SURFACE

A CURVED SURFACE is a non-planar surface representing such areas the shell of a ship.

```
  *)
  ENTITY curved_surface
    SUPERTYPE OF (bezier_surface XOR
                  bspline_surface)
    SUBTYPE OF (bounded_surface);
  END_ENTITY;
  (*
```

1.2.2.4 ELEMENTARY SURFACE

An ELEMENTARY SURFACE is a simple surface such as a planar, conical or cylindrical surface. Integration point: Geometry model; Elementary Surface entity.

```
  *)
  ENTITY elementary_surface
    SUPERTYPE OF (planar_surface)
    SUBTYPE OF (bounded_surface);
  END_ENTITY;
  (*
```

1.2.2.5 B-SPLINE SURFACE

A B-SPLINE SURFACE identifies a particular mathematical representation for a curved surface. Integration point: Geometry Model; B-spline Surface entity.

```
*)
ENTITY bspline_surface
  SUBTYPE OF (curved_surface);
END_ENTITY;
*)
```

1.2.2.6 BEZIER SURFACE

A BEZIER SURFACE identifies a particular mathematical representation for a curved surface. Integration point: Geometry Model; Bezier Surface entity.

```
*)
ENTITY bezier_surface
  SUBTYPE OF (curved_surface);
END_ENTITY;
(*
```

1.2.2.7 PLANAR SURFACE

A PLANAR SURFACE is a surface with no curvature anywhere within its boundaries. Integration point: Geometry model; Plane entity.

```
*)
ENTITY planar_surface
  SUBTYPE OF (elementary_surface);
  oriented_by_unit_vector   : unit_vector;
  located_by_position_point : position_point;
WHERE
  is_uniquely_determined_by (located_by_position_point,
                             oriented_by_unit_vector);
END_ENTITY;
(*
```

1.2.2.8 SURFACE EDGE

A SURFACE EDGE is a sequence of molded curves bounding a surface.

```
*)
ENTITY surface_edge;
  identified_by_edge_sequence_number : UNIQUE edge_sequence_number;
  defined_by_molded_curve            : molded_curve;
```

```
  bounding_bounded_surface           : bounded_surface;
END_ENTITY;
(*
```

1.2.2.9 MOLDED CURVE

A MOLDED CURVE is a curve on a surface, a curve bounding a surface or a curve formed by the intersection of two surfaces.
Integration point: Geometry Model; Curve, Curve-On Surface, Intersection-Curve entities.

```
*)
ENTITY molded_curve;
  identified_by_curve_id     : UNIQUE curve_id;
  defining_path_segment      : OPTIONAL SET [1:#] OF path_segment;
  defining_part_flange       : OPTIONAL SET [1:#] OF part_flange;
  defining_surface_edge      : OPTIONAL SET [1:#] OF surface_edge;
  defined_on_bounded_surface : OPTIONAL SET [1:#] OF bounded_surface;
  defined_by_curve_geometry  : curve_geometry;
  locating_node              : OPTIONAL SET [1:#] OF node;
END_ENTITY;
(*
```

1.2.2.10 POSITION POINT

POSITION POINT provides the X, Y and Z coordinates positioning a planar surface.
Integration point: Geometry Model; Point entity.

```
*)
ENTITY position_point;
  having_z_position         : REAL;
  having_y_position         : REAL;
  having_x_position         : REAL;
  locating_planar_surface   : OPTIONAL SET [1:#] OF planar_surface;
WHERE
  is_uniquely_determined_by_3 (having_x_position,
                               having_y_position,
                               having_z_position);
END_ENTITY;
(*
```

1.2.2.11 UNIT VECTOR

A UNIT VECTOR is a vector with a magnitude of one, indicating direction in space.
Integration point: Geometry Model; Vector, Direction entities.

```
*)
ENTITY unit_vector;
  having_z_value                 : REAL;
  having_y_value                 : REAL;
  having_x_value                 : REAL;
  orienting_planar_surface       : OPTIONAL SET [1:#] OF planar_surface;
  orienting_shape_orientation    : OPTIONAL SET [1:#] OF
                                        shape_orientation;
  orienting_parametric_opening : OPTIONAL SET [1:#] OF
                                        parametric_opening;
  orienting_nc_text              : OPTIONAL SET [1:#] OF nc_text;
WHERE
  is_uniquely_determined_by_3 (having_x_value,
                               having_y_value,
                               having_z_value);
  (having_x_value **2 +
   having_y_value **2 +
   having_z_value **2 ) = 1.0;
END_ENTITY;
(*
```

1.2.2.12 TRANSFORMATION MATRIX

A TRANSFORMATION MATRIX is a 4 x 4 matrix specifying the translation and orientation (rotation) of a library part.
Integration point: Geometry Model; Transformation entity.

```
*)
ENTITY transformation_matrix;
  orienting_library_part    : OPTIONAL SET [1:#] OF library_part;
END_ENTITY;
(*
```

1.2.2.13 CURVE GEOMETRY

The CURVE GEOMETRY provides the mathematical representation for a curve in space.

```
*)
ENTITY curve_geometry;
  defining_molded_curve                    : SET [1:#] OF molded_curve;
  of_web_non_parametric_endcut             : OPTIONAL SET [1:#] OF
                                             non_parametric_endcut;
  of_lower_flange_non_parametric_endcut : OPTIONAL SET [1:#] OF
                                             non_parametric_endcut;
  of_upper_flange_non_parametric_endcut : OPTIONAL SET [1:#] OF
```

```
                                                    non_parametric_endcut;
END_ENTITY;
(*
```

1.2.2.14 COMPARTMENT

A COMPARTMENT is an enclosed space within a ship. An example of a COMPARTMENT is Auxiliary Machinery Room 2.

```
*)
ENTITY compartment;
  identified_by_compartment_id   : UNIQUE compartment_id;
  identified_by_compartment_name : OPTIONAL compartment_name;
  bounded_by_bounded_surface     : SET [1:#] OF bounded_surface;
END_ENTITY;
(*
```

1.3 PARTS

1.3.1 PLATE PARTS

1.3.1.1 PLATE PART TYPES

1.3.1.1.1 NODE ID

A NODE ID is a unique identifier of a node.

```
*)
TYPE node_id = string(6);
END_TYPE;
(*
```

1.3.1.1.2 PATH SEGMENT ID

A PATH SEGMENT ID is the unique identifier of a path segment.

```
*)
TYPE path_segment_id = string(6);
END_TYPE;
(*
```

1.3.1.1.3 EGDE PREPARATION DESCRIPTION

An EDGE PREPARATION DESCRIPTION is the description of the physical geometry or condition at a plate or shape edge.

```
*)
TYPE edge_preparation_description = string(100);
END_TYPE;
(*
```

1.3.1.2 PLATE PART ENTITIES

1.3.1.2.1 PLATE PART

A plate part is a part cut from flat material stock. The part may be used as is or subsequently bent to form a flange (flange part) or rolled.
Integration point: Product Structure Model; Product Item entity.

```
*)
ENTITY plate_part
  SUBTYPE OF (part);
  with_plate_surface_offset    : REAL;
  with_plate_part_thickness    : REAL;
  marked_with_nc_mark          : OPTIONAL SET [1:#] OF nc_mark;
  having_plate_part_edge       : SET [1:#] OF plate_part_edge;
  having_part_flange           : OPTIONAL SET [1:#] OF part_flange;
  cut_by_structural_opening    : OPTIONAL SET [1:#] OF
                                       structural_opening;
  defined_by_path_segment      : OPTIONAL SET [1:#] OF path_segment;
  joined_by_nodal_joint        : OPTIONAL SET [1:#] OF nodal_joint;
END_ENTITY;
(*
```

1.3.1.2.2 PLATE PART EDGE

A PLATE PART EDGE is one of an ordered sequence of path segments defining the outer contour of a plate part.

```
*)
ENTITY plate_part_edge;
  identified_by_edge_sequence_number : INTEGER;
  defined_by_path_segment            : path_segment;
  having_edge_preparation            : OPTIONAL edge_preparation;
  for_plate_part                     : plate_part;
  joined_by_path_joint               : OPTIONAL SET [1:#] OF
                                                path_joint;
WHERE
  is_uniquely_determined_by_3 (for_plate_part,
                               identified_by_edge_sequence_number,
                               defined_by_path_segment);
```

```
END_ENTITY;
(*
```

1.3.1.2.3 NODE

A NODE is the logical equivalent of a geometric point. It is a unique (topological) point R3 with dimensionality 0 and extent 0.

Integration point: Topology Model; Vertex entity, and Geometry Model; Point entity.

```
    *)
ENTITY node;
  identified_by_node_id        : UNIQUE node_id;
  defined_on_bounded_surface   : OPTIONAL SET [1:#] OF bounded_surface;
  at_end_of_path_segment       : OPTIONAL SET [1:#] OF path_segment;
  at_start_of_path_segment     : OPTIONAL SET [1:#] OF path_segment;
  located_on_molded_curve      : OPTIONAL SET [1:#] OF molded_curve;
  locating_parametric_opening : OPTIONAL SET [1:#] OF
                                                 parametric_opening;
  locating_library_part        : OPTIONAL SET [1:#] OF library_part;
  locating_shape_orientation   : OPTIONAL SET [1:#] OF
                                                 shape_orientation;
  locating_nodal_joint         : OPTIONAL SET [1:#] OF nodal_joint;
  locating_nc_text           : OPTIONAL SET [1:#] OF nc_text;
END_ENTITY;
(*
```

1.3.1.2.4 PATH SEGMENT

A PATH SEGMENT is a bounded portion of a molded curve defined by two nodes with the positive direction of the path from the first node to the second node.

Integration point: Geometry Model; Curve, Bounded-Curve, and Trimmed-Curve entities.

```
*)
ENTITY path_segment;
  identified_by_path_segment_id : UNIQUE path_segment_id;
  ending_on_node                : node;
  starting_on_node              : node;
  defined_on_molded_curve       : molded_curve;
  defining_plate_part_edge      : OPTIONAL SET [1:#] OF
                                                   plate_part_edge;
  defining_non_parametric_opening : OPTIONAL SET [1:#] OF
                                             non_parametric_opening;
  defining_trace_mark           : OPTIONAL SET [1:#] OF trace_mark;
  defining_plate_part           : OPTIONAL SET [1:#] OF plate_part;
  defining_shape_part           : OPTIONAL SET [1:#] OF shape_part;
```

```
  joined_by_path_joint          : OPTIONAL SET [1:#] OF path_joint;
  defining_shape_part_edge      : OPTIONAL SET [1:#] OF
                                                  shape_part_edge;
WHERE
  is_uniquely_determined_by (starting_on_node,
                             ending_on_node);
END_ENTITY;
(*
```

1.3.1.2.5 EDGE PREPARATION

EGDE PREPARATION is the physical description of a plate part edge or shape part edge. It is typically a bevel or chamfer as required by the welding process.

```
*)
ENTITY edge_preparation;
  described_by_edge_preparation_description : UNIQUE
                                    edge_preparation_description;
  for_plate_part_edge       : OPTIONAL SET [1:#] OF plate_part_edge;
  for_structural_opening    : OPTIONAL SET [1:#] OF structural_opening;
  for_shape_part_edge       : OPTIONAL SET [1:#] OF shape_part_edge;
END_ENTITY;
(*
```

1.3.1.2.6 PART FLANGE

A PART FLANGE is a plate part that has a roll or knuckle along a path segment to form a flange.

```
*)
ENTITY part_flange;
  with_flange_angle         : REAL;
  with_flange_radius        : REAL;
  for_plate_part            : OPTIONAL SET [1:#] OF plate_part;
  defined_on_molded_curve   : molded_curve;
  with_endcut               : OPTIONAL endcut;
END_ENTITY;
(*
```

1.3.2 SHAPE PARTS

1.3.2.1 SHAPE PART TYPES

1.3.2.1.1 SHAPE REFERENCE POINT

A SHAPE REFERENCE POINT identifies the position of a shape part relative to a path segment, that is to the left, centered, or right of the path segment.

```
*)
TYPE shape_reference_point = ENUMERATION OF
     (left   ,
      center ,
      right  );
END_TYPE;
(*
```

1.3.2.1.2 CLEARANCE LENGTH

A CLEARANCE LENGTH is the measurement for a specific shape-clearance.

```
*)
TYPE clearance_length = real;
END_TYPE;
(*
```

1.3.2.1.3 SHAPE PART TYPE

A SHAPE PART TYPE identifies the type of shape part. The types of shape parts used in shipbuilding are I-beam, T-bar, Angle-bar, Flat- bar, and Channel.

```
*)
TYPE shape_part_type = ENUMERATION OF
     (I-BEAM    ,
      T-BAR     ,
      ANGLE-BAR,
      FLAT-BAR ,
      CHANNEL   );
END_TYPE;
(*
```

1.3.2.1.4 STANDARD SHAPE ID

A STANDARD SHAPE ID is the unique identifier of a shape-part with standard cross section. The identifier uses the unified numbering system in accordance with ASTM and SAE such as WT6-26.5 .

```
*)
TYPE standard_shape_id = string(10);
END_TYPE;
(*
```

1.3.2.1.5 CROSS SECTION CODE

The CROSS SECTION CODE identifies the variables defining the cross section of a shape part. For example; tf = average flange thickness, r = fillet radius, t = clear web height between fillets, etc.

```
*)
TYPE cross_section_code = string(2);
END_TYPE;
(*
```

1.3.2.1.6 CROSS SECTION TYPE

The CROSS SECTION TYPE indicates the type of cross section, that is, whether the cross section is standard or non-standard.

```
*)
TYPE cross_section_type = ENUMERATION OF
     (non-standard,
      standard    );
END_TYPE;
(*
```

1.3.2.1.7 N/C MARK ID

An N/C MARK ID is the unique identifier of an N/C Mark.

```
*)
TYPE nc_mark_id = string(6);
END_TYPE;
(*
```

1.3.2.1.8 N/C MARK TYPE

An N/C MARK TYPE indicates the type of N/C-mark, that is, whether the mark is nc-text or a trace-mark.

```
*)
TYPE nc_mark_type = ENUMERATION OF
     (trace_mark,
      nc_text   );
END_TYPE;
(*
```

1.3.2.1.9 N/C TEXT PARAMETER CODE

The N/C TEXT PARAMETER CODE provides the variables associated with N/C-text, ie. c = character height.

```
*)
TYPE nc_text_parameter_code = string(2);
END_TYPE;
(*
```

1.3.2.1.10 N/C TEXT STRING

The N/C TEXT STRING provides the actual character data to be marked by the N/C marking machine on the part.

```
*)
TYPE nc_text_string = string(50);
END_TYPE;
(*
```

1.3.2.1.11 ENDCUT PARAMETER CODE

An ENDCUT PARAMETER CODE identifies a variable which defines an attribute of an endcut, ie. p1 = web snipe length.

```
*)
TYPE endcut_parameter_code = string(2);
END_TYPE;
(*
```

1.3.2.1.12 ENDCUT TYPE

An ENDCUT TYPE indicates the type of endcut, that is, whether the endcut is parametric or non-parametric.

```
*)
TYPE endcut_type = ENUMERATION OF
     (non-parametric,
      parametric    );
END_TYPE;
(*
```

1.3.2.1.13 PARAMETRIC ENDCUT ID

A PARAMETRIC ENDCUT ID is the unique identifier of a parametric (standard) endcut. For example, T86S.

```
*)
TYPE parametric_endcut_id = string(6);
END_TYPE;
(*
```

1.3.2.2 SHAPE PART ENTITIES

1.3.2.2.1 SHAPE PART

A SHAPE PART is a rolled, extruded, or built up structural shape. Integration point: Product Structure Model; Product-Item entity.

```
*)
ENTITY shape_part
  SUBTYPE OF (part);
  located_by_shape_reference_point : shape_reference_point;
  offset_by_shape_surface_offset   : REAL;
  identified_with_shape_part_type  : shape_part_type;
  having_shape_part_edge           : SET [1:#] OF shape_part_edge;
  ending_with_endcut               : endcut;
  starting_with_endcut             : endcut;
  oriented_by_shape_orientation    : SET [1:#] OF shape_orientation;
  ending_with_shape_clearance      : shape_clearance;
  starting_with_shape_clearance    : shape_clearance;
  identified_with_cross_section    : cross_section;
  cut_by_structural_opening        : OPTIONAL SET [1:#] OF
                                              structural_opening;
  penetrating_cutout_hole          : OPTIONAL SET [1:#] OF
                                              cutout_hole;
  marked_by_nc_mark                : OPTIONAL SET [1:#] OF nc_mark;
  defined_on_path_segment          : SET [1:#] OF path_segment;
  joined_by_nodal_joint            : OPTIONAL SET [1:#] OF nodal_joint;
WHERE
  has_at_most_one_of (penetrating_cutout_hole,
                      cut_by_structural_opening);
END_ENTITY;
(*
```

1.3.2.2.2 SHAPE CLEARANCE

A SHAPE CLEARANCE is the distance between the end nodes defining a path segment, and the actual extreme starting and ending points of the shape part defined on that path segment. In effect, the actual length of a shape part may be ¡= the length of the path segment it is defined on.

```
*)
ENTITY shape_clearance;
  with_clearance_length     : UNIQUE clearance_length;
  at_end_of_shape_part      : OPTIONAL SET [1:#] OF shape_part;
  at_start_of_shape_part    : OPTIONAL SET [1:#] OF shape_part;
END_ENTITY;
(*
```

1.3.2.2.3 CROSS SECTION

A shape part CROSS SECTION provides a description of the cross section geometry (a cut perpendicular to the linear axis) of the shape part. CROSS SECTION descriptions may be parametric or non-parametric.

```
*)
ENTITY cross_section
  SUPERTYPE OF (non_standard_cross_section XOR
                standard_cross_section);
  identified_by_cross_section_type : UNIQUE cross_section_type;
  for_shape_part                   : OPTIONAL SET [1:#] OF shape_part;
END_ENTITY;
(*
```

1.3.2.2.4 SHAPE PART EDGE

A SHAPE PART EDGE is one of an ordered set of path segments defining the edge of a shape part.

```
*)
ENTITY shape_part_edge;
  identified_by_edge_sequence_number : INTEGER;
  for_shape_part                     : UNIQUE shape_part;
  joined_by_path_joint               : OPTIONAL SET [1:#] OF path_joint;
  having_edge_preparation            : OPTIONAL edge_preparation;
  defined_on_path_segment            : path_segment;
WHERE
  is_uniquely_determined_by_3 (identified_by_shape_part,
                               identified_by_edge_sequence_number,
                               defined_on_path_segment);
END_ENTITY;
(*
```

1.3.2.2.5 SHAPE ORIENTATION

A SHAPE ORIENTATION defines the orientation of a shape part at a node. Multiple SHAPE ORIENTATIONs can be used to represent a twisted shape part.

```
*)
ENTITY shape_orientation;
  orienting_shape_part    : OPTIONAL SET [1:#] OF shape_part;
  defined_by_unit_vector : unit_vector;
  defined_at_node        : node;
```

```
WHERE
  is_uniquely_determined_by (defined_unit_vector,
                             defined_at_node);
END_ENTITY;
(*
```

1.3.2.2.6 STANDARD CROSS SECTION
A STANDARD CROSS SECTION is a parametric description whose parameters are provided through the unified numbering system established in accordance with ASTM and SAE.

```
*)
ENTITY standard_cross_section
  SUBTYPE OF (cross_section);
  with_standard_shape_id                : standard_shape_id;
  described_by_cross_section_parameter : SET [1:#] OF
                                           cross_section_parameter;
END_ENTITY;
(*
```

1.3.2.2.7 NON-STANDARD CROSS SECTION
A NON-STANDARD CROSS SECTION is a non-parametric description. Specific parameters associated with web and flanges, for a specific shape part type, are provided by this description.

```
*)
ENTITY non_standard_cross_section
  SUBTYPE OF (cross_section);
  with_lower_flange_thickness : OPTIONAL REAL;
  with_lower_flange_width     : OPTIONAL REAL;
  with_upper_flange_thickness : OPTIONAL REAL;
  with_upper_flange_width     : OPTIONAL REAL;
  with_web_thickness          : REAL;
  with_web_height             : REAL;
END_ENTITY;
(*
```

1.3.2.2.8 CROSS SECTION PARAMETER
A CROSS SECTION PARAMETER describes an attribute of a shape part cross section. These include web and flange dimensions, fillet radius, etc.

```
*)
ENTITY cross_section_parameter;
  having_cross_section_value : REAL;
```

```
  having_cross_section_code   : UNIQUE cross_section_code;
  for_standard_cross_section : OPTIONAL SET [1:#] OF
                                 standard_cross_section;
WHERE
  is_uniquely_determined_by (having_cross_section_code,
                             having_cross_section_value);
END_ENTITY;
(*
```

1.3.2.2.9 N/C MARK

An N/C MARK is a piece of text or contour marked on a plate part or shape part by a cam device such as a burning machine or robot. Such marking is typically performed by a punch marker, zinc marker or an ink-jet marker.

```
*)
ENTITY nc_mark
  SUPERTYPE OF (nc_text XOR
                trace_mark);
  identified_by_nc_mark_id   : UNIQUE nc_mark_id;
  identified_by_nc_mark_type : nc_mark_type;
  marking_plate_part         : OPTIONAL SET [1:#] OF plate_part;
  marking_shape_part         : OPTIONAL SET [1:#] OF shape_part;
WHERE
  must_be_at_least_one_of (nc_text,
                           trace_mark);
  has_at_least_one_of (marking_plate_part,
                       marking_shape_part);
  has_at_most_one_of (marking_plate_part,
                      marking_shape_part);
END_ENTITY;
(*
```

1.3.2.2.10 N/C TEXT

An N/C TEXT is a type of N/C-mark consisting of a character string.

```
*)
ENTITY nc_text
  SUBTYPE OF (nc_mark);
  defined_with_nc_text_string : nc_text_string;
  having_nc_text_parameter    : SET [1:#] OF nc_text_parameter;
  oriented_by_unit_vector     : unit_vector;
  located_at_node             : node;
END_ENTITY;
```

```
(*
```

1.3.2.2.11 N/C TEXT PARAMETER
An N/C TEXT PARAMETER describes an attribute of an N/C text mark, ie. size, font, etc.

```
*)
ENTITY nc_text_parameter;
  defined_with_nc_text_parameter_code : UNIQUE nc_text_parameter_code;
  defined_with_nc_text_parameter_value : nc_text_parameter_value;
  for_nc_text                : OPTIONAL SET [1:#] OF nc_text;
WHERE
  is_uniquely_determined_by (defined_with_nc_text_parameter_code,
                             defined_with_nc_text_parameter_value);
END_ENTITY;
(*
```

1.3.2.2.12 TRACE MARK
A TRACE MARK is a numerically controlled (N C) marking contour.

```
*)
ENTITY trace_mark
  SUBTYPE OF (nc_mark);
  with_mark_interval         : mark_interval;
  with_mark_length           : mark_length;
  defined_on_path_segment    : path_segment;
END_ENTITY;
(*
```

1.3.2.2.13 ENDCUT
The ENDCUT provides the flange/web cut details at the ends of a shape part.
Integration point: Form Features model.

```
*)
ENTITY endcut
  SUPERTYPE OF (transition_cut AND
               (non_parametric_endcut XOR
               parametric_endcut));
  defined_with_endcut_type  : UNIQUE endcut_type;
  at_end_of_shape_part      : OPTIONAL SET [1:#] OF shape_part;
  at_start_of_shape_part    : OPTIONAL SET [1:#] OF shape_part;
  for_part_flange           : OPTIONAL SET [1:#] OF part_flange;
WHERE
```

```
  must_be_at_least_one_of (non_parametric_endcut,
                           parametric_endcut);
END_ENTITY;
(*
```

1.3.2.2.14 ENDCUT PARAMETER

An ENDCUT PARAMETER describes an attribute of a parametric endcut. For example, snipe and radius are two attributes of an endcut.

```
*)
ENTITY endcut_parameter;
  having_endcut_parameter_value : REAL;
  having_endcut_parameter_code  : UNIQUE endcut_parameter_code;
  for_parametric_endcut         : OPTIONAL SET [1:#] OF
                                    parametric_endcut;
WHERE
  is_uniquely_determined_by (having_endcut_parameter_value,
                             having_endcut_parameter_code);
END_ENTITY;
(*
```

1.3.2.2.15 PARAMETERIC ENDCUT

A PARAMETRIC ENDCUT is an endcut with the cut geometry specified by reference to a standard endcut and its associated endcut parameters.

```
*)
ENTITY parametric_endcut
  SUBTYPE OF (endcut);
  identified_by_parametric_endcut_id : parametric_endcut_id;
  defined_by_endcut_parameter        : SET [1:#] OF endcut_parameter;
END_ENTITY;
(*
```

1.3.2.2.16 NON-PARAMETRIC ENDCUT

A NON-PARAMETRIC ENDCUT is an endcut with non-standard cut geometry.

```
*)
ENTITY non_parametric_endcut
  SUBTYPE OF (endcut);
  with_web_curve_geometry   : curve_geometry;
  with_lower_flange_curve_geometry : OPTIONAL curve_geometry;
  with_upper_flange_curve_geometry : OPTIONAL curve_geometry;
```

```
END_ENTITY;
(*
```

1.3.2.2.17 TRANSITION CUT

A TRANSITION CUT defines the cuts to produce a transition between two different size shape parts

```
*)
ENTITY transition_cut
  SUBTYPE OF (endcut);
END_ENTITY;
)*
```

1.3.3 LIBRARY PARTS

1.3.3.1 LIBRARY PART TYPES

1.3.3.1.1 LIBRARY ID

A LIBRARY ID is the unique identifier of a library containing library parts.

```
*)
TYPE library_id = string(32);
END_TYPE;
(*
```

1.3.3.1.2 LIBRARY PART ID

A LIBRARY PART ID is the unique identifier of a library part within a library.

```
*)
TYPE library_part_id = string(6);
END_TYPE;
(*
```

1.3.3.1.3 LIBRARY PART TYPE

A LIBRARY PART TYPE identifies the type of library part; p = parametric and n = non-parametric.

```
*)
TYPE library_part_type = string(1);
END_TYPE;
(*
```

1.3.3.1.4 LIBRARY VERSION

The LIBRARY VERSION identifies the version of the library being used, ie. Rev. A, Rev. B, etc.

```
*)
TYPE library_version = string(4);
END_TYPE;
(*
```

1.3.3.1.5 PART PARAMETER CODE

The parametric library PART PARAMETER CODE identifies the variables which define a parametric library part, ie. t = thickness.

```
*)
TYPE part_parameter_code = string(2);
END_TYPE;
(*
```

1.3.3.2 LIBRARY PART ENTITIES

1.3.3.2.1 PART LIBRARY

A PART LIBRARY is a collection of standard parts. also refered to as a standard parts catalog, which provides a pre-defined part for use in the ship product model.

```
*)
ENTITY part_library;
  identified_by_library_version : library_version;
  identified_by_library_id      : UNIQUE library_id;
  containing_library_part       : SET [1:#] OF library_part;
WHERE
  is_uniquely_determined_by (identified_by_library_id,
                             identified_by_library_version);
END_ENTITY;
(*
```

1.3.3.2.2 LIBRARY PART

A LIBRARY PART is a pre-defined part contained in a library. There are two types of LIBRARY PARTs, parametric and non-parametric. Chocks, brackets and gussets are examples of library parts.

```
*)
ENTITY library_part
  SUPERTYPE OF (non_parametric_library_part XOR
```

```
                  parametric_library_part)
  SUBTYPE OF (part);
  identified_by_library_part_id : UNIQUE library_part_id;
  identified_by_library_part_type : library_part_type;
  oriented_by_transformation_matrix : transformation_matrix;
  located_by_node             : node;
  defined_in_library          : SET [1:#] OF library;
  joined_by_nodal_joint       : OPTIONAL SET [1:#] OF nodal_joint;
WHERE
  must_be_at_least_one_of (non_parametric_library_part,
                           parametric_library_part);
END_ENTITY;
(*
```

1.3.3.2.3 NON-PARAMETRIC LIBRARY PART

A NON-PARAMETRIC LIBRARY PART is a type of library part whose geometry (shape, size) is the same for every occurence.

```
*)
ENTITY non_parametric_library_part
  SUBTYPE OF (library_part);
  defined_by_shape_part       : OPTIONAL SET [1:#] OF shape_part;
  defined_by_plate_part       : OPTIONAL SET [1:#] OF plate_part;
END_ENTITY;
(*
```

1.3.3.2.4 PARAMETRIC LIBRARY PART

A PARAMETRIC LIBRARY PART is a type of library part whose geometry is necessarily defined by both the library part ID and parameter values associated with a particular library part occurence. two parametric library parts with the same library part ID may or may not be physically identical.

```
*)
ENTITY parametric_library_part
  SUBTYPE OF (library_part);
  defined_by_part_parameter : SET [1:#] OF part_parameter;
END_ENTITY;
(*
```

1.3.3.2.5 PART PARAMETER

A parametric library PART PARAMETER describes the attributes associated with a parametric library part. These include shape and size.

```
*)
ENTITY part_parameter;
  defined_by_part_parameter_value : part_parameter_value;
  defined_by_part_parameter_code : part_parameter_code;
  defining_parametric_library_part : OPTIONAL SET [1:#] OF
                                          parametric_library_part;
WHERE
  is_uniquely_determined_by (defined_by_part_parameter_code,
                             defined_by_part_parameter_value);
END_ENTITY;
(*
```

1.4 STRUCTURAL JOINTS

A joint is a connection between plate parts and/or shape parts. Usually the connection is made by welding but may be bolted or riveted. Joints in this model carry all the necessary relationships and information for joining two or more parts.

1.4.1 STRUCTURAL JOINT TYPES

1.4.1.1 BOLT PARAMETER CODE

A BOLT PARAMETER CODE identifies a variable which is used to define an attribute of a bolt, ie. d = diameter, t = threads/inch, etc.

```
*)
TYPE bolt_parameter_code = String(2);
END_TYPE;
(*
```

1.4.1.2 BOLT PROCESS

The BOLTING PROCESS describes the process for a bolted joint. This includes references to standards or detailed drawings providing bolting pattern, tightening torque, etc.

```
*)
TYPE bolt_process = string(100);
END_TYPE;
(*
```

1.4.1.3 INSPECTION PROCEDURE

An INSPECTION PROCEDURE provides the description for the inspection of a joint. The INSPECTION PROCEDURE may include references to standard inspection procedures, ie. magnaflux, visual, x-ray, etc.

```
*)
TYPE inspection_process = string(100);
END_TYPE;
(*
```

1.4.1.4 JOINT ID

A JOINT ID is the unique identification of a joint.

```
*)
TYPE joint_id = string(6);
END_TYPE;
(*
```

1.4.1.5 JOINT TYPE

A JOINT TYPE indicates the type of joint, that is, whether it is a nodal joint or a path joint.

```
*)
TYPE joint_type = ENUMERATION OF
     (path_joint,
      nodal_joint);
END_TYPE;
(*
```

1.4.1.6 JOINING PROCEDURE

The JOINING PROCEDURE describes the joining of parts at a joint. Joining procedures may include references to standard joining procedures.

```
*)
TYPE joining_procedure = string(100);
END_TYPE;
(*
```

1.4.1.7 RIVET PARAMETER CODE

A RIVET PARAMETER CODE identifies the variables defining the attributes of a rivet. for example, d = diameter, l = length, etc.

```
*)
TYPE rivet_parameter_code = string(2);
END_TYPE;
(*
```

1.4.1.8 RIVET PROCESS

The RIVET PROCESS describes the process for a riveted joint. This includes references to standards or detailed drawings providing riveting pattern, etc.

```
*)
TYPE rivet_process = string(100);
END_TYPE;
(*
```

1.4.1.9 STANDARD DETAILS REFERENCE

The STANDARD DETAILS REFERENCE provides references to standard details, drawings and procedures identifying information for creating a joint.

```
*)
TYPE standard_details_reference = string(100);
END_TYPE;
(*
```

1.4.1.10 WELD TYPE

The WELD TYPE identifies the type of weld; ie. fillet, full penetration, etc.

```
*)
TYPE weld_type = string(6);
END_TYPE;
(*
```

1.4.1.11 WELD PROCESS

The WELD PROCESS describes the process for a welded joint. This includes references to standards and detailed drawings.

```
*)
TYPE weld_process = string(100);
END_TYPE;
(*
```

1.4.2 STRUCTURAL JOINT ENTITIES

1.4.2.1 JOINT

A JOINT is used to define the physical connection between shape parts and plate parts. JOINTs occur at path segments and nodes. The type of joint is identified as either path joint or nodal joint.

```
*)
ENTITY joint
  SUPERTYPE OF ((nodal_joint XOR
                path_joint) AND
                (bolt_joint XOR
                rivet_joint XOR
                weld_joint));
  identified_by_joint_id                 : UNIQUE joint_id;
  having_joint_type                      : joint_type;
  with_inspection_process                : OPTIONAL inspection_process;
  joined_by_joining_procedure            : joining_procedure;
  refering_to_standard_details_reference : UNIQUE
                                           standard_details_reference;
  penetrating_cutout_hole                : OPTIONAL SET [1:#] OF
                                             cutout_hole;
WHERE
  must_be_at_least_one_of (nodal_joint,
                           path_joint);
END_ENTITY;
(*
```

1.4.2.2 BOLT JOINT

A BOLT JOINT is a type of joint using threaded fastenings to secure two or more parts together.

```
*)
ENTITY bolt_joint
  SUBTYPE OF (joint);
  described_by_bolt_process : bolt_process;
  defined_by_bolt_parameter : SET [1:#] OF bolt_parameter;
END_ENTITY;
(*
```

1.4.2.3 BOLT PARAMETER

A BOLT PARAMETER describes an attribute of a bolt used in a bolt joint, ie. bolt diameter, bolt length, threads/inch etc.

```
*)
ENTITY bolt_parameter;
  having_bolt_parameter_value : REAL;
  having_bolt_parameter_code  : UNIQUE bolt_parameter_code;
  for_bolt_joint              : OPTIONAL SET [1:#] OF bolt_joint;
WHERE
  is_uniquely_determined_by (having_bolt_parameter_code,
```

```
                              having_bolt_parameter_value);
END_ENTITY;
(*
```

1.4.2.4 NODAL JOINT

A NODAL JOINT is a type of joint. A NODAL JOINT is a connection between at least two parts occuring at a node.

```
*)
ENTITY nodal_joint
  SUBTYPE OF (joint);
  joining_plate_part          : OPTIONAL SET [1:#] OF plate_part;
  joining_library_part        : OPTIONAL SET [1:#] OF library_part;
  joining_shape_part          : OPTIONAL SET [1:#] OF shape_part;
  located_at_node             : node;
END_ENTITY;
(*
```

1.4.2.5 PATH JOINT

A PATH JOINT is a type of joint. A PATH JOINT is a connection between at least two part (shape or plate) edges.

```
*)
ENTITY path_joint
  SUBTYPE OF (joint);
  joining_shape_part_edge     : OPTIONAL SET [1:#] OF shape_part_edge;
  joining_path_segment        : OPTIONAL SET [1:#] OF path_segment;
  joining_plate_part_edge     : OPTIONAL SET [1:#] OF plate_part_edge;
END_ENTITY;
(*
```

1.4.2.6 RIVET JOINT

A RIVET JOINT is a type of joint using rivets to securly fasten two or more parts together.

```
*)
ENTITY rivet_joint
  SUBTYPE OF (joint);
  with_rivet_process          : rivet_process;
  with_rivet_parameter        : SET [1:#] OF rivet_parameter;
END_ENTITY;
(*
```

1.4.2.7 RIVET PARAMETER

A RIVET PARAMETER contains an attribute of a rivet used in a riveted joint, ie. rivet diameter, rivet length, etc.

```
*)
ENTITY rivet_parameter;
  having_rivet_parameter_value : REAL;
  having_rivet_parameter_code  : UNIQUE rivet_parameter_code;
  for_rivet_joint              : OPTIONAL SET [1:#] OF rivet_joint;
WHERE
  is_uniquely_determined_by (having_rivet_parameter_code,
                             having_rivet_parameter_value);
END_ENTITY;
(*
```

1.4.2.8 WELD JOINT

A WELD JOINT is a type of joint using material fusion to join parts together along a seam. This is the most common type of joint used in shipbuilding today.

```
*)
ENTITY weld_joint
  SUBTYPE OF (joint);
  with_weld_process          : weld_process;
  of_weld_size               : weld_size;
  with_weld_type             : weld_type;
END_ENTITY;
(*
```

1.5 STRUCTURAL OPENINGS

1.5.1 STRUCTURAL OPENING TYPES

1.5.1.1 HOLE ID

A HOLE ID is the unique identifier for a structural-opening.

```
*)
TYPE hole_id = string(6);
END_TYPE;
(*
```

1.5.1.2 HOLE TYPE

A HOLE TYPE indicates the type of structural opening, that is, whether the opening is parametric or non-parametric.

```
*)
TYPE hole_type = ENUMERATION OF
     (non-parametric,
      parametric);
END_TYPE;
(*
```

1.5.1.3 OPENING PARAMETER CODE

The OPENING PARAMETER CODE identifies the variables which define a parametric opening, ie. r = radius.

```
*)
TYPE opening_parameter_code = string(2);
END_TYPE;
(*
```

1.5.1.4 PARAMETRIC OPENING ID

A PARAMETRIC OPENING ID is the unique identifier of a parametric (standard) opening. It is the name of a standard library opening or the name of the macro creating the opening.

```
*)
TYPE parametric_opening_id = string(4);
END_TYPE;
(*
```

1.5.1.5 DISTRIBUTION SYSTEM PART ID

A DISTRIBUTION SYSTEM PART ID is the unique identifier of a distribution part.

```
*)
TYPE distribution_system_part_id = string(6);
END_TYPE;
(*
```

1.5.1.6 PENETRATION PART ID

A PENETRATION PART ID is the unique identifier of a pentration part.

```
*)
TYPE penetration_part_id = string(6);
END_TYPE;
(*
```

1.5.2 STRUCTURAL OPENING ENTITIES

1.5.2.1 STRUCTURAL OPENING

A STRUCTURAL OPENING is an opening in a part to allow penetration of another part, penetration of a distribution system part, passage of a fluid or to lighten the part.

```
*)
ENTITY structural_opening
  SUPERTYPE OF (system_penetration_hole XOR
                access_lightening_hole OR
                cutout_hole AND
                (non_parametric_opening XOR
                parametric_opening));
  defined_by_hole_type        : hole_type;
  identified_by_hole_id       : UNIQUE hole_id;
  cutting_plate_part          : OPTIONAL SET [1:#] OF plate_part;
  cutting_shape_part          : OPTIONAL SET [1:#] OF shape_part;
  defined_on_bounded_surface  : OPTIONAL bounded_surface;
  having_edge_preparation     : OPTIONAL edge_preparation;
WHERE
  must_be_at_least_one_of_3 (access_lightening_hole,
                             cutout_hole,
                             system_penetration_hole);
  must_be_at_least_one_of (non_parametric_opening,
                           parametric_opening);
  has_at_least_one_of (cutting_plate_part,
                       cutting_shape_part);
  has_at_most_one_of (cutting_plate_part,
                      cutting_shape_part);
END_ENTITY;
(*
```

1.5.2.2 PARAMETRIC OPENING

A PARAMETRIC OPENING is a type of standard structural opening whose geometry opening is defined by reference to a parametric opening id along with parameters and an orientation for the particular opening occurence.

```
*)
ENTITY parametric_opening
  SUBTYPE OF (structural_opening);
  identified_by_parametric_opening_id : parametric_opening_id;
  located_at_node                     : node;
  defined_by_opening_parameter        : opening_parameter;
  oriented_by_unit_vector             : unit_vector;
```

```
END_ENTITY;
(*
```

1.5.2.3 NON-PARAMETRIC OPENING

A NON-PARAMETRIC OPENING is a type of structural opening whose opening is defined by a path-segment representing the hole contour.

```
*)
ENTITY non_parametric_opening
  SUBTYPE OF (structural_opening);
  defined_by_path_segment    : path_segment;
END_ENTITY;
(*
```

1.5.2.4 AIR ESCAPE

An AIR ESCAPE is a type of cutout-hole. Its function is to prevent air pockets from forming when fluids are being transfered into a tank.

```
*)
ENTITY air_escape
  SUBTYPE OF (cutout_hole);
END_ENTITY;
(*
```

1.5.2.5 ACCESS/LIGHTENING HOLE

An ACCESS/LIGHTENING HOLE is a type of structural-opening used primarily for accessing areas of a hull or for lightening structure. ACCESS/LIGHTENING HOLEs may be penetrated by distribution system parts (a non-tight type of penetration).

```
*)
ENTITY access_lightening_hole
  SUBTYPE OF (structural_opening);
  penetrated_by_distribution_system_part : OPTIONAL SET [1:#] OF
                                           distribution_system_part;
END_ENTITY;
(*
```

1.5.2.6 CUTOUT HOLE

A CUTOUT HOLE is a type of structural opening. It is generally a hole in a plate part or a shape part to allow the penetration by a shape part. Other uses are rathole, air escape and oil stop. These holes are not penetrated by shape parts.

```
  identified_by_distribution_system_part_id : UNIQUE
                                   distribution_system_part_id;
  penetrating_system_penetration_hole       : OPTIONAL SET [1:#] OF
                                     system_penetration_hole;
  penetrating_access_lightening_hole        : OPTIONAL SET [1:#] OF
                                      access_lightening_hole;
  penetrating_penetration_part              : OPTIONAL SET [1:#] OF
                                             penetration_part;
END_ENTITY;
(*

*)
END_SCHEMA; -- end of ipim_ship_structure_schema
(*
```

8 LAYERED ELECTRICAL PRODUCT MODEL

~~This model is not available at this time.~~

AMERICAN BINDERY
2
5-94