

Rapid Prototyping of Application-Specific Signal Processors (RASSP)

BUILD 2

ENTERPRISE DATA MODEL REPORT

Authorization: Subcontract No. P.O. TTM 748357

-

Responsible Organization:

Information Requirements and Analysis

Advanced Information Engineering

Approved By:

F. L. Bsharah J. F. Willis (Concurrence)

Project Engineer Manager

Information Requirements and Analysis Advanced Information Engineering

Advanced Information Engineering

Distribution: Authorized only to U.S. Government agencies and private individuals or enterprises eligible to obtain export controlled technical data in accordance with regulations implementing 10 U.S.C. 140c as of 31 March 1987. Other requests may be referred to the Director, Defense Advanced Research Projects Agency.

Warning: This document may contain technical data whose export is restricted by the Arms Export Control Act (Title U.S.C. Sec. 2751 et seq.) or Executive Order 12470. Violators of these export control laws are subject to severe criminal penalties.

Destruction Notice: Destroy by any method that will prevent disclosure of contents or reconstruction of the document.

Date: 20 March 1996

No. of Pages: xii +160



Revision/Change Log

Revision	Reason	Date	Driver (CR/DR)	Description
Baseline	-----	23 December 1994	-----	Initial Release, Version 1
A	Extended REDM to incorporate DM2.0 and Manufacturing Interface constructs.	20 March 1996	-----	Build 2 release

Contents

Section Page

-

Contents

Foreword

Introduction

1 Scope

2 Definitions and Abbreviations

3 Enterprise Information Requirements

3.1 Enterprise Objects

3.1.1 action

3.1.2 action_assignment

3.1.3 action_execution

3.1.4 action_execution_support_resource

3.1.5 action_item

3.1.6 action_method

3.1.7 action_method_relationship

3.1.8 action_status

3.1.9 address

3.1.10 ahead_or_behind

3.1.11 approval

3.1.12 approval_assignment

3.1.13 approval_date_time

3.1.14 approval_person_organization

3.1.15 approval_role

3.1.16 approval_status

3.1.17 approved_item

3.1.18 assembly_component_usage

3.1.19 assembly_component_usage_substitute

3.1.20 cage

3.1.21 calendar_date

3.1.22 classified_item

3.1.23 concurrent_action_method

3.1.24 configuration_design

3.1.25 configuration_item

3.1.26 context_dependent_unit

3.1.27 contract

3.1.28 contract_assignment

3.1.29 conversion_based_unit

3.1.30 coordinated_universal_time_offset

3.1.31 correspondence

3.1.32 data_template

3.1.33 date

3.1.34 date_and_time

3.1.35 date_time_select

3.1.36 dated_effectivity

3.1.37 day_in_month_number

3.1.38 day_in_week_number

3.1.39 day_in_year_number

3.1.40 derived_unit

3.1.41 derived_unit_element

3.1.42 dimensional_exponents

3.1.43 discrepant_product

3.1.44 document

3.1.45 document_reference

3.1.46 document_type

3.1.47 drawing

3.1.48 enhancement_product

3.1.49 enterprise

3.1.50 file_folder

3.1.51 group

3.1.52 group_relationship

3.1.53 hardware_software

3.1.54 hour_in_day

3.1.55 identifier

3.1.56 label

3.1.57 local_time

3.1.58 lot_effectivity

3.1.59 make_from_usage_option

3.1.60 measure_with_unit

3.1.61 minute_in_hour

3.1.62 month_in_year_number

3.1.63 named_unit

3.1.64 next_assembly_usage_occurrence

3.1.65 node_location

3.1.66 ordered_action

3.1.67 ordinal_date

3.1.68 organization

3.1.69 organizational_address

3.1.70 organizational_project

3.1.71 part

3.1.72 pca_component_usage

3.1.73 person

3.1.74 person_and_organization

3.1.75 person_organization_select

3.1.76 personal_address

3.1.77 physical_unit

3.1.78 planned_effectivity

3.1.79 procedure

3.1.80 process_action_method_relationship

3.1.81 product

3.1.82 product_anomaly

3.1.83 product_anomaly_disposition

3.1.84 product_change

3.1.85 product_classification

3.1.86 product_concept

3.1.87 product_concern

3.1.88 product_definition

3.1.89 product_definition_relationship

3.1.90 product_definition_usage

3.1.91 product_flaw

3.1.92 product_flaw_classification

3.1.93 product_issue

3.1.94 product_process_step

3.1.95 product_requiring_change

3.1.96 product_responsibility

3.1.97 product_state

3.1.98 product_version

3.1.99 product_version_group

3.1.100 program

3.1.101 promissory_usage_occurrence

3.1.102 publication

3.1.103 quantified_assembly_component_usage

3.1.104 recommended_support_resource

3.1.105 related_change

3.1.106 requested_action

3.1.107 reuse_part

3.1.108 role

3.1.109 second_in_minute

3.1.110 security_classification

3.1.111 security_classification_assignment

3.1.112 security_classification_level

3.1.113 sequential_method

3.1.114 serial_action_method

3.1.115 serial_concurrent_action_method

3.1.116 serial_numbered_effectivity

3.1.117 si_prefix

3.1.118 si_unit

3.1.119 si_unit_name

3.1.120 signal_processor_design

3.1.121 software_application

3.1.122 specification

3.1.123 specified_higher_usage_occurrence

3.1.124 specified_item

3.1.125 supplier

3.1.126 support_equipment

3.1.127 support_resource_select
3.1.128 system
3.1.129 system_user
3.1.130 text
3.1.131 unit
3.1.132 week_in_year_number
3.1.133 week_of_year_and_day_date
3.1.134 year_number

3.2 Enterprise Object Assertions

3.2.1 action to action_method
3.2.2 action_assignment to action
3.2.3 action_execution to ordered_action
3.2.4 action_execution_support_resource to action_execution
3.2.5 action_item to product_version
3.2.6 action_method to requested_action
3.2.7 action_method_relationship to action_method
3.2.8 action_status to action_execution
3.2.9 approval to approval_status
3.2.10 approval_assignment to approval
3.2.11 approval_date_time to approval
3.2.12 approved_item to product_version
3.2.13 approval_person_organization to approval
3.2.14 approval_person_organization to approval_role
3.2.15 assembly_component_usage_substitute to assembly_component_usage
3.2.16 classified_item to product_version
3.2.17 configuration_design to configuration_item
3.2.18 configuration_design to product_version
3.2.19 configuration_item to product_concept
3.2.20 contract_assignment to contract
3.2.21 contract_assignment to product_version
3.2.22 conversion_based_unit to measure_with_unit
3.2.23 date_and_time to date
3.2.24 date_and_time to local_time

3.2.25 dated_effectivity to date_and_time

3.2.26 derived_unit to derived_unit_element

3.2.27 derived_unit_element to named_unit

3.2.28 document to document_type

3.2.29 document_reference to document

3.2.30 file_folder to product_version

3.2.31 group_relationship to group

3.2.32 lot_effectivity to measure_with_unit

3.2.33 make_from_usage_option to measure_with_unit

3.2.34 named_unit to dimensional_exponents

3.2.35 node_location to hardware_software

3.2.36 ordered_action to requested_action

3.2.37 organization to cage

3.2.38 organizational_address to organization

3.2.39 organizational_project to organization

3.2.40 person_and_organization to organization

3.2.41 person_and_organization to person

3.2.42 personal_address to person

3.2.43 physical_unit to configuration_design

3.2.44 planned_effectivity to configuration_design

3.2.45 planned_effectivity to product_definition_usage

3.2.46 product_anomaly_disposition to action_execution

3.2.47 product_anomaly_disposition to product_anomaly

3.2.48 product_change to product_anomaly_disposition

3.2.49 product_change to product_requiring_change

3.2.50 product_change to product_version

3.2.51 product_classification to product

3.2.52 product_definition to product_version

3.2.53 product_definition_relationship to product_definition

3.2.54 product_flaw_classification to product_flaw

3.2.55 product_process_step to product

3.2.56 product_requiring_change to action_execution

3.2.57 product_requiring_change to product_anomaly

3.2.58 product_responsibility to organizational_project

- 3.2.59 product_responsibility to product*
- 3.2.60 product_state to action_execution*
- 3.2.61 product_state to product_version*
- 3.2.62 product_version to product*
- 3.2.63 product_version_group to group*
- 3.2.64 product_version_group to product_version*
- 3.2.65 quantified_assembly_component_usage to measure_with_unit*
- 3.2.66 recommended_support_resource to action_item*
- 3.2.67 related_change to product_anomaly*
- 3.2.68 related_change to product_requiring_change*
- 3.2.69 security_classification to security_classification_level*
- 3.2.70 security_classification_assignment to security_classification*
- 3.2.71 serial_numbered_effectivity to physical_unit*
- 3.2.72 specific_higher_usage_occurrence to assembly_component_usage*
- 3.2.73 specific_higher_usage_occurrence to next_assembly_component_usage*
- 3.2.74 specified_item to product_version*
- 3.2.75 supplier to date*
- 3.2.76 supplier to product*
- 3.2.77 system_user to hardware_software*

Annex A RASSP Enterprise Data Model

A.1 RASSP Enterprise data model EXPRESS

A.2 RASSP Enterprise Data Model EXPRESS-G

Annex B Bibliography

Foreword

This document has been prepared by the Information Requirements and Analysis (IR&A) group of Rockwell's Advanced Information Engineering (AIE) organization.

This document identifies and defines the enterprise information requirements of Build 2 for the Rapid Prototyping of Application Specific Signal Processors (RASSP) Enterprise Framework. The RASSP Enterprise Data Model (REDM) is documented in this report.

The REDM is utilized to ensure that RASSP applications may be integrated. The Application Interpreted Models that reference the REDM are documented in separate reports and are available to the reader from their local RASSP representative (see Bibliography at the end of this document).

Introduction

The Build 2 Enterprise Data Model was developed by leveraging the Build 1 Enterprise Data Model's data structures and extending their constructs. The model enhancements include data constructs from Intergraph's Desktop Manager 2.0 Product Data Manager and South Carolina Research Authority's (SCRA) Manufacturing Interface and Proposed Parts Taxonomy. The Build 2 Enterprise Data Model Report identifies and defines the enterprise information requirements for the RASSP Enterprise Framework. The information requirements are represented in the RASSP Enterprise Data Model (REDM). The EXPRESS and EXPRESS-G modeling language is used to document the REDM. For Build 1, the REDM was developed via feedback from the RASSP Enterprise Team Members and incorporation of the Build 0 Application Interpreted Model. Specifically, the SCRA STEP Configuration Management Suitability Report (MMC-RASSP-2.01.00) provided guidance in the development process. For Build 2, the REDM was extended to include not only configuration management constructs but the design to manufacturing interface constructs. By identifying the mapping between manufacturing and design areas, modularity of the manufacturing interface systems is provided. SCRA's two reports RASSP PCA Manufacturing Interface Definition (MMC-RASSP-3.01.00) and Proposed Parts Taxonomy for RASSP Manufacturing Library Information (MMC-RASSP-4.01.00) were used in development of the Build 2 REDM. The REDM represents the enterprise information regarding configuration management and design to manufacturing interface principles.

Industries have a need to communicate to their suppliers, customers, clients, and contractors any product problems or anomalies, the corrections for these problems and any resulting corrective actions or changes. The products supported by the RASSP configuration management process are those for which RASSP wishes to maintain a change history such as files, discrete parts or components, assemblies, documents, and signal processors designs. The planning model representing the high level concepts of the RASSP Enterprise Framework is depicted in Figure 1.



Figure 1 - RASSP Enterprise Planning Model

Clause 1 defines the scope of the Build 2 RASSP Enterprise Data Model. Clause 2 provides generic definitions for data covered by the REDM. The enterprise information requirements represented by the REDM are specified in clause 3 using a generic terminology that is appropriate at the enterprise level. A graphical representation of the enterprise information requirements, referred to as the RASSP Enterprise Data Model, is given in annex A.

1 Scope

This report specifies the use of the information resources, as defined by the STEP standards and the RASSP enterprise, necessary for the scope and information requirements for the configuration management and manufacturing interface processes. Configuration management for an item (product) includes the identification of the reason for a change, its cause, the approval and performance of the resulting changes to the item, and the authorization of corrective actions to prevent reoccurrence. Manufacturing interface includes data constructs necessary to define the design to manufacturing interface. The identified information provides configuration management support throughout the life cycle of an application specific signal processor and its relationship to manufacturing. This support includes areas such as design, manufacture, production, and technical publication generation.

The following are within the scope of the Enterprise Data Model:

- Configuration Management

- Identification of the item requiring change;

- The classification of each item requiring change as either discrepant or needing enhancement;

- The identification of an anomaly in the form of a flaw or an issue that results from corrective, perfecting, adaptive, or preventative needs. An identified anomaly applies to a product or one or more versions of a file that requires a change;

- The specification of the tasks required implementation of a change and inspection of that changed product to verify that the change requirements have been properly implemented.

- Manufacturing Interface

- Identification of the types of manufacturing data necessary to manufacture a digital signal processor directly from an engineer's design;

The following are outside the scope of this report:

- The usage of the change information in planning and administration functions;

- The scope of management concern.

2 Definitions and Abbreviations

This report makes use of the following terms and definitions:

anomaly: A description of either a product problem or enhancement that may result in a change requirement. The product problems are deviations from the expected product characteristics. A product enhancement identifies the need for new and or improved product characteristics. Product characteristics are the form, fit, and function properties of a product as well as any other descriptive traits.

authorization: The decision making mechanism through which the appropriate level of permission is granted to proceed with the execution of planned actions or resource allocations. A commitment or acknowledgment to perform a particular process step or series of steps.

change management: A procedure used by the functional organizations within an enterprise. The purpose is to determine which functions are impacted by a change activity and coordinate the tasks that will be involved throughout the entire change procedure.

The management of a change process is conducted in two parts: 1.) the design activity which involves all the design and administrative activities involved in the disposition of a change need and 2.) the actual implementation activity which involves the actual change process to an item requiring change. Change management includes the conceptual design, final design process, testing procedures and final delivery.

corrective action: An action taken to prevent a product anomaly from reoccurring. Corrective action may include any or all of the following steps; localization, isolation, disassembly, re-assembly, alignment, and checkout.

change requirements: The reason for the condition of changing, altering or modifying, transformation, replacing of one thing for another substitution and a transition from one state, condition, phase to another of that which is required.

product data: A single article or unit included in a collection, enumeration, or series that collectively defines a product and is specified separately from the product.

updates applicable to either product improvements and/or major modifications: Update reviews should be initiated as a result of discrepancies reported on previous reviews, to provide an audit trail for follow up improvements and corrective actions. The update review should assess the present status of the fielded system against the baseline established by the previous fielded history review.

product functionality: A description of the requirement that is satisfied by the product.

related change: A change to a product that is required because of a problem, enhancement need, or corrective action associated with a related anomaly.

support resource: A product required to design, build, operate, and maintain another product. A resource may be a facility, tool, person, or documentation.

process step: A unit of specific work behavior with a clear beginning and ending. The process step describes the performance of a meaningful function.

Unit of Functionality: A grouping of objects (entities, attributes, enumerations, etc.). The Unit of Functionality (UoF) is used to organize and summarize one or more concepts of operation into reusable capabilities.

3 Enterprise Information Requirements

This clause specifies the enterprise information constructs required for the configuration management and design to manufacturing interface of a RASSP product (application specific signal processor).

The RASSP Enterprise Data Model (REDM) is an enterprise information model employing STEP standards and RASSP Enterprise Framework terminology. Using the STEP community's terminology allows for the creation of an enterprise model that is easily traceable to the STEP standard. This traceability is essential when the implementation and integration of the RASSP Enterprise Framework with the STEP community is addressed. The REDM, for Build 2, was based on the the REDM from Build 1, the Application Interpreted Model (AIM) from Build 1, and SCRA documents MMC-RASSP-3.01.00, RASSP PCA Manufacturing Interface Definition, and MMC-RASSP-4.01.00, Proposed Parts Taxonomy for RASSP Manufacturing Library Information.

The enterprise information requirements are specified as enterprise objects and enterprise assertions. These assertions pertain to individual enterprise objects and to relationships between enterprise objects. The information requirements are defined using the terminology of the RASSP and STEP communities.

The entities (and their associated attributes and rules) enumerated in Table 1 are not defined in this document. Their definitions and rules may be found in the MMC-RASSP-3.01.00 RASSP PCA Manufacturing Interface Definition document.

CAD_data	SMDCAD_data	SMD_header
SMD_records	SMD_rotation_flag	board_extremes
board_id	board_ident	cad_prog_name
change_no	change_number	date_of_creation
det_12_bite_bf	det_1_bite_f	det_2_bite_bf
det_NOR_rel_lev	det_artwork	det_assy
det_assy_doc_rev	det_assy_name	det_compon_cnt
det_compon_descr	det_compon_nam	det_compon_p_n
det_ece	det_eff	det_pfn
det_qty_used	det_ref_desig	det_request_cnt
det_u_m	edge_con_bot	edge_con_id
edge_con_left	edge_con_right	edge_con_top
eight_bite_bf	end_of_sub_file	factory_interface_information
five_bite_bf	flags	header_sub_file
height_of_pad	hole_data	hole_diameter
hole_size	holes	layer_data
line_point_pairs	line_segments	lines
ll_x_cord	ll_y_cord	lower_left_x_coord_rect_pad
lower_left_y_coord_rect_pad	mfg_interface_requirements	mask_clearance
mask_flag	mask_space	node_number
non_plated_holes	non_plated_pads	num_layers
number_of_layers	number_of_point_on_line	number_plated_hole_sizes
oscc_ploutput_data	oscc_ploutput_data	oscc_ploutput_data_file
oscc_smoutput_data_file	pads	pin_number
plated_hole_size	plated_holes	plated_pads
rect_pads_flag	rectangular_pads	ref_des
rev_letter	rotation_flag	shape_name
silk_screen	silk_screen_bot	silk_screen_data

silk_screen_top	time_record	two_bite_bf
type_id	type_layup	upper_right_x_coord_rect_pad
upper_right_y_coord_rect_pad	ur_x_cord	ur_y_cord
vapes	width_of_line	width_of_pad
x_component_center	x_hole_center	x_pad_center
x_point_on_line	y_component_center	y_hole_center
y_pad_center	y_point_on_line	

Table 1 - PCA Manufacturing Interface Entities

The entities (and their associated attributes and rules) enumerated in Table 2 are not defined in this document. Their definitions and rules may be found in the MMC-RASSP-4.01.00 Proposed Parts Taxonomy for RASSP Manufacturing Library Information document.

assembly_spec_level	bom_quantity_unit
chemical_class	chemical_subclass
chemical_type	electronic_functional_unit_logic
electronic_part_class	electronic_part_package
electronic_part_subclass	electronic_part_type
engineering_unit	hardware_class
hardware_subclass	hardware_type
lead_material	lead_plating
lead_solderability	pcb_package
polarity_name	printed_circuit_board_class
printed_circuit_board_subclass	printed_circuit_board_type
pca_component	assembly_specification
bill_of_material_item	chemical
commercial_form_factor	component_specification
electronic_functional_parameters	electronic_functional_unit
electronic_part	functional_unit
hardware	military_form_factor
packaged_part	packaged_part_lead
pca_specification	pcb_specification

Table 2 - Manufacturing Library Entities

NOTES

1 - A graphical representation of the enterprise information requirements is given in annex A.

3.1 Enterprise Objects

3.1.1 action

An action is the specific effort to realize a specific result. An action is a type of product.

— method.

3.1.1.1 method

The method is the procedure used to carry out the action.

3.1.2 action_assignment

An `action_assignment` is an association of an action with product data.

— `assigned_action`.

3.1.2.1 `assigned_action`

The `assigned_action` is the action which is to be associated with the product data.

3.1.3 `action_execution`

An `action_execution` is an action which has been carried out.

— `order`.

3.1.3.1 `order`

An `order` is the `action_order` against which the `action_execution` was made.

3.1.4 `action_execution_support_resource`

The `action_execution_support_resource` is the actual `support_resources` used/consumed in each execution of an action.

— `executed_action`;

— `supporting_resource`;

3.1.4.1 executed_action

The `executed_action` is the execution of an action that is performed by a `support_resource`.

3.1.4.2 supporting_resource

The `supporting_resource` is the support resource (person or organization) that is executing the action.

3.1.5 action_item

An `action_item` is the association of an action to a `product_version`.

— items.

3.1.5.1 items

Items are a set of `product_versions` which are associated to particular actions that are or are to be carried out.

3.1.6 action_method

An `action_method` is a potential means of satisfying the requirements that are highlighted in a `requested_action`.

— consequence;

— purpose;

— requests.

3.1.6.1 consequence

A consequence is an informal description of the effects of the `action_method`.

3.1.6.2 purpose

The purpose is an informal description of the rationale behind the `action_method`.

3.1.6.3 requests

The requests is `requested_actions` which could be satisfied by this `action_method`.

3.1.7 action_method_relationship

An `action_method_relationship` is an association between two `action_methods`.

— name;

— description;

— `relating_action_method`;

— related_action_method.

3.1.7.1 name

A name is the word, or group of words, by which the action_method_relationship is referred to.

3.1.7.2 description

The description is text that relates the nature of the action_method_relationship.

3.1.7.3 relating_action_method

The relating_method is one of the related actions.

3.1.7.4 related_action_method

The related_method is the other related action.

3.1.8 action_status

An action_status is the ranking which gives an indication of the state of an action.

EXAMPLE - Effectivity from a particular date or across specific batches are examples of action_statuses.

— status;

— assigned_action.

3.1.8.1 status

The status of the action in terms of what state the action is in.

3.1.8.2 assigned_action

The assigned_action is the action_execution that has an assigned status..

3.1.9 address

Aa address is the place where people and organizations may be reached.

— mail_stop;

— postal_box;

— street;

— street_number;

— town;

— region;

— postal_code;

— country;

— facsimile_number;

— telephone_number;

— electronic_mail_address;

— telex_number.

3.1.9.1 mail_stop

The mail_stop is an organization defined address for internal mail delivery.

3.1.9.2 postal_box

The postal_box: is the number of a postal box.

3.1.9.3 street

The street is the name of a street.

3.1.9.4 street_number

The `street_number` is the number of a building on a street.

3.1.9.5 town

The `town` is the name of a town.

3.1.9.6 region

The `region` is the name of a region.

EXAMPLE - The counties of Great Britain and the states of North America are examples of regions.

3.1.9.7 postal_code

The `postal_code` is the code that is used by the country's postal service.

3.1.9.8 country

The `country` is the name of a country.

3.1.9.9 facsimile_number

The `facsimile_number` is the number at which facsimiles may be received.

3.1.9.10 telephone_number

The telephone_number is the number at which telephone calls may be received.

3.1.9.11 electronic_mail_address

The electronic_mail_address is the electronic address at which electronic mail may be received.

3.1.9.12 telex_number

The telex_number is the number at which telex calls may be received.

Formal propositions:

WR1: at least one of the attributes shall have a value.

3.1.10 ahead_or_behind

An ahead_or_behind type is used to specify whether a given time is ahead of or behind coordinated universal time.

3.1.11 approval

An approval is a confirmation of the quality of the product data which it is related to.

— status;

— level.

EXAMPLE - One possible level of approval is "released for production"; this explicitly identifies the approved usage. Another possible level is "preliminary design completed"; this only implies the approved usage which will depend upon company—specific procedures.

3.1.11.1 status

The status of the approval in terms of whether or not that approval has been granted.

3.1.11.2 level

The level is the type or level of approval in terms of the usage that the approval is for. This usage may be implied rather than explicit.

3.1.12 approval_assignment

An approval_assignment is an association of a approval with product data.

— assigned_approval.

3.1.12.1 assigned_approval

The assigned_approval is the approval which is to be associated with the product data.

3.1.13 approval_date_time

A `approval_date_time` is the association of a date and/or time with an approval.

— `date_time`;

— `dated_approval`.

3.1.13.1 date_time

The date and/or time which is to be associated with the approval.

3.1.13.2 dated_approval

The approval which is to be associated with the date and/or time.

3.1.14 approval_person_organization

A `approval_person_organization` is an association of a person and/or organization with an approval.

— `person_organization`;

— `authorized_approval`;

— `role`.

3.1.14.1 person_organization

The person_organization is the person and/or organization which authorizes the approval.

3.1.14.2 authorized_approval

The authorized_approval is the approval which is authorized by the person and/or organization.

3.1.14.3 role

The role is the capacity in which a person and/or organization authorize an approval. The role is the name of the performed function

3.1.15 approval_role

An approval_role is is a function performed with respect to an approval.

— role.

3.1.15.1 role

The role is the name of the performed function.

3.1.16 approval_status

An approval_status is is the ranking which gives an indication of the state of an approval.

EXAMLPE - 'Approved' and 'disapproved' are examples of approval_statuses.

— name.

3.1.16.1 name

The name is the ranking of the approval.

3.1.17 approved_item

A approved_item assigns an approval to a particular product_version.

— items.

3.1.17.1 items

Items are a set of approved_items which identify the versions of particular products to which the approval is assigned.

3.1.18 assembly_component_usage

The assembly component usage relates a constituent to its assembly. The assembly_component_usage entity is a subtype of the product_definition_usage entity that establishes a relationship between product_definitions within one of the following three product structures:

— bill—of—material (BOM) structure;

— parts list structure;

— promissory use structure.

The `assembly_component_usage` entity has four subtypes:

— The `quantified_assembly_component_usage`;

— The `next_assembly_usage_occurrence`;

— The `specified_higher_usage_occurrence`;

— `promissory_usage_occurrence`.

The `quantified_assembly_component_usage` represents the relationship between a constituent and an assembly where, for discrete constituents, several occurrences of the constituent are represented by the single constituent and a quantity representing the number of occurrences of it. The quantity represents a unit of measure other than a unitless number for non—discrete constituents. The `next_assembly_usage_occurrence` represents a relationship between a component and its immediate assembly in a product structure. The `specified_higher_usage_occurrence` shall be used to represent the explicit relationship between a descendent component and any ancestor higher level assembly. The `promissory_usage_occurrence` shall be used to represent intended relationships between a lower—level constituent and a higher level assembly, when intermediate constituents and their relationships are yet undetermined.

In a BOM graph structure, `product_definition` entities represent nodes and `next_assembly_usage_occurrence` or `quantified_assembly_component_usage` entities represent links.

In a parts list tree structure, a `product_definition` entity represents the root node. `Next_assembly_usage_occurrence` entities represent nodes at each intermediate level of the structure. The `specified_higher_usage_occurrence` entities enable links to higher levels of the structure.

In a promissory use graph structure, `product_definition` entities represent nodes, and `promissory_usage_occurrence` entities represent links between the nodes.

— `reference_designator`;

— `product_definition_relationship.relying_product_definition`;

— `product_definition_relationship.related_product_definition`.

3.1.18.1 reference_designator

The `reference_designator` is the identifier for the `assembly_component_usage`, in addition to the `id` attribute inherited from the `product_definition_usage`.

NOTE — The `reference_designator` attribute may be constrained to be unique by an application protocol.

3.1.18.2 product_definition_relationship.relying_product_definition

The `product_definition_relationship.relying_product_definition` is an assembly for which the `related_product_definition` is its constituent.

3.1.18.3 product_definition_relationship.related_product_definition

The `product_definition_relationship.related_product_definition` is a constituent for which the `relying_product_definition` is its parent assembly.

3.1.19 assembly_component_usage_substitute

The `assembly_component_usage_substitute` specifies that one constituent can be used as a substitute for another within a given assembly context.

The instance of the substitute constituent does not require the same spatial relationship or the same quantity. A substitute constituent does not require equivalent form, fit, and function of the constituent for which it is a substitute.

This entity defines one-way substitution only. Within a given context, if A is specified as a substitute for B, B is not assumed to be a substitute for A, unless explicitly stated so in another instance of the entity.

The `assembly_component_usage_substitute` entity establishes an exclusive relationship between the referenced and substitute constituents.

The `assembly_component_usage_substitute` entity may be used to eliminate the re-identification of all higher level assemblies when a new version of a lower level constituent is created.

— base;

— substitute.

Formal Propositions:

UR1: The combination of the base and substitute attributes shall be unique.

WR1: The value of the `relating_product_definition` attribute of both the base and the substitute attributes shall be the same; i.e., they should refer to the same assembly `product_definition`.

WR2: The base and substitute attributes shall not be the same.

3.1.19.1 base

The base is an `assembly_component_usage` for which the substitute may be used.

3.1.19.1 substitute

The substitute is an `assembly_component_usage` which may be used for the base.

3.1.20 cage

The cage is a code used to uniquely identifies a commercial or government entity and/or enterprise.

— `cage_code`.

3.1.20.1 cage_code

The `cage_code` is the unique and alternate identifier of an organization.

3.1.21 calendar_date

A `calendar_date` is a date which is identified by a day in a month of a year.

— `day_component`;

— `month_component`.

3.1.21.1 day_component

The day_component is the day element of the date.

3.1.21.2 month_component

The month_component is the month element of the date.

3.1.22 classified_item

A classified_item applies security_classification to a particular product_version.

— items.

3.1.22.1 items

Items are a set of classified_items which identify the versions of particular products to which the security_classification_is assigned.

3.1.23 concurrent_action_method

A concurrent_action_method is a process_action_method_relationship where individual action_methods are complete when the collection of action_methods is complete.

The concurrent_action_method may be used to define either a peer relationship or a parent to child relationship between two action_methods. For a parent to child relationship, the parent is defined as the

related action_method. For a peer relationship, the distinction between related and relating are not significant.

Informal propositions:

IP1: The individual action_methods in this collection shall be completed during completion of the longest action_method in the collection.

3.1.24 configuration_design

The configuration design relates a configuration controlled item and a product design intended to implement that item. Thus, the configuration_design entity shall represent the association of a configuration_item with a product_version to specify that the corresponding design is for the specific configuration_item.

NOTE - organizations establish this association before any actual units are planned and before any details of the design have been established.

— configuration;

— design.

Formal propositions:

UR1: The combination of the value of the configuration attribute and the value of the design attribute shall be unique.

3.1.24.1 configuration

A configuration_item which specifies a product_version as a candidate for manufacturing actual units

associated with the configuration_item.

3.1.24.2 design

A product_version representing a design which is a candidate for use in manufacturing actual units associated with the configuration attribute.

3.1.25 configuration_item

A configuration_item is used to manage the composition of constituents for actual units of manufacture.

All configuration management within an organization is done using these configuration_items.

Configuration management is the identification of a product_version that realizes the configuration_item.

The product that is planned for manufacture is referred to as the configuration_item. It is usually visible to customers of the organization that does the configuration management. A configuration_item may be an entire product_concept or some portion thereof.

A configuration_item can be established prior to the existence of a corresponding product_version.

The association between a configuration_item and a corresponding product_version is established using a configuration_design.

A configuration_item is associated with a single product_concept.

An organization determines which products are to be under its configuration management control. These products become the configuration items of the organization. These are high level functional elements which act as the focal points for managing the effectivity of constituent lower level parts and assemblies.

— item_concept;

— purpose.

Formal propositions:

UR1: The value of the identification attribute shall be unique.

3.1.25.1 item_concept

A product_concept associated with the configuration_item.

3.1.25.2 purpose

A descriptive label providing a reason to create the item_concept.

3.1.26 context_dependent_unit

A context_dependent_unit is a unit which is not related to the SI system.

EXAMPLE — The number of parts in an assembly is a physical quantity measures in units that may be called "parts" but which cannot be related to an SI unit.

— name.

3.1.26 name

The word, or group of words, by which the context_dependent_unit is referred to.

3.1.27 contract

A contract is a binding agreement.

NOTE — Contracts may be enforceable by law

— name;

— purpose;

— kind.

3.1.27.1 name

The word, or group of words, by which the contract is referred to.

3.1.27.2 purpose

An informal description of the reasons for the contract.

3.1.27.3 kind

The contract's type.

3.1.28 contract_assignment

A `contract_assignment` is an association of a contract with product data.

— `assigned_contract`;

— `product`.

3.1.28.1 assigned_contract

The contract which is to be associated with the product data.

3.1.28.2 product

The product data which is to be associated with the contract.

3.1.29 conversion_based_unit

A `conversion_based_unit` is a unit that is defined on a `measure_with_unit`.

EXAMPLE - An inch is a `converted_unit`. It is from the Imperial system, its name is "inch" and it can be related to the SI unit, millimetre, through a `measure_with_unit` whose value is 25.4 millimetre. A foot is also a `converted_unit`. It is from the Imperial system, its name is "foot" and it can be related to an SI unit, millimetre, either directly or through the unit called "inch".

— name;

— conversion_factor.

3.1.29.1 name

The word, or group of words, by which the conversion_based_unit is referred to.

3.1.29.2 conversion_factor

The physical quantity from which the converted_unit is derived.

3.1.30 coordinated_universal_time_offset

A coordinated_universal_time_offset is used to relate a time to coordinated universal time by an offset (specified in hours and minutes) and a direction.

— hour_offset;

— minute_offset;

— sense.

3.1.30.1 hour_offset

The number of hours by which a time is offset from coordinated universal time.

3.1.30.2 minute_offset

The number of minutes by which a time is offset from coordinated universal time.

3.1.30.3 sense

The direction of the offset.

3.1.31 correspondence

A type of document which is used for communication between parties.

3.1.32 data_template

A type of product which defines in a skeleton manner, the makeup and format of a technical report, document, input/output screen or any set of desired information.

3.1.33 date

A date is the identification of a moment in time occurring between midnight of one day and midnight of the day following.

— year_component.

3.1.33.1 year_component

The year in which the date occurs.

3.1.34 date_and_time

A `date_and_time` is a moment of time on a particular day.

— `date_component`;

— `time_component`.

3.1.34.1 date_component

The date element of the date time combination.

3.1.34.2 time_component

The time element of the date time combination.

3.1.35 date_time_select

A `time_date_select` type allows a date and/or `local_time` to be referenced.

3.1.36 dated_effectivity

The dated effectivity specifies that a `product_definition_usage` is effective for a series of actual units produced during a given time period.

— `effectivity_start_date`;

— `effectivity_end_date`.

3.1.36.1 effectivity_start_date

The date and time at which the `product_definition_usage` identified by the `design_usage` attribute becomes effective.

3.1.36.2 effectivity_end_date

The date and time at which the `product_definition_usage` identified by the `design_usage` attribute is no longer effective. If no value is given the end date for the effectivity is not yet determined.

3.1.37 day_in_month_number

A `day_in_month_number` is the position of the specified day in a month.

3.1.38 day_in_week_number

A `day_in_week_number` is the value of the day as defined in ISO 8601 (clause 5.2.3).

NOTE - Monday is day number 1, Tuesday is day number 2, Wednesday is day number 3, Thursday is day number 4, Friday is day number 5, Saturday is day number 6, and Sunday is day number 7.

Formal propositions:

WR1: the value of the integer shall be between 0 and 7.

3.1.39 day_in_year_number

A day_in_year_number is the position of the specified day in a year.

EXAMPLE - The 27th day of March is day 86 in years that are not leap years and day 87 in leap years.

3.1.40 derived_unit

A derived_unit is an expression of units.

— elements.

Formal propositions:

WR1: there shall be either more than one member in the elements set or the value of the exponent of the single element of the elements set shall not be equal to one.

3.1.40.1 elements

The group of units and their exponents that define the `derived_unit`.

3.1.41 `derived_unit_element`

A `derived_unit_element` is one of the unit quantities which makes up a `derived_unit`.

EXAMPLE - Newtons per square millimetre is a derived unit. It has two elements, Newton whose exponent has a value of 1 and millimeter whose exponent is -2 .

— `unit`;

— `exponent`.

3.1.41.1 `unit`

The fixed quantity which is used as the mathematical factor.

3.1.41.2 `exponent`

The power that is applied to the unit attribute.

3.1.42 `dimensional_exponents`

The dimensionality of any quantity can be expressed as a product of powers of the dimensions of base quantities. The dimensional_exponents entity defines the powers of the dimensions of the base quantities. All the physical quantities are founded on seven base quantities.

NOTE - Length, mass, time, electric current, thermodynamic temperature, amount of substance, and luminous intensity are the seven base quantities.

EXAMPLE - A length of 2 millimetres has a length exponent of 1. The remaining exponents are equal to 0. A velocity of 2 millimetres per second has a length exponent of 1 and a time exponent of -1 . The remaining exponents are equal to 0.

— length_exponent;

— mass_exponent;

— time_exponent;

— electric_current_exponent;

— thermodynamic_temperature_exponent;

— amount_of_substance_exponent;

— luminous_intensity_exponent.

3.1.42.1 length_exponent

The power of the length base quantity.

3.1.42.2 mass_exponent

The power of the mass base quantity.

3.1.42.3 time_exponent

The power of the time base quantity.

3.1.42.4 electric_current_exponent

The power of the electric current base quantity.

3.1.42.5 thermodynamic_temperature_exponent

The power of the thermodynamic temperature base quantity.

3.1.42.6 amount_of_substance_exponent

The power of the amount of substance base quantity.

3.1.42.7 luminous_intensity_exponent

The power of the luminous intensity base quantity.

3.1.43 discrepant_product

Identifies a product_version that fails to satisfy design nominal criteria.

— failure_rate.

3.1.43.1 failure_rate

The failure_rate is the number of failures a product has failed to operate correctly.

3.1.44 document

A document is an unambiguous reference to a formal standard or document. A document is a type of product.

— kind;

— size.

3.1.44.1 kind

The sort of data that the document describes.

3.1.44.2 size

The size is the relative measurement of the document.

EXAMPLE - 12,345 bytes or 8 pages are examples of a size of a document.

3.1.45 document_reference

A document_reference is an association of a document with product data.

— assigned_document.

3.1.45.1 assigned_document

The document which is to be associated with the product data.

3.1.46 document_type

A document_type is the sort of data that the formal standards or documents are being used to describe in a particular context..

EXAMPLE - 'Material', surface finish', and 'heat treatment process' are all pieces of data that can be described implicitly, by reference to other documents (such as DIN documents), rather than explicitly every time they are used.

— product_data_type.

3.1.46.1 product_data_type

The product_data_type is the name of the sort of data that the document is being used to describe.

3.1.47 drawing

A drawing is a type of document which graphical represents a product.

3.1.48 enhancement_product

An enhancement_product is the identification of a need for new or improved product functionality.

3.1.49 enterprise

A type of organization that identifies a supplier/manufacturer/consumer of a product_version (in-house or external).

3.1.50 file_folder

The association of a product_version to a file or folder.

— representative_product;

— file_type.

3.1.50.1 representative_product

The representative_product is the product which is represented by the file or folder.

3.1.50.2 file_type

The file_type defines whether the file_folder instance is a file or a folder.

3.1.51 group

The group is a collector of products.

— group_name.

3.1.51.1 group_name

The word, or group of words, by which the group is referred to.

3.1.52 group_relationship

A group_relationship is an association between two group instances.

— relating_group;

— related_group;

— group_relationship_name;

— description.

3.1.52.1 relating_group

The relating_group is one of the related groups.

3.1.52.2 related_group

The related_group is the other related group.

3.1.52.3 group_relationship_name

The word, or group of words, by which the group_relationship is referred to.

3.1.52.4 description

The description is text that relates the nature of the group_relationship.

3.1.53 hardware_software

A hardware_software is a type of system. It defines a physical implementation of a computer system architecture.

3.1.54 hour_in_day

A hour_in_day is the hour element of a specified time on a 24 hour clock.

EXAMPLE - The hour of 3 o'clock in the afternoon is 15.

Formal propositions:

WR1: the value of the integer shall be between 0 and 23.

3.1.55 identifier

An identifier is an alphanumeric string which allows an individual thing to be identified. It may not provide natural language meaning.

EXAMPLE - A part_number would be an identifier.

3.1.56 label

A label is the term by which something may be referred to. It is a string which represents the human-interpretable name of something and shall have a natural language meaning.

EXAMPLE - "Smith", "Widget Inc.", and "Materials Test Laboratory" are examples of labels.

3.1.57 local_time

A local_time is a moment of occurrence measured by hour, minute, and second. It represents one instant of time on a 24 hour clock.

NOTE - This construct is used to represent a moment in time whereas time measures represent amounts of time.

EXAMPLE - 1500 hours is an instant in time whereas 15 hours is an amount of time.

— hour_component;

— minute_component;

— second_component;

— zone.

3.1.57.1 hour_component

The hour_component is the number of hours.

3.1.57.2 minute_component

The minute_component is the number of minutes.

3.1.57.3 second_component

The second_component is the number of seconds.

3.1.57.4 zone

The zone is the time zone where the local_time was taken.

Formal propositions:

WR1: the seconds attribut shall only exist if the minute attribute exists.

3.1.58 lot_effectivity

This lot effectivity specifies that a product_definition_usage is effective for a specific quantity of actual products. The product lot has an identifier. If the lot contains individual units, these need not be identified.

— effectivity_lot_id;

— effectivity_lot_size.

3.1.58.1 effectivity_lot_id

The effectivity_lot_id is an identifier for the lot of the actual product. The product_definition_usage identified by the inherited design_usage attribute is effective for this lot.

3.1.58.2 effectivity_lot_size

The effectivity_lot_size is a measure of the size of the effective lot.

3.1.59 make_from_usage_option

The make_from_usage_option identifies that a product is made from another product through machining or some other unspecified process.

In situations in which a product is made from another product using a sequence of processes, the intermediate products will be related using the make_from_usage_option entity.

A product to be modified can be an assembly.

NOTE 1 - Generally, the assembly_component_usage differs from the make_from_usage_option in that the constituents of an assembly are used in the assembly without any change.

The make_from_usage_option represents the fact that any actual unit of one design can be manufactured by consuming or modifying an actual unit of another design;

NOTE 2 - Typically the consumed product is referred to as stock or raw material.

The make_from_usage_option_group is used to represent one specific combination of products that can be made from a single product;

NOTE 3 - Typically the single product is referred to as stock or raw material.

The relationship concept represented by the make_from_usage_option applies to designs, represented by product_definitions, rather than the actual units of the designs. A make_from_usage_option relationship is independent of any specific manufactured instances of actual units, and is represented by the attribute references, inherited from the supertype entity, to the relating_product_definition and related_product_definition.

A product_definition may be the relating_product_definition of many make_from_usage_option relationships, and a product_definition may be the related_product_definition of many make_from_usage_option relationships. Further, there may be multiple make_from_usage_option instances referencing the same relating_product_definition and related_product_definition pair of product_definitions.

EXAMPLE 6 - 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 represented by separate instances of product_definitions. Two instances of the make_from_usage_option entity exist, one between the relating_product_definition shaft and the related_product_definition forging, the other between the relating_product_definition shaft and the related_product_definition casting.

— ranking;

— ranking_rationale;

— quantity;

— product_definition_relationship.relatating_product_definition;

— product_definition_relationship.related_product_definition.

Formal propositions:

WR1: The value of ranking shall be positive.

WR2: The value of quantity shall be positive.

3.1.59.1 ranking

The ranking is an integer which ranks the preference for use of the related_product_definition input product_definition among all make_from_usage_option instances with the same value for the inherited relating_product_definition attribute. This is a positive integer value that only has meaning when comparing it with corresponding values for make_from_usage_options sharing the same relating_product_definition product_definition. It is a relative ranking value, not an absolute ranking. A lower value indicates a higher preference for the related_product_definition product_definition, and a higher value indicates a lower preference.

NOTE - Special care is required when assigning these values. If different organizations use different ranges of values, and if populated data sets from these organizations are merged, and multiple make_from_usage_— options from both organizations then exist in the merged file for a single relating_product_definition product_definition, then non—comparable values for this attribute may result.

3.1.59.2 ranking_rationale

The ranking_rationale is the text which describes the rationale used for the ranking.

EXAMPLE 7 - Examples of ranking_rationale are cost and long lead time.

3.1.59.3 quantity

The quantity is the number of physical instances of the relating_product_definition product_definition that can be made from one unit of a related_product_definition product_definition.

3.1.59.4 product_definition_relationship.relati ng_product_definition

A product_definition_relationship.relati ng_product_definition is a product_definition made from the related_product_definition product.

3.15.59.5 product_definition_relationship.related_product_definition

A product_definition_relationship.related_product_definition is a product_definition from which the relating_product_definition is made.

3.1.60 measure_with_unit

A `measure_with_unit` is the specification of a physical quantity.

— `value_component`;

— `unit_component`.

3.1.60.1 `value_component`

The value of the physical quantity when expressed in the specified units.

3.1.60.2 `unit_component`

The unit in which the physical quantity is expressed.

Formal propositions:

WR1: the unit shall be a valid unit for the kind of measure.

3.1.61 `minute_in_hour`

A `minute_in_hour` is the minute element of a specified time.

Formal propositions:

WR1: the value of the integer shall be between 0 and 59.

3.1.62 month_in_year_number

A month_in_year_number is the position of the specified month in a year as defined in ISO 8601 (clause 5.2.1).

NOTE - January is month number 1, February is month number 2, March is month number 3, April is month number 4, May is month number 5, June is month number 6, July is month number 7, August is month number 8, September is month number 9, October is month number 10, November is month number 11, and December is month number 12.

3.1.63 named_unit

A named_unit is a unit quantity associated with the word, or group of words, by which the unit is identified.

— dimensions.

3.1.63.1 dimensions

dimensions: the exponents of the base properties by which the named_unit is defined.

3.1.64 next_assembly_usage_occurrence

The next_assembly_usage_occurrence is the relationship between a child constituent and its immediate parent assembly in a product structure. It represents the use of individual occurrences of constituents. The use of the same constituent may be represented by another distinct next_assembly_usage_occurrence instance for the purpose of assigning a position and orientation for the constituent.

NOTE - An application algorithm can derive an indented parts list for a product by sequentially tracing through a structure of next_assembly_usage_occurrence instances. A similar algorithm can be used to calculate the position and orientation of each occurrence of every constituent relative to its higher level assemblies within a BOM.

— product_definition_relationship.relying_product_definition;

— product_definition_relationship.related_product_definition.

3.1.64.1 product_definition_relationship.relying_product_definition

The product_definition_relationship.relying_product_definition is an assembly for which the related_product_definition is its immediate constituent.

3.1.64.2 product_definition_relationship.related_product_definition

The product_definition_relationship.related_product_definition is a constituent for which the relating_product_definition is its immediate parent assembly.

3.1.65 node_location

A node_location is the network node where a hardware_software system resides.

— system;

— protocol;

— node_address.

3.1.65.1 system

The system is the hardware_software system that reside at the location.

3.1.65.2 protocol

The protocol is convention used to define the address of the hardware_software system.

3.1.65.3 node_address

The node_address is the address location of the hardware_software system.

3.1.66 ordered_action

An ordered_action is the formal notification that authority has been given to perform an action. An action_order is the result of the processing of requested_actions.

NOTE - The distinction between a requested_action and an ordered_action is the level of authority that is associated with it. Anyone can submit a requested_action whereas only authorized people or organizations can submit ordered_actions that are to be acted upon. A request asks for action whereas an order demands action.

— name;

— description;

— analysis;

— comment;

— requests.

3.1.66.1 name

A name is the word, or group of words, by which the `ordered_action` is referred to.

3.1.66.2 description

The description is the text that relates the nature of the `ordered_action`.

3.1.66.3 analysis

The analysis is an informal description of the results of the analysis that was carried out on the elements of the requests set.

EXAMPLE - The fact that two different requests are asking for the same effect could be recorded in this attribute.

3.1.66.4 comment

The comment is an informal description of any other pertinent information.

3.1.66.5 requests

The requests are the `requested_action` that this `ordered_action` relates to.

3.1.67 ordinal_date

An ordinal_date is a date which is identified by a day of a year.

— day_component.

Formal propositions:

WR1: the day_component shall be between 1 and 365 if the year_component is not a leap year; otherwise the day_component shall be between 1 and 366.

3.1.67.1 day_component

The day_component is the day element of the date.

3.1.68 organization

An organization is an administrative structure.

— cage_code.

3.1.68.1 cage_code

The cage_code is the unique and alternate identifier of an organization.

3.1.69 organizational_address

A `organizational_address` is an address where organizations are located.

— `organizations`.

3.1.69.1 organizations

The `organizations` are the organizations located at the address.

3.1.70 organizational_project

An `organizational_project` is project for which one or more organizations are responsible.

— `name`;

— `description`;

— `responsible_organization`.

3.1.70.1 name

The `name` is the word, or group of words, by which the `organizational_project` is referred to.

3.1.70.2 description

The description is the text that relates the nature of the organizational_project.

3.1.70.3 responsible_organization

The responsible_organization is the organizations which are responsible for the project.

3.1.71 part

A part is a product that is intended to be produced or employed in a production process. A part is the type of product that is a discrete product of the organization.

— part_type;

— part_function_type;

— part_configuration_identifier.

3.1.71.1 part_type

The part_type is the further classification of a part.

3.1.71.2 part_function_type

The part_functon_type is the further functional classification of a part.

3.1.71.3 part_configuration_identifier

The part_configuration_identifier is the identification of the configuration of the part.

3.1.72 pca_component_usage

A `pca_component_usage` is a type of an `assembly_component_usage`.

3.1.73 person

A person is an individual human being.

— `last_name`;

— `first_name`;

— `middle_name`;

— `prefix_titles`;

— `suffix_titles`.

Formal propositions:

WR1: either the `last_name` or the `first_name` shall be defined.

3.1.73.1 last_name

The `last_name` is the person's surname.

3.1.73.2 first_name

The `first_name` is the first element of the person's list of forenames.

3.1.73.3 middle_name

The `middle_name` is the person's other forenames, if there are any.

3.1.73.4 prefix_titles

The `prefix_titles` is the word, or group of words, which specify the person's social and/or professional standing and appear before his/her names.

3.1.73.5 suffix_titles

The `suffix_titles` is the word, or group of words, which specify the person's social and/or professional standing and appear after his/her names.

3.1.74 person_and_organization

A `person_and_organization` is a person in an organization.

— person;

— organization;

— person_organization_role.

3.1.74.1 person

The person is the person who is related to the organization.

3.1.74.2 organization

The organization is the organization to which the person is related.

3.1.74.3 person_organization_role

The person_organization_role is the capacity or position a person plays in the organization to which the person is related.

3.1.75 person_organization_select

The person_organization_select type allows a person and/or organization to be referenced.

3.1.76 personal_address

A personal_address is an address where a person resides.

— people.

3.1.76.1 people

The people are the people who reside at the address.

3.1.77 physical_unit

A uniquely identifiable physical manifestation of a product_version design. A tracked instance of a product_version (that is, a serialized unit or lot).

— configuration.

3.1.77.1 configuration

The configuration is the configuration_design which is associated to a physical instantiation of a product_version.

3.1.78 planned_effectivity

The planned effectivity defines common effectivity attributes for items under configuration control. The planned_effectivity entity is used by an organization to specify effectivity of product_definition_usages.

EXAMPLE 13 - A user may want to specify that certain product_definition_usages are to be effective for a configuration_item. A 200 HP engine is to be effective starting on a certain date. This information is captured prior to any production plans exist for the 200 HP engine in a planned_effectivity entity.

Configuration management is the association of the appropriate versions of a product to build a configuration_item. This association is referred to as planned_effectivity.

There are three ways to apply planned_effectivity. They are:

a) serial_numbered_effectivity, where the planned_effectivity is based on serial numbered instances of manufactured products.

b) dated_effectivity, where the planned_effectivity is based on dates when instances of the product are manufactured.

c) lot_effectivity, where the planned_effectivity is based on instances of lots of products manufactured.

The subtypes of this entity represent different situations in which the specified design_usage is effective for actual units of a configuration_item.

— configuration;

— design_usage;

— identification.

Formal propositions:

UR1: The combination of the value of the configuration attribute, the value of the design_usage attribute, and the value of the identification attribute shall be unique.

WR1: The design_usage shall refer to a constituent of the product_version referenced by the configuration_design.

3.1.78.1 configuration

The configuration is a configuration_design whose product_version is contained in the set of product_definition_usages that constitute the configuration_item of the configuration_design.

3.1.78.2 design_usage

A design_usage is a product_definition_usage instance which the planned_effectivity entity specifies as being effective.

3.1.78.3 identification

The identification is an identifier for the planned_effectivity.

3.1.79 procedure

A procedure is a type of document that describes procedures to be followed.

3.1.80 process_action_method_relationship

A process_action_method_relationship is an action_method_relationship that is specified as part of a process. The process_action_method_relationship establishes a collection of action_methods.

The process_action_method_relationship may be used to define either a peer relationship or a parent to child relationship between two action_methods. For a parent to child relationship, the parent is defined as the related action_method.

3.1.81 product

A product is a physically realizable object that is produced by a natural process or manufacture.

EXAMPLE - Production, construction, manufacture, and fabrication are all examples of processes.

EXAMPLE - The ball-point pen, its cap, and the assembly of the cap and the ball-point pen are all physically realizable objects.

— id;

— name;

— description;

— frame_of_reference.

Formal propositions:

UR1: every product's identification shall be unique.

3.1.81.1 id

The id is the identification of the product.

EXAMPLE — Part numbers and stock item numbers are examples of product identifiers.

3.1.81.2 name

The name is the word, or group of words, by which the product is referred to.

EXAMPLE — ``Ball—point pen", ``cap", and ``nib" are examples of product.names.

3.1.81.3 description

The description is the text that relates the nature of the product.

3.1.81.4 frame_of_reference

The frame_of_reference is the context within which the product was defined.

3.1.82 product_anomaly

The product_anomaly is the identification of a nonconformance or a deviation from design nominal conditions for a product.

— anomaly_cause;

— anomaly_type;

— detection_method;

— product_anomaly_description;

— product_anomaly_identifier.

3.1.82.1 anomaly_cause

An anomaly_cause specifies a narrative identifying the reason why the nonconformance occurred.

3.1.82.2 anomaly_type

An anomaly_type specifies the type of product_anomaly as being either an product_issue, product_concern, or an product_flaw.

3.1.82.3 detection_method

The detection_method specifies the procedure that a system, sub-system or assembly was evaluated and determined to be nonconforming.

3.1.82.4 product_anomaly_description

The description specifies a narrative account describing the nonconformance.

3.1.82.5 product_anomaly_identifier

A product_anomaly_identifier specifies the unique identification of a product_issue, product_concern, or a product_flaw that is associated with a product.

3.1.83 product_anomaly_disposition

The product_anomaly_disposition is the actual resolution applied to a product_anomaly.

— anomalized_product;

— disposition_actions.

3.1.83.1 anomalized_product

The anomalized_product specifies the identification of a product_anomaly.

3.1.83.2 disposition_actions

The disposition_actions specifies the performance of an action_execution for answering the disposition of a product to the satisfaction of the controlling interest.

3.1.84 product_change

An product_change is the creation of a new product that results from an anomaly or concern about a baseline product.

NOTE - This entity identifies the new product as well as the baseline product that the new version was based upon, due to an anomaly or concern as well as the authorization that accounts for the product_change.

— baseline_product;

— baseline_product_disposition;

— reasons;

— resulting_product.

3.1.84.1 baseline_product

The baseline_product specifies the product that undergoes a change process and results in a new product.

3.1.84.2 baseline_product_disposition

The baseline_product_disposition specifies the resolution that is being applied to a baseline_product to satisfy an anomaly.

3.1.84.3 reasons

The reason specifies the rationale of why a product_change took place.

3.1.84.4 resulting_product

The resulting_product specifies the product that results from a change process.

3.1.85 product_classification

A product_classification is an association of security_classification with product data.

— items.

3.1.85.1 items

The items is the product data which is assigned a security_classification.

3.1.86 product_concept

The product_concept is the idea of a product as defined by customer needs. The product_concept and its features may be identified as configuration items to control their manufacture. A product concept may exist before a product has been defined. A product concept identifies a selection of product features or capabilities.

A product concept identifies a deliverable product as perceived by the customer. A product concept is often used to identify a selection of product features or capabilities.

A product concept may be composed of several configuration items.

Note - A product_concept will often correspond to the highest level item(s) manufactured by an organization for a customer. It may be characterized by a set of product features identified by the customers or derived from customers' needs. The definition of product concepts is often driven by marketing.

EXAMPLE - If an organization manufactures cars and engines for cars, the cars will be represented by product_concept instances. If another organization manufactures engines for cars, then the engines will be represented as product_concept in that organization.

— product_concept_context.

3.1.86.1 product_concept_context

The product_concept_context is a market context in which the product_concept is defined.

3.1.87 product_concern

The product_concern is a type of product_anomaly that expresses a concern for a particular product.

3.1.88 product_definition

A product definition is the identification of a characterization of a product_version in a particular application context.

NOTE - A product_definition is characterized by properties which refer to it.

EXAMPLE - A product's physical design may be one product_definition whilst the functional design of the same product may be a different product_definition. Both product_definitions would be related to the same product_version but would be used in different application contexts.

— description;

— version;

— frame_of_reference.

3.1.88.1 description

The description is the text that relates the nature of the product_definition.

3.1.88.2 version

The version is the product_version to which the product_definition relates.

3.1.88.3 frame_of_reference

The frame_of_reference is the product_definition_context in which the product_definition or product_definition data is used.

3.1.89 product_definition_relationship

A product_definition_relationship is an association between two product_definitions. An association may exist between product_definitions that relate to different products or between different definitions of the same product.

EXAMPLE - The relationships within a bill of materials structure are examples of product_definition_relationships that associate different products. The relationship between a sketch and a detailed design is an example of a product_definitionrelationship that associates different definitions of a single product.

A single product_definition may be used more than once within the description of a product.

NOTE - The same component could be used more than once in the same assembly. Each usage of the component would be specified as an instance of the product_definition_relationship entity.

— id;

— name;

— description;

— relating_product_definition;

— related_product_definition.

3.1.89.1 id

The id is the identification of the product_definition_relationship .

3.1.89.2 name

The name is the word, or group of words, by which the product_definition_relationship is referred by.

3.1.89.3 description

The description is the text that relates the nature of the product_definition_relationship.

3.1.89.4 relating_product_definition

The relating_product_definition is one of the product_definitions which is a part of the relationship.

EXAMPLE - If the product_definition_relationship is an assembly component relationship the relating_product_definition may be the assembly.

3.1.89.5 related_product_definition

The related_product_definition is the other product_definition which is a part of the relationship.

EXAMPLE - In an assembly the related_product_definition may be the product_definition that is an element of the assembly.

3.1.90 product_definition_usage

The product_definition_usage is a subtype of the product_definition_relationship entity for use within the context of product structure definition and management. This subtype adds meaning to the two attributes: relating_product_definition, related_product_definition.

The subtypes of this entity represent different kinds of product structure relationships between the referenced pair of product_definitions. One subtype, make_from_usage_option, represents the relationship between a product and another product, where one product is made from the other. The other subtype, assembly_component_usage, represents the relationship between an assembly and one of its constituents.

- product_definition_relationship.id;

- product_definition_relationship.relatating_product_definition;

- product_definition_relationship.related_product_definition.

Formal propositions:

UR1: The inherited id, relating_product_definition and related_product_definition, uniquely identifies an instance of product_definition_usage.

WR1: The graph structure of product_definition nodes and product_definition_usage links shall be acyclic. Each product_definition shall not be a descendant of itself in the graph structure.

3.1.90.1 product_definition_relationship.id

The `product_definition_relationship.id` is an identifier for a usage of a `product_definition`. It is used to distinguish between two instances of `product_definition_usage` where the pair of `product_definition` attributes are the same

EXAMPLE 5 - If four identical bolts are used to attach two plates, there may be a need to identify one specific bolt for some purpose. It needs to be torqued to a greater degree than the rest. The `id` attribute then is used to identify this specific bolt's requirement, even though all four bolt `product_definition_usages` will have the same attribute pair of `product_definitions`.

3.1.90.2 `product_definition_relationship.relying_product_definition`

The `product_definition_relationship.relying_product_definition` is a `product_definition` that is made from or serves as the assembly for the `related_product_definition`.

3.1.90.3 `product_definition_relationship.related_product_definition`

The `product_definition_relationship.related_product_definition` is a `product_definition` from which the `relying_product_definition` is made or which is the component in the `relying_product_definition` assembly.

3.1.91 `product_flaw`

The description of a nonconformance or flaw in, on or about a `product_version`.

— `product_flaw_type`.

3.1.91.1 `product_flaw_type`

The `product_flaw_type` is the further classification of the type of flaw that is associated to a product.

3.1.92 product_flow_classification

The `product_flow_classification` is the specification of one or more flaw categories that a `product_flow` belongs to.

— `classified_product`;

— `flaw_class_identifier`.

3.1.92.1 classified_product

The `classified_product` specifies the `product_flow` being classified.

3.1.92.2 flaw_class_identifier

The `flaw_class_identifier` specifies the classification or type of a `product_flow` as inherent, natural failure, or caused by another `product_version`.

3.1.93 product_issue

The identification of special issues or concerns that are not flaws but may require further action.

3.1.94 product_process_step

The `product_process_step` is the association of a product with the `process_step` to be performed on or by the product.

— products.

3.1.94.1 products

The product which is to be associated with the process_step.

3.1.95 product_requiring_change

Identifies a product_version that does not satisfy a particular requirement. A product_requiring_change is a product_version that is changed because of the identification of a flaw or need for capability enhancement .

— requiring_change_product;

— anomalized_products;

— product_change_requirement_type.

3.1.95.1 requiring_change_product

The requiring_change_product specifies the action_execution that will satisfy the change requirement.

3.1.95.2 anomalized_products

The anomalized_products specifies the product_anomaly that that will be addressed by the change.

3.1.95.3 product_change_requirement_type

The `product_change_requirement_type` specifies whether the reason for a product change is either a discrepancy or enhancement.

3.1.96 product_responsibility

The `product_responsibility` specifies the association of a `organizational_project` to a product.

— `project`;

— `product`.

3.1.96.1 project

The `project` is the `organizational_project` that is associated to the product.

3.1.96.2 product

The `product` specifies the product that is associated to an `organizational_project`.

3.1.97 product_state

The `product_state` specifies the lifecycle state of a product.

— `state_name`;

— product;

— action_transition.

3.1.97.1 state_name

The state_name is the word, or group of words, by which the product_state is referred to.

3.1.97.2 product

The product specifies the product_version that has the associated lifecycle state.

3.1.97.3 action_transition

The action_transition specifies the action_execution that transitioned the product_version to a given lifecycle state.

3.1.98 product_version

A product_version is an identified version of a product that differs from other versions in some significant way. However, it is insufficiently different to be regarded as a different product.

NOTE - At any given time there may be multiple active and obsolete versions for the same product.

— version_id;

— description;

— of_product.

Formal propositions:

UR1: the version_id of each product_version that is related a single product (through their ofproduct attributes) shall be unique within the collection of product_versions which are related to that product.

3.1.98.1 version_id

The version_id is the unique identification of the product_version in the context of the product that it relates to.

EXAMPLE — Part version number is an example of a product_version identifier.

3.1.98.2 description

The description is the text the relates the nature of the product_version.

NOTE - The descriptions of different versions of a single product could identify differences in the purpose and function of each version.

3.1.98.3 of_product

The of_product is the product that the product_version is a version of.

NOTE - A product is associated with one or more product_versions through the inverse of this relationship.

3.1.99 product_version_group

A product_version_group is the association of a product_version to a group.

— group;

— version.

3.1.99.1 group

The group is the group in which a product belongs to.

3.1.99.2 version

The version is the product_version which belongs to the group.

3.1.100 program

A type of organization denoting a particular organized thrust or development effort.

EXAMPLE - The B-1B Aircraft Program is an example of a program.

3.1.101 promissory_usage_occurrence

The promissory usage occurrence is the intention to use constituent `product_definition` in an assembly `product_definition`. It is used when the product structure is not completely defined. In such a situation, it is still possible to relate an assembly to a constituent to capture the intent that the constituent will be eventually used. The `promissory_usage_occurrence` represents the relationship between a constituent and an ancestor assembly within an overall product structure without any specification of the intermediate assemblies being represented.

— `product_definition_relationship.relatering_product_definition`;

— `product_definition_relationship.related_product_definition`.

3.1.101.1 `product_definition_relationship.relatering_product_definition`

The `product_definition_relationship.relatering_product_definition` is an assembly for which the `related_product_definition` is a constituent, and the details of the product structure are not completely defined.

3.1.101.2 `product_definition_relationship.related_product_definition`

The `product_definition_relationship.related_product_definition` is a constituent for which the `relatering_product_definition` is an assembly, and the details of the product structure are not completely defined.

3.1.102 `publication`

A publication is a type of document that published for distribution.

3.1.103 `quantified_assembly_component_usage`

The `quantified_assembly_component_usage` establishes the relationship between an assembly and one of its constituents, when there is a need to specify the quantity of the child constituent used in the assembly.

NOTE - Generally for production planning or material planning purposes several occurrences of a constituent are lumped together and a quantity is specified to account for the several occurrences. A typical example would be the specifying of an occurrence of a rivet used for joining airplane structures and denoting the number of such rivets used on the entire plane. If each of the occurrences of the rivets used is to be specified, then the `next_assembly_usage_occurrence` entity may be used. As many instances of the `next_assembly_usage_occurrence` as the number of occurrences of the rivets will exist.

— `quantity`;

— `product_definition_relationship.relating_product_definition`;

— `product_definition_relationship.related_product_definition`.

3.1.103.1 quantity

The `quantity` is a measure of how many or how much of the constituent is used in the assembly.

3.1.103.2 `product_definition_relationship.relating_product_definition`

The `product_definition_relationship.relating_product_definition` is an assembly for which the `related_product_definition` is its constituent, and where the quantity of the constituent may be specified.

3.1.103.3 `product_definition_relationship.related_product_definition`

The `product_definition_relationship.related_product_definition` is an assembly for which the `relating_product_definition` is its parent assembly, and where the quantity of the constituent may be

specified.

3.1.104 recommended_support_resource

A support_resource that is recommended/required to assist, accomodate/facilitate, the performance of an action_item such as design, production, training, operation, and/or maintenance.

— recommended_action;

— supporting_resource;

— recommended_role.

3.1.104.1 recommended_action

The recommended_action is the recommendation of an action to be performed on a product_version by a support_resource.

3.1.104.2 supporting_resource

The supporting_resource is the support resource (person or organization) that is recommended to perform the action.

3.1.104.3 recommended_role

The recommended_role is the recommended position or capacity in which the person or organization plays when performing the action.

3.1.105 related_change

A `related_change` is a type of `product_requiring_change` that identifies a `product_requiring_change` due to an anomaly with another `product_requiring_change` .

— `anomalized_product`;

— `related_change_product`.

3.1.105.1 anomalized_product

An `anomalized_product` specifies the identification of a product anomaly that has identified an additional `product_requiring_change`.

3.1.105.2 related_change_product

The `related_change_product` is a product that has been identified as needing to be changed due to the change of another product.

3.1.106 requested_action

A `requested_action` is a formal notification of a desire for action to be taken.

— `id`;

— `version`;

— purpose;

— description.

3.1.106.1 id

The id is the means of identification of the requested_action.

3.1.106.2 version

The version is the identification of the version of the requested_action.

3.1.106.3 purpose

The purpose is an informal description of the reason for the requested_action.

3.1.106.4 description

The description is an informal definition of the requested_action.

3.1.107 reuse_part

A reuse_part is a RASSP part that may be reused for different signal processor designs.

3.1.108 role

A role is the support resource context in which a user performs a given process_step on a product.

— role_name.

3.1.108.1 role_name

The role_name is the nomenclature used to describe the role that a user plays in the performance of a task. A role_name may be considered a user job classification.

EXAMPLE - Examples of role_names are "designer", "manager", and "checker".

3.1.109 second_in_minute

A second_in_minute is the second element of a specified time.

Formal propositions:

WR1: the value of the integer shall be between 0 and 59.

3.1.110 security_classification

A security classification is the level of confidentiality that is required for the purpose of product data protection.

— name;

— purpose;

— security_level.

3.1.110.1 name

The name is the word, or group of words, by which the security_classification is referred to.

3.1.110.2 purpose

The purpose is an informal description of the intent of the security_classification.

3.1.110.3 security_level

The security_level is the category of the security_classification.

3.1.111 security_classification_assignment

A security_classification_assignment is an associaton of a security_classification with product data.

— assigned_security_classification.

3.1.111.1 assigned_security_classification

The assigned_security_classification is the security_classification which is to be associated with the product

data.

3.1.112 security_classification_level

A security_classification_level is a category of security.

EXAMPLE - 'Confidential', 'secret', 'and top secret' are examples of security_classification_levels.

— name.

3.1.112.1 name

The name is the word, or group of words, by which the security_classification_level is spoken of or referred to.

3.1.113 sequential_method

A sequential_method is a serial_action_method where each of the action_methods are completed in a specified order. The sequence is such that a sequential_method with a lower index is completed before those with a higher index.

— sequence_position.

Informal propositions:

IP1: There shall be only one sequential_method with the same sequence_position within a given context or action_method.

NOTE - This means that for any instance of `sequence_method`, there only exists one `sequential_method` with the same `sequence_position`.

3.1.113.1 `sequence_position`

The `sequence_position` is the relative position of the `sequential_method` within the ordered collection of `action_methods`.

3.1.114 `serial_action_method`

A `serial_action_method` is a `process_action_method_relationship` where individual `action_methods` are complete when the collection of `action_methods` is complete. The `action_methods` in the collection must be completed in a manner whereby one `action_method` must be complete before the next `action_method` is initiated.

The `serial_action_method` may be used to define either a peer relationship or a parent to child relationship between two `action_methods`. For a parent to child relationship, the parent is defined as the related `action_method`. For a peer relationship, the distinction between related and relating are defined by the application resource or the application protocol.

NOTES - The sequential ordering of parent to child relationships is specified through `sequential_method`.

EXAMPLE - A peer relationship `serial_action_method` has two `action_methods` that define the process of turning on a light or turning off a light. The existing state of the light is off. The first `action_method` is turning on the light. The first `action_method` must be completed before the second `action_method` is initiated. The second `action_method` is the related `action_method`. The `serial_action_method` specifies the ordered completion of activities that do not overlap during execution.

Informal propositions:

IP1: Individual `action_methods` in a collection shall be completed so that one `action_method` is completed before the next `action_method` is initiated.

3.1.115 serial_concurrent_action_method

A `serial_concurrent_action_method` is a `process_action_method_relationship` where individual `action_methods` shall be complete when the entire collection is complete. These `action_methods` may occur in an overlapping manner until all `action_methods` are completed.

The `serial_concurrent_action_method` may be used to define either a peer relationship or a parent to child relationship between two `action_methods`. For a parent to child relationship, the parent is defined as the `related_action_method`.

Informal propositions:

IP1: The individual `action_methods` in a collection may be completed in a concurrent, serial, or overlapping manner.

3.1.116 serial_numbered_effectivity

This serial numbered effectivity specifies that a `product_definition_usage` is effective for one or more actual units that result from a production planning activity, where each such actual unit has its own individual identifier. These identifiers are used to define a range. It is assumed that these identifiers are assigned during actual manufacturing of a product and have a well defined ordering algorithm.

— `effectivity_start_id`;

— `effectivity_end_id`.

3.1.116.1 effectivity_start_id

The first of one or more actual units to result from a production planning activity. The product_definition_usage identified by the design_usage attribute is effective for these actual units.

3.1.116.2 effectivity_end_id

The ending identifier of a bounded sequence of actual units. If no value is given the range of the serial_numbered_effectivity is open. If the values of the effectivity_start_id and effectivity_end_id are the same, the serial_numbered_effectivity applies to a single actual unit.

3.1.117 si_prefix

An si_prefix is the name of a prefix that may be associated with an si_unit. The definitions of SI prefixes are specified in ISO 1000 (clause 3).

3.1.118 si_unit

An si_unit is the fixed quantity used as a standard in terms of which items are measured as defined by ISO 1000 (clause 2).

— prefix;

— name.

3.1.118.1 prefix

The prefix is the SI prefix.

3.1.118.1 name

The name is the word, or group of words, by which the `si_unit` is referred to.

3.1.119 `si_unit_name`

A `si_unit_name` is the name of an SI unit. The definitions of the names of the SI units are specified in ISO 1000 (clause 2).

3.1.120 `signal_processor_design`

A `signal_processor_design` is a type of configuration item. It is the focus of configuration management for the signal processor design process. A `signal_processor_design` is represented by one or many electronic files and is representative of a part. An object (i.e., part) may only become a `signal_processor_design` when it is used in a design of a signal processor.

3.1.121 `software_application`

A `software_application` is a type of part. It is programming code that may be identified by a part number and is written in a specific software programming language.

3.1.121.1 `software_language`

The `software_language` is the word, or group of words, which identify the programming language which was utilized to produce the `software_application`.

3.1.122 specification

A specification is a type of document that defines a product in detail and/or its requirements.

3.1.123 specified_higher_usage_occurrence

The `specified_higher_usage_occurrence` represents the relationship between a specific use of a constituent with respect to a non-immediate/non-parent ancestor assembly within the product structure; For a general product structure, in order to identify the usage of any constituent within an assembled product, it is necessary to identify the path between the assembled product and the constituent. The `specified_higher_usage_occurrence` entity provides this capability.

The `specified_higher_usage_occurrence` specifies the relationship between a constituent and an assembly where the assembly is not the immediate parent for the constituent.

If a `specified_higher_usage_occurrence` is specified, the entire path between the constituent and the assembly is also identified using successive instances of `specified_higher_usage_occurrence`. Successive instances of `specified_higher_usage_occurrence` identify all the intermediate constituent and assembly relationships that exist between the assembly and its constituent specified by the primary `specified_higher_usage_occurrence`.

The relationship between the constituent and the assembly of the `specified_higher_usage_occurrence` to be specified is captured by the relationship of the inherited attributes `relating_product_definition` and `related_product_definition`.

The two attributes (`upper_usage` and `next_usage`) within the primary instance of the entity `specified_higher_usage_occurrence` will respectively specify the `next_assembly_usage_occurrence` and an `assembly_component_usage` which together will provide the definition of the path from the constituent to the assembly for which the `specified_higher_usage_occurrence` is being specified. To ensure that the `next_assembly_usage_occurrence` and the `assembly_component_usage` together constitute the entire path desired for the `specified_higher_usage_occurrence`, it is essential that the instance of the `related_product_definition` attribute of the `assembly_component_usage` entity referenced by the `upper_usage` be the same as the instance of the `relating_product_definition` attribute of the `next_assembly_usage_occurrence` entity referenced by `next_usage`. The attribute `related_product_definition`

of the next_assembly_usage_occurrence shall be the same instance of the attribute relating_product_definition of the specified_higher_usage_occurrence being specified. The attribute relating_product_definition of the assembly_component_usage entity referenced by the attribute upper_usage shall be the same instance as the attribute relating_product_definition of the specified_higher_usage_occurrence being specified.

If the assembly_component_usage referenced by the attribute upper_usage is not a next_assembly_usage_occurrence it will be a specified_higher_usage_occurrence. This specified_higher_usage_occurrence shall have its attributes upper_usage and next_usage defined as described in the previous paragraph to specify further the path of the primary specified_higher_usage_occurrence. This recursive specification shall continue until the attribute upper_usage references an assembly_component_usage entity that is a next_assembly_usage_occurrence. At this point, the primary specified_higher_usage_occurrence is fully specified both in terms of its constituents/assembly relationship and the entire path between them.

In order to be able to completely specify a specified_higher_usage_occurrence all the necessary assembly_component_usage instances shall have been defined.

The specified_higher_usage_occurrence entity supports the representation of parts list tree structures. Typically, it is used to define portions of parts lists that contain a specific constituent within an assembly for which certain properties are to be associated.

— upper_usage;

— next_usage;

— product_definition_relationship.relatating_product_definition;

— product_definition_relationship.related_product_definition.

Formal propositions:

UR1: The combination of the upper_usage and next_usage attributes shall be unique.

WR1: The instance of `specified_higher_usage_occurrence` shall not be the same as the instance of `upper_usage`.

WR2: The `relating_product_definition` (i.e., assembly) of the `specified_higher_usage_occurrence` shall be the same instance product definition as `relating_product_definition` (i.e., assembly) for the `upper_usage`.

WR3: The `related_product_definition` (i.e., constituent) of the `specified_higher_usage_occurrence` shall be the same instance of product definition as the `related_product_definition` for the `next_usage`.

WR4: The `related_product_definition` (i.e., component) for the `upper_usage` shall be the same instance of product definition as the `relating_product_definition` (i.e., assembly) for the `next_usage`.

WR5: The type of the `upper_usage` attribute cannot be the `promissory_usage_occurrence` type.

3.1.123.1 upper_usage

The `upper_usage` is an `assembly_component_usage` that has the same instance of the attribute `relating_product_definition` as this `specified_higher_usage_occurrence` and the same instance of the attribute `related_product_definition` as the `relating_product_definition` of the `next_assembly_usage_occurrence` referenced by the attribute `next_usage`.

3.1.123.2 next_usage

The `next_usage` is a `next_assembly_usage_occurrence` that has the same instance of the attribute `related_product_definition` as this `specified_higher_usage_occurrence` and the same instance of the product definition referenced by the attribute `relating_product_definition` as the product definition referenced by the attribute `related_product_definition` of the attribute `upper_usage`.

3.1.123.3 product_definition_relationship.relying_product_definition

The `product_definition_relationship.relying_product_definition` is the inherited attribute for the assembly product definition of the `specified_higher_usage_occurrence`.

3.1.123.4 product_definition_relationship.related_product_definition

The product_definition_relationship.related_product_definition is the inherited attribute for the constituent product definition of the specified_higher_usage_occurrence.

3.1.124 specified_item

A specified_item assigns a document to a particular product_version.

— items.

3.1.124.1 items

Items are a set of specified_items which identify the versions of particular products to which the document is assigned.

3.1.125 supplier

A supplier is a type of organization. In the supplier capacity, the organization provides a product, on a given date, to the enterprise.

3.1.125.1 product

The product is the product that is supplied.

3.1.125.2 source_date

The `source_date` is the date in which the product was supplied by the organization.

3.1.126 support_equipment

A device recommended/required to facilitate design, production, training, operation, and/or maintenance of a `product_version`.

— `name`.

3.1.126.1 name

The `name` is the word, or group of words, by which the `support_equipment` is referred to.

3.1.127 support_resource_select

The `support_resource_select` type allows a piece of equipment, person and/or organization to be referenced with respect to their supporting role in an `action_execution`.

The `support_resource_select` recommends or requires the facilitating design, production, training, operation, and/or maintenance of a `product_version`. A `support_resource` may be personnel, `support_equipment`, or organization.

3.1.128 system

A type of `product_version` that is a regularly interacting or interdependent group of products forming a unified whole under the influence of related forces.

3.1.129 system_user

A system_user is a type of person where the person is associated to a hardware_software system.

— system;

— user_id;

— password;

— access_level.

3.1.129.1 system

The system is the hardware_software system that is associated to the person.

3.1.129.2 user_id

The user_id is the identification of the person that is recognizable by the hardware_software system.

3.1.129.3 password

The password is the hardware_software system recognizable confirmation of the user_id.

3.1.129.4 access_level

The `access_level` is an indication of what the person (with a successful login of `user_id` and password) can perform in the `hardware_software` system..

3.1.130 text

A text is an alphanumeric string of characters which is intended to be read and understood by a human being. It is for information purposes only.

3.1.131 unit

A unit is a physical quantity, with a value of one, which is used as a standard in terms of which other quantities are expressed.

3.1.132 week_in_year_number

A `week_in_year_number` is the value of the calendar week as defined in ISO 8601 (clause 3.1.7).

Formal propositions:

WR1: the value of the integer shall be between 1 and 53.

3.1.133 week_of_year_and_day_date

A `week_of_year_and_day_date` is a date which is identified by a day in a week of a year.

— `week_component`;

— `day_component`.

Informal propositions:

`valid_year_and_day`: the combination of the `day_component` and the `week_component` shall be between 1 and 365 if the `year_component` is not a leap year, otherwise the combination of the `day_component` and the `week_component` shall be between 1 and 366.

3.1.133.1 week_component

The `week_component` is the week element of the date.

3.1.133.2 day_component

The `day_component` is the day element of the date.

3.1.134 year_number

A `year_number` is the year as defined by the Gregorian Calendar.

3.2 Enterprise Object Assertions

This subclause specifies the enterprise object assertions for the RASSP Enterprise Data Model. Object assertions specify the relationships between enterprise objects, the cardinality of the relationships, and the rules required for the integrity and validity of the enterprise objects. The enterprise assertions and their definitions are given below.

3.2.1 action to action_method

Each instance of an action defines the method of zero, one, or many action_method instances.

3.2.2 action_assignment to action

Each instance of an action defines the association to zero, one, or many action_assignment instances.

3.2.3 action_execution to ordered_action

Each instance of an ordered_action authorizes zero, one, or many action_execution instances.

3.2.4 action_execution_support_resource to action_execution

Each instance of a action_execution is the executed action for zero, one or many action_execution_support_resource instances.

3.2.5 action_item to product_version

Each instance of an `action_item` defines a set of one or more `product_version` instances.

3.2.6 `action_method` to `requested_action`

Each instance of an `action_method` requests a set of one or more `requested_action` instances.

3.2.7 `action_method_relationship` to `action_method`

Each `action_method_relationship` is the `related_action_method` for zero, one, or many `action_method` instances. Each `action_method_relationship` is the `relating_action_method` for zero, one, or many `action_method` instances.

3.2.8 `action_status` to `action_execution`

Each instance of an `action_execution` has a status defined by a zero, one or more `action_status` instances.

3.2.9 `approval` to `approval_status`

Each instance of `approval_status` is the status for exactly one `approval`.

3.2.10 `approval_assignment` to `approval`

Each instance of approval is assigned to zero, one or many approval instances.

3.2.11 approval_date_time to approval

Each instance of approval shall be referenced by exactly one approval_date_time. This enforces the requirement for every approval to have a date on which the approval obtained its specific status.

3.2.12 approved_item to product_version

Each instance of an approved_item is for a set of one or more product_version instances.

3.2.13 approval_person_organization to approval

Each instance of approval shall have one or more approval_user_organization referencing it. This rule enforces the requirement for an approval to be authorized by one or more people within their organizations.

3.2.14 approval_person_organization to approval_role

Each instance of an approval_role is the role for zero, one or more approval_person_organization instances.

3.2.15 assembly_component_usage_substitute to assembly_component_usage

Each instance of a `assembly_component_usage` may be the base for zero, one, or more `assembly_component_usage_substitute`. Each instance of a `assembly_component_usage` may substitute for zero, one, or more `assembly_component_usage_substitute`.

3.2.16 `classified_item` to `product_version`

Each instance of a `classified_item` classifies a set of one or more `product_version` instances.

3.2.17 `configuration_design` to `configuration_item`

Each instance of a `configuration_item` defines the configuration for zero, one, or many `configuration_design` instances.

3.2.18 `configuration_design` to `product_version`

Each instance of a `product_version` is the design for zero, one, or many `configuration_design` instances.

3.2.19 `configuration_item` to `product_concept`

Each instance of a `product_concept` is the item concept for zero, one, or many `configuration_item` instances.

3.2.20 `contract_assignment` to `contract`

Each instance of a `contract` is assigned to zero, one or many `contract_assignment` instances.

3.2.21 contract_assignment to product_version

Each instance of a product_version is assigned to zero, one or many contract_assignment instances.

3.2.22 conversion_based_unit to measure_with_unit

Each instance of a measure_with_unit defines the conversion factor of zero, one or many conversion_based_unit instances.

3.2.23 date_and_time to date

Each instance of a date is the component for zero, one, or many date_and_time instances.

3.2.24 date_and_time to local_time

Each instance of a local_time is the component for zero, one, or many date_and_time.

3.2.25 dated_effectivity to date_and_time

Each instance of a date_and_time defines the effectivity_start_date for zero, one, or more dated_effectivity instances. Each instance of a date_and_time may define the effectivity_end_date for zero, one, or more dated_effectivity instances.

3.2.26 derived_unit to derived_unit_element

Each instance of derived unit requires a set of one or more derived_unit_elements.

3.2.27 derived_unit_element to named_unit

Each instance of a named_unit is the unit for zero, one, or many derived_unit_element instances.

3.2.28 document to document_type

Each instance of a document_type is the kind for zero, one, or many document instances.

3.2.29 document_reference to document

Each instance of a document is assigned to zero, one, or many document_reference instances.

3.2.30 file_folder to product_version

Each product_version is electronically represented by zero, one or many associated file_folder instances.

3.2.31 group_relationship to group

Each group_relationship is the related_group for zero, one, or many group instances. Each group_relationship is the relating_group for zero, one, or many group instances.

3.2.32 lot_effectivity to measure_with_unit

Each instance of a measure_with_unit defines the lot size of zero, one, or many lot_effectivity instances.

3.2.33 make_from_usage_option to measure_with_unit

Each instance of a measure_with_unit defines the quantity of zero, one, or many make_from_usage_option instances.

3.2.34 named_unit to dimensional_exponents

Each instance of a dimensional_exponents defines the dimensions of zero, one or more named_unit instances.

3.2.35 node_location to hardware_software

Each instance of a hardware_software is defined by the location of zero, one or more node_location instances.

3.2.36 ordered_action to requested_action

Each instance of an ordered_action authorizes a set of one or more requested_action instances.

3.2.37 organization to cage

Each instance of a cage defines the cage code for zero, one or more organization instances.

3.2.38 organizational_address to organization

Each instance of an organizational_address defines the location for a set of one or more organization instances. Each instance of a organization is located at zero, one, or many organizational_address instances.

3.2.39 organizational_project to organization

Each instance of a organizational_project is the responsibility of a set of one or many organization instances.

3.2.40 person_and_organization to organization

Each instance of an organization defines zero, one, or many person_and_organization instances.

3.2.41 person_and_organization to person

Each instance of a person defines zero, one, or many person_and_organization instances.

3.2.42 personal_address to person

Each instance of an personal_address defines the location for a set of one or more person instances. Each instance of a person is located at zero, one, or many personal_address instances.

3.2.43 physical_unit to configuration_design

Each instance of a configuration_design defines the configuration for zero, one, or many physical_unit instances.

3.2.44 planned_effectivity to configuration_design

Each instance of a configuration_design defines the configuration for zero, one, or many planned_effectivity instances.

3.2.45 planned_effectivity to product_definition_usage

Each instance of a product_definition_usage defines the design_usage for zero, one, or many planned_effectivity instances.

3.2.46 product_anomaly_disposition to action_execution

Each instance of a product_anomaly_disposition is dispositioned by a set of one or more action_execution instances.

3.2.47 product_anomaly_disposition to product_anomaly

Each instance of a product_anomaly is resolved by zero, one or many product_anomaly_disposition instances.

3.2.48 product_change to product_anomaly_disposition

Each product_anomaly_disposition defines the baseline product disposition for zero, one or many product_change instances.

3.2.49 product_change to product_requiring_change

Each instance of a product_requiring_change defines the baseline product for zero, one, or many product_change instances.

3.2.50 product_change to product_version

Each instance of a product_version defines the resulting product for zero, one, or many product_change instances.

3.2.51 product_classification to product

Each instance of product classification requires a set of one or more products.

3.2.52 product_definition to product_version

Each product_version is characterized by zero, one, or many product_definition instances.

3.2.53 product_definition_relationship to product_definition

Each product_definition_relationship is the related_product_definition for zero, one, or many product_definition instances. Each product_definition_relationship is the relating_product_definition for zero, one, or many product_definition instances.

3.2.54 product_flow_classification to product_flow

Each product_flow is classified by zero, one, or many product_flow_classification instances.

3.2.55 product_process_step to product

Each instance of product process step requires a set of one or more products.

3.2.56 product_requiring_change to action_execution

Each `action_execution` requires zero, one, or many `product_requiring_change` instances.

3.2.57 `product_requiring_change` to `product_anomaly`

Each `product_requiring_change` requires a set of one or many `product_anomaly` instances. Each `product_anomaly` defines a set of one or many `product_requiring_change` instances.

3.2.58 `product_responsibility` to `organizational_project`

Each `organizational_project` defines the project for zero, one or more `product_responsibility` instances.

3.2.59 `product_responsibility` to `product`

Each `product` defines the product for zero, one or more `product_responsibility` instances.

3.2.60 `product_state` to `action_execution`

Each `action_execution` defines the `action_transition` of zero, one or more `product_state` instances.

3.2.61 `product_state` to `product_version`

Each `product_version` has a lifecycle state defined by zero, one or more `product_state` instances.

3.2.62 product_version to product

Each product is versioned by zero, one or more product_version instances.

3.2.63 product_version_group to group

Each group is the container for zero, one or more product_version_group instances.

3.2.64 product_version_group to product_version

Each product_version is grouped by (belongs to) zero, one or more product_version_group instances.

3.2.65 quantified_assembly_component_usage to measure_with_unit

Each instance of a measure_with_unit defines the quantity of zero, one, or many quantified_assembly_component_usage instances.

3.2.66 recommended_support_resource to action_item

Each instance of a action_item is the recommended action for zero, one or many recommended_support_resource instances.

3.2.67 related_change to product_anomaly

Each instance of a product_anomaly references zero, one or many related_change instances.

3.2.68 related_change to product_requiring_change

Each instance of a product_requiring_change defines the related product that is changing for zero, one or many related_change instances.

3.2.69 security_classification to security_classification_level

Each instance of a security_classification_level is categorized by zero, one, or many security_classification instances.

3.2.70 security_classification_assignment to security_classification

Each instance of a security_classification is assigned to zero, one, or many security_classification_assignment instances.

3.2.71 serial_numbered_effectivity to physical_unit

Each instance of a physical_unit defines the effectivity start unit for zero, one, or many serial_numbered_effectivity instances. Each instance of a physical_unit may define the effectivity end unit for zero, one, or many serial_numbered_effectivity instances.

3.2.72 specific_higher_usage_occurrence to assembly_component_usage

Each instance of an assembly_component_usage defines the upper usage for zero, one or many specific_higher_usage_occurrence instances.

3.2.73 specific_higher_usage_occurrence to next_assembly_component_usage

Each instance of an next_assembly_component_usage defines the next usage for zero, one or many specific_higher_usage_occurrence instances.

3.2.74 specified_item to product_version

Each instance of a specified_item defines the reference of a set of one or more product_version instances.

3.2.75 supplier to date

Each instance of a date defines the source date for zero, one or many supplier instances.

3.2.76 supplier to product

Each instance of a product defines the product supplied by zero, one or many supplier instances.

3.2.77 system_user to hardware_software

Each instance of a hardware_software system defines the association of zero, one or many system_user instances.

THIS PAGE INTENTIONALLY BLANK

Annex A RASSP Enterprise Data Model

A.1 RASSP Enterprise data model EXPRESS

```
SCHEMA RASSP_Build2_Enterprise_Data_Model;
```

```
TYPE day_in_month_number = INTEGER;
```

```
END_TYPE;
```

```
TYPE day_in_week_number = INTEGER;  
END_TYPE;
```

```
TYPE day_in_year_number = INTEGER;  
END_TYPE;
```

```
TYPE hour_in_day = INTEGER;  
END_TYPE;
```

```
TYPE identifier = STRING;  
END_TYPE;
```

```
TYPE integer = INTEGER;  
END_TYPE;
```

```
TYPE label = STRING;  
END_TYPE;
```

```
TYPE minute_in_hour = INTEGER;  
END_TYPE;
```

```
TYPE month_in_year_number = INTEGER;  
END_TYPE;
```

```
TYPE second_in_minute = INTEGER;  
END_TYPE;
```

```
TYPE text = STRING;
```

```
END_TYPE;
```

```
TYPE week_in_year_number = INTEGER;
```

```
END_TYPE;
```

```
TYPE year_number = INTEGER;
```

```
END_TYPE;
```

```
TYPE ahead_or_behind = ENUMERATION OF
```

```
(ahead,
```

```
behind);
```

```
END_TYPE;
```

```
TYPE assembly_spec_level = ENUMERATION OF
```

```
(ASSEM,
```

```
NEXAS,
```

```
USEON);
```

```
END_TYPE;
```

```
TYPE bom_quantity_unit = ENUMERATION OF
```

```
(AR,
```

```
CM,
```

```
EA,
```

```
FT,
```

```
GAL,
```

```
pdd_IN,
```

```
KM,
```

```
L,
```

```
LB,
```

```
ML,  
MM,  
NS,  
OZ,  
PT,  
QT);  
END_TYPE;
```

```
TYPE chemical_class = ENUMERATION OF  
(CHEM);  
END_TYPE;
```

```
TYPE chemical_subclass = ENUMERATION OF  
(BAGT,  
CAGT,  
CLAGT,  
IAGT,  
MAGT,  
TAGT);  
END_TYPE;
```

```
TYPE chemical_type = ENUMERATION OF  
(BOADH,  
BOEPA,  
BOSOL,  
BOWEL,  
CLSOL,  
COCON,  
COLUB,
```



```
COSEL,  
COSIL,  
INSCO,  
MABLK,  
MABLU,  
MAENF,  
MAGRE,  
MARED,  
MAWHI,  
MAYEL,  
THCON);  
END_TYPE;
```

```
TYPE electronic_functional_unit_logic = ENUMERATION OF  
(ALU,  
ANA,  
CLK,  
CNT,  
COM,  
DMUX,  
DRAM,  
HYB,  
MUX,  
PIO,  
PLD,  
RAM,  
ROM,  
SEQ,  
SRAM);
```

END_TYPE;

TYPE electronic_part_class = ENUMERATION OF

(BATTE,

CAP,

CON,

IND,

LAMP,

RES,

ROTMA,

SEMI,

SWTCH,

TRADU,

UCT,

XFMR);

END_TYPE;

TYPE electronic_part_package = ENUMERATION OF

(AXIAL,

CAN,

CHIP,

COB,

DIP,

FEATR,

FP,

GCC,

JCC,

LCC,

PGA,

RDL,
SIP,
SOIC,
TOROI);
END_TYPE;

TYPE electronic_part_subclass = ENUMERATION OF

(ACMAC,
ADAPT,
ANTEN,
BALLA,
BELL,
BUSBR,
COAX,
CORE,
DCMAC,
DIG,
DIODE,
pdd_FIXED,
FLUOR,
FUS,
GLOW,
HALL,
HYB,
INCAN,
JUM,
KEY,
LIN,
MIC,

MIXED,
NONR,
PLUG,
POWER,
RECH,
RECPT,
RELAY,
RIGID,
SCR,
SIGNL,
SPK,
SWTCH,
SYNCH,
TERM,
TRANS,
pdd_VAR,
WAVG);
END_TYPE;

TYPE electronic_part_type = ENUMERATION OF

(ACMAC,
ADAPT,
ALKA,
ANA,
BALLA,
BELL,
BIZ,
BRIDGE,
CAP,

COAX,
CSW,
CUST,
DARN,
DARP,
DCMAC,
DIO,
DIPOL,
FERBD,
FETN,
FETP,
FLUOR,
FUS,
GLOW,
HALL,
HTR,
HYB,
INCAN,
IND,
JUM,
KEY,
LDAC,
LED,
LSI,
MIC,
MONPO,
MSI,
NICAD,
OCPL,

OSW,
OVEN,
PCA,
PIS,
POT,
RCNC,
RCNO,
RES,
RHE,
ROSW,
RP_DB,
RP_DH,
RP_DI,
RP_DT,
RP_SB,
RP_SH,
RP_SI,
RP_ST,
SCR,
SOCKET,
SPK,
SSI,
SYNCH,
TMSTR,
TPCP,
TPCS,
TRAP,
TRAS,
TRNN,

```
TRNP ,  
UNI ,  
VHSIC ,  
VLSI ,  
VRSTR ,  
WAVG ,  
XTL ,  
ZDI ) ;  
END_TYPE ;
```

```
TYPE engineering_unit = ENUMERATION OF
```

```
(T ,  
G ,  
M ,  
K ,  
m ,  
u ,  
n ,  
p ,  
f ) ;
```

```
END_TYPE ;
```

```
TYPE hardware_class = ENUMERATION OF
```

```
(HDWR) ;
```

```
END_TYPE ;
```

```
TYPE hardware_subclass = ENUMERATION OF
```

```
(BOLT ,  
BRACK ,
```

BRVT ,
CLAMP ,
CLIP ,
CRVT ,
EJECT ,
EYELE ,
FRAME ,
FWSHR ,
HANDL ,
HTSNK ,
INSUL ,
INWAS ,
LOBAR ,
LWSHR ,
NUT ,
PAD ,
PIN ,
PLATE ,
PUBAR ,
RING ,
SCREW ,
SHIEL ,
SLEEV ,
SPACE ,
SPREA ,
SPRIN ,
SPRPN ,
TIE ,
TRVT ,

WASHR);

END_TYPE;

TYPE hardware_type = ENUMERATION OF

(HDWR,

HTSNK,

HTSRK,

INSLV,

INSPR,

INWAS,

LOBAR,

MCLAM,

MCLIP,

PCLAM,

PCLIP,

PLTIE,

STTIE,

TPTIE,

WASH);

END_TYPE;

TYPE lead_material = ENUMERATION OF

(STL,

OTH);

END_TYPE;

TYPE lead_plating = ENUMERATION OF

(GLD,

OTH);

```
END_TYPE;
```

```
TYPE lead_solderability = ENUMERATION OF
```

```
(NOWAV,
```

```
NRFLO,
```

```
NWRF,
```

```
WAV);
```

```
END_TYPE;
```

```
TYPE pcb_package = ENUMERATION OF
```

```
(TYPE1,
```

```
TYPE2,
```

```
TYPE3,
```

```
TYPE4,
```

```
TYPE5,
```

```
TYPE6);
```

```
END_TYPE;
```

```
TYPE polarity_name = ENUMERATION OF
```

```
( );
```

```
END_TYPE;
```

```
TYPE printed_circuit_board_class = ENUMERATION OF
```

```
(PCB);
```

```
END_TYPE;
```

```
TYPE printed_circuit_board_subclass = ENUMERATION OF
```

```
(FLEX,
```

```
MOLD,
```

```
RFLEX,  
RIGID,  
HYB);  
END_TYPE;
```

```
TYPE printed_circuit_board_type = ENUMERATION OF  
(FLEX,  
MOLD,  
RFLEX,  
RIGID,  
HYB);  
END_TYPE;
```

```
TYPE si_prefix = ENUMERATION OF  
(exa,  
peta,  
tera,  
giga,  
mega,  
kilo,  
hecto,  
deca,  
deci,  
centi,  
milli,  
micro,  
nano,  
pico,  
femto,
```

atto);

END_TYPE;

TYPE si_unit_name = ENUMERATION OF

(metre,

gram,

second,

ampere,

kelvin,

mole,

candela,

radian,

steradian,

hertz,

newton,

pascal,

joule,

watt,

coulomb,

volt,

farad,

ohm,

siemens,

weber,

tesla,

henry,

degree_Celsius,

lumen,

lux,

```
becquerel,  
gray,  
sievert);  
END_TYPE;
```

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

```
TYPE pca_component = SELECT  
(chemical,  
hardware,  
printed_circuit_board,  
electronic_part);  
END_TYPE;
```

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

```
TYPE support_resource_select = SELECT  
(person,  
organization,  
support_equipment);  
END_TYPE;
```

```
TYPE unit = SELECT
```

```
(derived_unit,
```

```
named_unit);
```

```
END_TYPE;
```

```
ENTITY CAD_data
```

```
SUBTYPE OF (factory_interface_information,file_folder);
```

```
layers : SET [1:?] OF layer_data;
```

```
silk_screen : silk_screen_data;
```

```
holes : SET [1:2] OF hole_data;
```

```
sub_file_1 : header_sub_file;
```

```
END_ENTITY;
```

```
ENTITY SMDCAD_data
```

```
SUBTYPE OF (factory_interface_information,file_folder);
```

```
header_record : SMD_header;
```

```
SMD_records : LIST [1:?] OF SMD_records;
```

```
END_ENTITY;
```

```
ENTITY SMD_header;
```

```
SMD_header_field_2 : five_bite_bf;
```

```
SMD_header_field_1 : board_id;
```

```
SMD_header_field_4 : time_record;
```

```
SMD_header_field_3 : date_of_creation;
```

```
SMD_header_field_1 : ref_des;
```

```
END_ENTITY;
```

```
ENTITY SMD_records;
```

```
SMD_record_field_4 : five_bite_bf;
SMD_header_field_3 : x_component_center;
SMD_header_field_5 : y_component_center;
SMD_header_field_7 : SMD_rotation_flag;
SMD_header_field_2 : eight_bite_bf;
SMD_header_field_6 : eight_bite_bf;
END_ENTITY;
```

```
ENTITY SMD_rotation_flag;
END_ENTITY;
```

```
ENTITY action
SUPERTYPE OF (action_execution)
SUBTYPE OF (product);
method : action_method;
END_ENTITY;
```

```
ENTITY action_assignment
ABSTRACT SUPERTYPE OF (action_item ANDOR product_process_step);
assigned_action : action;
END_ENTITY;
```

```
ENTITY action_execution
SUBTYPE OF (action);
order : ordered_action;
END_ENTITY;
```

```
ENTITY action_execution_support_resource;
executed_action : action_execution;
```

```
supporting_resources : support_resource_select;  
END_ENTITY;
```

```
ENTITY action_item  
SUBTYPE OF (action_assignment);  
items : SET [1:?] OF product_version;  
END_ENTITY;
```

```
ENTITY action_method  
SUBTYPE OF (product);  
requests : SET [1:?] OF requested_action;  
purpose : text;  
consequence : text;  
END_ENTITY;
```

```
ENTITY action_method_relationship  
SUPERTYPE OF (process_action_method_relationship);  
relating_action_method : action_method;  
related_action_method : action_method;  
name : label;  
description : text;  
END_ENTITY;
```

```
ENTITY action_status;  
assigned_action : action_execution;  
status : label;  
END_ENTITY;
```

```
ENTITY address
```



```
SUPERTYPE OF (organizational_address ANDOR personal_address);  
telex_number : OPTIONAL label;  
electronic_mail_address : OPTIONAL label;  
telephone_number : OPTIONAL label;  
facsimile_number : OPTIONAL label;  
country : OPTIONAL label;  
postal_code : OPTIONAL label;  
region : OPTIONAL label;  
town : OPTIONAL label;  
postal_box : OPTIONAL label;  
street : OPTIONAL label;  
street_number : OPTIONAL label;  
mail_stop : OPTIONAL label;  
END_ENTITY;
```

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

```
ENTITY approval_assignment  
ABSTRACT SUPERTYPE OF (approved_item);  
assigned_approval : approval;  
END_ENTITY;
```

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

```
ENTITY approval_person_organization;  
authorized_approval : approval;  
role : approval_role;  
person_organization : person_organization_select;  
END_ENTITY;
```

```
ENTITY approval_role;  
role : label;  
END_ENTITY;
```

```
ENTITY approval_status;  
name : label;  
END_ENTITY;
```

```
ENTITY approved_item  
SUBTYPE OF (approval_assignment);  
items : SET [1:?] OF product_version;  
END_ENTITY;
```

```
ENTITY assembly_component_usage  
SUPERTYPE OF (quantified_assembly_component_usage ANDOR pca_component_usage  
ANDOR  
ONEOF(promissory_usage_occurrence,specified_higher_usage_occurrence,next_assembly_  
SUBTYPE OF (product_definition_usage);  
reference_designator : OPTIONAL identifier;  
END_ENTITY;
```

```
ENTITY assembly_component_usage_substitute;  
base : assembly_component_usage;
```

```
substitute : assembly_component_usage;  
UNIQUE  
URL: base,substitute;  
END_ENTITY;
```

```
ENTITY assembly_specification  
SUBTYPE OF (pca_specification);  
name : STRING;  
revision : STRING;  
spec_identifier : STRING;  
cage_code : STRING;  
level : assembly_spec_level;  
END_ENTITY;
```

```
ENTITY bill_of_material_item;  
bom_item_type : pca_component;  
quantity : INTEGER;  
item_number : INTEGER;  
bom_item_descr : STRING;  
part_number : STRING;  
ref_designator : SET [1:?] OF STRING;  
qunit : bom_quantity_unit;  
applicable_spec : pca_specification;  
END_ENTITY;
```

```
ENTITY board_extremes;  
board_ext_field_0 : two_bite_bf;  
board_ext_field_1 : ll_x_cord;  
board_ext_field_3 : ur_x_cord;
```

```
board_ext_field_4 : ur_y_cord;  
board_ext_field_2 : ll_y_cord;  
END_ENTITY;
```

```
ENTITY board_id;  
END_ENTITY;
```

```
ENTITY board_ident;  
board_id_field_2 : five_bite_bf;  
board_id_field_1 : board_id;  
board_id_field_3 : rev_letter;  
board_id_field_5 : flags;  
board_id_field_0 : two_bite_bf;  
board_id_field_4 : two_bite_bf;  
END_ENTITY;
```

```
ENTITY cad_prog_name;  
END_ENTITY;
```

```
ENTITY cage;  
cage_code : identifier;  
END_ENTITY;
```

```
ENTITY calendar_date  
SUBTYPE OF (date);  
day_component : day_in_month_number;  
month_component : month_in_year_number;  
END_ENTITY;
```

ENTITY change_no;

END_ENTITY;

ENTITY change_number;

change_no_field_0 : two_bite_bf;

change_no_field_1 : change_no;

END_ENTITY;

ENTITY chemical

SUBTYPE OF (part);

class : chemical_class;

subclass : chemical_subclass;

chem_type : chemical_type;

END_ENTITY;

ENTITY classified_item

SUBTYPE OF (security_classification_assignment);

items : SET [1:?] OF product_version;

END_ENTITY;

ENTITY commercial_form_factor;

eia_jdc : STRING;

END_ENTITY;

ENTITY component_specification

SUBTYPE OF (pca_specification);

name : STRING;

revision : STRING;

spec_identifier : STRING;

END_ENTITY;

ENTITY concurrent_action_method

SUBTYPE OF (process_action_method_relationship);

END_ENTITY;

ENTITY configuration_design;

design : product_version;

configuration : configuration_item;

UNIQUE

URL: configuration,design;

END_ENTITY;

ENTITY configuration_item

SUPERTYPE OF (signal_processor_design)

SUBTYPE OF (product);

item_concept : product_concept;

purpose : label;

UNIQUE

URL: identification;

END_ENTITY;

ENTITY context_dependent_unit

SUBTYPE OF (named_unit);

name : label;

END_ENTITY;

ENTITY contract;

name : label;

```
kind : text;  
purpose : text;  
END_ENTITY;
```

```
ENTITY contract_assignment;  
product : product_version;  
assigned_contract : contract;  
END_ENTITY;
```

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

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

```
ENTITY correspondence  
SUBTYPE OF (document);  
END_ENTITY;
```

```
ENTITY data_template  
SUBTYPE OF (product);  
END_ENTITY;
```

```
ENTITY date
SUPERTYPE OF (ONEOF(ordinal_date,calendar_date,week_of_year_and_day_date));
year_component : year_number;
END_ENTITY;
```

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

```
ENTITY date_of_creation;
date_field_0 : two_bite_bf;
date_field_1 : date;
END_ENTITY;
```

```
ENTITY dated_effectivity
SUBTYPE OF (planned_effectivity);
effectivity_end_date : OPTIONAL date_and_time;
effectivity_start_date : date_and_time;
END_ENTITY;
```

```
ENTITY derived_unit;
elements : SET [1:?] OF derived_unit_element;
END_ENTITY;
```

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


ENTITY det_12_bite_bf;
END_ENTITY;

ENTITY det_1_bite_f;
END_ENTITY;

ENTITY det_2_bite_bf;
END_ENTITY;

ENTITY det_NOR_rel_lev;
END_ENTITY;

ENTITY det_artwork;
END_ENTITY;

ENTITY det_assy;
END_ENTITY;

ENTITY det_assy_doc_rev;
END_ENTITY;

ENTITY det_assy_name;
END_ENTITY;

ENTITY det_compon_cnt;
END_ENTITY;

ENTITY det_compon_descr;

END_ENTITY;

ENTITY det_compon_nam;

END_ENTITY;

ENTITY det_compon_p_n;

END_ENTITY;

ENTITY det_ece;

END_ENTITY;

ENTITY det_eff;

END_ENTITY;

ENTITY det_pfn;

END_ENTITY;

ENTITY det_qty_used;

END_ENTITY;

ENTITY det_ref_desig;

END_ENTITY;

ENTITY det_request_cnt;

END_ENTITY;

ENTITY det_u_m;

END_ENTITY;

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

```
ENTITY discrepant_product
SUBTYPE OF (product_requiring_change);
failure_rate : SET [1:?] OF REAL;
END_ENTITY;
```

```
ENTITY document
SUPERTYPE OF (specification ANDOR correspondence ANDOR drawing ANDOR
procedure ANDOR publication)
SUBTYPE OF (product);
kind : document_type;
size : integer;
UNIQUE
URL: id;
END_ENTITY;
```

```
ENTITY document_reference
ABSTRACT SUPERTYPE OF (specified_item);
assigned_document : document;
END_ENTITY;
```

```
ENTITY document_type;  
product_data_type : label;  
END_ENTITY;
```

```
ENTITY drawing  
SUBTYPE OF (document);  
END_ENTITY;
```

```
ENTITY edge_con_bot;  
END_ENTITY;
```

```
ENTITY edge_con_id;  
edge_field_2 : edge_con_right;  
edge_field_4 : edge_con_left;  
edge_field_3 : edge_con_bot;  
edge_field_1 : edge_con_top;  
edge_field_0 : two_bite_bf;  
END_ENTITY;
```

```
ENTITY edge_con_left;  
END_ENTITY;
```

```
ENTITY edge_con_right;  
END_ENTITY;
```

```
ENTITY edge_con_top;  
END_ENTITY;
```

```
ENTITY eight_bite_bf;
```

END_ENTITY;

ENTITY electronic_functional_parameters;

power : REAL;

component_value : STRING;

value_unit : engineering_unit;

max_wrkg_volt : REAL;

pos_tol : REAL;

neg_tol : REAL;

base_value : REAL;

END_ENTITY;

ENTITY electronic_functional_unit

SUBTYPE OF (functional_unit);

logic : electronic_functional_unit_logic;

generic_part_number : STRING;

parameters : electronic_functional_parameters;

END_ENTITY;

ENTITY electronic_part

SUPERTYPE OF (packaged_part)

SUBTYPE OF (part);

function : functional_unit;

class : electronic_part_class;

subclass : electronic_part_subclass;

part_type : electronic_part_type;

END_ENTITY;

ENTITY end_of_sub_file;

END_ENTITY;

ENTITY enhancement_product

SUBTYPE OF (product_requiring_change);

END_ENTITY;

ENTITY enterprise

SUBTYPE OF (organization);

END_ENTITY;

ENTITY factory_interface_information

SUPERTYPE OF (oscc_smoutput_data_file ANDOR oscc_ploutput_data_file ANDOR
CAD_data ANDOR SMDCAD_data);

END_ENTITY;

ENTITY file_folder

SUPERTYPE OF (oscc_smoutput_data_file ANDOR oscc_ploutput_data_file ANDOR
CAD_data ANDOR SMDCAD_data)

SUBTYPE OF (physical_unit);

file_type : label;

representative_product : product_version;

END_ENTITY;

ENTITY five_bite_bf;

END_ENTITY;

ENTITY flags;

END_ENTITY;

ENTITY functional_unit

```
SUPERTYPE OF (electronic_functional_unit);  
END_ENTITY;
```

```
ENTITY group;  
group_name : text;  
END_ENTITY;
```

```
ENTITY group_relationship;  
relating_group : group;  
related_group : group;  
group_relationship_name : text;  
description : text;  
END_ENTITY;
```

```
ENTITY hardware  
SUBTYPE OF (part);  
class : hardware_class;  
subclass : hardware_subclass;  
hard_type : hardware_type;  
END_ENTITY;
```

```
ENTITY hardware_software  
SUBTYPE OF (system);  
END_ENTITY;
```

```
ENTITY header_sub_file;  
header_record_1 : board_ident;  
header_record_8 : edge_con_id;  
header_record_6 : number_of_layers;
```

```
header_record_7 : board_extremes;  
header_record_5 : type_id;  
header_record_10 : silk_screen;  
header_record_4 : time_record;  
header_record_3 : date_of_creation;  
header_record_2 : change_number;  
header_record_11 : plated_hole_size;  
header_record_9 : mask_clearance;  
END_ENTITY;
```

```
ENTITY height_of_pad;  
END_ENTITY;
```

```
ENTITY hole_data;  
plated_hole_sub_file : plated_holes;  
non_plated_hole_sub_file : non_plated_holes;  
END_ENTITY;
```

```
ENTITY hole_diameter;  
END_ENTITY;
```

```
ENTITY hole_size;  
END_ENTITY;
```

```
ENTITY holes;  
hole_field_12 : two_bite_bf;  
hole_field_6 : two_bite_bf;  
hole_field_2 : x_hole_center;  
hole_field_4 : hole_diameter;
```



```
hole_field_1 : two_bite_bf;  
hole_field_3 : y_hole_center;  
hole_field_7 : shape_name;  
hole_field_9 : rotation_flag;  
hole_field_8 : pin_number;  
hole_field_14 : node_number;  
hole_field_13 : ref_des;  
hole_field_5 : five_bite_bf;  
hole_field_10 : five_bite_bf;  
hole_field_11 : five_bite_bf;  
END_ENTITY;
```

```
ENTITY layer_data;  
layer_sub_file_4 : vapes;  
layer_sub_file_1 : plated_pads;  
layer_sub_file_2 : non_plated_pads;  
layer_sub_file_3 : lines;  
END_ENTITY;
```

```
ENTITY line_point_pairs;  
line_y_point : y_point_on_line;  
line_x_point : x_point_on_line;  
END_ENTITY;
```

```
ENTITY line_segments;  
line_field_1 : two_bite_bf;  
line_field_2 : number_of_point_on_line;  
line_field_3 : width_of_line;  
line_fields : LIST [1:?] OF line_point_pairs;
```

```
line_field_4 : mask_flag;  
line_field_n : node_number;  
END_ENTITY;
```

```
ENTITY lines;  
end_of_sub_file_3_record : end_of_sub_file;  
line_segment_records : LIST [1:?] OF line_segments;  
END_ENTITY;
```

```
ENTITY ll_x_cord;  
END_ENTITY;
```

```
ENTITY ll_y_cord;  
END_ENTITY;
```

```
ENTITY local_time;  
zone : coordinated_universal_time_offset;  
hour_component : hour_in_day;  
minute_component : OPTIONAL minute_in_hour;  
second_component : OPTIONAL second_in_minute;  
END_ENTITY;
```

```
ENTITY lot_effectivity  
SUBTYPE OF (planned_effectivity);  
effectivity_lot_size : measure_with_unit;  
effectivity_lot_id : identifier;  
END_ENTITY;
```

```
ENTITY lower_left_x_coord_rect_pad;
```

END_ENTITY;

ENTITY lower_left_y_coord_rect_pad;

END_ENTITY;

ENTITY make_from_usage_option

SUBTYPE OF (product_definition_usage);

quantity : measure_with_unit;

ranking_rationale : text;

ranking : INTEGER;

END_ENTITY;

ENTITY manufacturing_interface_requirements;

oscc_smoutput_data : oscc_smoutput_data_file;

oscc_ploutput_data : oscc_ploutput_data_file;

: CAD_data;

SMDCAD_data : LIST [1:2] OF SMDCAD_data;

END_ENTITY;

ENTITY mask_clearance;

mask_cl_field_0 : two_bite_bf;

mask_cl_field_1 : mask_space;

mask_cl_field_2 : rect_pads_flag;

END_ENTITY;

ENTITY mask_flag;

END_ENTITY;

ENTITY mask_space;

END_ENTITY;

ENTITY measure_with_unit;

unit_component : unit;

value_component : REAL;

END_ENTITY;

ENTITY military_form_factor;

slash_number : STRING;

style_or_type : STRING;

specification : STRING;

ff_case : STRING;

dash_number : STRING;

END_ENTITY;

ENTITY named_unit

SUPERTYPE OF (ONEOF(si_unit,context_dependent_unit,conversion_based_unit));

dimensions : dimensional_exponents;

END_ENTITY;

ENTITY next_assembly_usage_occurrence

SUBTYPE OF (assembly_component_usage);

END_ENTITY;

ENTITY node_location;

system : hardware_software;

protocol : text;

node_address : text;

END_ENTITY;

ENTITY node_number;

END_ENTITY;

ENTITY node_number;

END_ENTITY;

ENTITY non_plated_holes;

end_of_non_plated_sub_1_record : end_of_sub_file;

non_plated_hole_records : LIST [1:?] OF holes;

END_ENTITY;

ENTITY non_plated_pads;

end_of_sub_file_2_record : end_of_sub_file;

non_plated_pad_records : LIST [1:?] OF pads;

END_ENTITY;

ENTITY num_layers;

END_ENTITY;

ENTITY number_of_layers;

num_lay_field_1 : num_layers;

num_lay_field_0 : two_bite_bf;

END_ENTITY;

ENTITY number_of_point_on_line;

END_ENTITY;

ENTITY number_plated_hole_sizes;

END_ENTITY;

ENTITY ordered_action;

requests : SET [1:?] OF requested_action;

name : label;

description : text;

comment : text;

analysis : text;

END_ENTITY;

ENTITY ordinal_date

SUBTYPE OF (date);

day_component : day_in_year_number;

END_ENTITY;

ENTITY organization

SUPERTYPE OF (ONEOF(enterprise,program) ANDOR supplier)

SUBTYPE OF (product);

cage_code : cage;

END_ENTITY;

ENTITY organizational_address

SUBTYPE OF (address);

organizations : SET [1:?] OF organization;

END_ENTITY;

ENTITY organizational_project;

responsible_organizations : SET [1:?] OF organization;

description : text;

name : label;

END_ENTITY;

ENTITY oscc_ploutput_data;

END_ENTITY;

ENTITY oscc_ploutput_data;

plout_field_1 : det_1_bite_f;

plout_field_2 : det_request_cnt;

plout_field_4 : det_ece;

plout_field_3 : det_assy;

plout_field_6 : det_compon_cnt;

plout_field_5 : det_eff;

plout_field_7 : det_pfn;

plout_field_9 : det_assy_name;

plout_field_8 : det_compon_p_n;

plout_field_10 : det_compon_nam;

plout_field_11 : det_qty_used;

plout_field_13 : det_ref_desig;

plout_field_15 : det_assy_doc_rev;

plout_field_17 : det_12_bite_bf;

plout_field_19 : det_2_bite_bf;

plout_field_12 : det_u_m;

plout_field_14 : det_compon_descr;

plout_field_16 : det_artwork;

plout_field_18 : det_NOR_rel_lev;

END_ENTITY;

ENTITY oscc_ploutput_data_file

```
SUBTYPE OF (factory_interface_information,file_folder);
oscc_ploutput_records : SET [1:?] OF oscc_ploutput_data;
END_ENTITY;
```

```
ENTITY oscc_smoutput_data_file
SUBTYPE OF (factory_interface_information,file_folder);
oscc_smoutput_records : SET [1:?] OF oscc_ploutput_data;
END_ENTITY;
```

```
ENTITY packaged_part
SUBTYPE OF (electronic_part);
commercial_class : commercial_form_factor;
military_class : military_form_factor;
leads : SET [1:?] OF packaged_part_lead;
cage : STRING;
pin_count : INTEGER;
package_type : electronic_part_package;
END_ENTITY;
```

```
ENTITY packaged_part_lead;
lead_identification : STRING;
material : lead_material;
plating : lead_plating;
polarity : polarity_name;
solderability : lead_solderability;
END_ENTITY;
```

```
ENTITY pads;
pad_field_1 : two_bite_bf;
```



```
pad_field_6 : two_bite_bf;
pad_field_12 : two_bite_bf;
pad_field_3 : y_pad_center;
pad_field_2 : x_pad_center;
pad_field_4 : height_of_pad;
pad_field_5 : width_of_pad;
pad_field_7 : shape_name;
pad_field_8 : pin_number;
pad_field_9 : rotation_flag;
pad_field_13 : ref_des;
pad_field_14 : node_number;
pad_field_10 : mask_flag;
pad_field_11 : hole_size;
END_ENTITY;
```

ENTITY part

```
SUPERTYPE OF (software_application ANDOR reuse_part ANDOR electronic_part
ANDOR hardware ANDOR printed_circuit_board ANDOR chemical)
```

```
SUBTYPE OF (product);
```

```
part_configuration_identifier : identifier;
```

```
part_function_type : text;
```

```
part_type : text;
```

```
END_ENTITY;
```

ENTITY pca_component_usage

```
SUBTYPE OF (assembly_component_usage);
```

```
END_ENTITY;
```

ENTITY pca_specification

SUPERTYPE OF (assembly_specification ANDOR component_specification ANDOR test_specification ANDOR pcb_specification)

SUBTYPE OF (specification);

END_ENTITY;

ENTITY pcb_specification

SUBTYPE OF (pca_specification);

spec_identifer : STRING;

revision : STRING;

name : STRING;

END_ENTITY;

ENTITY person

SUPERTYPE OF (system_user)

SUBTYPE OF (product);

last_name : label;

first_name : label;

suffix_titles : OPTIONAL LIST [1:?] OF label;

prefix_titles : OPTIONAL LIST [1:?] OF label;

middle_names : OPTIONAL LIST [1:?] OF label;

UNIQUE

UR1: id;

END_ENTITY;

ENTITY person_and_organization;

person : person;

organization : organization;

person_organization_role : text;

END_ENTITY;

```
ENTITY personal_address
SUBTYPE OF (address);
people : SET [1:?] OF person;
END_ENTITY;
```

```
ENTITY physical_unit
SUPERTYPE OF (file_folder)
SUBTYPE OF (product_version);
configuration : configuration_design;
UNIQUE
UR1: configuration;
END_ENTITY;
```

```
ENTITY pin_number;
END_ENTITY;
```

```
ENTITY planned_effectivity
SUPERTYPE OF
(ONEOF(serial_numbered_effectivity,lot_effectivity,dated_effectivity));
configuration : configuration_design;
design_usage : product_definition_usage;
identification : identifier;
UNIQUE
UR1: identification,configuration,design_usage;
END_ENTITY;
```

```
ENTITY plated_hole_size;
plated_hole_size_field_0 : two_bite_bf;
plated_hole_size_field_1 : number_plated_hole_sizes;
```

END_ENTITY;

ENTITY plated_holes;

end_of_plated_sub_record : end_of_sub_file;

plated_hole_records : LIST [1:?] OF holes;

END_ENTITY;

ENTITY plated_pads;

end_of_sub_file_1_record : end_of_sub_file;

plated_pad_records : LIST [1:?] OF pads;

END_ENTITY;

ENTITY printed_circuit_board

SUBTYPE OF (part);

class : printed_circuit_board_class;

subclass : printed_circuit_board_subclass;

pcb_type : printed_circuit_board_type;

package_type : pcb_package;

END_ENTITY;

ENTITY procedure

SUBTYPE OF (document);

END_ENTITY;

ENTITY process_action_method_relationship

SUPERTYPE OF

(ONEOF(serial_concurrent_action_method, concurrent_action_method, serial_action_me

SUBTYPE OF (action_method_relationship);

END_ENTITY;

ENTITY product

SUPERTYPE OF (part ANDOR action_method ANDOR action ANDOR configuration_item
ANDOR product_concept ANDOR document ANDOR person ANDOR organization ANDOR
data_template ANDOR system);

description : text;

frame_of_reference : label;

name : label;

id : identifier;

UNIQUE

URL: id;

END_ENTITY;

ENTITY product_anomaly

SUPERTYPE OF (product_issue ANDOR product_concern ANDOR product_flaw);

product_anomaly_identifier : identifier;

product_anomaly_description : text;

detection_method : text;

anomaly_type : text;

anomaly_cause : text;

INVERSE

products : SET[1:?] OF product_requiring_change FOR anomalized_products;

END_ENTITY;

ENTITY product_anomaly_disposition;

anomalized_product : product_anomaly;

disposition_actions : SET [1:?] OF action_execution;

END_ENTITY;

ENTITY product_change;

baseline_product : product_requiring_change;

```
baseline_product_disposition : product_anomaly_disposition;
resulting_product : product_version;
reasons : SET [1:?] OF text;
END_ENTITY;
```

```
ENTITY product_classification
SUBTYPE OF (security_classification_assignment);
items : SET [1:?] OF product;
END_ENTITY;
```

```
ENTITY product_concept
SUBTYPE OF (product);
product_concept_context : label;
UNIQUE
URL: identification;
END_ENTITY;
```

```
ENTITY product_concern
SUBTYPE OF (product_anomaly);
END_ENTITY;
```

```
ENTITY product_definition;
version : product_version;
description : text;
frame_of_reference : label;
END_ENTITY;
```

```
ENTITY product_definition_relationship
SUPERTYPE OF (product_definition_usage);
```

```
related_product_definition : product_definition;
relating_product_definition : product_definition;
id : identifier;
name : label;
description : text;
END_ENTITY;
```

```
ENTITY product_definition_usage
SUPERTYPE OF (ONEOF(make_from_usage_option,assembly_component_usage))
SUBTYPE OF (product_definition_relationship);
UNIQUE
URI:
SELF\product_definition_relationship.id,SELF\product_definition_relationship.rel.
END_ENTITY;
```

```
ENTITY product_flaw
SUBTYPE OF (product_anomaly);
product_flaw_type : text;
END_ENTITY;
```

```
ENTITY product_flaw_classification;
classified_product : product_flaw;
flaw_class_identifier : identifier;
END_ENTITY;
```

```
ENTITY product_issue
SUBTYPE OF (product_anomaly);
END_ENTITY;
```

```
ENTITY product_process_step
SUBTYPE OF (action_assignment);
products : SET [1:?] OF product;
END_ENTITY;
```

```
ENTITY product_requiring_change
SUPERTYPE OF (ONEOF(discrepant_product,enhancement_product) ANDOR
related_change)
SUBTYPE OF (product_version);
anomalized_products : SET [1:?] OF product_anomaly;
product_change_requirement_type : text;
requiring_change_product : action_execution;
END_ENTITY;
```

```
ENTITY product_responsibility;
project : organizational_project;
product : product;
END_ENTITY;
```

```
ENTITY product_state;
product : product_version;
action_transition : action_execution;
state_name : label;
END_ENTITY;
```

```
ENTITY product_version
SUPERTYPE OF (product_requiring_change ANDOR physical_unit);
of_product : product;
description : text;
version_id : identifier;
```


UNIQUE

URL: version_id, of_product;

END_ENTITY;

ENTITY product_version_group;

group : group;

version : product_version;

END_ENTITY;

ENTITY program

SUBTYPE OF (organization);

END_ENTITY;

ENTITY promissory_usage_occurrence

SUBTYPE OF (assembly_component_usage);

END_ENTITY;

ENTITY publication

SUBTYPE OF (document);

END_ENTITY;

ENTITY quantified_assembly_component_usage

SUBTYPE OF (assembly_component_usage);

quantity : measure_with_unit;

END_ENTITY;

ENTITY recommended_support_resource;

recommended_action : action_item;

supporting_resource : support_resource_select;

recommended_role : role;

END_ENTITY;

ENTITY rect_pads_flag;

END_ENTITY;

ENTITY rectangular_pads;

rect_field_6 : mask_flag;

rect_field_5 : upper_right_y_coord_rect_pad;

rect_field_3 : lower_left_y_coord_rect_pad;

rect_field_1 : two_bite_bf;

rect_field_4 : upper_right_x_coord_rect_pad;

rect_field_2 : lower_left_x_coord_rect_pad;

END_ENTITY;

ENTITY ref_des;

END_ENTITY;

ENTITY related_change

SUBTYPE OF (product_requiring_change);

related_change_product : product_requiring_change;

anomalized_product : product_anomaly;

END_ENTITY;

ENTITY requested_action;

id : identifier;

version : label;

purpose : text;

description : text;

END_ENTITY;

ENTITY reuse_part

SUBTYPE OF (part);

END_ENTITY;

ENTITY rev_letter;

END_ENTITY;

ENTITY role;

role_name : label;

END_ENTITY;

ENTITY rotation_flag;

END_ENTITY;

ENTITY security_classification;

security_level : security_classification_level;

name : label;

purpose : text;

END_ENTITY;

ENTITY security_classification_assignment

ABSTRACT SUPERTYPE OF (classified_item ANDOR product_classification);

assigned_security_classification : security_classification;

END_ENTITY;

ENTITY security_classification_level;

name : label;

END_ENTITY;

ENTITY sequential_method

SUBTYPE OF (serial_action_method);

sequence_position : NUMBER;

END_ENTITY;

ENTITY serial_action_method

SUPERTYPE OF (sequential_method)

SUBTYPE OF (process_action_method_relationship);

END_ENTITY;

ENTITY serial_concurrent_action_method

SUBTYPE OF (process_action_method_relationship);

END_ENTITY;

ENTITY serial_numbered_effectivity

SUBTYPE OF (planned_effectivity);

effectivity_start_id : physical_unit;

effectivity_end_id : OPTIONAL physical_unit;

END_ENTITY;

ENTITY shape_name;

END_ENTITY;

ENTITY si_unit

SUBTYPE OF (named_unit);

name : si_unit_name;

prefix : OPTIONAL si_prefix;

END_ENTITY;

ENTITY signal_processor_design
SUBTYPE OF (configuration_item);
END_ENTITY;

ENTITY silk_screen;
ss_flags_field_0 : two_bite_bf;
ss_flags_field_1 : silk_screen_bot;
ss_flags_field_2 : silk_screen_top;
END_ENTITY;

ENTITY silk_screen_bot;
end_of_ss_sub_1_record : end_of_sub_file;
ss_sub_1 : lines;
END_ENTITY;

ENTITY silk_screen_data;
silk_screen_sub_file_bot : silk_screen_bot;
silk_screen_sub_file_top : silk_screen_top;
END_ENTITY;

ENTITY silk_screen_top;
end_of_ss_sub_2_record : end_of_sub_file;
ss_sub_2 : lines;
END_ENTITY;

ENTITY software_application
SUBTYPE OF (part);

software_language : text;

END_ENTITY;

ENTITY specification

SUPERTYPE OF (pca_specification)

SUBTYPE OF (document);

END_ENTITY;

ENTITY specified_higher_usage_occurrence

SUBTYPE OF (assembly_component_usage);

next_usage : next_assembly_usage_occurrence;

upper_usage : assembly_component_usage;

UNIQUE

URL: upper_usage,next_usage;

END_ENTITY;

ENTITY specified_item

SUBTYPE OF (document_reference);

items : SET [1:?] OF product_version;

END_ENTITY;

ENTITY supplier

SUBTYPE OF (organization);

product : product;

source_date : date;

END_ENTITY;

ENTITY support_equipment;

name : label;

END_ENTITY;

ENTITY system

SUPERTYPE OF (hardware_software)

SUBTYPE OF (product);

END_ENTITY;

ENTITY system_user

SUBTYPE OF (person);

system : hardware_software;

access_level : level;

user_id : text;

password : text;

END_ENTITY;

ENTITY test_specification

SUBTYPE OF (pca_specification);

spec_identifier : STRING;

revision : STRING;

name : STRING;

END_ENTITY;

ENTITY time_record;

time_field_0 : two_bite_bf;

time_field_1 : local_time;

END_ENTITY;

ENTITY two_bite_bf;

END_ENTITY;

```
ENTITY type_id;  
type_id_field_0 : two_bite_bf;  
type_id_field_1 : type_layup;  
type_id_field_2 : cad_prog_name;  
END_ENTITY;
```

```
ENTITY type_layup;  
END_ENTITY;
```

```
ENTITY upper_right_x_coord_rect_pad;  
END_ENTITY;
```

```
ENTITY upper_right_y_coord_rect_pad;  
END_ENTITY;
```

```
ENTITY ur_x_cord;  
END_ENTITY;
```

```
ENTITY ur_y_cord;  
END_ENTITY;
```

```
ENTITY vapes;  
rectangular_pad_records : LIST [1:?] OF rectangular_pads;  
end_of_sub_file_4_record : end_of_sub_file;  
END_ENTITY;
```

```
ENTITY week_of_year_and_day_date  
SUBTYPE OF (date);
```



```
week_component : week_in_year_number;  
day_component : OPTIONAL day_in_week_number;  
END_ENTITY;
```

```
ENTITY width_of_line;  
END_ENTITY;
```

```
ENTITY width_of_pad;  
END_ENTITY;
```

```
ENTITY x_component_center;  
END_ENTITY;
```

```
ENTITY x_hole_center;  
END_ENTITY;
```

```
ENTITY x_pad_center;  
END_ENTITY;
```

```
ENTITY x_point_on_line;  
END_ENTITY;
```

```
ENTITY y_component_center;  
END_ENTITY;
```

```
ENTITY y_hole_center;  
END_ENTITY;
```

```
ENTITY y_pad_center;
```

END_ENTITY;

ENTITY y_point_on_line;

END_ENTITY;

END_SCHEMA;

A.2 RASSP Enterprise Data Model EXPRESS-G

The EXPRESS-G diagrams for the RASSP Enterprise Data Model are shown in the following pages. Table A.1 shows the position of each page in order to assemble the REDM as a single diagram.

109	111	113	115	117
119	121	123	125	127
129	131	133	135	137
139	141	143	145	147
149	151	153	155	157

Table A.1 - RASSP Enterprise Data Model Page Positions

Annex B Bibliography

ISO 10303-11 *Industrial automation - Product data representation and exchange - Part 11: The EXPRESS Language Reference Manual.*

ISO/DIS 10303-41 *Industrial automation systems and integration - Product data representation and exchange: Integrated generic resources: Fundamentals of product description and support.*

ISO/DIS 10303-44 Industrial automation systems and integration - Product data representation and exchange: Integrated generic resources: Product structure configuration.

ISO/WD 10303-49 Industrial automation systems and integration - Product data representation and exchange: Integrated generic resources: Process plan structure.

MMC-RASSP-2.01.00 STEP Configuration Management Suitability Report.

MMC-RASSP-3.01.00 RASSP PCA Manufacturing Interface Definition.

MMC-RASSP-4.01.00 Proposed Parts Taxonomy for RASSP Manufacturing Library Information.

NA-94-1387 Rapid Prototyping of Application Specific Signal Processors (RASSP) Build 0 Information Model Report.

NA-94-1621 Rapid Prototyping of Application Specific Signal Processors (RASSP) Build 1 Application Reference Model Report.

NA-94-1622 Rapid Prototyping of Application Specific Signal Processors (RASSP) Build 1 Enterprise Data Model Report.

NA-94-1623 Rapid Prototyping of Application Specific Signal Processors (RASSP) Build 1 Application Interpreted Model Report.

usage_occurrence))

thod))

ating_product_definition,SELF\product_definition_relationship.related_product_def.

inition;

