# Mapping From Manufacturing Information Systems (MIS) to CIS/2

Frank Wang, Chuck Eastman

## Basic Rules and Principles of General Items Mapping

***1.Unique ID***

If a MIS object requires a global unique identifier, the recommendation is to use the DMC identifier as its unique ID. In order to use a DMC identifier, the application must be a DMC (Data Management Conformant) application, which capable to make an instance unique among files. A DMC identifier is composed by three different identifiers, which are an application identifier, an installation identifier and an object identifier.

For exporting from non-DMC applications, the assigned identifier of objects must be uniquely at least in a same model. The exporting translator may assign an explicit *item_property* to an object as a unique identifier.

***2. Member ID***

A regular member identification, which is a unique identifier within the schema, will use the predefined integer identifier from a *structural_frame_item* entity.

***3. Mapping a mark item***

Although the CIS/2 LPM500 does not provide all type of marks that are demanded form the Fabtrol's system, there are still various ways to map a customize mark item within the CIS/2 schema. A suggestion is to define an additional ***item_property_assigned*** instance that attaches onto a descriptive structural frame item and uses it as an explicit mark attribute.

***4. A key part of an assembly***

A user can articulate a key part of an assembly by defining a ***structural_frame_item_relationship*** instance to create explicit relationship between structural items. Other issues about the hierarchies of assemblies are discussed in Dave Bartliff's report, "*Use of CIS/2 For Manufacturing Information System*s".

**5.** ***Mapping a Drawing***

a. Mapping a normal drawing

For each individual drawing, a user must create an associated ***group_assignment*** instance to collect the content of a drawing. The *group_assignment* entity refers to a *group* item, which inherits a ***media_file*** entity to represent a drawing. According to the CIS/2 schema definition, a *media_file* item provides predefined fields for a user to classify a drawing.

b. A revision of a drawing

When a drawing has been revised, it is recommended to make the revision *action* item and associated with a defined group. An *action* is a generic instance that is defined in Part 41, which allows a user to describe customized operations or activities as an action.

**Legends**

**A:**

1. 'O'- MIS object can map directly into CIS/2 without explicit item assigned.
2. '? '- MIS object can map into CIS/2 with explicit item assigned.
3. 'X'- MIS object cannot map into CIS/2, agreement must be made between exporting and importing translators.

***B:***

1. '**A**'- An explicit item_property _assigned is required.
2. '**I**'- An explicit item_property is required (non-structural frame item).
3. '**G'-** A group_assignment and group item is required.
4. '**R**'- An item_reference is required.
5. '**P**'- A structural_frame_item_priced is required.
6. '**V**'- A version_action_request item is required.

**Select a measure value and unit type for an item_property_assigned**

It is important to have an agreement on unit types for each item_property that are used for mapping MIS object to CIS/2. An *item_property* points to a *measure_with_units* item that requires a unit type and a measure value. Exporting translator must recognize the required unit standard on the importing translator. If unit types are differently used in the importing and exporting system, the conversion must be taken either on the importing side or the exporting side.

**Define a flavour for joint parts**

Joint parts such as bolts, nuts and washers should be defined explicitly in specific flavour files, so all parts of a joint will have a consistent naming standard.

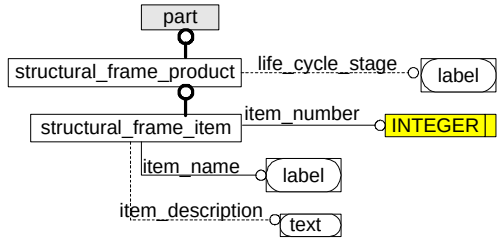
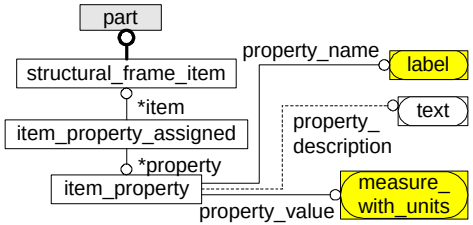
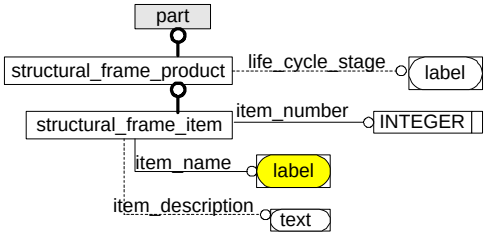
**Define a MIS flavour file**

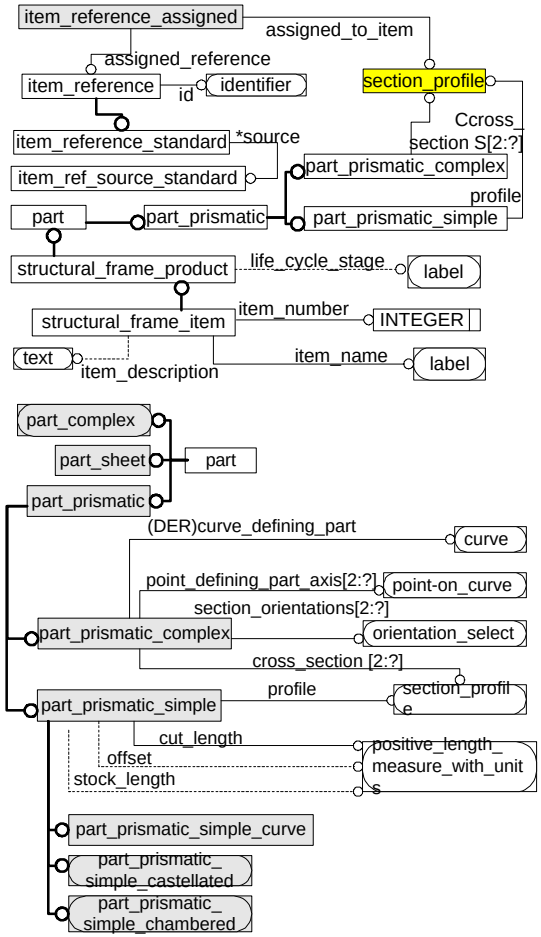
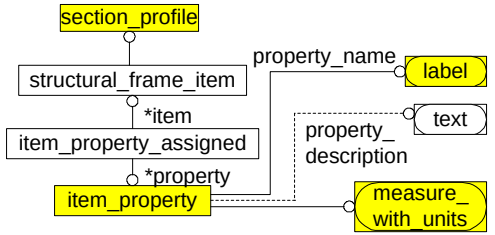
It is necessary to integrate many individual MIS related standards together as an MIS flavour file.

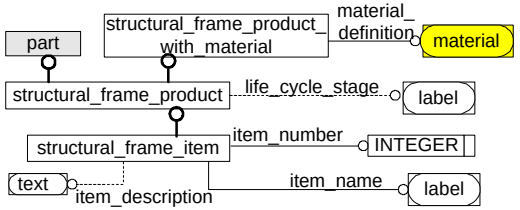
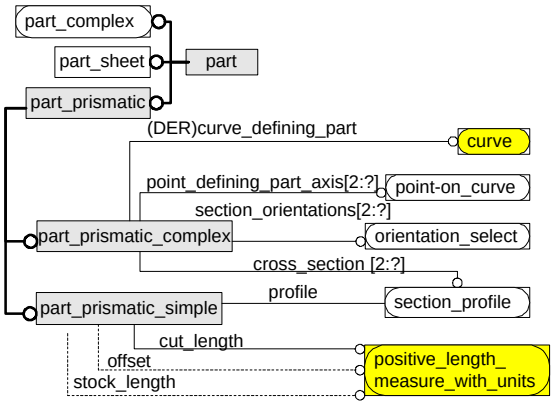
**Classification of assemblies**

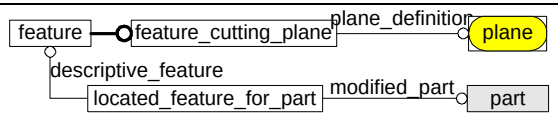
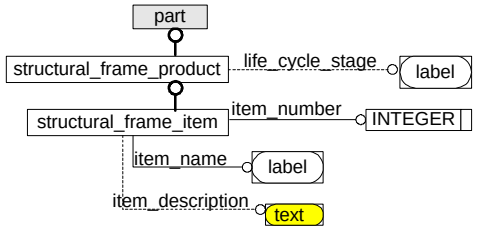
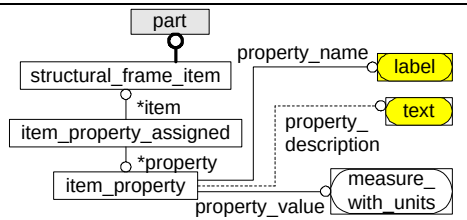
A classification standard or convention must be made to verify a type of an assembly, which is generic in CIS/2. In MIS data definition, there are many types of assembly, for example a bolt assembly, a bolt assembly group, a super assembly, an assembly and others. The classification for an assembly type must be defined on an agreement whether using the name field or the description filed for a *structural_fram_item* for non-DMC applications importing and exporting.

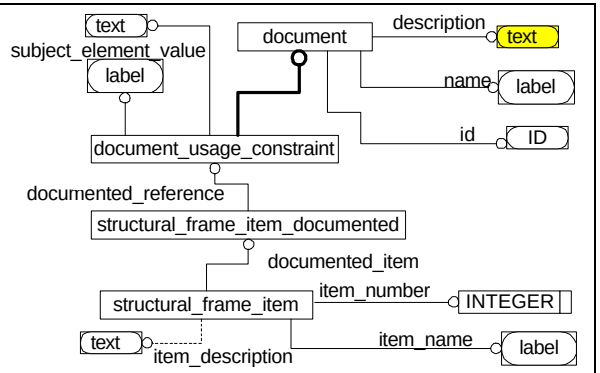
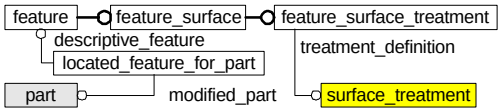
*Table one – Mapping methods between MIS and CIS/2*

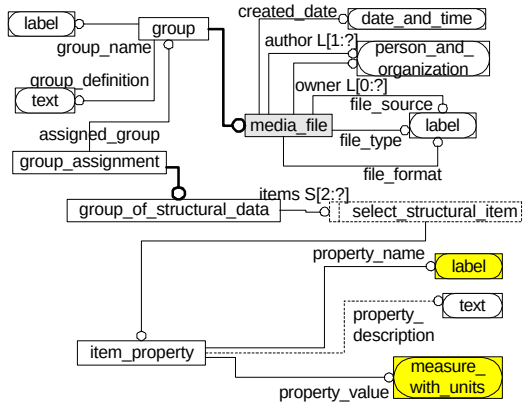
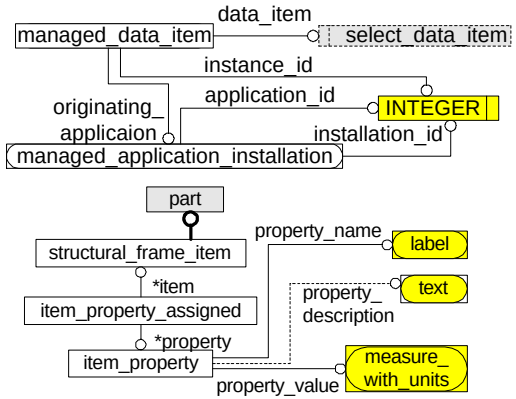
| No. | MIS Item Name | A | B | Mapping to CIS/2 Schema | C | EXPRESS-G DIAGRAM |
|---|---|---|---|---|---|---|
| **1.0** | **Part Details** | | | | | |
| 1.1 | Part ID | O | | For all regular item IDs: a direct mapping method is available with using the *item_id* field from a *structural_frame_item* instance.<br>`structural_frame_item(item_id: integer)` | |  |
| 1.2 | Part Mark | ? | A | To define extra information for a *structural_frame_item*, a user can create an *item_property_assigned* instance that attaches a customized attribute onto a descriptive item.<br>`item_property_assigned (property(item_property(property_name:label, OPTIONAL property_description: text, property_value: measure_with_uint)))` | |  |
| 1.3 | Part Name | O | | Use the *item_name* field in the *structural_frame_item* instance.<br>`structural_frame_item(OPTIONAL item_name: label)` | |  |

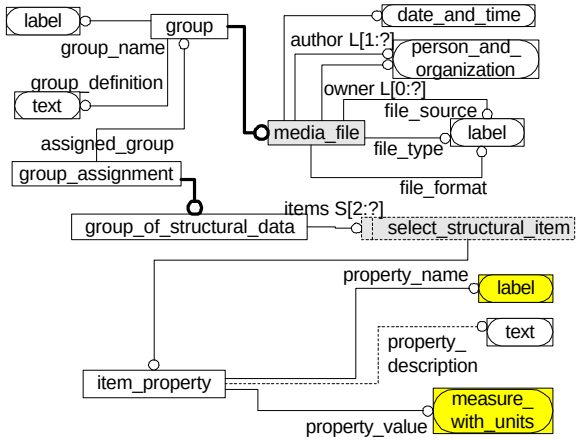
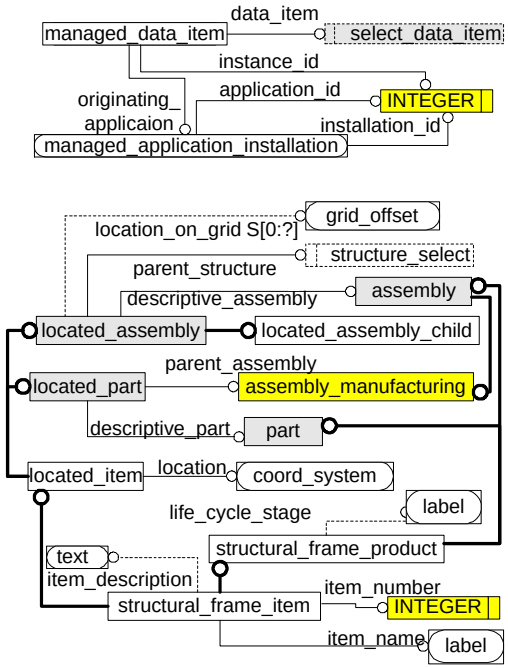
| | | | | | | |
|---|---|---|---|---|---|---|
| 1.4 | Part Section Type | ? | R | There are two ways to map a section profile of a part into CIS/2. The first way is to use the *item_reference_assigned* item, which will assign to an explicit *item_reference*. The identifier of the *item_reference* will indicate the name of the section profile for a part.<br><br>Normally, if a part is not a taped or complex part, the part can be defined as a *part_prismatic_simple_curved* (this was used by Berry Butler from Design Data). | * |  |
| 1.5 | Part Section Size | ? | A | **1. Define a part section size by using *item_property_assigned***<br>A size of a section profile is not defined implicitly in CIS/2 schema. To accommodate the size of the section profile, exporting application can use item_property_assigned to indicate the area of a profile section.<br><br>**2. Importing translator verifies the section profile size by using an internal mapping reference**<br>If the given section profile is using an | |  |

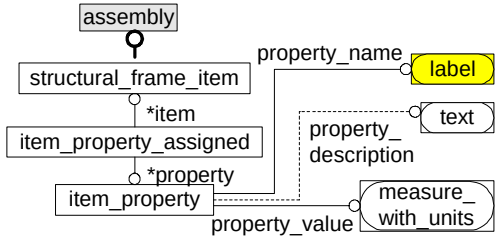
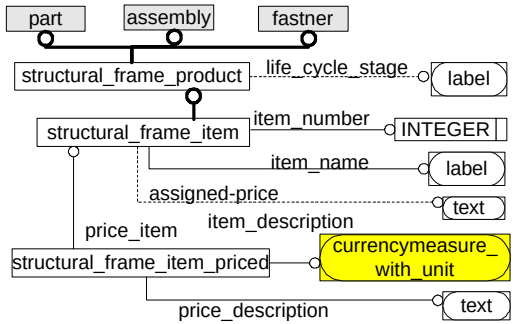
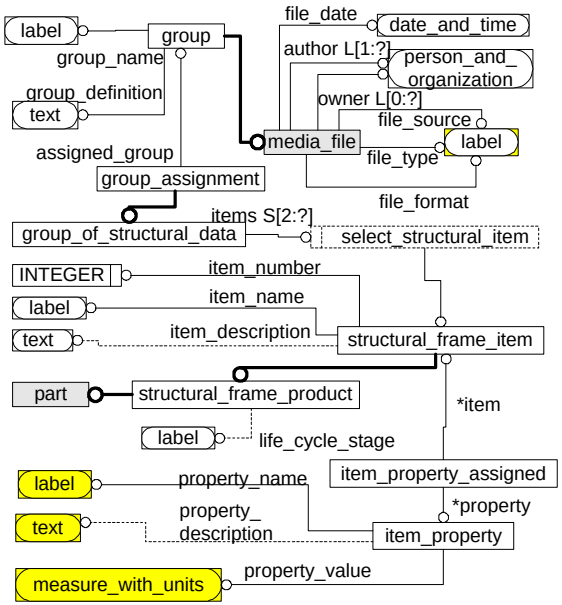
| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | *item_reference_assigned* item to carry the section profile type, Fabtrol or other translators who are using the MIS item standard may use an internal indexing document to bridge the section profile with a corresponding area value. | | |
| 1.6 | Part Material Grade | O ? | A | **1. Define the material grade by using the *structural_frame_product_with _material***<br>Most applications will map a Material Grade item with using an explicit *item_reference* that indicates the steel grade information.<br>If a steel grade is not mapped to an external reference document then it needs to be defined as a material of a steel member. However, to define the part material grade by using the structural_frame_product_with_material will require an exporting translator to write out more instances and the complexity will be higher.<br><br>**2. Write out a material grade item by using a *item_porperty_assigned* item**<br>If the material grade is defined explicitly as an item reference, the material grade can be defined as a item_property_assigned to additionally indicate the material grade to a descriptive part.<br><br>**3. Define material grade with using an item_reference_assigned**<br>It will be more continent to define a material grade referencing to an explicit material grade standard. | * |  |
| 1.7 | Part Length | O | | The Part Length is the rough cut for a descriptive part in MIS. A direct mapping is available for mapping a Part Length in CIS/2 by using the *cut_length* of a *part_prismatic_simple*.<br>The unit type must also be pass down to the importing translator.<br><br>`structural_frame_item(part(part_prismatic(part_prismatic_simple(cut_length: length_measure_with_units))))` | |  |

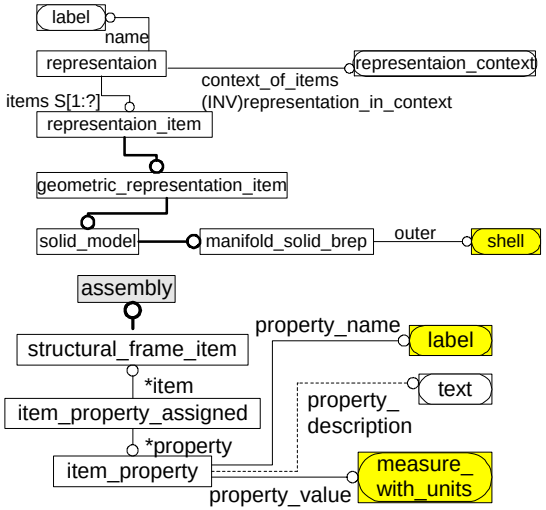
| | | | | | | |
|---|---|---|---|---|---|---|
| 1.8 | Part End 1 angle Y | O | | **1.Data provided by the exporting translator**<br>The exporting translator can export a calculated angle value by using an explicit *item_property_assigned* item to indicate the angle of a cutting plane.<br><br>**2. Data derived from the importing translator**<br>A Part End Angle is defined as a feature in CIS/2, which is a *feature_cutting_plane* entity. A directly mapping is available for a Part End to CIS/2.<br><br>The import translator must calculate the two normal or directions between the part and the cutting plane to derive the end angle of a part.<br><br>`structural_frame_item(feature(feature_cutting_plane: plane))^structural_frame_item(part)` | |  |
| 1.9 | Part End 1 angle Z | O | | See 1.8. | | See 1.8. |
| 1.10 | Part End 2 angle Y | O | | See 1.8 | | See 1.8. |
| 1.11 | Part End 2 angle Z | O | | See 1.8 | | See 1.8. |
| 1.12 | Part Notes | O | | A user can map the Part Notes by using the *description_field* in the *structural_frame_item* attribute.<br><br>`structural_frame_item(OPTIONAL item_description: text )` | |  |
| 1.13 | Part User Defined Field 1 | ? | A | **1. Define the part user defined field by using item_property_assigned**<br>To define extra information for a *structural_frame_item*, a user can create an *item_property_assigned* instance that attaches onto a descriptive item. Since the importing side only is interested in a user defined text field, it is not sufficient to write out many instances that will not used anywhere in the receiving MIS translator.<br><br>**2. Define the part user defined field by using the *structural_frame_item_documented***<br>The *document_reference* attribute of a *structural_frame_item_documented* is a *document_usage_constraint*, which is a defined as a Part | |  |

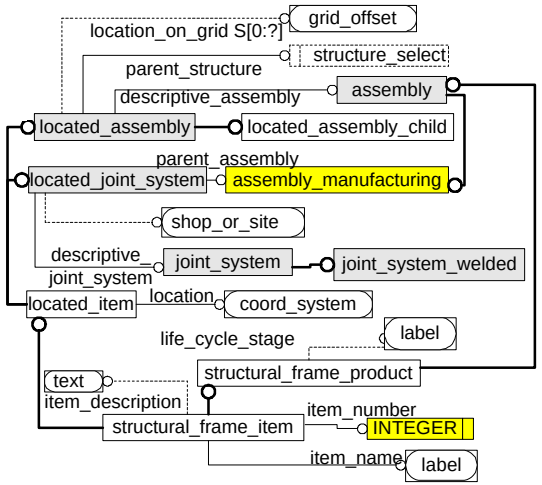
| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | 41 entity. A “part user defined field” in MIS can be defined as a document and as a source of a *document_usage_constraint* in CIS/2. The description field of a document is an text field where will become the user-defined filed for a part. | |  |
| 1.14 | Part User Defined Field 2 | ? | A | See 1.13 | | See 1.13 |
| 1.15 | Part User Defined Field 3 | ? | A | See 1.13 | | See 1.13 |
| 1.16 | Part Finish | O | | **1. Define a part finish by using the *surface_treatment*** A part finish is defined as a surface treatment in CIS/2.<br><br>`structural_frame_process(surface_treatment( surface_treatment_coat(methods: SELECT coating methods)))`<br><br>**2. Define a part finish by using an explicit *item_reference_assigned***<br>This method is similar to all standard references, which we can define a surface treatment flavour file. It is necessary to integrate many individual MIS related standards together as an MIS flavour file. | * |  |
| 1.17 | Part Drawing ID | | I | See 12.1 | | See 12.1 |

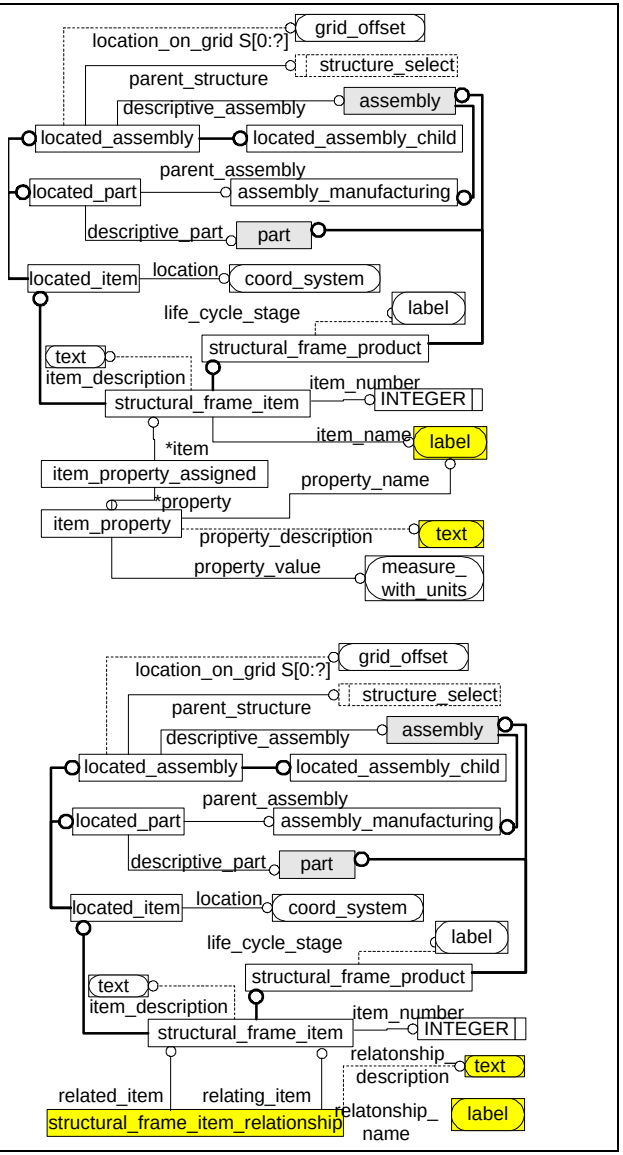
| | | | | | | |
|---|---|---|---|---|---|---|
| 1.18 | Part NC-DSTV ID | ? | I | A DSTV is a special type of a NC file. The mapping methods will be similar to a drawing mapping that are applying a *media_file* entity but making the *file_format* noted as "NC".<br>`media_file(file_format: label)`<br><br>*The item property is a structural item. All drawing IDs will use an item_property to assign a numerical number as its drawing ID. | * |  |
| **2.0** | **Part instance** | | | | | |
| 2.1 | Part Instance Unique ID | O ? | A | **1. For a DMC application**<br>If a MIS instance requires an UNIQUE ID, a suggestion is to use the *managed_data_item* identification as a uniquely ID.<br>`managed_data_item(data_id: integer)^`<br>`managed_application_installation(`<br>`application_id:integer,`<br>`installation_id:integer)`<br><br>**2. For a non-DMC application**<br>For non-DMC application: A translator can create an explicit *item_property_assigned* to a part and use the assigned value as its unique identifier. | * |  |
| 2.2 | Part ID | | | See 1.1 | | See 1.1 |

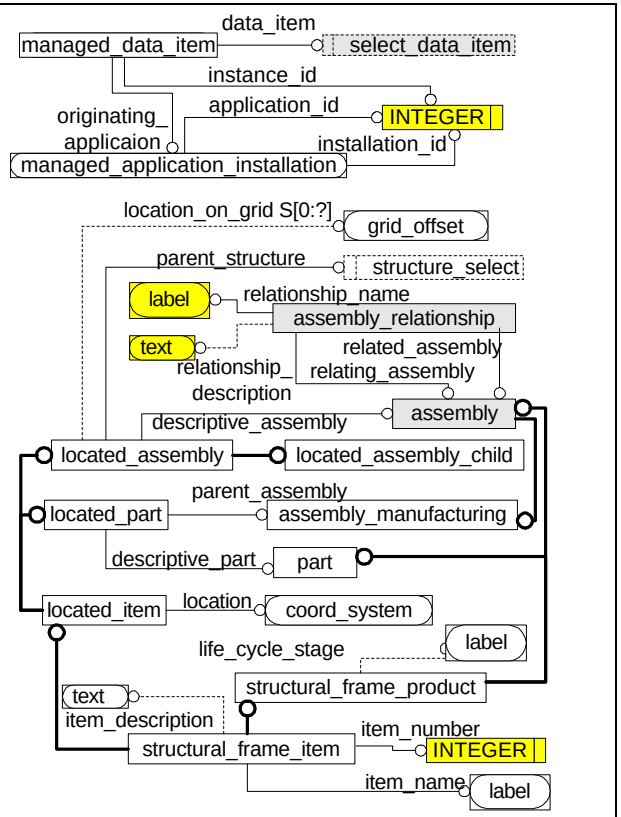
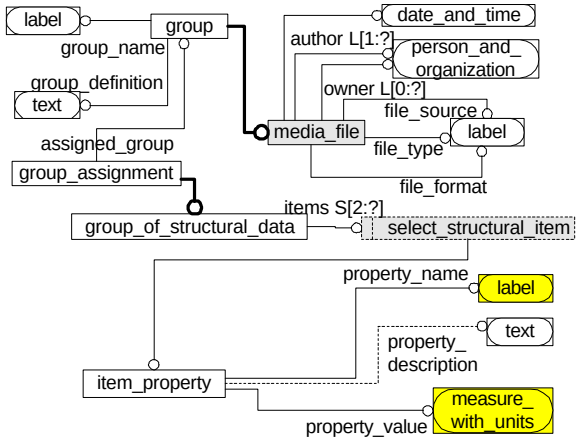
| | | | | | | |
|---|---|---|---|---|---|---|
| 2.3 | Part ABM Mark | ? | I | Assumed the mapping method for mapping a bill of material to CIS/2 is similar to a mapping a drawing (see 1.17, discussion is needed). A user will need to create a *group_assignment* that includes the content of an ABM. After the *group_assignment* is defined, a user needs to make an explicit *item_property_assigned* item to indicate the ABM mark, which is the similar mapping method to make a mark item (see 1.2).<br>To define extra information for a *structural_frame_item*, a user can create an *item_property_assigned* instance that attaches onto a descriptive item. The label field must classify that the item_property is used for an ABM mark label in the instance level.<br><br>`item_property_assigned(property(item_property(property_name:label, OPTIONAL property_description: text, property_value: measure_with_unit)))` | |  |
| 2.4 | Assembly Instance Unique ID *(parent)* | O<br>? | | **1. For a DMC application**<br>If a MIS instance needs an UNIQUE ID, a suggestion is to use the *managed_data_item* (DMC) identification as an ID.<br><br>`managed_data_item(data_id: integer)^ managed_application_installation( application_id:integer, installation_id:integer)`<br><br>**2. For a non-DMC application**<br>For non-DMC application: A translator can create an explicit *item_property_assigned* to an assembly and use the assigned value as its unique identifier. | *<br>1 |  |
| **3.0** | **Assembly Details** | | | | | |
| 3.1 | Assembly ID | O | | See 1.1 | | See 1.1 |

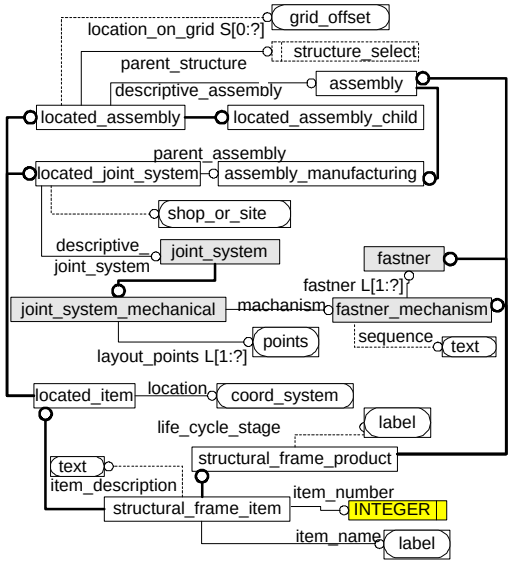
| | | | | | | |
|---|---|---|---|---|---|---|
| 3.2 | Assembly Mark | O | A | See 1.2 | | See 1.2 |
| 3.3 | Assembly Name | O | | See 1.3 | | See 1.3 |
| 3.4 | Assembly Type | ? | A | To define extra information for a *structural_frame_item*, a user can create an *item_property_assigned* instance that attaches onto a descriptive item. A type of an assembly is not implicitly defined in CIS/2.<br><br>`item_property_assigned(property(item_property(property_name:label, OPTIONAL property_description: text, property_value: measure_with_unit)))` | |  |
| 3.5 | Assembly Cost Code | O | P | For cost estimating items, a user can use a *structural_frame_item_priced* instance to marked down a price for a part or an assembly.<br>(*The total price of an assembly will not be able to derive from its related parts. This derivation must be done in another level mapping that is the instance file level mapping).<br><br>`structural_frame_item_priced(assigned_price : currency_measure_with_unit, price_item: structural_frame_item).` | * |  |
| 3.6 | Assembly BOQ category | ? | A | The category of a bill of quantity (BOQ) is a collection among all parts of an assembly. It can be represented as a group object in CIS/2, which is a similar way to define a drawing. | |  |

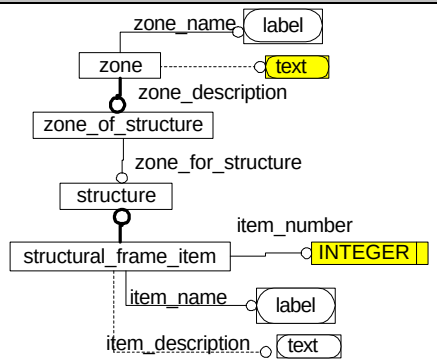
| | | | | | | |
|---|---|---|---|---|---|---|
| 3.7 | Assembly Gross Length * | X<br>? | A | There are two approaches for a B-rep calculation. The first approach is the exporting application must calculate the B-rep of an assembly and export it into an exporting file. A B-rep entity is available in CIS/2 schema since the CIS/2 includes the entire Part 42. The second approach is that the B-rep calculation will handle by the importing translator. The B-rep algorithm must be implemented in the translator.<br>The orientation of an assembly in a model space may not be the proper orientation for transporting purpose. A few transformations must apply onto the assembly to calculate the maximum gross length of the assembly. An application approach is more easily to accomplish the re-orientating of an assembly.<br><br>Exporting application may write out gross measurements of an assembly into an explicit *item_property_assigned* instance.<br><br>An exporting translator can create an explicit *item_property_assigned* for an assembly and use the assigned value to calculate the gross measurement in a particular dimension. | * |  |
| 3.8 | Assembly Gross Width * | X<br>? | A | See 3.7 | | See 3.7 |
| 3.9 | Assembly Gross depth * | X<br>? | A | See 3.7 | | See 3.7 |
| 3.10 | Assembly Drawing ID | ? | A | To define a drawing in CIS/2, please see 12.1.<br>A drawing is not *a structural_frame_item* in CIS/2 but a generic item. To assign an additional attribute to a generic item is to define an *item_property* instance that indicates the number of a drawing. Another solution is to regulate usages of all attributes for a *media_file* item so the number of a drawing can be accommodated within one of the data fields. | | See 12.1 |

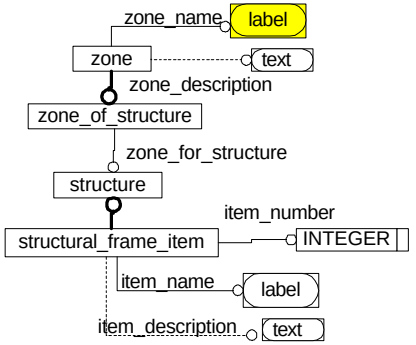
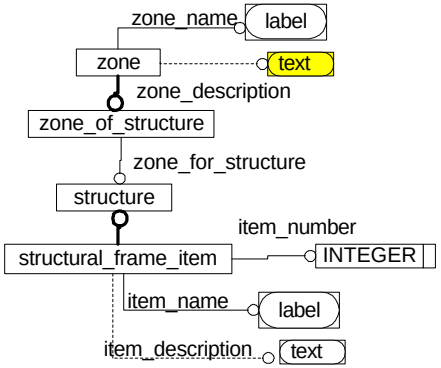
| | | | | | | |
|---|---|---|---|---|---|---|
| 3.11 | Weld ID [1] | O | | To define an identifier for a structural item, please see 1.1. A *shop_or_site* attribute can indicate whether a weld is operated on site or in shop from a *located_assembly*.<br><br>`located_joint_sysem item. located_joint_system(parent_assembly: assembly_manufacturing)` | |  |
| 3.13 | Weld ID [n] | O | | See 3.11 | | See 3.11 |
| **4.0** | **Assembly instance** | | | | | |
| 4.1 | Assembly Instance Unique ID | O ? | A | See 2.4 | | See 2.4 |
| 4.2 | Assembly ID | O | | See 3.1 | | See 1.1 |

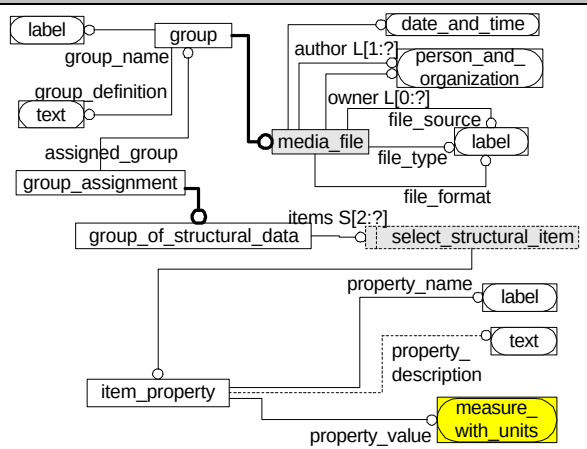
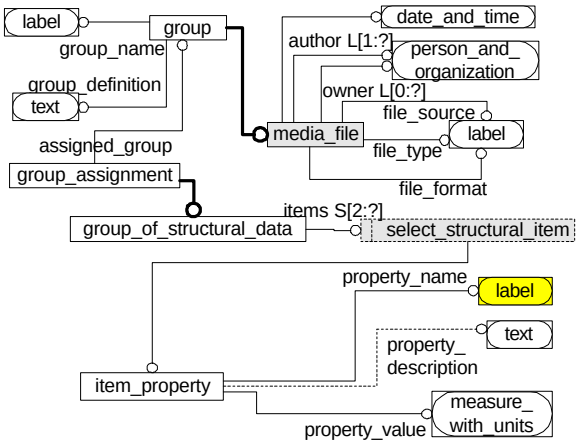
| | | | | | | |
|---|---|---|---|---|---|---|
| 4.3 | Assembly Main Part – Part Instance Unique ID | O | | **1. Using a *structural_frame_relationship* item to define a main part within an assembly**<br>Although the *design_assembly* includes a key member attribute that can directly identify a key member in an assembly, but the similar functionality is not available in the manufacturing model. However, a user can assign a *structural_frame_item_relationship* to identify a main part of an assembly (See 2.4).<br><br>`structural_frame_item_relationship(related_item: structural_frame_item ,relating_item: structural_frame_item,relationship_name:label, OPTIONAL relationship_description: text)`<br><br>**2. Define a main part ID by using an *item_property_assigned* item**<br>An exporting translator can generate an explicit *item_property_assigned* for an assembly and use the value filed to indicate the unique ID of its main part. | *<br>1 |  |

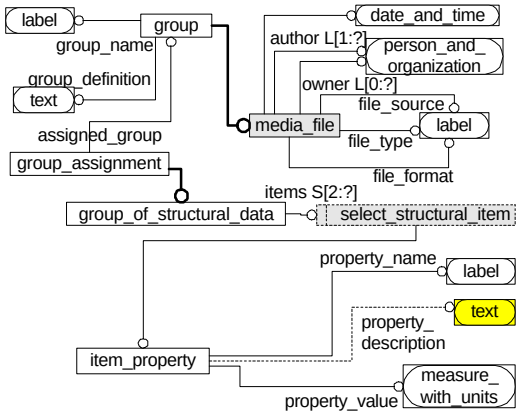
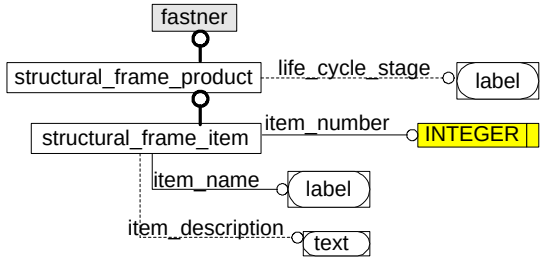
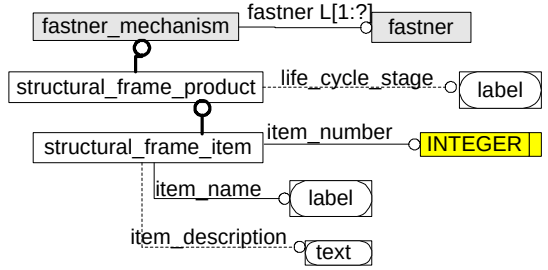
| | | | | | | |
|---|---|---|---|---|---|---|
| 4.4 | Super Assembly Instance Unique ID *(parent)* | O ? | A | **1. Using a *structural_frame_relationship* item to define a main assembly within a super assembly**<br>The CIS/2 schema provides an *assembly_relationship* field for users to define relationship between assemblies. This can be used to define a relationship between a super assembly and a child assembly.<br><br>`structural_frame_item_relationship(related_item: structural_frame_item ,relating_item: structural_frame_item,relationship_name:label, OPTIONAL relationship_description: text)` | * 1 |  |
| 4.5 | Assembly – Sequence ID | ? | I | **1. Definition based on 17.1 (Alternative 1)**<br>To define a sequence in MIS is similar to define a zone in CIS/2. To create a zone in CIS/2, a user needs to define a *structural* item to represent a sequence. The zone item will refer to a *structure* item, which is either as an assembly or a group of items. The ID of a sequence is using the *item_id* field of a *structural_fram_item* in a zone item (*see Sequence ID at 7.1 mapping alternative 1*).<br><br>`structural_frame_item(structure())`<br>`structureal_frame_item(zone(zone_of_structure(structure)))`<br>`structural_frame_item(item_id:integer)`<br><br>**2. Definition based on 17.1 (Alternative 2)**<br>Based on the second alternative of sequence definition | |  |

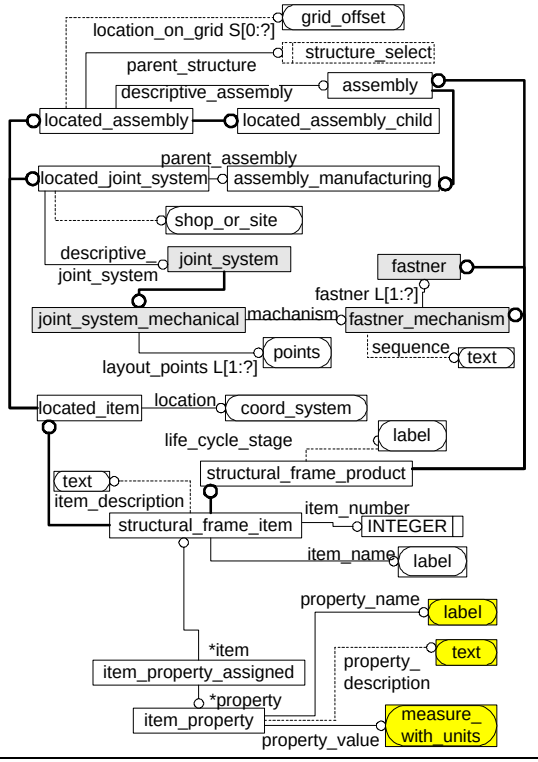
| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | (*see Sequence ID at 7.1 mapping alternative 2*) a zone will be grouped as a *group_assignment*. An explicit *item_property* will assign as an ID to the group. | | |
| 4.6 | Assembly – Lot ID | ? | I | **Using the *item_property_assigned* item to define an explicit ID field for indexing an Lot ID**<br>(See 8.1) | | |
| 4.7 | Bolt Assembly Group Instance ID [1] | O | | A bolt assembly is a joint system and it is assembled by many sub-pieces.<br>To define a regular item ID, see 1.2 | |  |
| 4.8 | Bolt Assembly Group Instance ID [n] | O | | See 4.7. | | See 4.7 |
| **5.0** | **Super-assembly** | | | | | |
| 5.1 | Super-assembly ID | O | | See 1.1 | | See 1.1 |
| 5.2 | Super-assembly Mark | ? | A | See 1.2 | | See 1.2 |
| 5.3 | Super-assembly Name | O | | See 1.3 | | See 1.3 |
| 5.4 | Super-assembly Type | ? | | See 3.4 | | See 3.4 |
| 5.5 | Weld ID [1] | O | | For all regular item IDs: a direct mapping is available with using the *item_id* field from a *structural_frame_item* instance.<br>*structural_frame_item(item_id:integer)*<br>A *shop_or_site* attribute can indicate whether a weld must be operated on site or in shop from a *located_assembly* instance. | | See 3.11 |

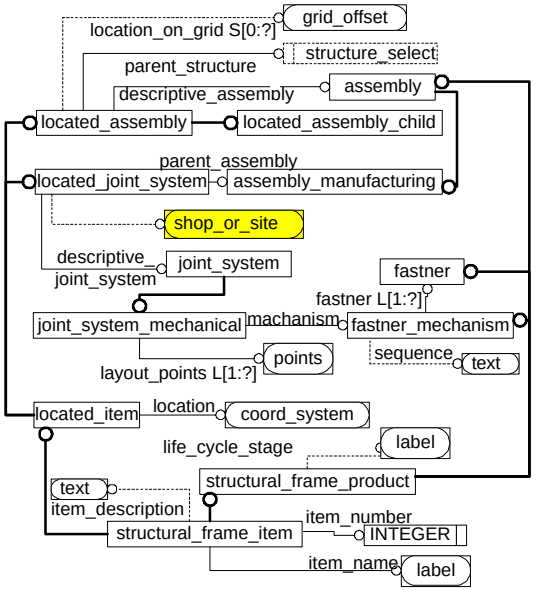
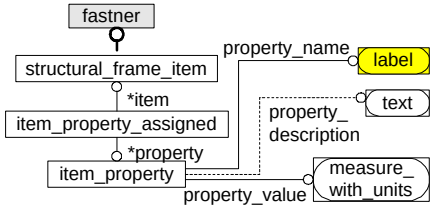
| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | `located_joint_sysem instance. located_joint_system(parent_assembly: assembly_manufacturing)` | | |
| 5.6 | Weld ID [n] | O | | See 5.5 | | See 3.11 |
| **6.0** | **Super-assembly instance** | | | | | |
| 6.1 | Super-assembly Instance Unique ID | O ? | A | See 4.4 | | See 4.4 |
| 6.2 | Super-assembly ID | O | | See 4.1 | | See 1.1 |
| 6.3 | Super-assembly Instance Unique ID *(parent)* | O ? | A | See 4.4 | | See 4.4 |
| **7.0** | **Sequence** | | | | | |
| 7.1 | Sequence ID | ? | I | **1. Sharing a numerical identifier with the descriptive structure**<br>Since a Sequence is mapped into a Zone in CIS/2, there is no numerical identifier for a *structural_item* in CIS/2. For a non-DMC file exporting, when defining a zone member, for example a *zone_of_structure*, the zone member points to a structure as an attribute called *zone_for_structure*, which describes the descriptive structure for the zone member. The structure is a *structural_frame_item* in CIS/2, which has a numerical identifier filed, called *item_number*. In order to assign a numerical identifier for an structural member, which is not allowed to use *item_property* in this case, one suggestion is to share the same numerical identifier with the descriptive structure item. It means that every declared *structure_frame_item* in a model must have a unique ID and every *zone* member, including its subtype member, must only refer to an individual structure. Therefore, the zone member (*structural_item*) will have a unique ID to represent a unique Sequence ID in MIS.<br><br>**2. Declared zone as a *group_assignment***<br>Another suggestion is to assign the declared zone as a *group_assignment* item. The *group_assignment* will include an explicit *item_property* within the aggregated item set. The assigned item_property will provide an identifier to the zone item. A disadvantage of this alternative is the importing translator needed to process a | |  |

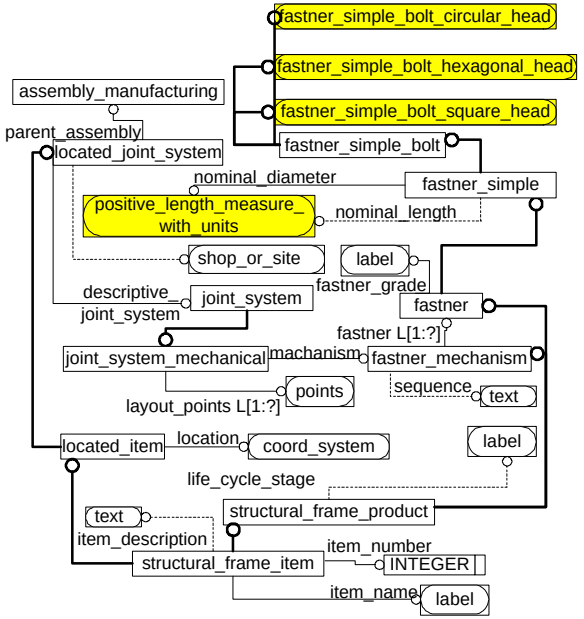
| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | searching through the aggregated data and find out the corresponding *item_property*, a *group_assignment* may have multiple *item_property* items included. | | |
| 7.2 | Sequence Name | ? | I | The name of a zone item is a direct accessible attribute in CIS/2, which is the *zone_name* attribute of a zone member.<br><br>`zone(zone:name(zone_of_structure(zone_for_structure: structure))` | |  |
| 7.3 | Sequence Description | ? | I | Same as item 7.2, the description filed will use for recording the MIS sequence description.<br><br>`zone(zone:description(zone_of_structure(zone_for_structure: structure))` | |  |

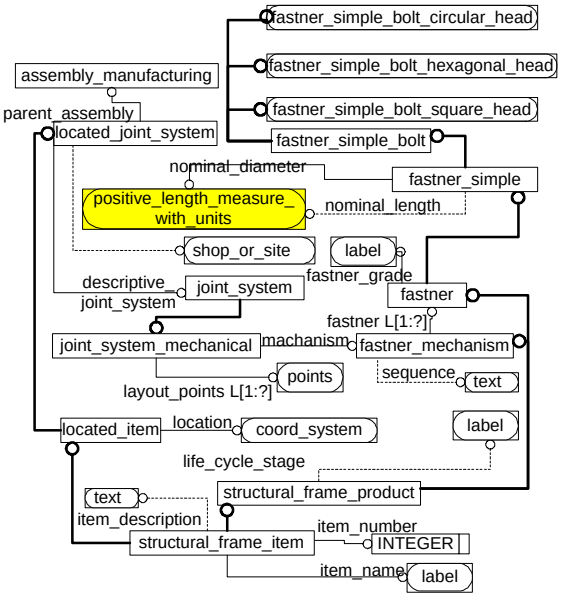
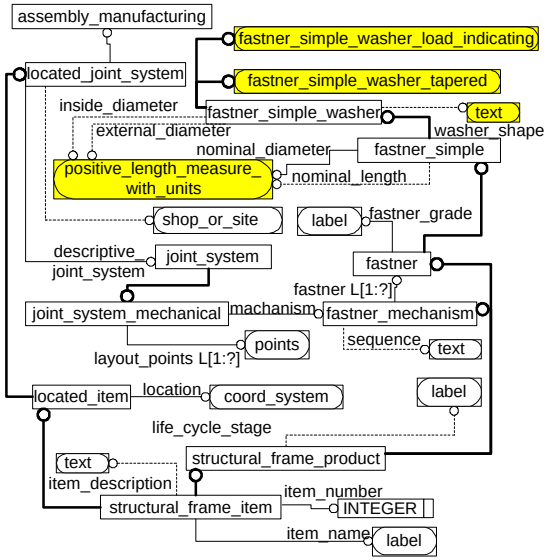
| **8.0** | **Lot** | | | | | |
|---|---|---|---|---|---|---|
| 8.1 | Lot ID | ? | I | A lot will be defined as a *group_assignment* in CIS/2. A *group_of_structural_data* is a subtype of *group_assignment* that collects several *structural_frame_items* together. The *group_of_structural_data* will refer to a *group*. The group item does not have any predefined ID fields to assign an identifier to a group item. Therefore, a user needs to use either an explicit *item_property* that will assign to a group item to identify an instance or the DMC identifier, if the exporting application is a DMC application.<br><br>The label and text field will be used to classify whether the group is a group of a lot item or a group used for other purposes.<br><br>Since a lot is represented as a *group_assignment*, it is necessary to assign an explicit *item_property* to the *group_assignment* to indicate the ID of a non-structural frame item.<br><br>`item_propery(property_name:label, property_description: OPTIONAL text,property_value: measure_with_unit)` | |  |
| 8.2 | Lot Name | ? | I | The *group_name* field of a group will use for labeling a lot name.<br><br>`group_of_structural_data(assigned_group:group(group_name: label))` | |  |

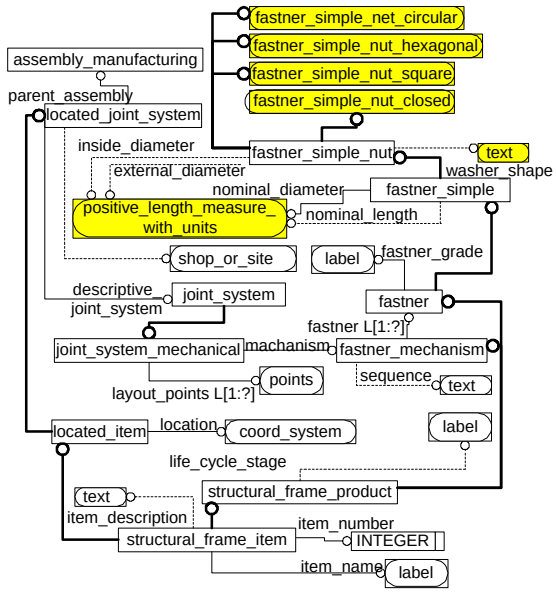
| | | | | | | |
|---|---|---|---|---|---|---|
| 8.3 | Lot Description | ? | I | The descriptive text filed will be used to represent the sequence description of a lot description.<br><br>Since a lot can map into a group in CIS/2. The group provides a predefined name field.<br><br>`group_of_structural_data(assigned_group:group(OPTIONAL group_description:text))` | |  |
| **9.0** | **Bolt Assembly Group Instance** | | | | | |
| 9.1 | Bolt Assembly Group Instance ID | O | | For all regular item IDs: a directly mapping is available with using the *item_id* field from a *structural_frame_item* instance.<br>To distinct a bolt assembly with other assemblies, it may be possible to use the *item_name* field to classify a bolt assembly.<br><br>*structural_frame_item(item_id:integer)* | |  |
| 9.2 | Bolt Assembly ID | O | | For all regular item IDs: a directly mapping is available with using the *item_id* field from a *structural_frame_item* instance.<br>*structural_frame_item(item_id:integer)* | |  |

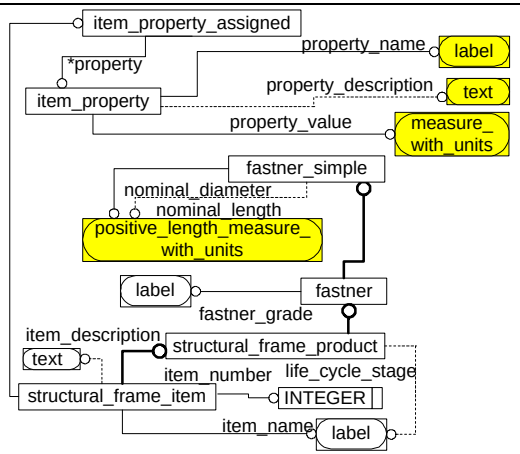
| | | | | | | |
|---|---|---|---|---|---|---|
| 9.3 | Bolt Assembly quantity | ? X | A | The importing translator must calculate the total quantity of bolts in an assembly in an instance level.<br><br>Question:<br>If the quantity means the quantity of a single assembly, then it is possible to assign an explicit *item_property_assigned* to carry the quantity value. | * |  |

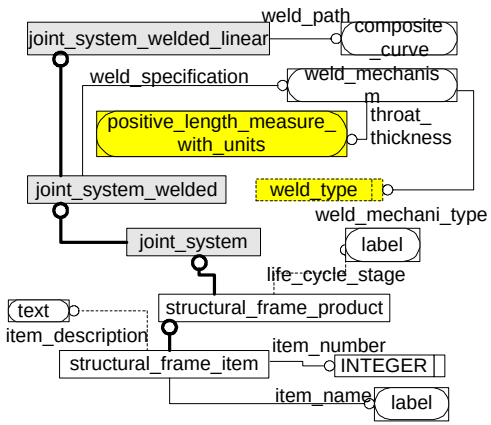
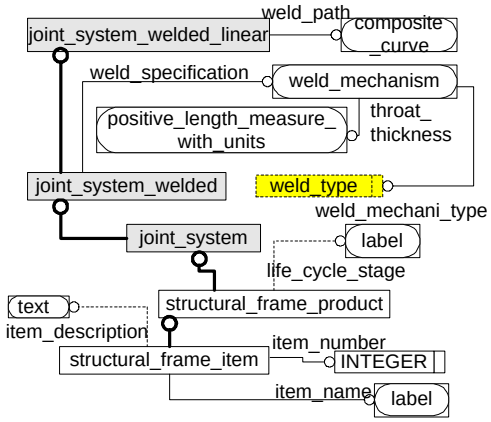
| | | | | | | |
|---|---|---|---|---|---|---|
| 9.4 | Bolt Assembly Shop_or_Site | O | | This information is available from the *located_joint_system* of a bolt assembly<br><br>`located_joint_system(OPTIONAL place of assembly: shop_or_site )` | |  |
| **10.0** | **Bolt Assembly** | | | | | |
| 10.1 | Bolt Assembly ID | O | | For all regular item IDs: a directly mapping is available with using the *item_id* field from a *structural_frame_item* instance.<br><br>`structural_frame_item(item_id:integer)` | |  |

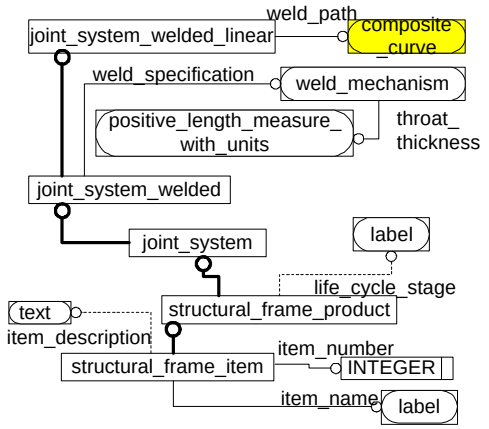
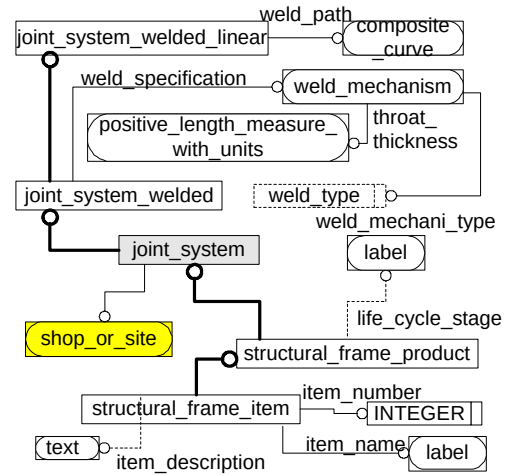
| | | | | | | |
|---|---|---|---|---|---|---|
| 10.2 | Bolt Type&Size | ? | A | **1. Defines a bolt type implicitly**<br>If a bolt is defined as using an *item_reference* for its specification then the name-and-type of a bolt will present as a name label for the *structural_frame_item.*<br>If the information is defined explicitly, there are two additional information must be given to define a type and size of a bolt, which are a descriptive bolt type and the diameter of the bolt (not the diameter of the head). A bolt type is defined as a name type in CIS/2.<br><br>`Type:`<br>`fastener_simple(OPTIONAL nonminal_length: positive_length_measure_with_units(fastener_simple_bolt(fastener_simple_bolt_`**`square*`**`_head())))  (*three types: square,hexagonal and circle)`<br>`Size:`<br>`fastener_simple(nonminal_diameter:positive_length_measure_with_units)`<br><br>**2. Assign an explicit item_property to indicate the bolt type**<br>An alternative is to assign an *item_property_assigned* to a fastener item. The assigned property will indicate the type the bolt.<br><br>**3. Define a bolt type by referenced a bolt flavor file**<br>A bolt type can be defined with using an *item_reference_assigned* item from a bolt flavour file. | * |  |

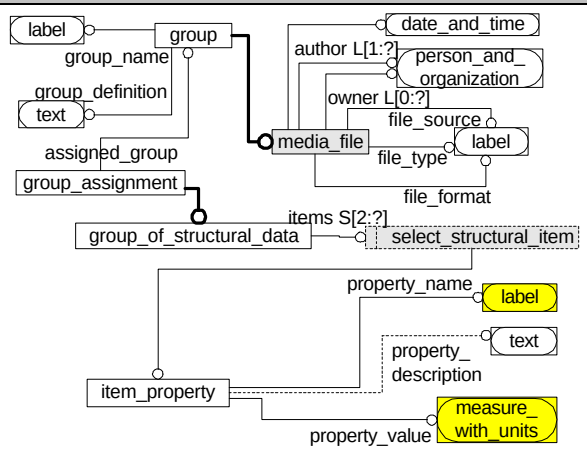
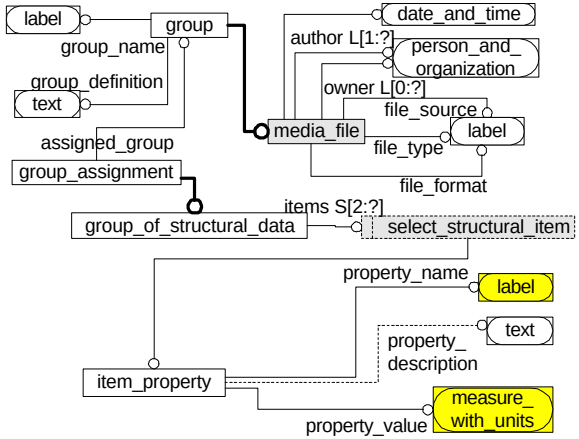
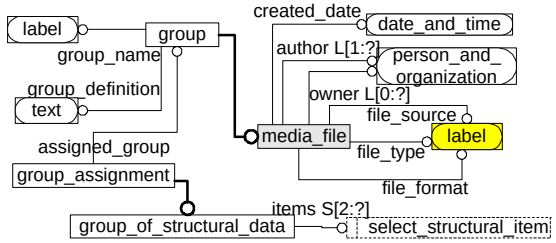
| | | | | | | |
|---|---|---|---|---|---|---|
| 10.3 | Bolt Length | O | | **1. Derived a bolt length internally**<br>The total length of a bolt is the nominal length plus the height of the head.<br><br>`fastener_simple(OPTIONAL nonminal_length: positive_length_measure_with_units(fastener_simple_bolt(fastener_simple_bolt_square_head(bolt_head_height:positive_length_measure_with_units))))`<br><br>**2. Derived a bolt length from a bolt flavour file**<br>The bolt length can be derived from a bolt flavour file based on the name of a bolt type. | |  |
| 10.4 | Washer Type&Size [1] | ? | A | **1. Define a washer type using CIS/2 entities**<br>If a washer is defined as using an *item_reference* for its specification then the name-and-type of a washer will present as a name label for the *structural_frame_item.* If the information is defined explicitly, there are two information must be given to define a type and size of a bolt, which are a descriptive washer type and the inner diameter of the washer. A washer type is defined as a name type in CIS/2.<br><br>`Type:`<br>`fastener_simple(OPTIONAL nonminal_length: positive_length_measure_with_units(fastener_simple_washer())`<br>`Size:`<br>`fastener_simple(fastener_simple_washer(inside_diameter:positive_length_measure_with_units)`<br><br>**3. Define a washer type by using an explicit item_proprty**<br>A washer or a nut is associated with a bolt item. The type of a nut or washer can be specified as an explicit | * |  |

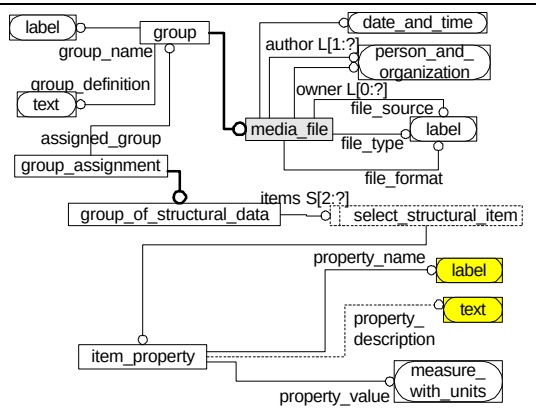
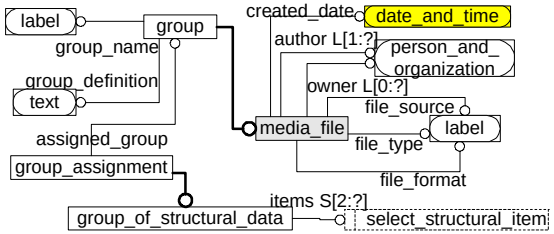
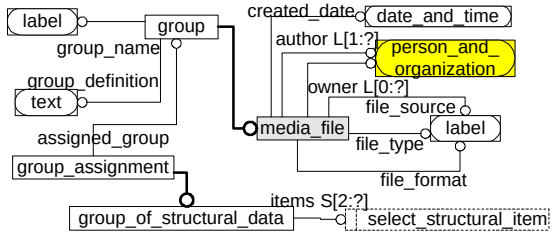
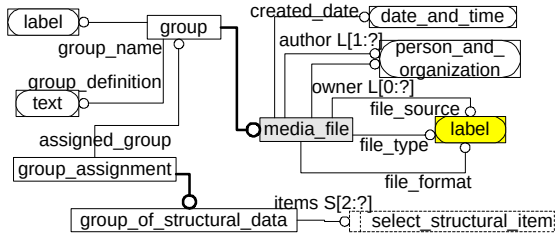
| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | item_property_assigned instead of retrieving a name of a type.<br><br>**3. Define a washer type from a washer flavour file**<br>The definition is similar to item 10.2, it is helpful to use a washer flavour file for retrieving washers information. | | |
| 10.5 | Washer Quantity [1] | X | | The estimated quantity of a particular washer can only be derived from an instances level. | | |
| 10.9 | Washer Type&Size [n] | ? | A | See 10.4 | * | See 10.4 |
| 10.10 | Washer Quantity [n] | X | | See 10.5 | | |
| 10.11 | Nut Type&Size [1] | ? | A | If a nut is defined as using an *item_reference* for its specification then the name-and-type of a nut will present as a name label for the *structural_frame_item.*<br>If the information is defined explicitly, there are two information must be given to define a type and size of a not, which are a descriptive nut type and the inner diameter of the nut. A nut type is defined as a name type in CIS/2.<br><br>`Type:`<br>`fastener_simple(fastener_simple_nut())`<br>`Size:`<br>`fastener_simple(nonminal_diameter:`<br>`positive_length_measure_with_units)` | * |  |

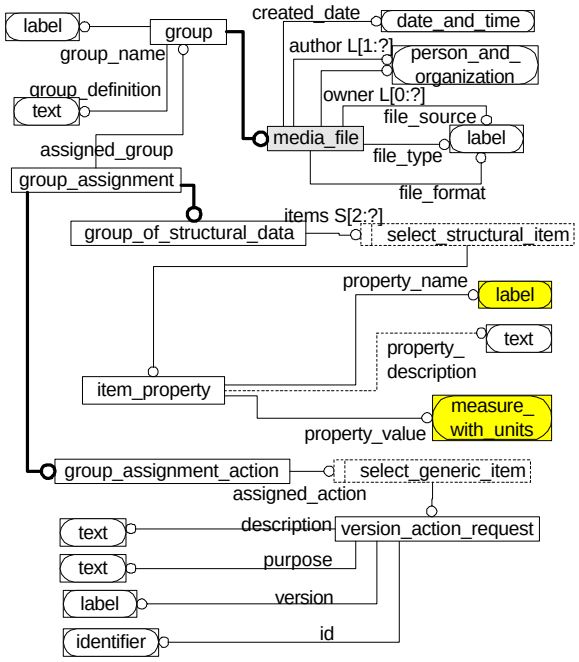
| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | |  |
| 10.12 | Nut Quantity [1] | X | | The estimated quantity of a particular type of nut can only be derived from an instances level. | * | |
| 10.13 | Nut Type&Size [n] | ? | A | See 10.11 | * | See 10.11 |
| 10.14 | Nut Quantity [n] | X | | See 10.12 | * | |
| **11.0** | **Weld** | | | | | |
| 11.1 | Weld ID | O | | For all regular item IDs: a directly mapping method is available with using the *item_id* field from a *structural_frame_item* instance.<br>*structural_frame_item(item_id:integer)*<br>A *shop_or_site* attribute can indicate whether a weld must be operated on site or in shop from a *located_assembly* instance.<br><br>A weld that belongs to an assembly is retrievable from the `located_joint_sysem instance.`<br>`located_joint_system(parent_assembly: assembly_manufacturing)` | | See 3.11 |

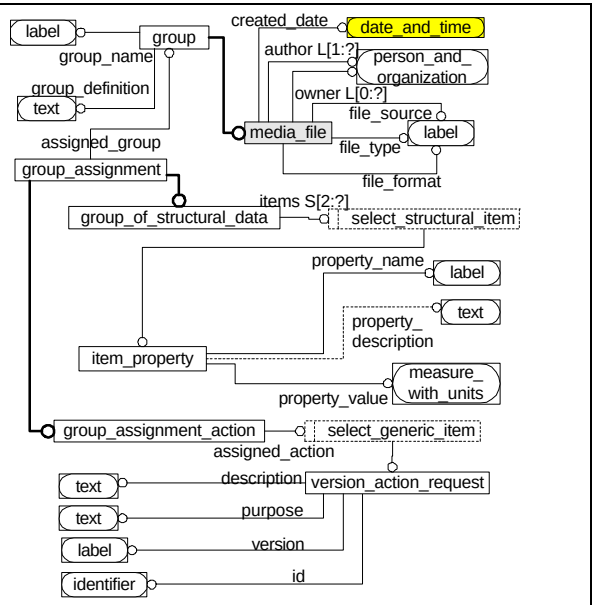
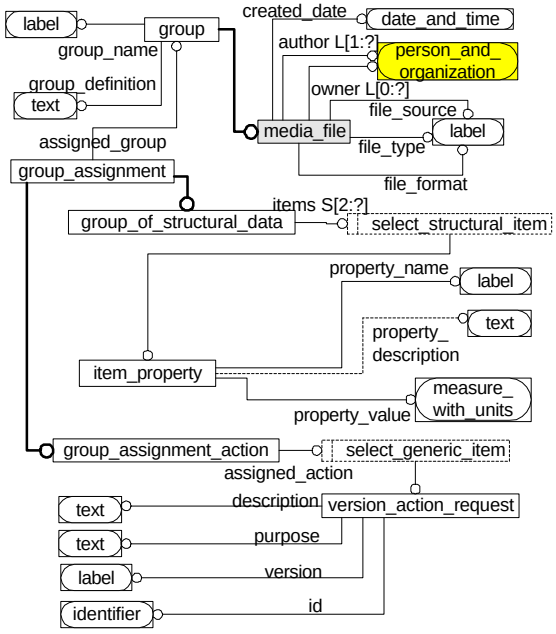
| | | | | | | |
|---|---|---|---|---|---|---|
| 11.2 | Weld Type | ?<br>O | A | Defined Implicitly:<br>A weld type in CIS/2 is defined as a *joint_system_welded* entity. A *joint_system_welded* will refer to a weld_ mechanism entity, which defines the type of the weld.<br><br>Defined Explicitly:<br>A weld type may be defined explicitly as an *item_reference* that refers to an external document. | |  |
| 11.3 | Weld Size | ? | A | The current CIS/2 does not allow a cross section area of a weld to de derived. Cross-section area of a weld must be given explicitly from an external reference. The thickness of a weld throat is available form the CIS/2 model that mans an explicit variable is demanded to derive the area. *Area=(variables × weld throat)*<br><br>The exporting application can export the size of a weld as an explicit item_property_assigned. | *<br>2 |  |

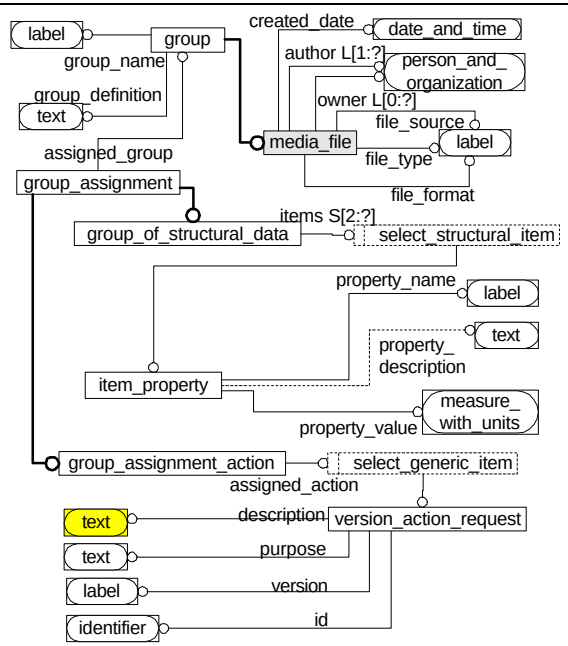
| | | | | | | |
|---|---|---|---|---|---|---|
| 11.4 | Weld Length | O ? | A | 1. The exporting translator exports curve elements.<br>The length of a weld is derivable from a linear geometrical line of a weld path. The estimation of the total length needs to be derived from length of individual line segments.<br><br>`structural_frame_item(joint_system(joint_system_weld(joint_system_weld_linear(weld_path: coposite_curve))))`<br><br>2. The exporting translator exports explicit attributes for a weld length.<br>The exporting translator exports a weld length as an explicit *item_property_assigned* data that indicates the length of a weld. | |  |
| 11.5 | Weld shop_or_site | O | | This value can be directly mapped by a *located_joint_system* of a weld. A *located_joint_system* has an attribute *place_of_assembly* and it points to *shop_or_site* selection type. | |  |

| 12.0 | **Assembly Drawing** | | | | | |
|---|---|---|---|---|---|---|
| 12.1 | Assembly Drawing ID | ? | I | 1. If exporting translator is a DMC translator.<br>If the exporting translator is a DMC translator, then it can export the drawing ID as the associated managed_data_item ID.<br><br>2. If exporting translator is not a DMC translator.<br>The content of a drawing is a *group_assignment* that aggregated many selected structural items together. A *group_assignment* points to a *group* item that inherits to a *media_file* item.<br>Since there is no relevant number attribute includes in a media file, the exporting translator can assign an *item_property* to the *group_assignment* object to represent the ID field of a drawing.<br><br>`group_assignment(group_of_structural_data(items S[1:?]: SELECT structural_item))(assigned_group:group(media_file())` | |  |
| 12.2 | Assembly Drawing Number | ? | I | Information of a drawing is predefined attributes of a *media_file* entity. A field to indicate the number of a drawing is missing from the predefined *media_file*'s attributes. A user may set up conventional field usage *(See Dave Bartliff's , Use of CIS/2 For Manufacturing Information Systems*) of how to use all predefined data fields from a *media_file* entity to cover a drawing number or create an additional *item_property* that carries a number as a drawing number.<br><br>`(item_property(property_name:label, OPTIONAL property_description: text, property_value: measure_with_unit))` | |  |
| 12.3 | Assembly Drawing Title | O | | A title of a drawing can be directly mapped into CIS/2 entity with using the *group_name*, a text field attribute of a group.<br><br>`group(group_name:text)` | |  |

| | | | | | | |
|---|---|---|---|---|---|---|
| 12.4 | Assembly Drawing Graphics Filename + Path | O ? | I | **1. Sharing the *file_source* field to indicate the file path**<br>Exporting translator may use the *file_source* field to store the path of a file.<br><br>**2. Define the file path by using the *item_property***<br>Exporting translator exports an explicit *item_property* that describes a path name of a file. | |  |
| 12.5 | Assembly Drawing Create Date | O | | The creation date of a drawing is available from the *file_date* attribute of a *media_file* entity.<br><br>`media_file(file_date: date_and_time)` | |  |
| 12.6 | Assembly Drawing Created By | O | | The owner and author of a drawing are available from the media entity.<br><br>`media(author L[1:?]: person_and_organization, owner L[0:?])` | |  |
| 12.7 | Assembly Drawing Size | O | | Assign an explicit property to a *structural_frame_item* by using *item_property_assigned*. The size of a drawing is described as the paper size of a drawing.<br><br>`item_property_assigned(property(item_property(property_name:label, OPTIONAL property_description: text, property_value: measure_with_unit )))` | |  |

| | | | | | | |
|---|---|---|---|---|---|---|
| 12.8 | Assembly Drawing Current Revision Mark | ? | A | **1. Using an *item_property* to identify a revision mark**<br>A direct way to map a revised drawing is to assign an explicit *item_property* with a customized value to the revised drawing item (described by Dave Bartliff).<br><br>**2. Using a *version_action_request* item to represent a revision**<br>A more official way has a more accurate logic of mapping a revised drawing but may increase the overall complicity of a mapping, which consider revision as an action applied onto a drawing. When grouping items together to form *a group_assignment* item, a user can includes an ANDOR type entity *group_assignment_actioned* with the *group_assignment*. The *group_assignment_actioned* allows user to select any generic items for explicit actions. A suggestion is to use the *version_action_request* to describe a revising of a drawing.<br>A user may use the *version_label* to describe a mark.<br><br>`(group_assignment(group_of_structural_data(item S[1:?]: SELECT structural_frame_item))(group_assignment_actioned(assigned_action: select_generic_item(version_action_request(id: identifier)))` | * 3 |  |

| | | | | | | |
|---|---|---|---|---|---|---|
| 12.9 | Assembly Drawing Current Revision Date | ? | V | A revision of a drawing is defined in item 12.8<br>1.Dect way to map the revision is to borrow a given attribute field to specify a revised date.<br>2. Use the date from the *file_date* attribute of *the group(media_file())* item, this group item is not the same group instance as the original group instance that represents the original drawing).<br><br>`media_file(file_date: date)`<br>`select_generic_item(version_action_request(`<br>`version: label))` | |  |
| 12.10 | Assembly Drawing Current Revision By | ? | V | A revision of a drawing is defined in item 12.8<br>The date will be available from the *file_date* of the revised media file (not the original one).<br><br>`mdia_file(author L[1:?]:`<br>`person_and_organization, owner L[0:?])` | |  |

| | | | | | | |
|---|---|---|---|---|---|---|
| 12.11 | Assembly Drawing Current Revision Notes | ? | V | A user can use the descriptive text field within the *version_action_request*<br><br>`revision_action_request entity. select_generic_item(version_action_request( description: text))` | |  |
| **13.0** | **Part Drawing** | | | | | |
| 13.1 | Part Drawing ID | | I | See 12.1 | | See 12.1 |
| 13.2 | Part Drawing Number | | I | See 12.2 | | See 12.2 |
| 13.3 | Part Drawing Title | O | | See 12.3 | | See 12.3 |
| 13.4 | Part Drawing Graphics Filename + Path | | I | See 12.4 | | See 12.4 |
| 13.5 | Part Drawing Create Date | O | | See 12.5 | | See 12.5 |
| 13.6 | Part Drawing Created By | O | | See 12.6 | | See 12.6 |
| 13.7 | Part Drawing Size | O | | See 12.7 | | See 12.7 |
| 13.8 | Part Drawing Current Revision Mark | | V<br>A | See 12.8 | | See 12.8 |
| 13.9 | Part Drawing Current Revision Date | | V | See 12.9 | | See 12.9 |
| 13.10 | Part Drawing Current Revision By | | V | See 12.10 | | See 12.10 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 13.11 | Part Drawing Current Revision Notes | | V | See 12.11 | | See 12.11 |
| **14.0** | **Erection Drawing** | | | | | |
| 14.1 | Erection Drawing ID | | I | See 12.1 | | See 12.1 |
| 14.2 | Erection Drawing Number | | I | See 12.2 | | See 12.2 |
| 14.3 | Erection Drawing Title | O | | See 12.3 | | See 12.3 |
| 14.4 | Erection Drawing Graphics Filename + Path | O | | See 12.4 | | See 12.4 |
| 14.5 | Erection Drawing Create Date | O | | See 12.5 | | See 12.5 |
| 14.6 | Erection Drawing Created By | O | | See 12.6 | | See 12.6 |
| 14.7 | Erection Drawing Size | O | | See 12.7 | | See 12.7 |
| 14.8 | Erection Drawing Current Revision Mark | ? | V | See 12.8 | | See 12.8 |
| 14.9 | Erection Drawing Current Revision Date | ? | V | See 12.9 | | See 12.9 |
| 14.10 | Erection Drawing Current Revision By | ? | V | See 12.10 | | See 12.10 |
| 14.11 | Erection Drawing Current Revision Notes | ? | V | See 12.11 | | See 12.11 |
| **15.0** | **NC-DSTV Filename** | | | | | |
| 15.1 | NC-DSTV file ID | | | See 12.1 | | See 12.1 |
| 15.2 | NC-DSTV filename+path | | | 12.4 | | |
| 15.3 | Part header | | | | ? | |
| 15.4 | Part contours | | | | ? | |
| 15.5 | Hole and PopMark date | | | | ? | |

**Table 1 MIS items mapping with using item_property**

| Item using item_property | | Value Type (SQL) | CIS/2 Item | measure | Unit |
|---|---|---|---|---|---|
| 1.2 | Part Mark ID | INTEGER | part_prismatic_simple | count_measure | context_dependent_unit |
| 1.5 | Part Section Size | REAL | part_prismatic_simple | area_measure | area_uint |
| 1.17 | Part Drawing ID | INTEGER | group_of_structural_data | count_measure | context_dependent_unit |
| 1.18 | Part NC-DSTV ID | INTEGER | group_of_structural_data | count_measure | context_dependent_unit |
| | | | | | |
| 2.3 | Part ABM Mark | TEXT | part_prismatic_simple | (name_field) | context_dependent_unit |
| | | | | | |
| 3.2 | Assembly Mark ID | INTEGER | assembly_manufacturing | count_measure | context_dependent_unit |
| 3.4 | Assembly Type | TEXT | assembly_manufacturing | (name_field) | context_dependent_unit |
| 3.6 | Assembly BOQ Category | TEXT | assembly_manufacturing | (description_field) | context_dependent_unit |
| | | | | | |
| 4.5 | Assembly-Sequence ID | INTEGER | group_of_structural_data | count_measure | context_dependent_unit |
| 4.6 | Assembly-Lot ID | INTEGER | group_of_structural_data | count_measure | context_dependent_unit |
| | | | | | |
| 5.2 | Supper-Assembly Mark | TEXT | assembly_manufacturing | (name_field) | context_dependent_unit |
| 5.4 | Super-Assembly Type | TEXT | assembly_manufacturing | (name_field) | context_dependent_unit |
| | | | | | |
| 7.1 | Sequence ID | INTEGER | group_of_structural_data | count_measure | context_dependent_unit |
| | | | | | |
| 8.1 | Lot ID | INTEGER | group_of_structural_data | count_measure | context_dependent_unit |
| | | | | | |
| 12.1 | Assembly Drawing ID | INTEGER | group_of_structural_data | count_measure | context_dependent_unit |
| 12.2 | Assembly Drawing Number | TEXT | group_of_structural_data | (name_field) | context_dependent_unit |
| 12.4 | Assembly Drawing Graphic Filename + Path | TEXT | group_of_structural_data | (description_field) | context_dependent_unit |
| 12.8 | Assembly Drawing Current Revision Mark | TEXT | group_of_structural_data | (name_field) | context_dependent_unit |
| | | | | | |
| 13.1 | Part Drawing ID | INTEGER | group_of_structural_data | count_measure | context_dependent_unit |
| 13.2 | Part Drawing Number | TEXT | group_of_structural_data | | |
| 13.4 | Part Drawing Graphics Filename + Path | TEXT | group_of_structural_data | (description_field) | context_dependent_unit |
| 13.8 | Part Drawing Current Revision Mark | TEXT | group_of_structural_data | (name_field) | context_dependent_unit |
| | | | | | |
| 14.1 | Erection Drawing ID | INTEGER | group_of_structural_data | count_measure | context_dependent_unit |
| 14.2 | Erection Drawing Number | TEXT | group_of_structural_data | | |
| 14.4 | Erection Drawing Graphic Filename + Path | TEXT | group_of_structural_data | (description_field) | context_dependent_unit |
| 14.8 | Erection Drawing Current Revision Mark | TEXT | group_of_structural_data | (name_field) | context_dependent_unit |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| 15.1 | NC-DSTV file ID | INTEGER | group_of_structural_data | count_measure | context_dependent_unit |
| 15.2 | NC-DSTV filename + path | TEXT | group_of_structural_data | (description_field) | context_dependent_unit |
| 15.3 | Part header | TEXT | group_of_structural_data | (description_field) | context_dependent_unit |
| 15.4 | Part contours | (?) | group_of_structural_data | (description_field) | context_dependent_unit |
| 15.5 | Hole and PopMark date | TEXT | group_of_structural_data | (description_field) | context_dependent_unit |

**Table 2 Mapping MIS items with using structural_frame_item_documented**

| | Structural_frame_item_documented | CIS/2 item | Value Type |
|---|---|---|---|
| 1.14 | Part User Defined Field 1 | document.definition | TEXT |
| 1.15 | Part User Defined Field 2 | document.definition | TEXT |
| 1.16 | Part User Defined Field 2 | document.definition | TEXT |

**Table 3 Derived type MIS items**

| MIS item | | Value Type (SQL) | CIS/2 item | Value Type | Unit |
|---|---|---|---|---|---|
| 1.8 | Part Angle END 1 Angle Y | REAL | part | plane_angle_measure | plane_angle _unit |
| 1.9 | Part Angle END 1 Angle Z | REAL | part | plane_angle_measure | plane_angle _unit |
| 1.10 | Part Angle END 2 Angle Y | REAL | part | plane_angle_measure | plane_angle _unit |
| 1.11 | Part Angle END 2 Angle Z | REAL | part | plane_angle_measure | plane_angle _unit |
| | | | | | |
| 3.7 | Assembly Gross Length | REAL | assembly | positive_length_measure | length_unit |
| 3.8 | Assembly Gross Width | REAL | assembly | positive_length_measure | length_unit |
| 3.9 | Assembly Gross Height | REAL | assembly | positive_length_measure | length_unit |
| | | | | | |
| 9.3 | Bolt Assembly Quantity | INTEGER | joint_system_mechanical | count_measure | context_dependent_unit |
| | | | | | |
| 10.5 | Washer Quantity [1~N] | INTEGER | fastener_simple_washer | count_measure | context_dependent_unit |
| 10.12 | Nut Quantity [1~N] | INTEGER | fastener_simple_nut | count_measure | context_dependent_unit |

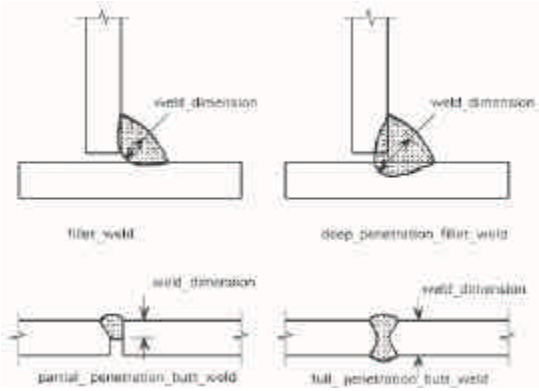
**Table 4 MIS items using structural_frame_item_priced**

| MIS item | | CIS/2 item | Value Type | Unit |
|---|---|---|---|---|
| 3.5 | assembly_cost_code | structural_frame_item | real | currency_unit |

**Other Notes:**

*1. How to keep track of a hierarchically relationship between a parent assembly and a child assembly is an important aspect. The parent and child relationship is defined in the design model as a particular attribute, but it is not defined in the manufacturing model in CIS/2. There are two scenarios to handle the exporting of assemblies.

(1) Assembly structured by the importing translator-
One solution is to make an explicit *assembly_relationship* instance to indicate the relationship between a key *child-assembly* with the parent assembly. The problem is the assembly demanded in the Fabtrol's system is not entirely consistent with assemblies typically exported from applications. Before data can be imported into the Fabtrol's system, there are many intermediate processes that must be processed. Those processes include disassembling exported assemblies and reassembling among pieces. The purpose and method of defining assemblies may not consist between the Frabtrol's system and exporting applications. An assembly is normally based on erection and shipping purpose in the Fabtrol's system, which is not similar to the intension of modeling an assembly. The front end of Fabtrol's system must be able to process a certain reassembling and disassembling routines based on individual purpose of the Fabtrol's system. Another way to solve this problem is to ask each import applications to obtain the reassembling and disassembling capability in their applications so the exported assemblies will consist with the assemblies that the Fabtrol's system is demanded.




*2. The right figure is copied from the CIS/2 schema definition (LPM500 V.4 P.377). The *weld_dimension* is the only variable that is available for calculating the cross section area of a particular weld in CIS/2. A convenient way to retrieve the remaining variables for calculating a cross section area is to create an external *item_reference* that refers to various cases of weld type. The referenced file shall include pre-calculate cross-section information based on its weld-type definition. It will save some computational time for a complicated geometrical derivation. Instead of using an external *item_reference* file to indicate explicit variables, a user can generate additional *item_property_assigned* to a weld instance or setup a convention of how to share an existed field.

(2) Assembly structured by the exporting application-
If assemblies generated in the detailing application can match requirements Fabtrol's system, then any intermediate adjustment is not necessary.

*3. The revision of a drawing is a minor ambiguity in the MIS specification. Drawings are representations and projections of a structural model. Any changing on a 2D printout drawing will not effect any modification to the electronic model.
Although one can literally map a revision of a drawing with using the grouping logic in CIS/2, but the content is not consistent. Extra information is required from Fabtrol to described how a drawing revision is used in the workflow and their system.
Instead of using the *version_action_request* item to define a revision of a drawing, a user can also use the *actioned_directive* entity as an explicit *action* item. An *actioned_directive provides* a name attribute that allows a user to specify a name for a person who did the revised job. The *group_assignment* instance will become an "ANDOR" instance that includes a *group_assignment_actioned* instance and a *group_of_structural_data* instance.