DDLm updates clarifying and revising the typing system

This document describes proposed updates to the DDLm dictionary, focusing on definitions by which the types of attribute values are specified. Particular attention is devoted to values of compound types (corresponding to CIF2 Lists and Tables), and mechanisms are introduced for defining value types by reference to the types defined for values of other attributes.

Change 1: clarification to the description of `_type.contents`

The 2012-05-07 definition of `_type.contents` is unclear about how it applies to definitions of items having `_type.container` of 'Table'. It appears to be the intent in that case for `_type.contents` to define the expected construction of the values (as opposed to the indices) within (Table) values for the defined attribute. That this interpretation is indeed intended is clarified by appending the following text to the `_description.text` attribute of the definition of `_type.contents`:

> Where the defined attribute is a container of the 'Table' variety, this attribute describes the construction of the value elements within (Table) values of the defined attribute.

Change 2: definitions for "internal" purposes

Changes described later in this document create a use for defining attributes that serve only internal purposes within their dictionaries. To enable such definitions to be recognized and distinguished, a new "Internal" state is added to those permitted for attribute `_type.purpose`. In CIF form, these values for `_enumeration_set.state` and `_enumeration_set.detail` are added to the definition of `_type.purpose`:

```
Internal
; Used to type items that serve only internal purposes of the dictionary
in which they appear.  The particular purpose served is not defined by
this state.
;
```

Change 3: defining attributes' contents by reference

As of the 2012-05-07 definition of `_type.contents`, there is no way to define the details of the content of an attribute having a compound type whose elements are also of compound type, except to the extent that multi-dimensional Lists, Arrays, and Matrices constitute such content types. Furthermore, where two attributes of compound type take values that are inherently of the same kind, DDLm currently offers no way to define explicitly that they have the same content type; one must simply duplicate the type information. Two changes are introduced to better support such cases.

First, a new "ByReference" state is added to those permitted for attribute `_type.contents`. In CIF form, these values for `_enumeration_set.state` and `_enumeration_set.detail` are added to the definition of `_type.contents`:

```
ByReference
; The contents have the same form as those of the attribute referenced by
_type.contents_referenced_id.
;
```

Second, a new attribute _type.contents_referenced_id is added, as foreshadowed by the new _type.contents code:

```
save_type.contents_referenced_id
    _definition.id            '_type.contents_referenced_id'
    _definition.update         2015-04-24
    _definition.class          Attribute
    _description.text
;
     The value of the _definition.id attribute of an attribute definition
     whose type is to be used also as the type of this item.  Meaningful only
     when this item's _type.contents attribute has value 'ByReference'.
;
    _type.category_id          type
    _type.object_id            contents_referenced_id
    _type.purpose              Identify
    _type.container            Single
    _type.contents             Tag
     save_
```

Change 4: constraining indices of Table entries, including by reference

DDLm currently has an attribute _enumeration_set.table_id that is intended to serve for enumerating the indices that may be used in the values of an attribute having _type.container 'Table'. This approach does not work well because the ENUMERATION_SET category describes entry *values*, not indices, and because even if that is ignored, it requires dummy values to be introduced for the category key, _enumeration_set.state. In its one use in the DDLm dictionary, _enumeration_set.table_id is in fact applied at the wrong level. That approach is replaced with a more workable solution, as detailed next.

Attribute _enumeration_set.table_id is removed.

Two new attributes are introduced:

```
save_type.indices
    _definition.id            '_type.indices'
    _definition.update         2015-04-24
    _definition.class          Attribute
    _description.text
;
     Used to specify the syntax construction of indices of the entries in the
     defined object when the defined object has 'Table' as its
     _type.container attribute.  Values are a subset of the codes and
     constructions defined for attribute _type.contents, accounting
     for the fact that syntactically, indices are always case-sensitive
     quoted strings.

     Meaningful only when the defined item has _type.container 'Table'.
;
    _name.category_id          type
    _name.object_id            indices
    _type.purpose              State
    _type.container            Single
    _type.contents             Code
     loop_
    _enumeration_set.state
```

```
    _enumeration_set.detail
     Text         'a case-sensitive string/lines of text'
     Filename     'name of an external file'
     Code         'code used for indexing data or referencing data resources'
     Date         'ISO date format yyyy-mm-dd'
     Uri          'an universal resource identifier string, per RFC 3986'
     Version      'version digit string of the form <major>.<version>.<update>'
     ByReference
;
     Indices have the same form as the contents of the attribute identified by
     _type.indices_referenced_id
;
    _enumeration.default        Text
     loop_
    _description_example.case
    _description_example.detail
     'Code' 'indices belong to an enumerated set of pre-defined codes'
     'Uri'  'indices have the form of URIs'
     save_

save_type.indices_referenced_id
    _definition.id             '_type.indices_referenced_id'
    _definition.update          2015-04-24
    _definition.class           Attribute
    _description.text
;
     The _definition.id attribute of a definition whose type describes the
     form and construction of the indices of entries in values of the present item.

     Meaningful only when the defined item's _type.container attribute has
     value 'Table', and its _type.indices attribute has value 'ByReference'.
;
    _type.category_id           type
    _type.object_id             indices_referenced_id
    _type.purpose               Identify
    _type.container             Single
    _type.contents              Tag
     save_
```

Change 5: fixing `_import.get`

Among the attributes currently defined by DDLm, only `_import.get` uses the `_enumeration_set.table_id` attribute removed in change 4. Its definition is updated and two supporting internal attributes are introduced to accommodate the change. Furthermore, the current dREL expression is incorrect: it assumes a single available value for _import.file, _import.frame etc., when the intent is clearly that multiple values are available that should be inserted into each Table element of the list. Multiple values imply a loop category, so we move `_import.file`, etc. into a new loop category, `IMPORT_DETAILS` (revised definitions of `_import.file/frame/mode/if_dupl/if_miss` not presented here). In order to correctly order the tables in the overall list, a dataname giving the explicit ordering must also be added to this category (see `_import_details.order` below).

```
save_import.get
    _definition.id             '_import.get'
    _definition.update          2015-04-24
```

```
    _definition.class            Attribute
    _description.text
;
     A list of tables of attributes defined individually in the category
     IMPORT_DETAILS, used to import definitions from other dictionaries.
;
    _name.category_id            import
    _name.object_id              get
    _type.purpose                Import
    _type.container              List
    _type.contents               ByReference
    _type.contents_referenced_id '_import_details.single'
     loop_
    _method.purpose
    _method.expression
     Evaluation
;
     imp_order_list = []
     loop id as import_details {
         imp_order_list ++= id.order
     }
     sort(imp_order_list)
     final_val = []
     for ord in imp_order_list {
         final_val ++= import_details[ord].single
     }
    _import.get = final_val
;
     save_

save_IMPORT_DETAILS
    _definition.id               IMPORT_DETAILS
    _definition.scope            Category
    _name.category_id            IMPORT
    _name.object_id              IMPORT_DETAILS
    _definition.class            Loop
    _category.key_id             '_import_details.order'
    loop_
        _category_key.name      '_import_details.order'
    _description.text
;
    Items in IMPORT_DETAILS describe individual attributes of an import operation.
;
save_

save_import_details.single
    _definition.id              '_import_details.single'
    _definition.update           2015-04-24
    _definition.class            Attribute
    _description.text
;
     A Table mapping attributes defined individually in category IMPORT to
     their values; used to import definitions from other dictionaries.
;
    _name.category_id            import_details
    _name.object_id              single
    _type.purpose                Internal
    _type.container              Table
    _type.contents               Text
```

```
    _type.indices                  ByReference
    _type.indices_referenced_id    '_import_details.single_index'
    loop_
      _method.purpose
      _method.expression
      Evaluation
;
      with id as import_details
      import_details.single = {"file":id.file_id
                              "save":id.frame_id,
                              "mode":id.mode,
                              "dupl":id.if_dupl,
                              "miss":id.if_miss}
;
     save_

save_import_details.single_index
    _definition.id             '_import_details.single_index'
    _definition.update          2015-04-24
    _definition.class           Attribute
    _description.text
;
     One of the indices permitted in the entries of values of attribute
_import_details.single.
;
    _name.category_id           import
    _name.object_id             single_index
    _type.purpose               Internal
    _type.container             Single
    _type.contents              Code
     loop_
    _enumeration_set.table_id
    _enumeration_set.detail
       file 'filename/URI of source dictionary'
       save 'save framecode of source definition'
       mode 'mode for including save frames'
       dupl 'option for duplicate entries'
       miss 'option for missing duplicate entries'
     save_

save_import_details.order
      _definition.id               '_import_details.order'
      _name.category_id            'import_details'
      _name.object_id              'order'
      _definition.class            Attribute
      _description.text
;
      The order in which the import described by the referenced row should be
      executed.
;
      _type.container              Single
      _type.contents               Integer
save_
```