International Tables for Crystallography (2006). Vol. G, Section 3.1.4, pp. 75–76.

(http://www.iucr.org/iucr-top/cif) should be consulted for current practice. However, a short overview of the existing procedures is helpful in describing how the community can participate in extending the standard.

3.1.3.1. Dictionary maintenance groups

Each published dictionary authorized by COMCIFS has a group of specialists appointed or invited to extend and maintain the dictionary to serve the changing needs of the subdiscipline that sponsors the dictionary. Members of these dictionary maintenance groups (DMGs) may suggest extensions or corrigenda on their own initiative or may pass on requests for extensions from individual crystallographers. A DMG will typically debate and review any suggested amendments and produce a draft revised dictionary for approval by COMCIFS.

3.1.3.2. mmCIF review cycle

The macromolecular CIF dictionary covers a very broad and active field, and a more formal procedure exists for the submission and review of proposed extensions. Possible new definitions are submitted using *pro forma* dictionary templates to a member of an editorial board appointed by the mmCIF dictionary maintenance group. Accepted proposals are approved by the DMG and released for general community review in provisional extension dictionaries as circumstances require. The extension dictionary is revised as necessary and is finally incorporated within the parent mmCIF dictionary after COMCIFS approval has been granted.

3.1.3.3. New dictionaries

A completely new dictionary to cover a subdiscipline not otherwise catered for may be commissioned by COMCIFS or may arise from community action, occasionally sponsored by an IUCr Commission. A working group is appointed to create the dictionary and relevant example files or software. The working group is expected to test the new dictionary extensively within its own community before submitting it to COMCIFS for initial approval. It is the responsibility of COMCIFS to check the dictionary for technical consistency and for compatibility with related dictionaries. COMCIFS may refer the dictionary back to the working group for further revisions. When the dictionary finally receives formal COMCIFS approval and is published, a dictionary maintenance group is formed to promote its further development (Section 3.1.3.1). The DMG usually includes one or more members of the initial working group and at least one voting member of COMCIFS.

3.1.4. Choice of data model

The following sections of this chapter describe the technical considerations in defining data items within a dictionary. Fundamental to this is the *data model* on which the dictionary is based. The STAR File upon which CIF is based is a very versatile data format and can accommodate a variety of data models. However, the use within CIF of a single level of looping enforces a rather flat data structure and a typical CIF maps most easily onto a relational database model. This is implicit in DDL1, which assigns different attributes to data items depending on whether they appear in data loops or not. Generally speaking, one may consider a list header and its associated data values as the head and body of a table of data values. The list header (or equivalently the table head) identifies the data items ranged by column within the table. For the dictionary entries relating to the data names in the list header, the `_category` attribute collects together data items which may be looped together in the same table, and the `_list_reference`, `_list_mandatory` and `_list_uniqueness` attributes work together to indicate the data items that must be present and collectively have a unique value to identify a specific row in a table of values.

Example 3.1.4.1. *Core dictionary definitions for the atom-site labels and bond distances in a CIF table of molecular geometry.*

```
data_geom_bond_atom_site_label_
    loop_ _name
                       '_geom_bond_atom_site_label_1'
                       '_geom_bond_atom_site_label_2'
    _category                    geom_bond
    _type                        char
    _list                        yes
    _list_mandatory              yes
    _list_link_parent          '_atom_site_label'
    _definition
;     The labels of two atom sites that form a bond.
      These must match labels specified as
      _atom_site_label in the atom list.
;

data_geom_bond_distance
    _name                       '_geom_bond_distance'
    _category                    geom_bond
    _type                        numb
    _type_conditions             esd
    _list                        yes
    _list_reference    '_geom_bond_atom_site_label_'
    _enumeration_range           0.0:
    _units                       A
    _units_detail               'angstroms'
    _definition
;     The intramolecular bond distance in angstroms.
;
```

For example, the following example from the core CIF dictionary (Chapter 4.1) shows a table of bond distances. The dictionary definitions are given in Example 3.1.4.1.

```
loop_
_geom_bond_atom_site_label_1
_geom_bond_atom_site_label_2
_geom_bond_distance
  O1  C2   1.342(4)
  O1  C5   1.439(3)
  C2  C3   1.512(4)
  C2  O21  1.199(4)
  C3  N4   1.465(3)
  C3  C31  1.537(4)
  C3  H3   1.00(3)
  N4  C5   1.472(3)
```

Within the dictionary, entries for all of `_geom_bond_distance`, `_geom_bond_atom_site_label_1` and `_geom_bond_atom_site_label_2` share the same `_category` attribute, namely ‘geom_bond’. (In the rest of this chapter, as elsewhere in the volume, we refer to categories by the upper-case form of their category attribute values; here, therefore, we are referring to the GEOM_BOND category.) The entry for `_geom_bond_distance` has a `_list_reference` value of `'_geom_bond_atom_site_label_'` indicating the data names that may be used to identify this particular table. The trailing underscore in this example indicates that all matching data names must be considered as components of a compound identifier; for this case the matching data names are `'_geom_bond_atom_site_label_1'` and `'_geom_bond_atom_site_label_2'`. The dictionary entry for `_geom_bond_atom_site_label_` has a `_list_mandatory` value of `yes`, indicating that these data items *must* be present within the table. In this way, the attributes specify the unique key within a database table (in this case, the key has multiple components: the labels of both contributing atom sites).

However, the mapping onto a relational database is not exact. In some cases CIFs may present data from a single category across several tables, or the implied key may not have a unique value unless concatenated with other fields in the table row. For many applications this is only of academic interest; but in some subdisciplines it is important that the data model is constrained strictly to a relational one, and for those applications dictionaries built on the DDL2 formalism are more appropriate.

Of the dictionaries presented in this volume, the core, powder, modulated structures and electron density dictionaries use the DDL1 formalism and the symmetry, macromolecular and image dictionaries use the DDL2 formalism. The core dictionary uses DDL1 so that it can be used alongside other less rigorous dictionaries. The powder dictionary is one case of this, where the need to tabulate and merge extensive lists of raw or processed data is not well served by a relational model. Modulated structures are also best served by a data model that is not rigorously relational. The macromolecular dictionary uses DDL2 because many of the major database applications in macromolecular crystallography are relational in nature, but in consequence it contains a copy of the core data items re-expressed in DDL2 formalism. The image dictionary is in DDL2 because it was designed to operate closely alongside the macromolecular dictionary. The symmetry dictionary is an interesting case. It was constructed in DDL2 format as an exercise in supplying an extension dictionary immediately suitable for direct incorporation into other DDL2-based dictionaries and also suitable for transformation to the simpler DDL1 formalism as necessary to complement existing DDL1 dictionaries.

While the main difference between DDL1 and DDL2 lies in the rigour with which relational data structures are enforced, DDL2 also offers a larger set of attributes for specifying hierarchical relationships between data names and for typing data values, and in consequence a complete DDL2-based dictionary is richer (and correspondingly more complex to construct) than an equivalent DDL1 description.

There may be no obvious reason for selecting one formalism over the other when planning a new data dictionary, and prospective authors must give considerable thought to the merits of both formalisms. However, once the choice has been made, the structure of the dictionary and its component definitions is profoundly affected. The constructions of the two types of dictionary are discussed separately in Sections 3.1.5 and 3.1.6 below.

3.1.5. Constructing a DDL1 dictionary

Dictionaries constructed according to DDL1 have quite a simple structure. The structure is summarized in this section; Sections 3.1.5.1–3.1.5.4 provide more detail. Each definition is encapsulated within its own data block. Fig. 3.1.5.1 outlines the contents of the core CIF dictionary. The order of the data blocks has no significance, but it is common practice to start the file with the data block that describes the name, version and revision history of the dictionary itself and then to arrange data blocks in alphabetical order, sorted first on category then on names within a category. This practice is not always followed – for example, the powder dictionary is ordered by theme. The choice of order in a dictionary is only used for presentation and dictionary parsers should not assume or rely on any order of data blocks.

The name of a data block is usually constructed from the name of the data item it describes, *e.g.* `data_refln_phase_meas`. Where the data block describes an entire category instead of a single data item, the category name is followed by matching square brackets, which may contain an alphabetic code representing the dictionary name if it is an extension to the core dictionary (*e.g.* `data_refln_[]`, `data_audit_link_[ms]`). Where the data block defines several data names, the initial common portion of the names is used with a trailing underscore (*e.g.* `data_refln_`).

```
data_on_this_dictionary
    _dictionary_name             cif_core.dic
    _dictionary_version          2.3.1
    _dictionary_update           2005-06-27
    _dictionary_history
;
   1991-05-27  Created from CIF Dictionary text. SRH
   . . .
;
```

(*a*)

```
data_atom_site_[]
    _name                          '_atom_site_[]'
    _category                        category_overview
    _type                            null
    loop_ _example
    _example_detail   .   .

data_atom_site_adp_type
    _name                          '_atom_site_adp_type'
    _category                        atom_site
    _type                            char
    _definition                    'A standard code ...'

data_atom_site_aniso_B_
    loop_ _name                    '_atom_site_aniso_B_11'
                                     . . .
```

(*b*)

Fig. 3.1.5.1. Schematic structure of core CIF dictionary. (*a*) Dictionary identifiers. (*b*) Definitions of categories and data items.

A preliminary data block, by convention labelled with the header string `data_on_this_dictionary`, contains the dictionary identification information and revision history. The name of the dictionary itself (given by the data name `_dictionary_name`) is conventionally of the form `cif_`*`identifier`*`.dic`, where the *identifier* is a short code for the topic area of the dictionary (*e.g.* 'core' for the core dictionary, 'pd' for the powder dictionary, 'ms' for the modulated structures dictionary, 'rho' for the electron density dictionary).

Data names are classified by category. The `_category` attribute is a character string intended to indicate the 'natural grouping' of data items. If a data item occurs in a looped list, it must be grouped only with items from the same category. It is, however, permissible for a file to contain more than one looped list of the same category, provided that each loop has its own specific reference item identified by the `_list_reference` attribute of the data names included. Examples of this will be given below.

For each category, a data block is usually provided that contains information about the purpose of the category, generally illustrated with examples.

All other data blocks represent self-contained definitions of a single data item or a small set of closely related data items. The definition includes the physical units of and constraints on the values of the data labelled by the defined data name, and also information about relationships with other data items.

It is conventional, although not mandatory in DDL1 dictionaries, that the category name should appear as the leading component or components of a data name. For example, the data name `_exptl_crystal_colour` is a member of the core category EXPTL, while `_exptl_crystal_density_meas` is a member of the category EXPTL_CRYSTAL and `_exptl_crystal_face_perp_dist` is a member of the category EXPTL_CRYSTAL_FACE. However, it will

references