Example 3.1.9.1. *A standard CIF dictionary definition block.*

```
data_atom_site_attached_hydrogens
    _name              '_atom_site_attached_hydrogens'
    _category            atom_site
    _type                numb
    _list                yes
    _list_reference      '_atom_site_label'
    _enumeration_range   0:8
    _enumeration_default 0
    loop_ _example
          _example_detail  2    'water oxygen'
                           1    'hydroxyl oxygen'
                           4    'ammonium nitrogen'
    _definition
;       The number of hydrogen atoms attached
        to the atom at this site excluding any
        hydrogen atoms for which coordinates
        (measured or calculated) are given.
;
```

Example 3.1.9.2. *A modified data attribute for overlaying a public definition.*

```
data_atom_site_attached_hydrogens_restricted
    _name              '_atom_site_attached_hydrogens'
    _enumeration_range   0:4
```

As an example, consider the core CIF dictionary definition mentioned above of the number of hydrogen atoms that might be attached to an atom site (Example 3.1.9.1).

For a particular application, any structures reporting more than four attached hydrogen atoms might be considered as invalid. A validation program to satisfy this requirement might therefore build a composite dictionary from the public cif_core.dic, which contains the definition in Example 3.1.9.1, and the fragment of Example 3.1.9.2, processed in APPEND/OVERLAY modes.

3.1.9.2. Protocol implementation

At the time of publication (2005), there is no reference implementation for this protocol, and so the proper treatment of the fine details of merging and overlay operations is not available. The following guidelines outline the first steps in an implementation under DDL1.4.

The description assumes that a composite dictionary is to be assembled from two public dictionaries, a.dic and b.dic, and a local dictionary mod.dic that includes some modifications to the definitions in one or both of the public dictionaries (and is therefore processed in OVERLAY mode). It is assumed that the composite dictionary will be written to disk as a separate file, virtual.dic, although in practice applications may simply construct the image of the composite dictionary in memory.

(1) Each contributing dictionary fragment should have at most one data block containing the data names `_dictionary_name` and `_dictionary_version` (with, optionally, `_dictionary_update` and `_dictionary_history`). The `*_name` and `*_version` together identify the dictionary file uniquely and should match the corresponding entries in the IUCr register if this is a public dictionary. This information is conventionally stored in a data block named `data_on_this_dictionary`.

In DDL1.4, all four of the items `_dictionary_name`, `*_version`, `*_update` and `*_history` are scalars, *i.e.* may not be looped. Hence a new dictionary identifier section in virtual.dic may be constructed as follows.

(i) Create a data block `data_on_this_dictionary` at the beginning of virtual.dic.

(ii) If a name for the composite dictionary is supplied (*via* a command-line switch, for example), write this as the value of `_dictionary_name`; otherwise generate a pseudo-unique string (*e.g.* concatenate the computer identifier string, process number and current date string).

(iii) If a dictionary version number is supplied (*via* a command-line switch, for example), write this as the value of `_dictionary_version`; otherwise supply the value '1.0'.

(iv) Supply the current date in the format *yyyy-mm-dd* as the value of `_dictionary_update`.

(v) Create a composite `_dictionary_history` by concatenation of the individual `_dictionary_history` fragments. The application may add details of the current merge operation to the history field.

(2) There is no significance to the ordering of data blocks containing definitions in dictionaries, although they are conventionally sorted alphabetically. For convenience, data blocks should be written out in the order in which they are encountered in the input primitive dictionary files, except that definitions modified by subsequent entries remain in their initial location.

(3) In STRICT mode, if the same value of `_name` is present in two or more data blocks, the composite dictionary is invalid and the application should raise a fatal error. Otherwise the composite dictionary simply contains the aggregate definitions from multiple input dictionaries.

(4) In REPLACE mode, a stored definition block is discarded and replaced by a new definition of the item referenced by `_name`.

(5) For the OVERLAY mode (assumed in the present discussion), the following procedure is proposed. Load a data block from the first dictionary file. Locate the `_name` tag. (Because `_name` may be looped, a data block may contain definitions for more than one data name. For convenience, we consider only the case of a data block containing a single value of `_name`. In any event, it is possible to separate a set of looped definitions into individual data blocks, each defining only one of the data names in the initial `_name` loop.) Search the next dictionary file for a data block containing the same value of `_name`. Load the contents of that data block.

(i) If the new data block contains only data items that do not appear in the first data block, they are simply concatenated with those already present.

(ii) If the new data block contains a scalar data item already present in the first data block (*i.e.* with `_list no`), discard the stored attributes.

(iii) If the new data block contains data items that may be looped and that occur in the first data block, build a new composite table of values in the following way: (*a*) construct a valid loop header if necessary; (*b*) do not repeat identical sets of values (*i.e.* collapse identical table rows); (*c*) if it is possible to identify the category key, then raise a fatal error if there are identical instances of a key value [after the normalization of step (*b*) has occurred]; (*d*) else append new rows to the table.

When the new composite data block has been built according to these principles, search the next dictionary file specified and repeat.

3.1.10. Public CIF dictionaries

So far, seven CIF dictionaries have been published by the IUCr with COMCIFS approval. They are described in the remaining chapters in this part of the volume. This section provides an overview of the large-scale structure of these dictionaries and forms a general introduction to Chapters 3.2 to 3.8.

The public CIF dictionaries have been constructed by experts in a number of different crystallographic fields. They are intended to serve the individual fields in which they have been commissioned and therefore vary in character depending on the requirements

Table 3.1.10.1. *High-level grouping of categories by dictionary*

Category groups are organized into families by common function and purpose.

cif_core.dic	cif_pd.dic	cif_ms.dic	cif_rho.dic	mmcif_std.dic	cif_img.dic	cif_sym.dic
(*a*) Experimental measurements						
					ARRAY	
					AXIS	
CELL		CELL		CELL		
DIFFRN		DIFFRN		DIFFRN	DIFFRN	
EXPTL		EXPTL		EXPTL		
	PD_CALIB					
	PD_CHAR					
	PD_DATA					
	PD_INSTR					
	PD_MEAS					
	PD_PREP					
	PD_SPEC					
(*b*) Analysis						
	PD_CALC					
	PD_PEAK					
	PD_PROC					
				PHASING		
REFINE		REFINE		REFINE		
REFLN	REFLN	REFLN		REFLN		
(*c*) Structure						
ATOM		ATOM	ATOM	ATOM		
CHEMICAL				CHEMICAL		
				CHEM_COMP		
				CHEM_LINK		
				ENTITY		
GEOM		GEOM		GEOM		
	PD_PHASE					
				STRUCT		
SYMMETRY		SYMMETRY		SYMMETRY		SPACE_GROUP
VALENCE				VALENCE		
(*d*) Publication						
CITATION				CITATION		
COMPUTING				COMPUTING		
DATABASE				DATABASE		
JOURNAL				JOURNAL		
PUBL				PUBL		
				SOFTWARE		
(*e*) File metadata						
AUDIT		AUDIT		AUDIT		
	PD_BLOCK					

and practices of each field. Here we provide a general framework within which the category groups of each separate dictionary may be described.

3.1.10.1. Categories and category groups

The only formal unit of classification common to all CIF dictionaries is the *category*. For example, in the core CIF dictionary information about the chemical and physical properties of the different atomic species in a crystal cell is collected in a few data names such as **_atom_type_oxidation_number** which belong to the same category, in this case the ATOM_TYPE category. As described in Section 3.1.5.3, it is conventional (although not mandatory) that CIF data names begin with components corresponding to the name of the category to which they belong.

The term *category* as used in CIF dictionaries has a technical meaning which constrains its normal use in grouping items that are understood to have a 'natural' relationship. In a CIF, only items belonging to the same category may appear together in the same looped list. This means, for example, that data items describing collective properties of the atom sites in the lattice (such as the number of atoms of each atomic species in the unit cell) must be assigned to a different category from the data items that describe the properties of the individual sites. Hence the properties of individual sites (such as the positional coordinates defined by **_atom_site_fract_x** *etc.*) belong to the ATOM_SITE category, while the transformation matrix between Cartesian and fractional components (expressed by a collection of data names such as **_atom_sites_fract_tran_matrix_11**) belong to the ATOM_SITES category. Clearly, the category names have been chosen to be similar to reflect their close relationship, while the EXPTL category containing data names such as **_exptl_crystal_colour** is named quite differently. It is natural to wish to describe related categories in a common higher level of classification, and indeed *category groups* exist as formal components of DDL2-structured dictionaries. We shall, however, refer informally to 'category groups' in discussions of DDL1 dictionaries as collections of categories with a close relationship that is usually implicit in their names.

3.1.10.2. Overview of category classification

Table 3.1.10.1 provides an informal classification at a high level of the category groups represented in each of the CIF dictionaries in this volume. Related category groups are clustered within the table in families sharing a common function. The five families (*a*) to (*e*) in Table 3.1.10.1 refer to: the crystallographic experiment itself; the processing and analysis of data from the experiment; the derived structural model; the reporting and publication of the results; and general auditing of the file itself, its purpose, authorship, history and links to other data sets, *i.e.* the file metadata. Detailed discussions of the individual categories (and formal category groups for DDL2 dictionaries) will be found in the relevant chapters in the rest of this part of the volume.

Table 3.1.10.1 shows the different characters of the seven dictionaries. The macromolecular dictionary (mmcif_std.dic; Chapter 4.5) contains an embedded version of the core dictionary

(cif_core.dic; Chapter 4.1) in DDL2 format and so includes all the categories defined in the core. However, it extends the description of the structural model extensively by introducing families of categories for the description of chemical components of a macromolecular structure (ENTITY) and for the detailed description of the structure itself (STRUCT). New categories are also introduced to describe the phasing of the structure and the SOFTWARE category allows the inclusion of more details of computational techniques than the core COMPUTING category does.

The other dictionaries are purely extensions which either introduce new data names (and occasionally new categories) into existing category groups or, where necessary, introduce completely new groups of categories.

The powder dictionary (cif_pd.dic; Chapter 4.2) contains several new category groups reflecting the need for substantially different methods of describing the experiment and analysing the data, as well as a need for the structural model to be able to handle multiple crystalline phases. The modulated structures dictionary (cif_ms.dic; Chapter 4.3) introduces no new category groups, but does introduce several new data names and categories within the existing framework. The electron density dictionary (cif_rho.dic; Chapter 4.4) introduces two new categories within an existing category group. The image CIF dictionary (cif_img.dic; Chapter 4.6) has several new categories that characterize arrays of data from two-dimensional X-ray detectors and the consequent detailed descriptions of the relevant axes within the experimental setup. The symmetry dictionary (cif_sym.dic; Chapter 4.7) was commissioned specifically to replace the symmetry categories in the core dictionary with a more detailed treatment.

References

Bray, T., Paoli, J. & Sperberg-McQueen, C. M. (1998). *Extensible Markup Language (XML) 1.0*. W3C Recommendation 10-February-1998. http://www.w3.org/TR/1998/REC-xml-19980210.

CrossRef (2004). *Query spec.* http://www.crossref.org/03libraries/25query_spec.html.

Hall, S. R., Allen, F. H. & Brown, I. D. (1991). *The Crystallographic Information File (CIF): a new standard archive file for crystallography. Acta Cryst.* A**47**, 655–685.

Hall, S. R., Spadaccini, N., Castleden, I. R., du Boulay, D. & Westbrook, J. D. (2002). *StarDDL: towards the unification of Star dictionaries. Acta Cryst.* A**58** (Suppl.), C256.

W3C (2004). *Extensible Markup Language (XML)*. http://www.w3c.org/XML/.