2.2.6. CIF metadata and dictionary compliance

The development of several CIF dictionaries and fields of application has rapidly progressed beyond the specific purpose of describing a small-molecule or inorganic crystal structure for which CIF was devised. With these, a variety of application-specific metadata approaches have evolved to characterize the role of a particular CIF within a family of possible applications. These approaches use data definitions in dictionaries in which enumerated codes identify the file relationships. The mmCIF dictionary (see Chapter 3.6) allows informal identification of 'external reference files' which act as libraries of standard molecular geometry. The pdCIF dictionary (see Chapter 3.3) specifies identifiers that may be included within data blocks of external files containing calibration results. It is the responsibility of the file users to manage a lookup table or database between the referenced identifiers and the location of the files to which they pertain.

Two categories of data items currently exist in the core dictionary to allow a file to indicate its relationship to CIF dictionaries and other data files. (Equivalent categories are also present in the mmCIF dictionary.) AUDIT_CONFORM is a category of data names identifying the dictionaries that hold definitions of the data names in the current CIF. Particularly where the referenced dictionaries include any of the various public dictionaries described in Part 3 of this volume, this serves to establish the discipline within the broad fields of crystallography, structural biology and structural chemistry to which the data are most relevant.

The category AUDIT_LINK allows an informal textual description of the relationship between the data blocks within the current file. It is 'informal' in the sense that the relevant data items are free-text in nature. It would surely be useful to have a catalogue of more specific designations to allow automated software to track such relationships as the separate reference and modulated structures in an incommensurate compound, or the multiple trial refinements of a protein structure. The challenge is to determine and classify such standard relationships between data blocks.

In the future it is hoped that a common approach to metadata will be developed to enable all CIF instantiations to be uniquely identified and interrelated. Development of standard descriptions of the relationships between structural entities of this sort (reference geometries, calibration results, partial refinements, modulated superposed structures *etc.*) will be an important stage in the formalization of complete CIF metadata, and will become an important step towards categorization of data entities needed for interoperability between different file formats and across a wide range of scientific disciplines.

2.2.7. Formal specification of the Crystallographic Information File

Version 1.1 specification

BY S. R. HALL, N. SPADACCINI, I. D. BROWN, H. J. BERNSTEIN, J. D. WESTBROOK AND B. MCMAHON

This section presents the documents *File syntax* (Sections 2.2.7.1–3) and *Common semantic features* (Section 2.2.7.4) that together comprise the formal CIF specification as approved by COMCIFS.

2.2.7.1. Syntax

2.2.7.1.1. *Introduction*

(1) This document describes the full syntax of the Crystallographic Information File (CIF).

2.2.7.1.2. *Definition of terms*

(2) The following terms are used in the CIF specification documents with the specific meanings indicated here.

(2.1) A **CIF** is a file conforming to the specification herein stated, containing either information on a crystallographic experiment or its results (or similar scientific content), or descriptions of the data identifiers in such a file.

(2.2) A **data file** is understood to convey information relating to a crystallographic experiment.

(2.3) A **dictionary file** is understood to contain information about the data items in one or more data files as identified by their data names.

(2.4) A **data name** is a case-insensitive identifier (a string of characters beginning with an underscore character) of the content of an associated data value.

(2.5) A **data value** is a string of characters representing a particular item of information. It may represent a single numerical value; a letter, word or phrase; extended discursive text; or in principle any coherent unit of data such as an image, audio clip or virtual-reality object.

(2.6) A **data item** is a specific piece of information defined by a data name and an associated data value.

(2.7) A **tag** is understood in this document to be a synonym for data name.

(2.8) A **data block** is the highest-level component of a CIF, containing data items or save frames. A data block is identified by a **data-block header**, which is an isolated character string (that is, bounded by white space and not forming part of a data value) beginning with the case-insensitive reserved characters `data_`.

(2.9) A **block code** is the variable part of a data-block header, *e.g.* the string `foo` in the header `data_foo`.

(2.10) A **save frame** is a partitioned collection of data items within a data block, started by a **save-frame header**, which is an isolated character string beginning with the case-insensitive reserved characters `save_`, and terminated with an isolated character string containing only the case-insensitive reserved characters `save_`.

(2.11) A **frame code** is the variable part of a save-frame header, *e.g.* the string `foo` in the header `save_foo`.

2.2.7.1.3. *File syntax*

(3) The syntax of CIF is a proper subset of the syntax of STAR Files as described by Hall (1991) and Hall & Spadaccini (1994). The general structure is described below in Section 2.2.7.1.4 and a number of subsections list specific restrictions to the STAR syntax that are in force within CIF. A formal language grammar using computer-science notation is included as Section 2.2.7.2.

2.2.7.1.4. *General features*

(4) A CIF consists of **data names** (tags) and associated values organized into **data blocks**. A data block may contain **data items** (associated data names and data values) and/or it may contain **save frames**.

(5) **Save frames** may only be used in dictionary files.

Implementation note: At a purely syntactic level there is no way to distinguish between dictionary and data files. (It is also to be noted that not all dictionary files contain save frames.) A fully validating parser must therefore be able to detect the start and termination of save frames, the uniqueness of the frame code within a data block and the uniqueness of data names within a frame code. It is, however, legitimate for an application-based parser designed to handle only the contents of data files to consider the presence of a save frame as an error.