5.3. Syntactic utilities for CIF

BY B. MCMAHON

5.3.1. Introduction

Since the introduction of the Crystallographic Information File (CIF), the crystallographic community has produced a wide variety of tools and applications to handle CIFs. Many changes have been made to existing programs to input and output CIF data sets, and occasionally changes may have been made to internal crystallographic calculations to provide a better fit to the view of the data expressed by the standard CIF dictionaries. However, for most crystallographers with an involvement in programming, there is an understandable tendency to invest the minimum amount of effort needed to accommodate the new format. Their primary interest is in the understanding and discovery of the underlying physical model of a crystal structure.

This chapter reviews several general-purpose tools that have been developed for CIF to check, edit, extract or manipulate arbitrary data items, with little in the way of crystallographic computation. They are of interest to the end user who wishes to visualize a structure in three dimensions or submit an article to a journal but who does not want to be concerned about the details of CIF. They also include several utilities that are helpful for manipulating the contents of CIFs without the need to write a large and complex program. The programmer with an interest in writing complete and robust CIF applications should look at the comprehensive libraries described in Chapters 5.4 to 5.6.

Many of the programs described in this chapter operate purely at the syntactic level; they require no knowledge of the scientific meaning of the data items being manipulated. Others have some bearing on the *semantics* of the file contents, either explicitly through information about data types and interrelationships carried in external CIF dictionaries, or implicitly through the user's choice and deliberate manipulation of items based on an understanding of what they signify. Nevertheless, most utilities described here are characterized by an ability to handle CIFs of any content and provenance. The best example of a program able to handle arbitrary CIFs at a purely syntactic level is *Star_Base* (Spadaccini & Hall, 1994), described in Chapter 5.2.

It should be noted that not all the programs described here are fully compliant with the specification of Chapter 2.2, and others have implementation restrictions or known bugs. Many, especially the older programs, are no longer actively supported and need to be handled with care. However, they are included here for the record, and because they may provide useful ideas and suggestions to future developers in an area that can still accommodate a wider range of tools for different uses.

5.3.2. Syntax checker

A CIF must conform to a subset of the syntax rules of a general STAR File (Chapter 2.1), but with the additional restrictions and conventions described in Chapter 2.2. The syntax is rather simple and robust subroutines to create CIFs may easily be written by computer programmers. However, the use of ASCII character sets, deliberately expressive data names and simple layout conventions both permit and encourage users to edit the files with general text editors that cannot guarantee to retain syntactic integrity. Consequently, there is a definite use for a simple program that can check whether a file conforms to the specified syntax.

It is worth mentioning that programmable text editors such as *emacs* may be supplied with rules that can check syntax as a file is edited. A simple rule set (known as a *mode file*) has been developed (Winn, 1998) to indicate the different components of a CIF, as a first step towards a syntax-checking *emacs* mode.

The *Star.vim* utility of Section 5.2.4 provides a similar functionality for editing in the *vim* environment, although it is not capable of validation directly; nevertheless, the appearance of unexpected or irregular highlighted text can draw the user's attention to syntactic problems, a feature that is also useful in more extended editors such as *enCIFer* (Section 5.3.3.1).

5.3.2.1. *vcif*

A simple syntax checker for CIF is the program *vcif* (McMahon, 1998), which scans a text file and outputs informative messages about apparent errors. While conservative CIF parsing software will quit upon finding an error, *vcif* will attempt to read to the end of the file and list all clearly distinguished errors. However, its interpretation of errors depends on a close adherence to the CIF syntax specification and makes no assumption about the intended purpose of the character strings it reads. In consequence, a single logical error such as failing to terminate a multiple-line text string may cause the program to report many other apparent errors as it proceeds out of phase through the rest of the file.

5.3.2.1.1. *How to use vcif*

The program may be run under Unix or DOS by typing

```
vcif filename
```

where *filename* is the name of the file to test. If *filename* is given as the hyphen character -, the program will read standard input. Standard input will also be read if no file name is supplied; this allows the program to be used in a pipeline of commands.

A number of options may be supplied to the program to modify its behaviour. Without these options (*i.e.* invoked as above) a brief but informative message is written to the standard output channel for each occurrence of what the program perceives to be a syntax error.

For example, for the incorrect sample file of Fig. 5.3.2.1(*a*), the output is listed in Fig. 5.3.2.1(*b*).

Note that the sequence number of the line in which the error occurs is printed. The summary error message is output on a single line (longer lines have been wrapped and indented in Fig. 5.3.2.1 for legibility). Where the type of error necessarily affects only a single line, the program can recover and correctly identify errors on subsequent lines. Where possible, unexpected character strings are printed to help the user to identify the error. No attempt is made to assign any meaning to the data names or the data values in the

Affiliation: BRIAN MCMAHON, International Union of Crystallography, 5 Abbey Square, Chester CH1 2HU, England.

references