Acta Cryst. (1993). C**49**, 418–423

International Union of Crystallography
'How does my CIF become a printed paper?'

By B. McMahon

*The Technical Editor's Office, International Union of Crystallography, 5 Abbey Square, Chester CH*1 *2HU, England*

Introduction

In 1990 the IUCr adopted the Crystallographic Information File (CIF) as its approved standard for crystallographic data storage and information interchange. From the beginning of 1992, authors have been encouraged to submit structural papers to IUCr journals in CIF format, and over 200 such submissions were received in 1992. The CIF is a flexible and extensible file structure, which allows free intermixing of data fields with precise meaning and free-form text fields. It is thus possible to use the same file as a primary archive file for a crystal structure computer program, for a database, or as a submission for publication in a journal. However, there are some conflicts between these various objectives (a program may require specific values chosen from an enumerated list, while a paper may discuss at length some non-standard behaviour of an experimental parameter). To reconcile such conflicts as far as is possible, a certain amount of care is necessary in composing the CIF. The publication process, in particular, involves handling of the data within the CIF in a moderately complex way, and in a way, furthermore, that is unlikely to be self-evident to a prospective author. The purpose of this note is, therefore, to explain, in some detail, the way in which *Acta Crystallographica* transforms a CIF into a 'Regular Structural Paper' in Section C of the journal.

'What is in the CIF?'

A CIF is an example of a Self-Defining Text Archive and Retrieval (STAR) file, as described by Hall (1991). In brief, such a file may contain arbitrary data. Each item of data is identified by a 'tag', or data name, preceding it in the file. Thus, the appearance of the (tag, value) pair

```
_cell_length_a      8.79(2)
```

both identifies the presence of an item of data (here, the length of one side of the crystal unit cell) and yields its value. Where repetitive data occur, as for example in a list of atom positions, the tags are clustered together in a 'loop' declaration, and the data values then follow in the order specified. The example should make this clear:

```
loop_
 _atom_site_label
 _atom_site_fract_x
 _atom_site_fract_y
 _atom_site_fract_z
   O1  .4154(4)  .5699(1)  .3026(1)
   C2  .5630(5)  .5087(2)  .3246(1)
   C3  .5350(5)  .4920(2)  .3997(1)
   N4  .3570(3)  .5558(1)  .4167(1)
   C5  .3000(5)  .6122(2)  .3581(1)
```

All the data for a particular application are gathered together in a block, prefixed by a code such as `data_QQQQ`, where *QQQQ* is some arbitrary label. Within a data block, data names must appear once only, but several data blocks may appear in the file.

Lines in the CIF are 80 characters or less in length. Data names begin with an underscore character. The data may consist of numbers, letters or other printable characters from the ASCII character set. If spaces are to appear in the data string, delimiter characters must surround it: in the case of CIF, these are single or double quotes for strings less than 80 characters long, and semicolons at the beginning of a line for text extending over more than one line. Items (data name 'tags', data items or special `loop_` and `data_QQQQ` codes) are separated by spaces, tabs or newline characters. Comments may appear in the file, preceded by a hash character.

These simple rules essentially define the CIF structure. They are advantageous to the prospective author, in that they allow him (or her) to mix data of use only in local applications with the 'public' data that he wishes to exchange with colleagues or submit for publication. They also permit the CIF to be formatted in a neat fashion, so that a human reader may easily scan the listing of a CIF and be guided by simple layout clues as to the nature and position of data within the file.

However, they also pose novel and sometimes difficult problems for software which is designed to extract and manage the stored data. First, the file must be sound in its logical structure. Software that attempts to retrieve specific data relies on being able to identify a portion of a character stream as a data name, number or character item, or as a portion of continuous (but ultimately bounded) text. So the omission of a final semicolon from a text field can cause parsing software to lose its bearings. In a similar manner, a string of characters output from a Fortran program with a well defined `FORMAT` statement may cause havoc if it includes blanks that impel the CIF parser to treat the string as a number of distinct fragments. In practice, such errors in structure are usually easy to detect. The CIF data management program *QUASAR* (Hall & Sievers, 1990) (about which more later) may be run in a test mode to indicate structural errors of this type.

Secondly, there is no way of knowing in advance what the file does contain, and so processing software needs to be built with as few assumptions as possible regarding the presence or absence of a specific data item. This constraint is not unduly troublesome for the publisher, as traditionally the contents of a paper are not fully pre-ordained; but a program to handle crystal cell parameters, for instance, ought to be sufficiently robust that it can respond correctly if, say, the orthogonal angles of a monoclinic cell are quoted as 90, 90.0, or are absent altogether, or are represented by dummy placeholding characters ('?').

'How can a CIF represent an *Acta* paper?'

The original plan for the use of a CIF as a publication vehicle was to have a single text field, say `_publ_manuscript_text`, containing the entire text of a paper. It was further envisaged that fields might also be used that contained the coding for a formatted version of the paper, as generated by a word-processing package. Because the CIF is constrained to use the ASCII printable character set, such a coding would have to be in an ASCII-based formatting language, such as TeX or troff, or in an ASCII 'dump' format, such as SGML or RTF.

However, it was then realised that, at least for structural papers in *Acta* C, which conform rather closely to a set of rules detailed in *Notes for Authors*, a different approach could be followed. Most of the information required by the Commission on Journals on the experimental aspects of the paper and on the derived numeric results would already have been written to the CIF by the structure solution and refinement software being used. Rather than repeat all this information within the `_publ_manuscript_text` field, the individual data could be collected from the appropriate fields, and woven together in the order required for publication. There would still need to be free text fields for the author's comment and discussion, but these could be assigned to separate data fields, and the author need therefore enter only a small amount of text at the end (or beginning) of a CIF. The numeric data within the CIF, generated by some structure package, need never be retyped, and so transcription errors could be eliminated.

'How is the *Acta* paper built?'

The first step, then, is to extract from an incoming CIF those data fields which are required in the paper, and in the correct order. The program *QUASAR*, already mentioned, allows this to be done easily. *QUASAR* reads a request list of data items, scans the CIF, and outputs those items present in the file in the order of the request list. The output from *QUASAR* is itself in STAR format. If an item is not found, a message to this effect is output by *QUASAR* at the appropriate point. For example, if the request list included

```
_cell_angle_alpha
_cell_angle_beta
_cell_angle_gamma
```

and the CIF detailed a monoclinic cell, but only a value for β was included, the relevant portion of the *QUASAR* output would read

```
_cell_angle_alpha
          ?  # requested item not found
_cell_angle_beta          92.13(1)
_cell_angle_gamma
          ?  # requested item not found
```

(note the presence of the 'placeholder' query symbols). A request list is thus constructed that indicates all the items required for the *Acta* paper, and *QUASAR* generates a derivative CIF that has only these items (and indications of the absence of any of them) in the correct order.

Fig. 1 is a simplified version of part of the request list that is used for *Acta* C. A complete list can be obtained as the file **request.lst** from the IUCr *sendcif* mail server. [See *Acta Cryst.* (1992). C**48**, 408 for full details on accessing the mail server at the Technical Editor's office.]

```
data_QQQQ

_journal_coden_ASTM
_journal_coeditor_code
_journal_techeditor_code
_journal_year
_journal_volume
_journal_issue
_journal_page_first
_journal_page_last
_publ_section_title
_publ_author_name
_publ_author_address
_journal_date_recd_electronic
_journal_date_accepted
_publ_section_abstract
_publ_section_comment

_chemical_formula_analytical
_chemical_formula_moiety
_chemical_formula_sum
_chemical_formula_structural
_chemical_formula_weight
_symmetry_cell_setting
_symmetry_equiv_pos_as_xyz
_symmetry_space_group_name_H-M
_cell_length_a
_cell_length_b
_cell_length_c
_cell_angle_alpha
_cell_angle_beta
_cell_angle_gamma
_cell_measurement_reflns_used
_cell_measurement_theta_min
_cell_measurement_theta_max
_cell_volume
_cell_formula_units_Z
_exptl_crystal_density_diffrn
_exptl_crystal_density_meas
_exptl_crystal_density_method
_diffrn_radiation_type
_diffrn_radiation_wavelength
_exptl_absorpt_coefficient_mu
_exptl_absorpt_coefficient_mu_
_cell_measurement_temperature
_exptl_crystal_description
_exptl_crystal_size_max
_exptl_crystal_size_mid
 . . .
```

Fig. 1. Portion of a request list used to retrieve from a CIF the information that will be published in an *Acta* paper.

A complication

This is all very well for a simple CIF which contains only a single data block, where all the information relating to a structure and all the text of the paper are included in that data block. However, a large proportion of papers in *Acta* contain comparative (or at least collective) studies of several compounds, and it is necessary to allow such papers to be presented in CIF format. The strategy is as follows. Each individual structure should be described in a single data block. The text of the paper may appear in any of these data blocks, or in a separate data block.*

A composite request list is now constructed for *QUASAR*. The master request list, described above, is divided into a number of sections (currently three, representing the front matter and discursive text of the paper; the experimental section; and the references, figure captions and other end matter). Each portion of the request list is then applied in turn to each data block.

Consider a CIF which has two data blocks. One, which we shall call `data_A`, has the data for structure A together with the names and addresses of the authors. The other, `data_B`, has the text of the paper and the structural data for compound

* There is some attraction in the idea of requiring the textual data always to appear in its own block, and even to specify the name of this block (say, as `data_manuscript`). However, although this could simplify our processing, it is not mandatory (though we encourage authors to add this extra level of structure to their files).

```
data_A
_publ_section_title
_publ_author_name
_publ_author_address
_publ_section_abstract

data_B
_publ_section_title
_publ_author_name
_publ_author_address
_publ_section_abstract

data_A
_chemical_formula_sum
_chemical_formula_weight
_symmetry_cell_setting
_symmetry_equiv_pos_as_xyz
_symmetry_space_group_name_H-M
_cell_length_a
_cell_length_b
_cell_length_c
_cell_angle_alpha
_cell_angle_beta
_cell_angle_gamma

data_B
_chemical_formula_sum
_chemical_formula_weight
_symmetry_cell_setting
_symmetry_equiv_pos_as_xyz
_symmetry_space_group_name_H-M
_cell_length_a
_cell_length_b
_cell_length_c
_cell_angle_alpha
_cell_angle_beta
_cell_angle_gamma
```

Fig. 2. Composite request list designed to extract information from a two-structure CIF.

B (we are not recommending this as an ideal contribution!). Part of the composite request list that we create looks like Fig. 2. The output from *QUASAR* is reproduced as Fig. 3. Note how, if missing fields are ignored, the names, addresses and abstract appear at the beginning of the paper, followed by the experimental data for first one structure, then the other.

Some consequences of this strategy can now be seen. Because we have no knowledge of the meaning of the data block identifiers, we cannot determine in advance in what order to process them. In our example, if the abstract had been included in `data_A` and the authors' names in `data_B`, the abstract would have been printed before the authors' names! It would be fairly straightforward to change the order of data block names in the request list for our example, though this would require manual intervention in what is otherwise a fully automatic process, but the best approach is clearly to group together all related data within the same data block.

'How is the paper typeset?'

It is the output from the *QUASAR* run detailed above that represents the contents of the final paper. This output is itself in CIF format (strictly speaking, some deviations from the full CIF specification occur: the same data block name may appear more than once in the output file, and data names may also be repeated). A translation program, *ciftex*, is now run to translate the contents of the derived CIF to a file of TEX commands that may be processed by the TEX program (Knuth, 1984) to yield high-quality typeset material.

```
data_A
_publ_section_title
;
Two dimeric nickel(II) carboxylates
;

loop_
_publ_author_name
_publ_author_address
   'Morooka, Mitsuo'
; Department of Chemistry
 ...    Japan
;
   'Ohba, Shigeru'
; Department of Chemistry
   ...    Japan
;
   'Nakashima, Michio'
; Department of Chemistry
    ...    Japan
;
_publ_section_abstract ?  # requested item not present
#               -----end-of-data-block-----
data_B
_publ_section_title    ?  # requested item not present
_publ_author_name      ?  # requested item not present
_publ_author_address   ?  # requested item not present

_publ_section_abstract
;
The structures of two dimeric Ni^II^ carboxylates
and one dimeric Ni^II^ silanecarboxylate have been
determined.
;
#               -----end-of-data-block-----
data_A
_chemical_formula_sum              'C34 H54 N2 Ni2 O8'
_chemical_formula_weight                         736.2
_symmetry_cell_setting             monoclinic

loop_
_symmetry_equiv_pos_as_xyz
+x,+y,+z       -x,+y,1/2-z        -x,-y,-z
+x,-y,1/2+z    1/2+x,1/2+y,+      1/2-x,1/2+y,1/2-z
1/2-x,1/2-y,-z 1/2+x,1/2-y,1/2+z

_symmetry_space_group_name_H-M           'C 1 2/c 1'
_cell_length_a                             20.520(2)
_cell_length_b                             10.647(1)
_cell_length_c                             18.260(2)
_cell_angle_alpha                           90.00000
_cell_angle_beta                           91.015(8)
_cell_angle_gamma                           90.00000
#               -----end-of-data-block-----
data_B
_chemical_formula_sum          'C80 H68 Cl6 N2 Ni2 O8'
_chemical_formula_weight                        1515.5
_symmetry_cell_setting             triclinic

loop_
_symmetry_equiv_pos_as_xyz
+x,+y,+z    -x,-y,-z

_symmetry_space_group_name_H-M           'P -1'
_cell_length_a                             13.231(1)
_cell_length_b                             13.857(1)
_cell_length_c                             11.425(1)
_cell_angle_alpha                           99.48(1)
_cell_angle_beta                           104.63(1)
_cell_angle_gamma                          109.68(1)
#               -----end-of-data-block-----
```

Fig. 3. *QUASAR* output for the request list in Fig. 2.

The basic approach of *ciftex* is straightforward. It reads through the derived CIF, ignores missing data items, and translates simple (data name, data field) pairs into TEX macro calls with the data as argument. So the CIF entry

```
    _cell_length_a     8.79(2)
```

is translated to the TeX code sequence

```
\cella{8.79 (2)}
```

where `\cella` is a macro that is defined in a reference file, and which tells TeX to typeset its argument preceded by an italic *a*, an equals sign and a space, and followed by an ångström symbol. Note that a space has also appeared before the parentheses marking the e.s.d. in the argument — this is an adjustment to conform with house style that *ciftex* makes as it processes its input.

TeX is itself a programming language, and so the macros we create can modify their own behaviour, depending on their arguments. This can be beneficial. Consider this example. In the experimental section of the paper, we wish to print a chemical formula. However, there are at least four different `_chemical_formula_` data names given in the CIF Dictionary, and we do not know which, if any, actually appear in the CIF. We define macros such as `\chemformsum` (for `_chemical_formula_sum`) and `\chemform` (for `_chemical_formula_moiety`) which test for the truth value of a TeX variable. If this variable is true, the macro processes its argument (that is, prints the formula) and then sets the variable to false. Thus, if another chemical formula macro is encountered, it will recognise that a formula will already have been printed, and will discard its argument (that is, not print another formula).

Of course, such logic could be built into *ciftex*; but the advantage of expressing it in TeX is that the TeX macros can be redefined at runtime, without the need for recompilation of a program. Further, by restricting the amount of knowledge that *ciftex* requires for processing CIFs, we increase its generality.

On the other hand, there are things which TeX cannot do, or can achieve only with great difficulty, that can easily be coded into *ciftex*. For instance, *ciftex* will read the argument to the `_chemical_formula_` data names and itself parse the string for subscripts and superscripts. The relevant TeX code to indicate subscripts and superscripts is generated within *ciftex*. As a more complex example, symmetry codes applied to atoms in bond distance and angle listings are given within the CIF as numeric values (`4_556` *etc*., where the '4' refers to the fourth entry in the list of `_symmetry_equiv_pos_as_xyz`). These values apply to the data names `_geom_bond_site_symmetry_1` and so on. It is necessary to read the list of symmetry-equivalent positions, parse the numeric pointer to the symmetry operator required, retrieve the operator and store it in a footnote that will be printed at the foot of the relevant table, and then to indicate typographically which code is appropriate by appending a superscript roman numeral to the relevant atom label. This is all done within *ciftex*.

There is thus a choice of ways to process information from the CIF, and the adoption of a hard-coded approach in *ciftex* or of a TeX macro with inbuilt logic is made on purely pragmatic grounds on a case-by-case basis. This means that over time a body of expertise and knowledge must be built up in the Editorial staff which is not easy to pass on, except in a massive instruction manual. For this reason, it is likely that authors will rarely be in a position to appreciate all the details that they need to attend to in constructing a CIF that can be handled in the most efficient way possible. Nevertheless, some general pointers will be given elsewhere in this document.

'Is it as simple as that?'

Almost, but not quite. The procedure described above works for simple (that is, non-looped) associations of data name and data field. Where there are loops, the *ciftex* translator has to work a little harder. The strategy followed in this case depends on whether a given data name is recognised as a member of a 'loop' or a 'table'. There is no distinction within the CIF, but it is necessary to make a distinction in order to know how to format the output. There is a flag which achieves this in the 'map' file which *ciftex* reads to associate CIF data names and TeX codes. It is a single letter associated with each data name, 'T' if the data name is expected to occur in a table, 'N' if it is expected to occur only unlooped or in a non-tabular loop. For a non-tabular loop, the data names declared after the '`loop_`' entry in the CIF are remembered, and the corresponding TeX macros wrapped around each item of data. Thus, the list of authors and addresses might appear in the CIF as

```
loop_
  _publ_author_name
  _publ_author_address
  'Jones, A.'
  ADDRESS_OF_JONES
  'Smith, B.'
  ADDRESS_OF_SMITH
```

and this would be written to the TeX file as

```
\author{Jones}
\address{ADDRESS_OF_JONES}
\author{Smith}
\address{ADDRESS_OF_SMITH}
```

More interesting, though, is tabular material. In this case, each of the data names following the '`loop_`' declaration is mapped, not to a TeX macro name, but to some text that will be printed at the head of the column within the table that that data name represents. The number of different data names is counted, and the data items are identified by their position in the loop modulo the total number of data names (in effect, by their 'phase' in the loop). In the simplest case, a TeX command is output to build a table with *n* columns, where *n* is the number of different data names. Then the data items are counted as they are processed. After every *n*th data item, a TeX code is output indicating 'end of table row', and a further code is output before the next item (if there is one) that means 'beginning of new table row'. In all other cases, a code is output signifying 'move to next column'.

Some care is needed to ensure that missing data items are handled properly. Because of the nature of the CIF, no item in a loop can be truly missing, but non-assigned values can be indicated by the presence of a '?' character. So if such a '?' character is encountered, nothing is printed, but TeX code must still be generated to skip to the next column. However, if a complete column is missing from a table, we do not wish to print an empty column on the finished page, so the number of columns is decremented by one, and every item in the 'missing' column is skipped over completely.

This works for general tables. But in the geometry tables, it is not the house style of the journal to list the bonded atoms in separate columns; instead, they are tied together with chemical bond symbols. In these tables, therefore, the number of columns to be typeset has to be decremented by the number of bond symbols to be typeset, and the 'skip to next column' codes have to be replaced where necessary by chemical bond symbols. To complicate matters further, the symmetry codes are printed, again not in separate columns, but as superscripts to the atoms to which they apply, so again the number of table columns calculated has to be decremented. However, these are the most difficult cases to typeset, and most of the translation of looped material is easier to accomplish.

'OK, so I'm blinded by science. What can I do to help?'

The general philosophy we have been following is to make as few additional requests of the author as possible beyond the constraints laid out in the CIF specification paper of Hall, Allen & Brown (1991) – there is enough to do to ensure conformance with that! However, it might be useful to offer a few comments on individual data fields, with reference to the example file cited in the Hall *et al.* paper. The various entries are given in the order in which they appear in that example.

`_chemical_formula_` *fields*

These should be entered with bounding quotes, as in the example file, and with spaces as described in the CIF paper. They should not contain codes for subscripts or superscripts — these are automatically output by *ciftex*. (However, chemical formulae appearing elsewhere, as in `_publ_section_comment` or other free text fields, should have such codes.)

`_computing_` *fields*

These contain references for computer programs used at various stages of the experimental work, and thus should include an (author, date) entry corresponding to an entry in `_publ_section_references`. The example CIF does not do this. For example, the *SHELX* entry should read

```
'SHELXS86 (Sheldrick, 1986)'
```

`_symmetry_space_group_name_H-M`

The CIF Dictionary requires that a full name be given, so that $P2_1/n$ should be entered as `'P 1 21/n 1'`. To ensure compatibility with other software, this rule should be followed, although our system will also accept `'P 21/n'` in this case. Because it is the short form of the symbol that is published, *ciftex* translates the full form to its short form. This translation is based on pattern analysis, and not on table lookup, and is far from complete. Again, symbols for subscripts should not be entered in this field.

`_symmetry_equiv_pos_as_xyz`

This should be a loop of character strings, each representing one operation (*e.g.* `'x,y,z'`, `'1/2-x, y, 1/2+z'`). The latter example will also be treated correctly if the bounding quotes and spaces are dropped (to give `1/2-x,-y,1/2+z`) even though a parser might be inclined to treat this as a numeric entry (because of the leading '1'). Recall, too, that symmetry codes used in geometry tables refer back to this list.

`_reflns_observed_criterion`

The example CIF contains an excessively verbose entry here. Better would be

```
'F > 6.0\s(F)'
```

Note that the special typographic codes for Greek and other characters can be used in any character or text field.

`_publ_section_` *sections*

As text fields, these are delimited by semicolons in the first columns of the opening and trailing lines. Text may follow the opening semicolon on the same line (as in the `_publ_author_address` fields of the example CIF), or may start on the next line (as in `_publ_section_abstract`). A blank line should be used to separate paragraphs (and references in `_publ_section_references`). The text entries should *not* be double-spaced (*i.e.* a blank after every line) – this causes every line to be set as a new paragraph. The preprint version of the manuscript that is generated in Chester is programmed to output double-spaced type for ease of editing.

`*_special_details` *sections*

The CIF Dictionary has provision for a number of fields, such as `_refine_special_details`, whose main purpose is to indicate to users of the CIF any technical details of the refinement or other aspects of the experimental procedure. In general, these are expected to be of use to experimentalists wishing to reproduce the experiment, and are not intended for general publication. The general discussion of the experimental details should be contained in the `_publ_section_experimental` field. However, if it *is* wished to include the `*_special_details` fields in the published paper, each such field **must** be listed as an entry in the `_publ_manuscript_incl_extra_item` loop (see later).

Numeric data (*especially in tables*)

Some authors traditionally scale values of certain parameters, such as the fractional coordinates, to produce tables for publication that do not include embedded decimal points. It is easier to handle tables that give absolute values of quantities, decimal points and all, and we encourage authors to adopt this practice. However, if it is desired to have a table of coordinates scaled, let us say, by 10^4, the author should give each data item in the exponential notation [*i.e.* `4154E-4(4)` for .4154(4) – note that the convention is to insert the exponent term before the e.s.d.]. The Editorial staff will then manually remove all the '$\times 10^{-4}$' terms which appear.

It is not necessary for authors to insert leading zeroes before decimal points in the CIF, although they are encouraged to do so. Leading zeroes are automatically inserted by *ciftex*.

'I can't possibly remember all this!'

Can anyone? Still, help is at hand. A program, *cifms*, is now available from the *Acta* editorial office. It will process a CIF in the same way as *ciftex*, but instead of writing TeX code, it will output an ASCII-only 'preview' version of the manuscript generated. So, although the author will not be able to generate Greek characters or proper subscripts, he will be able to see whether and how his sub/superscript and Greek codes will be handled. He will also be able to see how the paper will be structured, and can experiment with the presentation of data in his CIF to fine tune the way it will be handled before it is sent to *Acta* (see Fig. 4). The present version requires a Unix system with a C compiler; the prospective user must also have access to a copy of *QUASAR*. A shell archive (shar) file containing *cifms* and some ancillary files is available from the IUCr mail server by sending the mail message

```
send cifms.shar
```

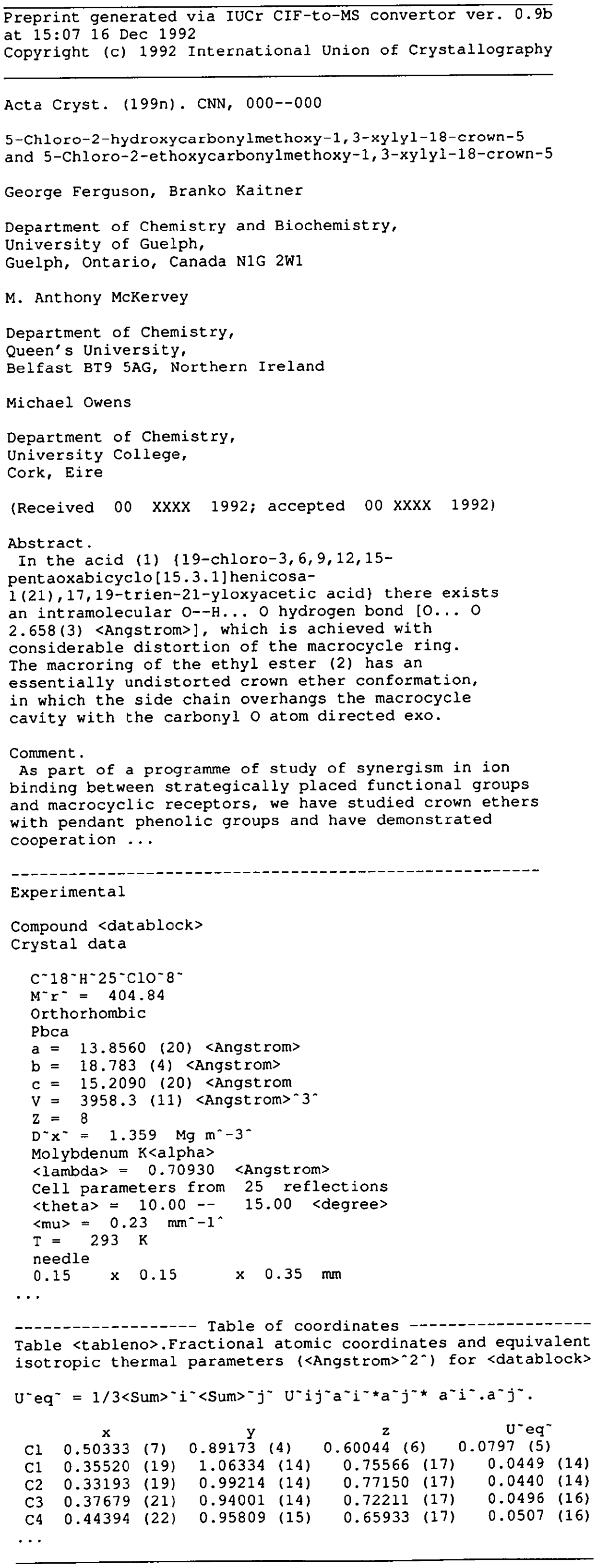
to **sendcif@iucr.ac.uk** (or to **sendcif@uk.ac.iucr** within the UK). If the program *QUASAR* is also required, it may be obtained by adding the line

```
send quasar.src
```

to such a message.

'I want to include more information in my *Acta* paper than is allowed for in *Notes for Authors*'

Notes for Authors gives a list of the data names which will be extracted from a CIF to form the paper that will be published in *Acta*. An author is free to include any additional comments in the most appropriate section



```
Preprint generated via IUCr CIF-to-MS convertor ver. 0.9b
at 15:07 16 Dec 1992
Copyright (c) 1992 International Union of Crystallography
------------------------------------------------------------

Acta Cryst. (199n). CNN, 000--000

5-Chloro-2-hydroxycarbonylmethoxy-1,3-xylyl-18-crown-5
and 5-Chloro-2-ethoxycarbonylmethoxy-1,3-xylyl-18-crown-5

George Ferguson, Branko Kaitner

Department of Chemistry and Biochemistry,
University of Guelph,
Guelph, Ontario, Canada N1G 2W1

M. Anthony McKervey

Department of Chemistry,
Queen's University,
Belfast BT9 5AG, Northern Ireland

Michael Owens

Department of Chemistry,
University College,
Cork, Eire

(Received 00 XXXX 1992; accepted 00 XXXX 1992)

Abstract.
 In the acid (1) {19-chloro-3,6,9,12,15-
pentaoxabicyclo[15.3.1]henicosa-
1(21),17,19-trien-21-yloxyacetic acid} there exists
an intramolecular O--H... O hydrogen bond [O... O
2.658(3) <Angstrom>], which is achieved with
considerable distortion of the macrocycle ring.
The macroring of the ethyl ester (2) has an
essentially undistorted crown ether conformation,
in which the side chain overhangs the macrocycle
cavity with the carbonyl O atom directed exo.

Comment.
 As part of a programme of study of synergism in ion
binding between strategically placed functional groups
and macrocyclic receptors, we have studied crown ethers
with pendant phenolic groups and have demonstrated
cooperation ...

-------------------------------------------------------
Experimental

Compound <datablock>
Crystal data

  C~18~H~25~ClO~8~
  M~r~ =  404.84
  Orthorhombic
  Pbca
  a =  13.8560 (20) <Angstrom>
  b =  18.783 (4) <Angstrom>
  c =  15.2090 (20) <Angstrom
  V =  3958.3 (11) <Angstrom>^3^
  Z =  8
  D~x~ =  1.359  Mg m^-3^
  Molybdenum K<alpha>
  <lambda> =  0.70930  <Angstrom>
  Cell parameters from  25  reflections
  <theta> =  10.00 --   15.00  <degree>
  <mu> =  0.23  mm^-1^
  T =   293  K
  needle
  0.15    x  0.15      x  0.35  mm
...

------------------ Table of coordinates ------------------
Table <tableno>.Fractional atomic coordinates and equivalent
isotropic thermal parameters (<Angstrom>^2^) for <datablock>

U~eq~ = 1/3<Sum>~i~<Sum>~j~ U~ij~a~i~*a~j~* a~i~.a~j~.

        x              y              z             U~eq~
 Cl  0.50333 (7)   0.89173 (4)   0.60044 (6)   0.0797 (5)
 C1  0.35520 (19)   1.06334 (14)    0.75566 (17)    0.0449 (14)
 C2  0.33193 (19)   0.99214 (14)    0.77150 (17)    0.0440 (14)
 C3  0.37679 (21)   0.94001 (14)    0.72211 (17)    0.0496 (16)
 C4  0.44394 (22)   0.95809 (15)    0.65933 (17)    0.0507 (16)
...
```

Fig. 4. Extracts from a *cifms* output.

of the running text (usually `_publ_section_comment` or `_publ_section_experimental`), and this is where complex tables that cannot simply be generated from the coordinates and geometry data fields should be placed. Note, by the way, that the permitted data name `_publ_section_table_legends` is of little use for standard structural papers, since the legends for the standard tables are generated automatically. It is really a leftover from earlier considerations of how the paper might be structured, and can usually be ignored.

However, there are cases when the CIF generated by the author's software may already contain data in recognised fields (that is, with data names that appear in the CIF Dictionary) that he wishes to have published. In such a case it would appear perverse to have to duplicate this information in the free text sections. The appropriate course of action is then to include a list of the additional data items that should be extracted from the CIF in a `_publ_manuscript_incl_extra_` loop. That is, if the author requires a table of contact distances to be published, he should add to his CIF the entry

```
loop_
 _publ_manuscript_incl_extra_item
      '_geom_contact_atom_site_label_1'
      '_geom_contact_atom_site_label_2'
      '_geom_contact_site_symmetry_1'
      '_geom_contact_site_symmetry_2'
      '_geom_contact_distance'
      '_geom_contact_publ_flag'
```

The bounding quotes around the extra items required are essential. This mechanism is explained in the CIF Dictionary, together with comments on the additional information that may be conveyed in `_extra_info` and `_extra_defn` fields (these latter fields are not mandatory). This facility was deliberately not emphasised in the CIF specification paper, because it can be a little problematic to know how to splice into the standard skeleton of an *Acta* structural paper some of the more esoteric data names. Nevertheless, it is a feature which can be of significant use if employed judiciously.

'Mais je veux parler en français!'

It is permissible to submit papers to *Acta* in French, German and Russian, and so these languages will also be supported for CIF submissions. At the time of writing, procedures have been developed for French and German papers.

A CIF containing a paper in a language other than English must, of course, have the recognised ('English') data names. Certain data fields, as for example the `entry_symmetry_cell_setting`, must contain the approved codes, which in this case are English words ('`monoclinic`' must be entered; it will automatically be translated by *Acta* software to 'monoclinique'). Beyond this, other character fields and free text fields should contain text in the language of submission (though the Commission on Journals still requires that the text of `_publ_section_abstract` be in English).

References

Hall, S. R. (1991). *J. Chem. Inf. Comput. Sci.* **31**, 326–333.

Hall, S. R., Allen, F. H. & Brown, I. D. (1991). *Acta Cryst.* A**47**, 655–685.

Hall, S. R. & Sievers, R. (1991). *QUASAR: a Program for Accessing a STAR File*. Univs. of Western Australia, Australia, and Bonn, Germany.

Knuth, D. E. (1984). *The TEXbook*. Reading, MA: Addison-Wesley.