mmCIF Software Tools

Shu-Hsin Hsieh
Steven Schirripa
John D. Westbrook
Nucleic Acid Database Project
Department of Chemistry
Rutgers, The State University of New Jersey
ndb.rutgers.edu

Abstract

A collection of software tools are presented which provide a programming interface for macromolecular Crystallographic Information File (mmCIF) dictionaries and data files. These tools are part of a software framework developed by the Nucleic Acid Database (NDB) [1, 2] to address practical problems encountered in processing and validating data in mmCIF format. This software framework includes both relational and object-oriented programming interfaces to mmCIF as well as a simple set of tools which facilitate the exchange of mmCIF data. In addition to the description of the architectures of the programming interfaces and software tools, this tutorial will provide practical applications of each programming interface and will focus on how these tools may be used to integrate mmCIF data into new and existing applications. Participants are encouraged to bring application programs for which they wish to construct an mmCIF interface.

1 Introduction

The Crystallographic Information File (CIF) [3] format has become a standard interchange format for the description, archiving, and publication of crystallographic experiments on small organic and inorganic molecules. The remarkable feature of the CIF interchange approach is the use of an electronic dictionary to provide a detailed description of each item of data in a CIF. The task of developing a dictionary of terminology for macromolecular crystallography and structure has been underway for several years [4] and is in near final form [5, 6].

The syntax of mmCIF is derived from the Self-defining Text Archive and Retrieval (STAR) format proposed by Hall [7]. mmCIF data files employ a simple grammar in which each data value or list of data values is accompanied by a data item name. Each item of data has a detailed definition, and these definitions are collected in the mmCIF dictionary. The specification for each mmCIF data definition is separately defined in another dictionary. The latter dictionary defines a Dictionary Description Language (DDL) which provides the framework for constructing the mmCIF dictionary definitions. Both of these dictionaries are expressed using the simple mmCIF syntax. Figure 1 shows an example of a fragment of a mmCIF data file, a mmCIF dictionary entry, and a DDL dictionary entry illustrating the uniform mode of expression used in each of the cases.

Data File

```
_cell.entry_id                    '5HVP'
_cell.length_a                    58.39
_cell.length_a_esd                 0.05
_cell.length_b                    86.70
_cell.length_b_esd                 0.12
_cell.length_c                    46.27
_cell.length_c_esd                 0.06
```

Dictionary Definition

```
save__cell.length_a
    _item_description.description
;Unit-cell length a corresponding to the structure reported.
;
 _item.name                   '_cell.length_a'
 _item.category_id             cell
 _item.mandatory_code          no
 _item_sub_category.id        'cell_length'
 _item_aliases.alias_name     '_cell_length_a'
 _item_aliases.dictionary     'cifdic.c94'
 _item_aliases.version        '2.0'
 _item_related.related_name   '_cell.length_a_esd'
 _item_related.function_code  'associated_esd'
 _item_type.code               float
 _item_type_conditions.code    esd
 _item_units.code             'angstroms'
   save_
```

DDL Definition

```
save__item_description.description
    _item_description.description
;   Text decription of the defined data item.
;
   _item.name                '_item_description.description'
   _item.category_id          item_description
   _item.mandatory_code       yes
   _item_type.code            text
    save_
```

Figure 1 Abbreviated examples of CIF data specifications, CIF dictionary definitions, and DDL definitions.

The uniform appearance of mmCIF data files and mmCIF dictionaries as collections of name and value pairs is dictated by STAR syntax rules; however, the underlying organization of the data and definitions is determined by the DDL. The DDL used in

developing the mmCIF dictionary [8, 5] is an extension of the DDL proposed by Cook and Hall [9, 10] for the core dictionary of crystallographic definitions. These DDL extensions were designed primarily to make the content of the mmCIF definitions more accessible to software that might use this information for data integrity validation.

2 The Organization of mmCIF

The basic element of information in a mmCIF data file is an individual data item such as a structure factor. Collections of related data items may be grouped together in subcategories. For instance, in the mmCIF dictionary the x, y, and z cartesian components are assigned to the *cartesian_coordinate* subcategory.

Data items may also be organized into categories. A category is a logical association among a group of data items requiring that the value(s) of one or more of the items in the group can be used to distinguish between different instances of the group. A category has many of the properties of a table in a relational database. For example, the mmCIF category *atom_site*, which holds the table of atomic positions, uses the data item *atom_id* as the unique identifier or key for each atomic position.

In organizing the data definitions in the mmCIF dictionary into categories, it was found that many different categories shared common sets of unique identifiers. For instance, the definition of protein secondary structure includes the residue labels that define the limits of each structural feature. These residue labels also occur in the definition of the molecular sequence and in the definition of each atomic position. In some cases, it is important that a secondary structure description references only those residue labels for which positions have been determined. The fact that the residue labels, although used in these different contexts, refer to the same item of information is defined within the mmCIF dictionary by specifying a parent/child relationships between the residue labels items in the different categories. Because parent/child relationships are prevalent at all levels of macromolecular structure description, it is an important feature of the mmCIF data description that the relationships between common data items in different categories is precisely described.

Collections of related categories in the mmCIF dictionary are organized into category groups. Category groups provide a mechanism for expressing associations among categories in the same manner as chapters organize related sections in books. For example, in the mmCIF dictionary all of the categories pertaining to refinement are assigned to a category group named *refine_group*.

The highest level of organization provided by mmCIF is the data block. Each data block acts like an independent database. A data block is used to contain the mmCIF dictionary, and it may be used to hold all of the information pertaining to a particular structural experiment.

3 The CIFLIB Class Library

CIFLIB [11, 12] is a software library that was developed to provide an application interface to information in CIF-format.

The entity relationship diagram of the CIFLIB class library in Figure 2 illustrates that CIFLIB follows the essentially relational organization used in the mmCIF dictionary. CIFLIB provides functions which perform the following types of operations:

- read and write operations on CIF format data files and dictionaries.
- read, write and update operations on individual data items and dictionary definition components.
- detailed integrity checks of CIF data and dictionaries as defined by the Dictionary Description Language (DDL) 2.1 [13, 14].
- efficient access to the CIF dictionary data model.
- robust syntactic and semantic error handling.

Figure 3 shows how this software library facilitates integrating the CIF interchange format with other applications. As the figure illustrates, CIFLIB provides complete access to the DDL, CIF dictionaries and CIF data files. This library can be used to build wrappers and filters around existing applications which need to access CIF data. Since CIFLIB provides complete access to the dictionary data model, the library can be conveniently used as an in-memory database or as a loader for an external database.

Accessing data in CIF format using CIFLIB is a multistep process. CIFLIB first reads the DDL dictionary. Once the DDL is loaded and checked against internally coded rules based on DDL 2.1, a CIF dictionary which is based on this DDL is read and checked. This process can be quite time consuming for large dictionaries, so a provision has been made to retain the state of any file which has been checked in an auxiliary file. This auxiliary file will be used in preference to the original file in subsequent file accesses if its modification date is more recent. Finally, CIF data files are read and checked

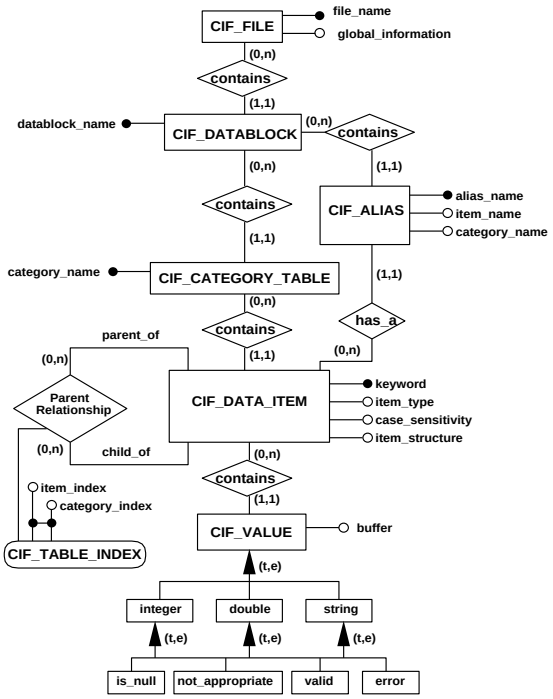

Figure 2 Extended entity relationship diagram of the CIFLIB class library.

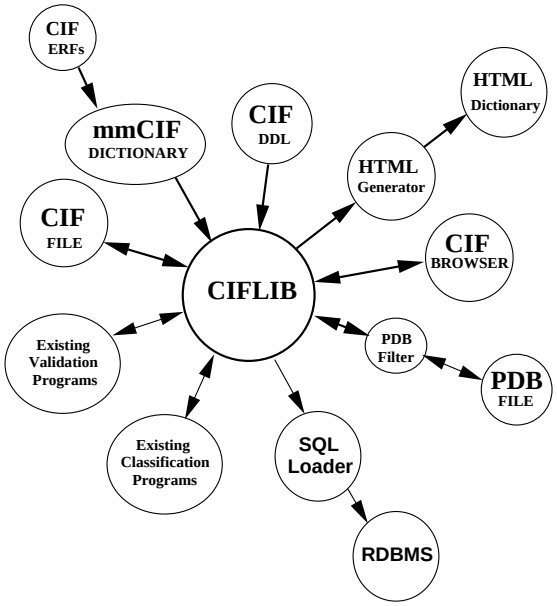

Figure 3 Functional diagram of CIFLIB illustrating the intended use of the library in supporting CIF access for a variety of application program types.

with respect to the CIF dictionary. In any file access, CIFLIB provides complete access to the data blocks containing the DDL, the CIF dictionary, and any number of blocks containing user data.

Each CIF file may be divided into data block sections. CIFLIB treats each data block as an independent database loaded into the data model defined in its associated dictionary. The CIF DDL is at the top of the chain and provides the data model for a CIF dictionary. The CIF dictionary in turn provides the data model for CIF data files. CIFLIB provides functions to read, write and merge data blocks. Any number of data blocks can be managed by the library.

Within each individual data block, category groups provide a mechanism for organizing categories into conceptually meaningful collections. CIFLIB provides functions to obtain the list of category groups defined within a data block as well as the names of the member categories of each group.

The library provides a set of functions for accessing category-level features within a data block. These functions provide a complete list of the categories specified within a data block, the list of data items specified within each category, and the number of rows of data in a category. The attributes of a category defined in the CIF dictionary such as the category description, category examples, member data items, key data items, and member subcategories can also be obtained.

Functions are provided to read, write and update individual data items, rows of data items, and columns of data items. These functions also check the integrity of item values with respect to their dictionary definitions. Access to all of the item attributes defined in the CIF dictionary is provided, and convenience functions are provided for the most commonly used attributes such as alias names, data type, default value, and enumeration.

CIFLIB provides a set of functions which give information about parent/child relationships, and provide access to the parent and child item values. The parent and child relationships returned by the functions in this section span a single generation; however, complicated hierarchies of parentage can be easily traced.

Although the extended DDL uses different conventions for naming data, it provides a mechanism to reference alternative data names. The mmCIF dictionary uses this feature to show the correspondence between the mmCIF data items and the existing core CIF data items. Because the mmCIF dictionary incorporates all of the definitions in the

core CIF dictionary, it is possible for software developed for the extended DDL to use the mmCIF dictionary to read, write and check data items derived from either dictionary.

4 The CIFLIB C Language Application Program Interface

A C language application program interface to CIFLIB [12] has been developed to provide a convenient functional interface to the CIFLIB class library. A reference manual which describes each interface function in detail is available [12].

CIFLIB provides a set of functions which access the error codes generated by those library functions which perform integrity checking. The CIFLIB functions which access and update individual item values return only a single error code. Functions providing read access return only the first error encountered in checking the target item. Similarly, functions providing update access return only the first error encountered in the checking process; however, all of the errors that may be detected during an I/O operation are appended to the warning or error lists maintained for each data block. Higher-level functions, which read and write files and data blocks, also append their diagnostic codes to internal error and warning lists. A set of functions has been provided to access and refresh these lists. Functions are also provided to translate individual error codes and to print the contents of an entire data block.

To demonstrate some of the functionality of the C API, a dictionary-to-HTML converter application was developed to provide flexible access to the contents of the mmCIF and DDL dictionaries on the World Wide Web [1]. The HTML mmCIF dictionary is organized so that a user can flexibly navigate through the hierarchy of the definitions and between all data item relationships. The first level of presentation is a page of category groups and group descriptions. The contents of each category group can be explored and specific categories within each group can be selected. Each category is presented on a page which includes all of the DDL attributes pertaining to the category description. From within the category presentation, individual data items can be selected. The data item presentation includes all of the relevant DDL attributes and selections for all related data items. Each of these levels of presentation is illustrated in Figures 4 - 7.

5 The CIFPARSE Function Library

In many instances it is desirable to be able to read and write information in CIF format without the

[1] The HTML dictionaries are available at http://ndbserver.rutgers.edu/mmcif/.

Figure 4 The CIF dictionary-to-HTML screen showing the category group organization of the dictionary.

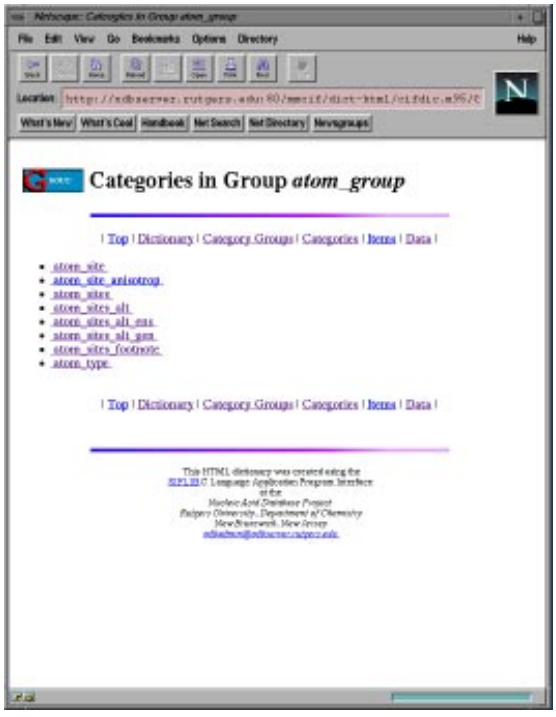

Figure 5 The CIF dictionary-to-HTML screen showing the categories in the category group *atom_group*.

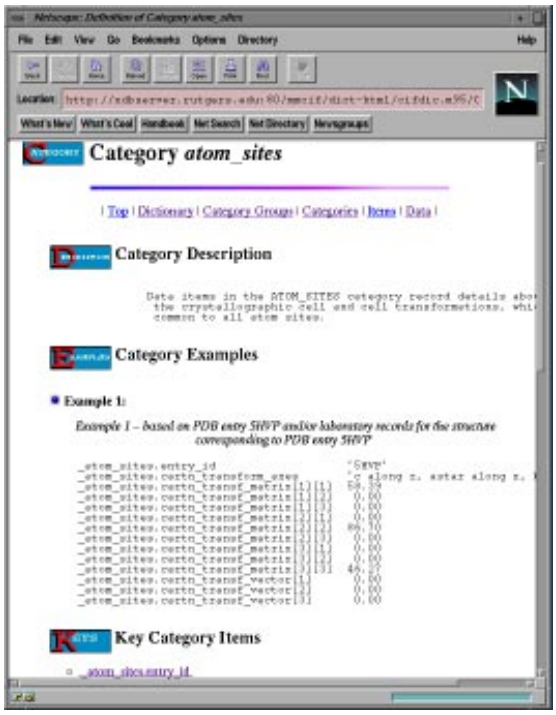

Figure 6 The CIF dictionary-to-HTML screen showing the description of the category *atom_sites*.

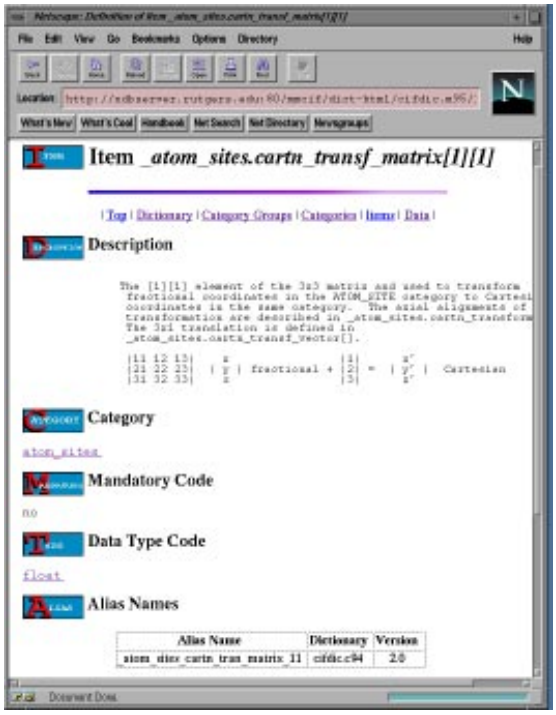

Figure 7 The CIF dictionary-to-HTML screen showing a portion of the description of an individual data item.

overhead of dictionary based integrity processing. The CIFPARSE function library was developed to provide convenient tools to access CIF data without any semantic checking (Figure 8). CIFPARSE uses a simple lex/yacc parser to read and check the syntax of mmCIF data files. CIF data are stored as strings in a simple data structure and a collection of access functions is provided to retrieve individual items of data.

This library is particularly useful for applications that exchange CIF data between program applications or for repetitive access to static CIF data that has been previously semantically validated. A reference manual which describes each function in detail is available [15].

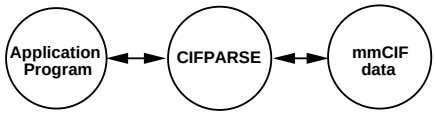

Figure 8 Functional diagram of the CIFPARSE function library illustrating the intended use of the library for processing mmCIF data files.

6 The CIFOBJ Class Library

The design of the CIFLIB class library closely follows the relational model that is specified in DDL 2.1. Categories in mmCIF dictionaries and data files are mapped into tabular data structures in the class, and the data access methods provided by the class are all row oriented. This design is satisfactory for applications which access CIF data in the CIF dictionary one category at a time; however, it is not particularly efficient for applications which access dictionary data item-wise. This pattern of access is common for applications which have to assemble all of the dictionary attributes of a data item for the purpose of integrity processing.

The CIFOBJ class library was developed in order to provide an object view of the mmCIF dictionary. The class library has two components as illustrated in Figure 9. The first component builds a persistent store of objects of type: item, subcategory, category, and dictionary. Each object is a container for all relevant attributes for that object type. CIFBOBJ accesses and checks the dictionary contents using methods provided by CIFLIB. The CIFOBJ loader class assembles the dictionary objects and passes these to the object storage manager.

The second component of the CIFOBJ class library provides methods for building dictionary objects from the persistent store. CIFOBJ implements methods to access all of the attributes for each object

type and returns this information as a string or an array of strings.

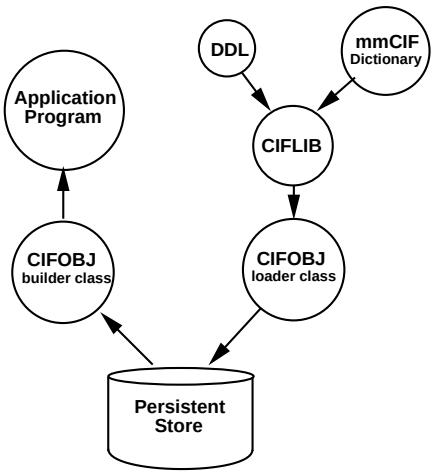

Figure 9 Functional diagram of CIFOBJ class library illustrating the dictionary object loader and the object builder components of the class.

7 Availability

All of the code described here was developed with the GNU C/C++ compiler and has been tested on a variety of Unix platforms that support the 2.72 or later versions of the GNU compiler (*e.g.* Silicon Graphics (IRIX 5.3, 6.1), SUN (SUNOS 4.13), and Hewlett-Packard (HP-UX 9.05)). These packages and example applications are available at http://ndbserver.rutgers.edu/software/ or ftp://ndbserver.rutgers.edu/pub/programs/. Reference documentation is available in both PostScript and HTML format at this site.

8 Acknowledgment

For their contributions to the development of the mmCIF dictionary we greatfully acknowledge the members of the Macromolecular CIF working group: Paula Fitzgerald at Merck Research Laboratories, Phil Bourne at San Diego Supercomputer Center, and Helen Berman at Rutgers. Thanks also go to Brian McMahon at the IUCr Office in Chester for many valuable discussions on core dictionary compatibility. This work was supported by the NSF (BIR9305135 & BIR9510703).

References

[1] H. M. Berman, W. K. Olson, D. L. Beveridge, J. D. Westbrook, A. Gelben, T. Demeny, S. Hsieh, A. R. Srinivasan, and B. Schneider. Nucleic Acid Database: A Comprehensive Relational Database of Three-Dimensional Structures of Nucleic Acids. Biophys. J., 63:751, 1992.

[2] Helen M. Berman, Anke Gelbin, Lester Clowney, Shu-Hsin Hsieh, Christine Zardecki, and John Westbrook. The Nucleic Acid Database: Present and Future. J. Res. Natl. Inst. Stand. Tech., 101:243, 1996.

[3] S. R. Hall, F. H. Allen, and I. D. Brown. A new standard archive file for crystallography. Acta Crystallogr., A47:655–685, 1991.

[4] P. M. D. Fitzgerald, H. M. Berman, P. E. Bourne, and K. Watenpaugh. Macromolecular CIF Working Group. International Union of Crystallography, 1992.

[5] P. Fitzgerald, H. M. Berman, P. Bourne, B. McMahon, K. Watenpaugh, and J. D. Westbrook. The Macromolecular Crystallographic Information File Dictionary. IUCr, http://ndbserver.rutgers.edu/mmcif, 1995.

[6] Philip E. Bourne, Helen M. Berman, Brian McMahon, Keith D. Watenpaugh, John Westbrook, and Paula M. D. Fitzgerald. The Macromolecular Crystallographic Information File (mmCIF). Methods in Enzymology, 1995. submitted.

[7] S. R. Hall. The STAR File: A new format for electronic data transfer and archiving. J. Chem. Inf. Comput. Sci., 31:326–333, 1991.

[8] P. M. D. Fitzgerald, H. M. Berman, P. E. Bourne, and K. Watenpaugh. The Macromolecular CIF Dictionary. ACA Annual Meeting, Albuquerque, New Mexico, 1993.

[9] A. F. P. Cook. Dictionary definition language in STAR file format. Technical report, ORAC Report, 1991.

[10] S. R. Hall and A. F. P. Cook. STAR dictionary definition language: Initial specification. J. Chem. Inf. Comput. Sci., 35:819–825, 1995.

[11] H. M. Berman and J. D. Westbrook. Now the Nucleic Acid Database Uses CIF. In P. E. Bourne, editor, Proceedings fo the First Macromolecular CIF Tools Workshop, Tarrytown, New York, 1993. National Science Foundation.

[12] John D. Westbrook, Shu-Hsin Hsieh, and P. M. D. Fitzgerald. CIFLIB, An Application Program Interface to CIF Dictionaries and Data Files. J. Appl. Cryst., 1996. in press.

[13] H. M. Berman and J. D. Westbrook. A Gentle Introduction to one Working Alternative DDL for Macromolecular Structure. In S. D. Wodak, editor, European Macromolecular Crystallographic Information (mmCIF) Workshop, Free University of Brussels, 1994. European Commission.

[14] J. D. Westbrook and S. R. Hall. A dictionary description language for macromolecular structure. J. Chem. Inf. Comput. Sci., 1996. to be submitted.

[15] Shu-Hsin Hsieh and John D. Westbrook. CIFPARSE: A function library for mmCIF data exchange. Rutgers University, 1996. (http://ndbserver.rutgers.edu/software/CIFPARSE/).