PROTEINS: Structure, Function, and Bioinformatics 59:687–696 (2005)

The CCPN Data Model for NMR Spectroscopy: Development of a Software Pipeline

Wim F. Vranken,[1] Wayne Boucher,[2] Tim J. Stevens,[2] Rasmus H. Fogh,[2] Anne Pajon,[1] Miguel Llinas,[3] Eldon L. Ulrich,[4] John L. Markley,[4] John Ionides,[1] and Ernest D. Laue[2,*]

[1]*Macromolecular Structure Database, European Bioinformatics Institute, Hinxton, Cambridge, United Kingdom*
[2]*Department of Biochemistry, University of Cambridge, Cambridge, United Kingdom*
[3]*Department of Chemistry, Carnegie Mellon University, Pittsburgh, Pennsylvania*
[4]*BioMagResBank, University of Wisconsin, Madison, Wisconsin*

ABSTRACT **To address data management and data exchange problems in the nuclear magnetic resonance (NMR) community, the Collaborative Computing Project for the NMR community (CCPN) created a "Data Model" that describes all the different types of information needed in an NMR structural study, from molecular structure and NMR parameters to coordinates. This paper describes the development of a set of software applications that use the Data Model and its associated libraries, thus validating the approach. These applications are freely available and provide a pipeline for high-throughput analysis of NMR data. Three programs work directly with the Data Model: CcpNmr Analysis, an entirely new analysis and interactive display program, the CcpNmr FormatConverter, which allows transfer of data from programs commonly used in NMR to and from the Data Model, and the CLOUDS software for automated structure calculation and assignment (Carnegie Mellon University), which was rewritten to interact directly with the Data Model. The ARIA 2.0 software for structure calculation (Institut Pasteur) and the QUEEN program for validation of restraints (University of Nijmegen) were extended to provide conversion of their data to the Data Model. During these developments the Data Model has been thoroughly tested and used, demonstrating that applications can successfully exchange data via the Data Model. The software architecture developed by CCPN is now ready for new developments, such as integration with additional software applications and extensions of the Data Model into other areas of research. Proteins 2005;59:687–696.**

Key words: data model, nuclear magnetic resonance, CcpNmr applications; format conversion; software; high throughput

INTRODUCTION

Nuclear magnetic resonance (NMR) spectroscopy has become an important tool in the analysis of macromolecular systems, and is used in many areas of research such as molecular structure determination, studies of dynamics, and high-throughput ligand screening. The continual development of the method over the past years has provided the NMR community with an extensive range of software that handles data from the initial processing of time domain spectra through to structure calculation and studies of dynamics. Unfortunately, this great variety of software employs many associated data formats, each of which describes the data in a different way. It is therefore often difficult to exchange data between programs used for similar or connecting stages along the analysis path, particularly when this needs to be done in a coordinated way. In turn, this tends to restrict the user to a specific set of programs that function together, and features available in other software packages are often not easily accessible.

In addition, despite the fact that most data exist in an electronic form at some point along the processing and analysis path, the data formats employed do not usually incorporate, or carry forward, information from previous steps. Data are therefore often lost or hard to retrieve when the time comes for it to be archived for deposition. This lack of data tracking is unacceptable for high-throughput (HTP) structural genomics projects where easy data exchange between different groups and institutions is essential. It is also highly desirable that different groups should be able to go back to the original data and re-process it, for example, for the development of new methods or for validation purposes. All of these features are also important in "low throughput" projects within individual laboratories, where they would facilitate work by different researchers on a particular project. Finally, a workable system to keep track of all the data ("data harvesting") would facilitate the deposition of data to archiving sites such as the PDB[1] and BioMagResBank (BMRB: see http://www.bmrb.wisc.edu/).

The Supplementary Materials referred to in this article can be found at http://www.interscience.wiley.com/jpages/0887-3585/suppmat

Grant sponsor: the EU; Grant numbers: QLG2-CT-2000-01313, EC Contract No. QLRI-CT-2001-00015; Grant sponsor: the BBSRC; Grant sponsor: the National Institutes of Health; Grant numbers: P41 LM005799, GM67965.

*Correspondence to: Ernest D. Laue, Department of Biochemistry, University of Cambridge, Cambridge CB2 1GA, UK. E-mail: e.d.laue@bioc.cam.ac.uk

Received 4 October 2004; Accepted 9 December 2004

Published online 6 April 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/prot.20449

For X-ray crystallography many of these problems have been addressed by the CCP4 project,[2] which provides a set of common libraries and programs so that users can readily benefit from the software developed by others. Within the NMR community the NMR-STAR format based on the STAR definition[3] has been instrumental in describing most of the data that needs to be deposited at the end of NMR projects. Programs like AQUA,[4] and reference information available at the BioMagResBank, have been developed to deal with the many problems caused by the different naming systems used in different NMR analysis software packages. More recently, the SPINS software[5] has employed a central database to track the data format files and important data items generated in a specified NMR analysis path. However, in all these efforts the approach is limited by the dependence on specifically defined data formats, which are textual descriptions of the data. The relationships between items of data are not immediately clear from such a description, and often the only precise definition is found embedded within the program code for parsing the data format file.

In an attempt to circumvent these problems of disparate data formats and computer programs, the Collaborative Computing Project for the NMR community (CCPN, see http://www.ccpn.ac.uk/) has developed a single object oriented data model. This Data Model for NMR has been developed as a community effort[6] to provide an abstract description of the concepts represented in macromolecular structure analysis, and their relationships to one another, in a manner that is independent of any particular data format.[6,7] The Data Model represents all of the information used in NMR analysis and related disciplines, covering areas such as molecular description, structure, and reference information that supports the NMR data. It is also specifically designed to handle intermediate information and results obtained during the analysis of NMR data.

The framework in which the Data Model has been developed makes possible the automatic generation of Application Programming Interfaces (APIs) in different programming languages.[7] These APIs provide an in-memory representation of the data as organized by the Data Model, which means that data can be manipulated in the same way regardless of both computer platform and programming language. Because the APIs maintain the integrity of the data according to the underlying Data Model, they allow a highly modular development of software libraries. In essence this means that new software for an API (in a particular language) can use all the other software that already exists for that API, and that data can be exchanged between applications in different languages via supported storage formats (currently XML and SQL).

This Data Model approach to software development is new, and before introducing this methodology it must be proven to be robust and reliable. To achieve this, a suite of applications for HTP NMR analysis (the CcpNmr suite) is being developed using an API for the Python programming language[8] (see Fig. 1). All these applications can interact with each other via the Data Model, thus providing a fully integrated "pipeline" for NMR processing and analysis. The software is already being used in everyday NMR work, establishing that the underlying Data Model is sound and functional, and has particular relevance for the analysis of NMR data in larger structural genomics projects.

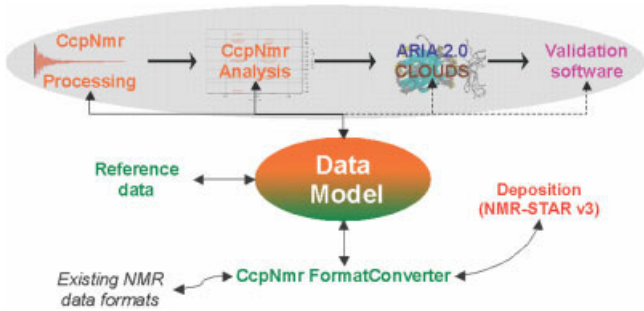

Fig. 1. Applications using the NMR Data Model. The source institution is indicated by color coding: the Macromolecular Structure Database group (EBI, Cambridge) (green), the University of Cambridge Biochemistry Department (orange), the Nilges group (Institut Pasteur, Paris) (blue), the CMBI (University of Nijmegen) (purple), the BioMagResBank (University of Wisconsin, Madison) (red), and the Llinas group (Carnegie Mellon University, Pittsburgh) (brown). Full lines indicate direct interaction between applications/formats, dotted lines indicate that indirect conversions take place.

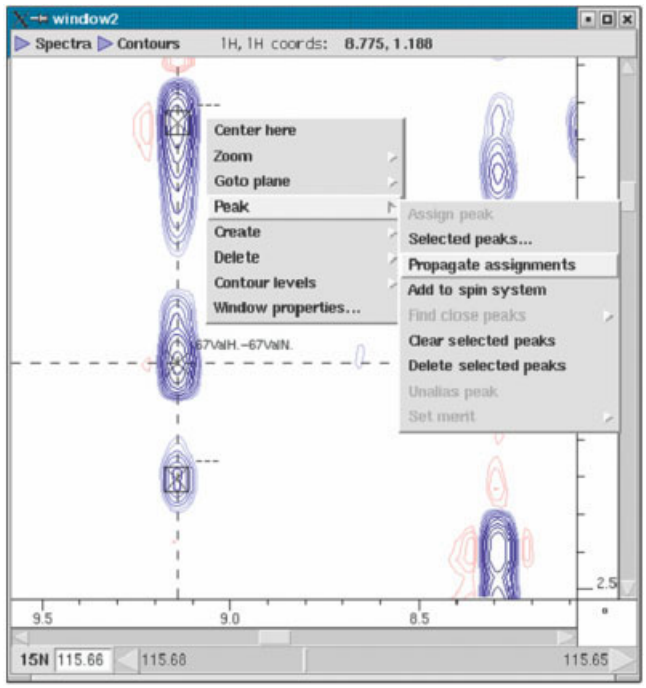

Fig. 3. A three-dimensional CcpNmr Analysis spectrum window with positive (blue) and negative (red) contours, crosspeaks, annotations and an open "right-mouse" menu. The bottom bar relates to the third dimension of the spectrum (in this case ^{15}N). The slice being viewed can be extended over a larger or smaller range of ^{15}N shifts or moved to a different region of the ^{15}N dimension as required.

For the analysis of NMR spectra, a completely new package, CcpNmr Analysis has been developed and in-memory conversions of the data allow access to structure calculation (ARIA[9]) and validation software (QUEEN[10]). In addition, a new version of CLOUDS[11] has been imple-

mented, in part to test and demonstrate the ease with which external programs can be integrated into the software pipeline. The CcpNmr Format Converter allows transfer of data from existing data formats to and from the Data Model, and also provides a general interface to the NMR-STAR v3 format for deposition of NMR data to the BioMagResBank. The Format Converter allows the user to produce data files that can be used with other existing NMR software at any stage during the analysis process. CcpNmr Processing will be based on the Azara package. Finally, an extensive library of reference material has been made available for creating molecules and for providing information on the naming systems used in NMR software.

The CcpNmr software described here is freely available, with companies being asked to support the project by paying a nominal fee for use of the software.

MATERIALS AND METHODS

The first API developed for the Data Model has been written in Python 2.2 and it is in this language that the CcpNmr applications are primarily written. Python is an ideal computer language for scientific software development. It is fully object-oriented, open source, cross platform, and well supported. It is interpreted rather than compiled, easy to learn, and is therefore a good language with which users can develop their own scripts. Python provides a wrapper around Tcl/Tk (see http://www.tcl.tk/software/tcltk) called Tkinter, so it can be immediately used for GUI development. Furthermore, performance critical parts of a program can readily be extended with code written in C.

The Format Converter is written solely in Python but the Analysis software is written in a mixture of Python and C. In CcpNmr Analysis, Python is used for the bulk of the code, including the graphical interface, general scripting, and user macros. However, the contour graphics handling is written in C code using OpenGL as the rendering library. This is particularly critical in order to get good performance for contouring of spectra, where contours are calculated and drawn on-the-fly. Most modern desktop computers have hardware-accelerated OpenGL, improving that aspect of performance.

One common chore in graphical program development is to remember the user-determined state in between sessions of work on a project, for example the colors of contours, whether a spectrum is visible in a given window, whether peaks should be drawn, where the windows appear on the screen, etc. This is often done using a proprietary key-value storage format. The Analysis program employs the Data Model machinery to describe the graphical state that was used, or, in other words, a Data Model package was developed that describes the graphical state of the program. Based on this package we can automatically provide an XML storage format of the graphical state, and no extra code is needed to parse or save this data. More importantly, this allows the use of the general API notifier system,[7] so that changes of graphical state made by the user in one module in the program can automatically be registered in other modules. Accordingly, the graphical state can automatically be kept consistent across the program, providing simplified program development. The Format Converter also uses the Data Model machinery to keep track of mapping between the data format files and the objects within the Data Model.

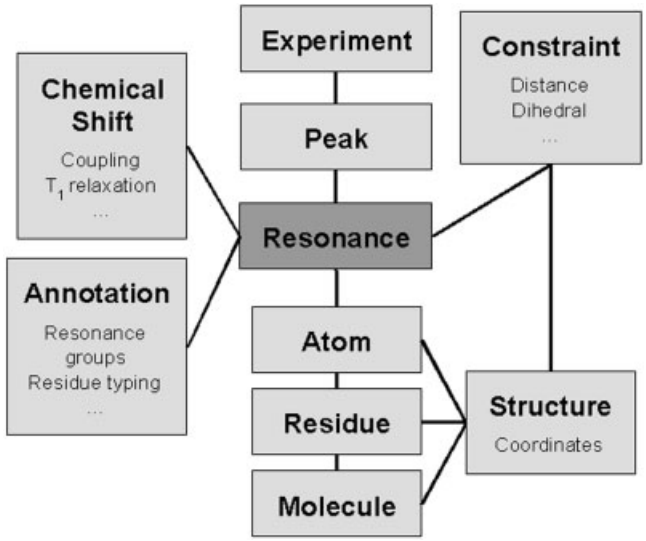

Fig. 2. Main concepts of the Data Model NMR package, and their relationship. Almost all relationships between different objects use the Resonance as an intermediary.

The common graphical parts of the CcpNmr applications have been built using a concurrently developed set of GUI libraries, which go beyond those normally available through Tkinter. These allow user interaction to proceed via a consistent, tested, and familiar set of graphical objects (widgets). This allows for a rapid development of user interfaces and is freely available to any external party wishing to develop graphical software for the Python API. The main purpose of Analysis is to provide a graphical NMR assignment program, but the CcpNmr applications can be run noninteractively via the Python command line interface. The C code has been developed on Linux, and has been successfully compiled and run on SGI Irix, Sun OS, and Mac OSX. It is also intended to port to Windows. The Format Converter, however, depends on only Python and Tcl/Tk, and has already been run successfully on Irix, Linux, and Windows.

RESULTS AND DISCUSSION

The CCPN Data Model for NMR

The Data Model is divided into "packages," each of which describes an area of information that is grouped together. For example, there is a "Molecule" package for molecular description, a "Coordinates" package that describes structure coordinates, and a "Sample" package to store sample information. The "NMR" package describes information specific to NMR, for example chemical shift lists, peak lists, experiment information, etc. The NMR package is linked, either directly or indirectly, to several other Data Model packages. The organization of these packages means

that, for example, a molecule description resulting from protein production can be centrally accessed by the X-ray and NMR packages, without the need for these X-ray or NMR packages to know about each other's existence. Also, packages are stored independently, so that only the required packages need to be loaded or saved. Indeed, only data in modified packages needs to be saved at a given time, which improves efficiency. Finally, because of this separation into packages, and because of its generic approach to describing data, the Data Model can be easily extended into other areas of research and distinct groups can develop the different parts. The NMR package of the Data Model has to be comprehensive enough to allow the description of less common NMR parameters and to also provide room for the incorporation of future concepts. The Data Model is therefore very complex and goes beyond the descriptions required to handle standard data. In order to describe some of the central concepts underlying the NMR Data Model more clearly it is presented here in a simplified manner (see Fig. 2). A more detailed discussion is given in the supplementary material.

The Data Model objects in the NMR package are mostly familiar concepts, for example, "Experiment" and "Peak," but there are some that do not have exact real-world counterparts. A good example of this is a "Resonance." The Data Model for NMR was designed to handle not only the final, fully assigned, data resulting from an NMR project, but also the intermediate information that changes as an assignment or structure calculation is in progress. The crucial element to achieve this, and also the most central concept in the NMR package, is the Resonance object. This object does not represent a particular atom, nor does it represent a particular value of an NMR parameter. Rather, it represents the connection between an atom (or NMR equivalent atoms) and the NMR parameters that they give rise to. This means that one Resonance can be connected to many chemical shift values (i.e., as they occur in different spectra/conditions for the same NMR equivalent atom[s]), and that NMR information can be linked to a Resonance without knowing which atom or group of atoms it relates to. The assignment of NMR data to a Resonance and the assignment of this Resonance to a particular atom (or set of atoms) are thus separate events, which do not have to be performed simultaneously (see Supplementary Material for an extensive description of the Resonance–Atom link). Resonances also serve to connect Peak assignments and Constraints to the other objects in the model. Finally, they serve as the basis for entering annotations or partial assignment information, such as which Resonances belong to the same group (or spin system), or which Resonance corresponds to which position in a residue.

To give an example of a practical implementation of the model, consider the "ShiftList" object. This object groups together Shift objects, which contain only a value and an error attribute. Each of these Shift objects has to be linked to one Resonance. What this means in practice is that one can have multiple ShiftLists (e.g., for data recorded at different temperatures), each with a separate set of Shifts. A signal that moves with changing temperature can now always be assigned to the same Resonance, but this one Resonance is connected to different Shift objects in the different ShiftLists. The separation of the assignment of the Resonance to an Atom means that these steps can be handled before any information is available describing which Atom the moving signal belongs to.

CcpNmr Analysis

An important step in proving the usefulness of the Data Model involves the writing of a new application for analysis of NMR data based directly on that Data Model. During the analysis of spectra and the assignment of signals, the user changes and adds intermediate data until a final assignment state is reached. This is a very dynamic process and poses specific problems for the Data Model architecture. Further demands on the architecture were made by extending the Data Model to include data particular to the Analysis program (e.g., information on windows, etc.), so that the scientific and application specific data could work seamlessly together. The development of this CcpNmr Analysis program was, finally, crucial in the development of a "software pipeline," starting with the experimental data and processing it, carrying out an assignment, doing structural calculations, and ultimately validating the results (see Fig. 1).

The assignment stage of NMR has traditionally been the least automated and most time-consuming part of this "pipeline." This means that this is an area in which much progress can be made, both in automation and in providing new and efficient graphical tools for tasks that cannot be automated. Accordingly, high-level design imperatives were set: to use the Data Model as the underlying model for all NMR data, to incorporate modern ideas about assignment, to use open, cross platform, software standards, to allow extensibility by third parties and to ensure good performance on modern desktop computers. Analysis is thus developed in a modular and flexible way so that its core assignment libraries can be manipulated and extended by additional modules (written in Python). In this way, new assignment tools and strategies can readily take advantage of the Data Model and the Analysis program itself, making use of all of the existing technology including the graphical interface. The complete setup, for example contour colors, polypeptide sequence creation, etc. is done via this interface, and no text files or configuration scripts have to be edited. Access to the Data Model is also handled in this manner, so that the end user does not have to be concerned with the details of the underlying data structure. If required, however, Data Model access is possible via the Python command line interface and the configurable macro system.

Many NMR assignment programs interact with NMR spectra plane by plane, but CcpNmr Analysis is inherently N-dimensional, and uses the same approach as ANSIG.[12] This means that in each graphical window there are scrollbars not only in the x-y plane but also in the orthogonal dimensions—the latter can be resized (as well as scrolled) so that as many, or as few, planes as desired can be viewed at once (Fig. 3). In addition to the scrollbars,

the user can also pan and zoom in the x-y plane by use of the mouse. A menu, which pops up with a right mouse click, contains common tasks, some of which can also be done with keyboard shortcuts. Such tasks include navigation functions to aid the selection of planes and positions in multi-dimensional spectra that correspond to particular points in other spectra. In addition, each graphical window can have an unlimited number of spectra drawn in it. For each window the program automatically determines those spectra for which there is at least one consistent mapping onto the window axes (this means, in the usual case, if the NMR active nucleus matches), and if there is more than one consistent mapping then the user can change from the automatically determined configuration to another one. Up to ten of the spectra (determined by the user) can be toggled on and off with keyboard shortcuts or with a mouse click.

There is a built-in peak-finding algorithm, based upon the one used in Sparky,[13] and a basic peak-fitting algorithm. Due to the architecture of Analysis, users can in the future add any such (or other) algorithm that is based on the Data Model. It will also allow NMR spectroscopists to readily implement new methods without the need to worry about writing a graphical interface. Ultimately, this will provide the user with choices on which procedure to use. The graphical spectrum windows allow the picking and subsequent selection of one or more peaks at a time. To make analysis efficient, many functions, for example, assignment and deletion, can be applied to a selected set of peaks. The treatment of aliasing in spectra and peaks is a particular feature of CcpNmr Analysis. The contours of a spectrum may be "tiled," that is plotted separately for each possible aliasing of the spectrum. All positions in such a plot correspond to just one unambiguous frequency position, so that any peak picked will automatically have the correct frequency. "Unaliasing" is also readily achieved if a peak is initially at a folded position and, where appropriate, the frequency correction is automatically transferred to other peaks assigned to the same Resonance.

The assignment proceeds via the Resonance object, which allows the user to interconnect NMR information during the intermediate stage of spectrum assignment. Partial assignment information can be stored in the Resonance, and Resonances can be grouped into spin systems (ResonanceGroup). These ResonanceGroups can contain information on the residue type, sequential assignment, etc., even before specific atom information is available for the Resonance. When a Resonance is assigned to atom(s), the corresponding assignment will automatically be displayed. All Resonances can be displayed and picked from a curated, ranked list.

Other features of Analysis include the display and use of BMRB reference chemical shift data, an assignment graph to display the extent of assignment and its connectivities within a project (Fig. 4), calculation of ambiguous distance constraints from NOESY spectra, and an interface to calculate heteronuclear NOEs. The CcpNmr FormatConverter (see below) is integrated with Analysis so that NMR data (e.g., peak lists) of various formats can be imported

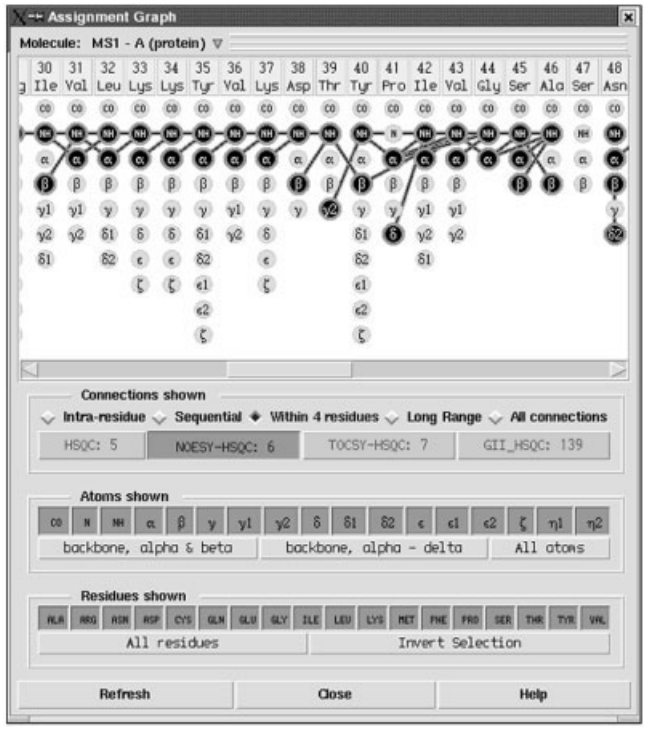

Fig. 4. The Analysis assignment graph. This window provides a graphical display of the extent of assignment and the inter-atom and residue connectivities that have been made within a project.

and exported from other NMR programs. Spectral data can be imported from various formats, currently Azara, Felix, NMRPipe, and UCSF (all of which use a so-called matrix, or blocked, binary format). Graphical representations from Analysis can also be output. The graphical windows can be output to file on disk in either PostScript or PDF formats, which can then be edited in appropriate desktop software (e.g., Adobe Acrobat) or sent to a printer. There are many tables in the graphical interface (e.g., of peaks, peak lists, etc.) and these can all be exported to (tab- or comma-separated) ASCII files, which can then be edited and further processed in appropriate desktop software (e.g., Microsoft Excel).

Overall, an important aim of the Analysis program is to remove the tediousness that has traditionally been associated with the manual assignment of NMR spectra, so that more attention can be paid to the important decisions. As more (semi-)automated procedures based on the Data Model are written, the Analysis program can make them available via a graphical user interface, giving the user the option to use a preferred method or to run different methods in parallel.

Integration of Existing Software

An important motivation for writing a new analysis program for NMR was to provide an easy and straightforward means for other groups to either develop or incorporate new tools for the automated analysis of NMR spectra or for structure calculation, without the need to write their own graphical interface or to integrate their work with

other programs. Incorporation of existing software can take place on several levels: a low-level rewrite of the existing software so that it essentially uses the Data Model for its own description of data, a high-level development of an in-memory data conversion layer that converts the Data Model objects to and from the structure used inside the application, or a specific layer that converts via intermediate files. In a demonstration of the low-level approach, CLOUDS,[11] the automated assignment and structure calculation protocol is being implemented as part of the CcpNmr Analysis software. The CLOUDS protocol is a means to automatically assign NMR spectra and determine protein structures via the generation of spatial distributions or "clouds" of hydrogen atoms, derived from proton–proton distances determined in an NOE experiment. The incorporation of CLOUDS into the CCPN framework shows that it is possible to integrate new NMR analysis and assignment strategies with the Data Model relatively quickly. This is mainly due to the modularity of the Data Model-based software, and the availability of loading, saving, data display, etc., routines that allow the programmer to concentrate on their particular algorithm.

The stage reached in a particular structure determination using CLOUDS is reflected in the current state of the Data Model objects. At each stage the state of the Data Model can be saved, viewed from within Analysis and manipulated if necessary, thus allowing human intervention and checking. Some parts of CLOUDS were rewritten entirely in Python as Data Model manipulations, but others, especially those that are calculation intensive (e.g., the molecular dynamics), are written in C with Python wrappers. Each step is a separate function and can be modified independently, or used independently in other types of analysis. The result of its incorporation in the CCPN framework is that CLOUDS can be made available without the need for special installation or generation of input data in a particular format (Fig. 5).

In order to connect the ARIA2.0[9] package to the Data Model, a high-level approach using an in-memory data conversion layer was employed, and ARIA2.0 can now automatically transfer its internal data to and from the Data Model and make use of its features. Integration of the QUEEN[10] program uses an approach involving intermediate files to achieve the same results. These conversions, on top of newly written code, reuse parts of the Format Converter code for crucial steps like the mapping from Resonances to Atoms. The interaction of these programs with the Data Model will be available from their next release onward.

CcpNmr Format Converter

A crucial step in validating the Data Model consisted of testing its ability to read existing data from current NMR applications. This also establishes whether the description of the data works on a practical level. For this purpose, a set of format conversion scripts for import and export of existing data were developed on top of the Data Model. The data format conversion problem illustrates one of the strengths of using a central Data Model (for a general description of the principle, see Fogh et al.[7]). The key advantage is that each import and export script only has to be written once for each data format and these will then immediately work with all the import and export scripts written for all other data formats. Although the initial effort involved is larger, none of the work is ever made redundant because all scripts remain valid even if new data formats appear. Using the Data Model as a stepping-stone has the added advantage that any ambiguity in the resonance assignments from the data format files is resolved during the conversion process, thus in effect adding a validation step. In contrast, traditional custom-made scripts are written to handle a particular conversion between specific programs. These scripts are usually particular to a laboratory and sometimes even to the researcher who wrote them.

The suite of scripts that was developed to handle transfer of data in existing formats into/from the Data Model is called the Format Converter. In practice, it permits the use of data from existing data format files in either the CcpNmr Analysis program or any other Data Model-based application. Information can also be written out from these applications to be used again in software that does not work with the Data Model.

The scripts are organized according to the software program (e.g., NMR-STAR, XEasy,[14] etc.) and the data component (e.g., chemical shifts, distance constraints, etc.). The first layer of scripts handles the reading and writing (format I/O) of specific files. During the reading process the file format is parsed and the data put into Python classes that are particular to that format, although some standardization is introduced for the naming of the data objects in each component. The writing process takes the data objects from the format-specific Python data classes and writes them to file in the correct manner. The second layer of scripts handles the import and export to and from the Data Model. During import the previously described format-specific Python data classes are interpreted, the correct Data Model objects created and the data values set inside them. During export the data values are read from the relevant objects inside the Data Model and the values are set inside the format-specific Python data classes. On top of the second layer is a GUI layer, which allows easy use of the scripts without the need to code in Python. A large part of the GUI functionality comes from modules that are available as part of the general CCPN software distribution.

The separation between the first two layers was introduced because, where possible, the code is generalized for several formats (in the DataFormat module) or for different components in a format (in generalIO modules). The specific format import is handled by a subclass of DataFormat, for example, SparkyFormat, which defines routines not described in DataFormat, or redefines routines for the particular format. At the format I/O level the organization is different: no subclassing is used but routines are imported from the generalIO module if necessary. This was done in order to make the scripts easier to manipulate by other users. Overall the format reading, data import and

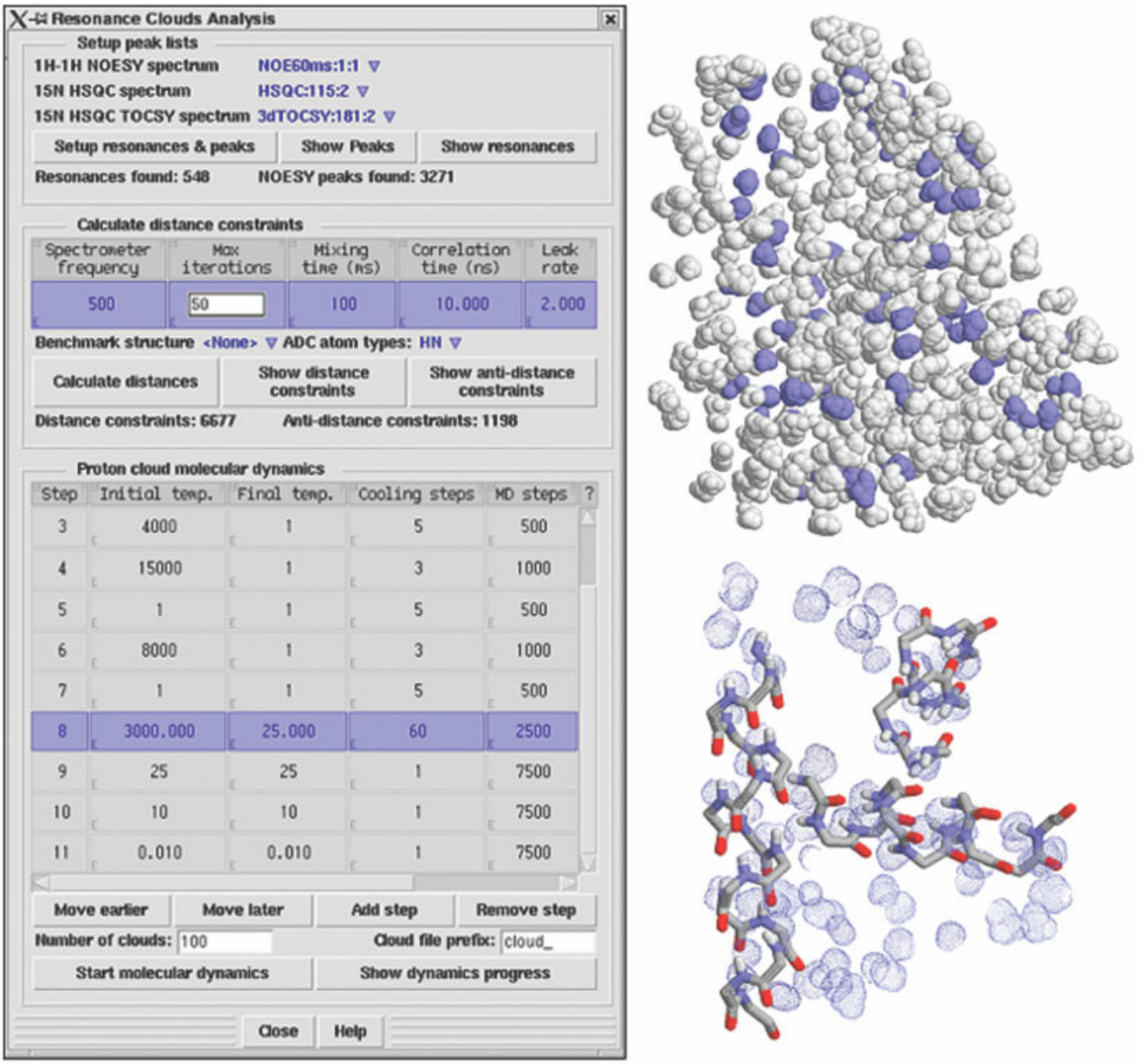

Fig. 5. CLOUDS integration. On the left, part of the graphical interface is shown. This window allows the user to generate distance constraints and set up the molecular dynamics protocol. The top right image shows the spatial distribution of proton "clouds" generated using NOE distance constraints. Amide protons are illustrated in blue. At the bottom right the structure of a homologous protein has been superimposed with the amide protons of a family of clouds. The molecular graphics were generated using RASMOL.

GUI parts are cleanly separated making the code readily extensible on each level.

Tables I and II describe the extent to which components and existing formats are currently handled by this set of scripts. In addition to the formats and components mentioned, support is provided for reading acquisition parameters from Bruker and Varian spectrometer files, editing the information and then writing out either NmrPipe[21] or Azara[24] processing scripts. New formats will continue to be added. The latest state of the software can be viewed at http://www.ebi.ac.uk/msd-srv/docs/NMR/. Further details of format conversion can be found in the Supplementary Data.

The Format Converter software is now being used to read in NMR-STAR restraint files from the BMRB,[25] followed by sequence and molecular system information from the corresponding PDB file, linking the "Resonances" created during the constraint import to the relevant atoms, and exporting the constraint files in DYANA/CYANA and/or CNS/XPLOR format. This procedure is being actively used for a joint project between the University of Utrecht and the BioMagResBank (pers. commun.

TABLE I. Overview of Structure Data Components That Can Be Imported (I) or Exported (O) From Different Data Formats

			Restraints		
	Sequence	Coordinates	Distance/H-bond	Dihedral and RDC	J coupling
CNS[15]/XPLOR[16]/ARIA[9]	I	I/O	I/O	I/O	I/O
Concoord[17]			I/O		
Dyana/Cyana[18]	I/O	I/O	I/O	I/O	
Fasta[19]	I/O				
NMR-STAR v2.1.1	I		I	I	
NMR-STAR v3	I/O	I/O	I/O	I/O	
PDB[1]	I	I/O	n/a	n/a	n/a

TABLE II. Overview of NMR Data Components That Can Be Imported (I) or Exported (O) From Different Data Formats

	Sequence	Chemical shifts	Peak lists	Project[1]
Ansig[12]	I/O	n/a	I/O	I
Felix[20]			I	
NmrDraw[21]			I/O	
NMR-STAR v2.1.1	I	I	I	I
NMR-STAR v3	I/O			I/O
NmrView[22]	I/O	I/O	I/O	I
Pronto[23]	I	I	I	
Sparky[13]	I/O	I	I/O	I/O
XEasy[14]	I/O	I/O	I/O	

[1]Project refers to a file or set of files that describe all data components.

with A. Nederveen [Utrecht] and J. Doreleijers [BioMagResBank]) for the development of the DOCR, FRED, and RECOORD databases, and the results are already employed in the DRESS database at the University of Nijmegen.[26]

A full reader and writer for the NMR-STAR v3 format is currently under development. Since complete deposition files have to be produced for this format the mapping involved is more complex, and there will be a need to select the right data objects for deposition. Because the atom names from the data format files have to be unambiguously defined when they are imported into the Data Model, the Format Converter serves as a validation tool that can be used to combine NMR information and link it together before final deposition in projects that have not used the Data Model.

Reference Data

To use the Data Model, reference information has to be available that allows the description of the molecular system under investigation, as well as all the atoms and bonds that are present. This requires a detailed library of the chemical structures of amino acids, nucleic acids, and common ligands, as well as their IUPAC atom names and the atom names for other naming systems in general use (e.g., XPLOR,[16] CYANA,[18] etc.).

The E-MSD database[27] developed by the Macromolecular Structure Database (MSD) group at the European Bioinformatics Institute (EBI) is a relational database that incorporates almost all the information available from the PDB archive. In the process of loading the PDB archive into this database, a set of reference data was generated that describes all the chemical components (amino acids, ligands, etc.) currently present in the PDB archive. This reference data was used as the primary source for the generation of ChemComp objects, which describe the chemical components as used within the Data Model.[7] In addition to the original data from the database, information was added that is relevant for NMR. Atom naming system information was included from reference files obtained from the BMRB, AQUA,[4] and CYANA. Also, data on which atoms are prochiral and which are NMR equivalent was included. One of the advantages of obtaining the chemical components data from the E-MSD database is that the three-letter codes used within PDB files are also stored and can be automatically used when creating a molecule based on a PDB file. ChemComp descriptions not contained within the normal distribution can be downloaded from http://www.ebi.ac.uk/msd-srv/docs/NMR/chemCompXml/main.html.

Isotope information was downloaded from the http://www.boku.ac.at/chemie/oc/periname.html and http://atom.kaeri.re.kr/ websites, converted into text files, entered into the Data Model and saved as a reference file that is included in the standard distribution.

Conclusion

The Data Model for NMR described here is an extensive, flexible, and well documented description of the data used in NMR spectroscopy. In comparison with existing data formats it allows a much more complex and layered description of these data. This will become increasingly important for automated procedures and eventual archiving of the data at deposition sites. The generic machinery developed around this Data Model[7] allows the autogeneration of code to handle the objects inside the Data Model and also provides integrated input/output code, greatly simplifying the task of developing new software. For the future, modeling of extensions into other areas of research[19] will expand the scope of the Data Model and open up avenues for development of Laboratory Information Management Systems (LIMS) with easy transfer of data between related areas of research (e.g., protein production to NMR spectroscopy).

The Data Model and its associated software framework provide an excellent environment for the development of

new applications. This is exemplified by the ease with which the software described in this paper was developed, and by the ease with which existing software can be interfaced with the Data Model, either in-memory or via format conversion of data files. These applications have allowed extensive testing with large amounts of data. Even though the actual description of the data is rather complex, the applications ensure that this does not complicate life for users, but instead allows them a greater flexibility. For example, a strength of the approach is that it is entirely possible to use the CcpNmr Analysis package for studies of small organic molecules in addition to proteins or nucleic acids, or to study a combination of these within one project.

At the time of writing the CcpNmr Analysis program has passed through its initial testing phase and has been made available in its first full release. The user base for the program is growing, and includes users who are new to the NMR field, thus demonstrating its efficiency and user-friendliness. The functionality of this first release is already advanced in comparison to many of the assignment programs it was intended to replace.

The Format Conversion software can be used to read data from files of existing NMR application programs and structure it according to the Data Model. This allows one to continue work within the Data Model framework, to link separate data files together in order to generate a file for deposition, or to simply convert to another file format. Currently, not all available formats are handled, and import/export is not possible for all components of all supported formats. This will be improved in time as external contributions are included and more routines are written to handle missing formats and components. The format conversion software is already being used in a project in which existing constraint lists deposited at the PDB[1] are being cleaned up and converted to the NMR-STAR format. Full support for reading and writing NMR-STAR v3 deposition files is currently under development.

The Python code written for the CcpNmr Analysis and Format Converter software serves as a good example of how the Data Model can be used to handle different types of NMR data in real application programs. In addition to CcpNmr Processing, other applications that are currently being modified to work with the Data Model are ARIA,[9] CLOUDS,[11] and QUEEN[10] (see Fig. 1). This will provide a complete software "pipeline" from processing to structure calculation that uses the Data Model and which will be particularly well suited to HTP projects. Again, it is important to note that this does not exclude the use of other programs: the Format Converter can be used at any time to export (and later import) files used by other software.

The future holds several exciting prospects. The current CcpNmr applications will continue to be developed and the Format Converter will be expanded to include new data formats as required. For Analysis, several more advanced features that extend its functionality are planned for the future. These include a means of determining and presenting relaxation parameters, the superposition of possible NOE interactions upon structural representations, assignment to multiple conformations of the same molecule, and integrated procedures for the analysis of residual dipolar couplings. Also, as the software has so far been focused on the assignment of multidimensional spectra for structural studies, there are plans to extend Analysis into the simpler, high-throughput one- and two-dimensional analyses that are increasingly being used in industrial screens and assays.

One important role of the CCPN project is to facilitate the development of tools for automated NMR assignment. This will involve the development of novel assignment strategies, and the integration of existing methods with the Data Model. As discussed above, to this end we are currently developing an implementation of CLOUDS in Analysis. By embedding CLOUDS within the CCPN framework, and using the Data Model, we hope to provide an accessible means for rapid structure determination. For the future, as more applications and scripts based on the Data Model become available, it will be possible to run different methods performing the same task in parallel and compare the results without the need to convert data at any stage. This will provide valuable validation of automated procedures.

We want to stress that the CCPN software is freely available for all users. We hope that larger companies will support the project by paying a nominal fee for use of the software—these funds are being used to support the project. Finally, the CCPN project is a collaborative effort and feedback and involvement of the community will be essential if it is to become a success, both with respect to the development of the Data Model itself and the applications that use it. Accordingly, we invite interested groups to collaborate on extending the current base set of applications and scripts to provide a software suite that is readily available to everyone and which can serve the NMR community.

Requirements and Availability

The CcpNmr applications described here require Python 2.2 (or higher) (see http://www.python.org/) containing the Tkinter module (which handles Tcl/Tk for the GUI). CcpNmr Analysis in addition requires a C compiler and access to OpenGL libraries for the relevant computer platform.

Distributions can be obtained from http://www.ccpn.ac.uk/, and are available for the Python API only, for the Python API with the FormatConverter software and for the Python API, FormatConverter and Analysis software combined. Relevant Python and Tcl/Tk distributions are also accessible from this page. Additional definitions for chemical components (chemComps) not included in the standard distribution can be downloaded from http://www.ebi.ac.uk/msd-srv/docs/NMR/chemCompXml/main.html.

The documentation is mostly auto-generated. For information on the Python API and Data Model diagrams, the Python code used by the Analysis and FormatConverter programs, see the appropriate directories under http://

www.ccpn.ac.uk/. More online information on the Format Converter can be found at http://www.ebi.ac.uk/msd-srv/docs/NMR/NMRtoolkit/main.html. The Data Model, Python API, the reference data and the Format Converter are distributed under an LGPL license (see http://www.gnu.org/licenses/licenses.html#LGPL). Analysis is distributed under the CCPN license (see http://www.ccpn.ac.uk/), which allows indefinite, royalty-free use, and requests a moderate annual fee only from for-profit users.

For further information on the Data Model, the API and automatic code generation see Fogh et al.[7]

Supporting Information Available

A more extensive description of the NMR Data Model package, information on the assignment procedure in CcpNmr Analysis and additional screen shots, as well as information on the CcpNmr FormatConverter handling of application data and the procedure for linking resonances to atoms.

ACKNOWLEDGMENTS

We thank the following people for suggestions and for testing. FormatConverter: Eiso AB, Alexandre Bonvin, Jules Jacobson, Jens Linge, Sander Nabuurs, Brian Smith, and Chris Spronk. Analysis: Jeremy Craven, Paul Driscoll, David Gill, Andrew Sanderson, Brian Smith, and Katherine Stott. The chemComp reference data in the E-MSD database is maintained by the MSD group, in particular Dimitris Dimitropoulos. We are particularly grateful to Aart Nederveen for extensive testing of the Format Converter software, to Jurgen Doreleijers for making his Python STAR reader and writer available, to Peter Güntert for making his CYANA reference library available, and to Wolfgang Rieping, Michael Habeck, and Michael Nilges for the in-memory conversion to ARIA 2.0.

We acknowledge funding from the EU for the NMR-QUAL (EC Contract No. QLG2-CT-2000-01313) and TEMBLOR (EC Contract No. QLRI-CT-2001-00015) grants and the BBSRC for funding of the CCPN project. Work at BMRB was supported by NIH grant P41 LM005799. Support to M. Llinas from the NIH, grant GM67965.

REFERENCES

1. The PDB team. The Protein Data Bank. In Bourne, PE, Weissig, H, editors. Structural bioinformatics. Hoboken: John Wiley & Sons, Inc.; 2003. p 181–198.
2. Collaborative Computational Project, Number 4. The CCP4 Suite: Programs for Protein Crystallography. Acta Crystallogr D Biol Crystallogr 1994;50:760–763.
3. Hall SRJ. The STAR File: a new format for electronic data transfer and archiving. Chem Inf Comput Sci 1991;31:326–333.
4. Doreleijers JF, Rullmann JAC, Kaptein RJ. Quality assessment of NMR structures: a statistical survey. Mol Biol 1998;281:149–164.
5. Baran MC, Moseley HN, Sahota G, Montelione GT. SPINS: Standardized ProteIn NMR Storage. A data dictionary and object-oriented relational database for archiving protein NMR spectra. J Biomol NMR 2002;24:113–121.
6. Fogh R, Ionides J, Ulrich E, Boucher W, Vranken W, Linge JP, Habeck M, Rieping W, Bhat TN, Westbrook J, and others. The CCPN project: an interim report on a data model for the NMR community. Nat Struct Biol 2002;9:416–418.
7. Fogh RH, Boucher W, Vranken WF, Pajon A, Stevens TJ, Bhat TN, Westbrook J, Ionides JMC, Laue ED. A framework for scientific data modeling and automated software development. Bioinformatics 2004. Forthcoming.
8. van Rossum, G. The Python language reference manual. Bristol, UK: Network Theory Ltd.; 2003.
9. Linge JP, Habeck M, Rieping W, Nilges M. ARIA: automated NOE assignment and NMR structure calculation. Bioinformatics 2003; 19:315–316.
10. Nabuurs SB, Spronk CAEM, Krieger E, Maassen H, Vriend G, Vuister GW. Quantitative evaluation of experimental NMR restraints. J Am Chem Soc 2003;125:12026–12034.
11. Grishaev A, Llinas M. CLOUDS, a protocol for deriving a molecular proton density via NMR. Proc Natl Acad Sci USA 2002;99:6707–6712.
12. Kraulis PJ. ANSIG: a program for the assignment of protein ^{1}H 2D NMR spectra by interactive graphics. J Magn Reson 1989;24: 627–633.
13. SPARKY, version 3.110. San Francisco: University of California; 2004.
14. Bartels C, Xia T-H, Billeter M, Güntert P, Wüthrich K. The program XEASY for computer-supported NMR spectral analysis of biological macromolecules. J Biomol NMR 1995;6:1–10.
15. Brünger AT, Adams PD, Clore GM, DeLano WL, Gros P, Grosse-Kunsteleve RW, Jiang J-S, Kuszewski J, Nilges M, Pannu NS, and others. Crystallography & NMR system (CNS): A new software system for macromolecular structure determination. Acta Crystallogr D Biol Crystallogr 1998;54:901–921.
16. Schwieters CD, Kuszewski JJ, Tjandra N, Clore GM. The Xplor-NIH NMR molecular structure determination package. J Magn Reson 2003;160:66–74.
17. de Groot BL, van Aalten DMF, Scheek RM, Amadei A, Vriend G, Berendsen HJC. Prediction of protein conformational freedom from distance constraints. Proteins 1997;29:240–251.
18. Güntert P, Mumenthaler C, Wüthrich K. Torsion angle dynamics for NMR structure calculation with the new program DYANA. J Mol Biol 1997;273:283–298.
19. Pearson WR. Rapid and sensitive sequence comparison with FASTP and FASTA. Meth. Enzymol 1990;183:63–98.
20. Felix. San Diego: Accelrys; 2004.
21. Delaglio F, Grzesiek S, Vuister GW, Zhu G, Pfeifer J, Bax A. NMRPipe: a multidimensional spectral processing system based on UNIX pipes. J Biomol NMR 1995;6:277–293.
22. Johnson BA, Blevins RAJ. NMRView: A computer program for the visualization and analysis of NMR data. Biomol NMR 1994;4:603–614.
23. Kjær M, Andersen KV, Poulsen FM. Automated and semi-automated analysis of homo- and heteronuclear multidimensional nuclear magnetic resonance of proteins: the program PRONTO. Meth Enzymol 1994;239:288–308.
24. Azara, version 2.7. Cambridge: Biochemistry department; 2002.
25. Doreleijers JF, Mading S, Maziuk D, Sojourner K, Yin L, Zhu J, Markley JL, Ulrich EL. BioMagResBank database with sets of experimental NMR constraints corresponding to the structures of over 1400 biomolecules deposited in the Protein Data Bank. J Biomol NMR 2003;26:139–146.
26. Nabuurs SB, Nederveen AJ, Vranken W, Doreleijers JF, Bonvin AM, Vuister GW, Vriend G, Spronk CA. DRESS: a Database of REfined Solution NMR Structures. Proteins 2004;55:483–486.
27. Golovin A, Oldfield TJ, Tate JG, Velankar S, Barton GJ, Boutselakis H, Dimitropoulos D, Fillon J, Hussain A, Ionides JM, and others. E-MSD: an integrated data resource for bioinformatics. Nucl Acid Res 2004;32:D211–216.
28. Pajon A, Ionides J, Diprose J, Fillon J, Fogh R, Ashton AW, Berman H, Boucher W, Cygler M, Deulery E, and others. Design of a data model for developing laboratory information management and analysis systems for protein production. Proteins. 2005;58: 278–284.