Converting Biomolecular Modelling Data Based on an XML Representation *

Yudong Sun[1,2] and Steve McKeever[1]

[1]Computing Laboratory, University of Oxford, Oxford OX1 3QD, UK

Summary

Biomolecular modelling has provided computational simulation based methods for investigating biological processes from quantum chemical to cellular levels. Modelling such microscopic processes requires atomic description of a biological system and conducts in fine timesteps. Consequently the simulations are extremely computationally demanding. To tackle this limitation, different biomolecular models have to be integrated in order to achieve high-performance simulations. The integration of diverse biomolecular models needs to convert molecular data between different data representations of different models. This data conversion is often non-trivial, requires extensive human input and is inevitably error prone. In this paper we present an automated data conversion method for biomolecular simulations between molecular dynamics and quantum mechanics/molecular mechanics models. Our approach is developed around an XML data representation called BioSimML (Biomolecular Simulation Markup Language). BioSimML provides a domain specific data representation for biomolecular modelling which can efficiently support data interoperability between different biomolecular simulation models and data formats.

1 Introduction

Computer based biomolecular modelling is being used as an effective methodology to study complex biological processes at a microscopic level. Computer simulations are used to investigate microscopic processes such as drug and protein interactions [1, 2]. Our study focuses on enzymes which are proteins that catalyse chemical reactions in biological cells. Monotopic membrane bound enzymes are an important class of drug target [3], such as cyclooxygenase are targets for non-steroidal anti-inflammatory drugs. Knowing the behaviours of these enzymes within membrane environment may aid the exploration of new drug targets and new inhibitor design. Moreover, understanding these enzymes requires modelling at all levels: from the quantum mechanical simulation of enzyme reaction mechanisms, through molecular mechanical descriptions of drug binding and molecular dynamics of protein conformational changes, to mesoscopic descriptions of drug diffusion through cell membrane to the enzymes. Therefore, the simulations need to cover different biomolecular modelling levels including quantum mechanics (QM), molecular mechanics (MM), and molecular dynamics (MD) [4].

QM/MM methods are applied at the most fundamental level to model the mechanisms of enzyme reactions, including chemical bond breaking and making [5]. Studying a biological system at atomic and subatomic levels, where QM/MM simulations are highly computationally demanding, ensures that long-timescale simulations are impractical. On the other hand, MD simulations are more time-efficient and enable longer timescales to be simulated [6]. One solution

*This work was funded by BBSRC, UK under the IntBioSim project.

[2]To whom correspondence should be addressed. E-mail: yudong.sun@comlab.ox.ac.uk

for enabling realistic QM/MM simulations is to select representative enzyme conformations from long-timescale MD simulations and then initiate QM/MM simulations to model enzyme reactions in the selected enzyme conformations. Different simulation models may use different simulation software with diverse data representations for molecular simulation data. In our case, the MD simulations are performed using a molecular simulation program CHARMM [7] and the QM/MM simulations by another simulation program TINKER [8]. CHARMM uses the PDB (Protein Data Bank) format [9] to describe 3D molecular structures while TINKER uses its proprietary XYZ format. Therefore, data conversion is necessary when transferring the details of an enzyme conformation from an MD simulation to a QM/MM simulation. The conversion includes rewriting the file with details of thousands of atoms to a new format, renaming each atom and atom type, and supplementing each atom with additional attributes. In the past, data conversion was manually handled. Some simulation packages provide own data converters such as the `pdbxyz` program of TINKER. Nevertheless, these converters do not always produce the correct output. Human input and curation is still indispensable in many cases. Manipulating the data of numerous atoms by hand is thus inevitably inefficient and error-prone.

To integrate biomolecular simulation models and realise efficient data conversion between different biomolecular models, we have developed a biomolecular simulation markup language called BioSimML to provide an XML based representation for a range of biomolecular data including molecular structure, atom name mapping, simulation parameters and metadata. BioSimML provides XML constructs with an associated toolkit to facilitate automatic data conversion in our multi-level biomolecular simulations. In this paper, we will present the BioSimML representation used for the implementation of automated data conversion between MD and QM/MM simulations.

The paper is structured as follows. Section 2 discusses the related work. Section 3 introduces the BioSimML document structure. Section 4 describes the data conversion between MD and QM/MM simulations. Section 5 summarises our achievements and provides a brief discussion of future work.

2 Related Work

Domain specific XML languages have been used to provide generic data representations for different purposes in scientific research such as in computational systems biology and computational chemistry [10]. CellML [11] is an XML-based standard format for defining and exchanging biological models. It captures mathematical descriptions of cellular functions such as biological pathways and electrophysiological models to facilitate the reuse of such models in the biological community, independent of the modelling software. CellML models are represented as a network of interconnected components. Each component contains variables denoting the attributes of the component and may contain mathematical equations to describe how the component behaves within the model. SBML (Systems Biology Markup Language) [12] is an XML-based format for representing the biochemical reaction networks common to computational biology research such as cell signalling pathways, metabolic pathways, and gene regulation. It represents a chemical reaction using a number of components: reaction compartment, reactant and product species, reaction descriptions, parameters, units of quantities, and mathematical rules to set parameters and quantities. CellML and SBML are not designed to represent biomolecular modelling or to describe biomolecular simulations.

CML (Chemical Markup Language) [13] is an XML representation of chemical information including compounds, molecules and molecular computations, crystallography, and spectra. It provides XML elements to describe molecules, atoms, electrons, and bonds as well as metadata. CMLReact [14] provides a set of additional components for CML to represent chemical reactions. However, CML is not designed for the integration of biomolecular models as it does not capture the atomic information used by some simulators such as the TINKER XYZ format, nor can it represent the mapping between different molecular data formats which is required for data conversion.

The PDB format provides a standard representation for 3D macromolecular structures, which has been widely used by molecular simulation packages. PDBML [15] is the XML representation for the PDB Exchange dictionary [16] that does not directly represent the PDB format. The PDB Exchange dictionary is a superset of the PDB crystallographic protein structure data, which describes the information of internal bookkeeping, non-crystallographic structure determination methods (e.g. nuclear magnetic resonance–NMR), details of crystallographic experiments and protein production. The PDB Exchange dictionary is described in the mmCIF (macromolecular Crystallographic Information File) format that is largely different from the PDB format. As the PDBML designers indicated, a PDB file cannot be converted to the mmCIF format in a completely automated manner because many mmCIF data items either are not present in the PDB format or are present in the non-parsable REMARK record of the PDB format [17]. PDBML is not a general description for other molecular data either, such as the TINKER XYZ format.

The CCP Data Model [18] was developed to cover all data needed for a macromolecular NMR spectroscopy, from the initial experimental data to the final validation. It is used for exchanging data between programs, for the storage of data and database deposition. The data model is implemented in the modelling language UML to create an abstract description of the data and the inter-data relationship. The CCP data model does not target the domain of biomolecular modelling.

The Open Babel project [19] provides a chemistry file translation program and a C++ library to support the interconversion between various file formats in molecular modelling and computational chemistry. The library has a function to convert between the PDB and TINKER XYZ format. However, our tests with Open Babel 2.0 and 2.1.1 have shown that the program cannot correctly convert the molecular file of arachidonic acid [20] in our QM/MM simulation discussed in Section 3 from the PDB to XYZ format.

To realise the data interoperability, we have developed the BioSimML markup language to support data conversion between different models and data representations. BioSimML provides XML representations for the essential data in biomolecular modelling. With the support of the associated toolkit, BioSimML has been successfully used to implement the automated data conversion between MD and QM/MM simulations and to accomplish the dynamic integration of multi-level biomolecular simulations running on a grid [21, 22].

3 BioSimML

BioSimML is a domain specific XML language that provides a structured representation for the essential data in biomolecular modelling. Figure 1 shows the structure of BioSimML. The

current BioSimML version has defined the XML constructs `pdb` and `xyz` to capture molecular structures originally described in the PDB format and the TINKER XYZ format. BioSimML also defines a construct `mdp` (molecular dynamics parameters) to capture the GROMACS [23] simulation parameters. GROMACS is one of the molecular dynamics packages that we have used in our multi-level biomolecular simulations. BioSimML also defines metadata components to describe the general information of a biomolecular simulation and the condition to integrate two biomolecular simulation models [22]. To enable the data conversion between biomolecular models and simulation packages, BioSimML also provides a representation for the mapping of atom name and type in one molecular data format to the notation of another data format.

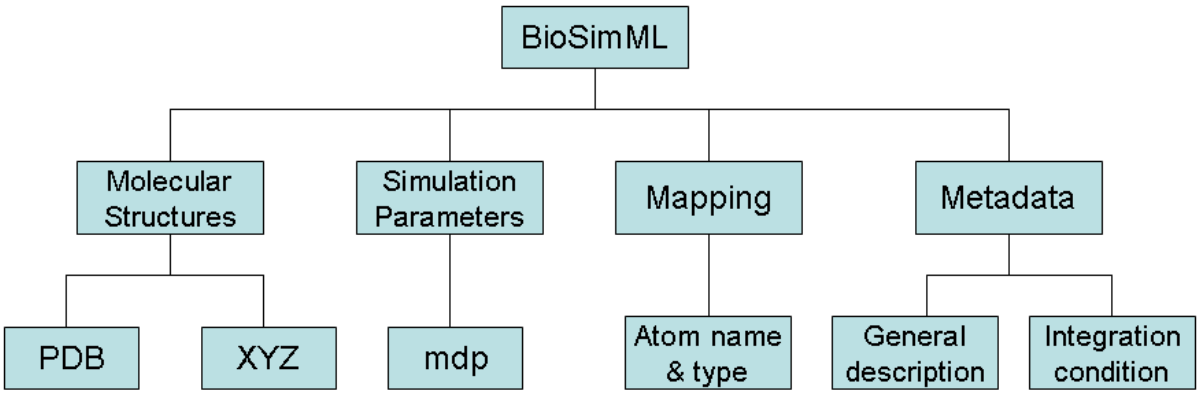

Figure 1: The structure of BioSimML markup language

The syntax of BioSimML is defined with an XML schema language RELAX NG [24] that uses the XML syntax to specify an XML document structure. The RELAX NG specification can be used for the validation of BioSimML captured data against the BioSimML syntax. Due to the length of this paper, we only present the BioSimML representations related to the data conversion discussed in the paper. For example, the RELAX NG specification for the BioSimML construct of the TINKER XYZ format is defined as following. The XYZ format describes the Cartesian coordinates of atoms in a molecular system.

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- BioSimML construct for TINKER XYZ format -->
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
 datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
  <start>
    <element name="xyz">
      <ref name="xyzPattern"/>
    </element>
  </start>

  <!-- outmost pattern of XYZ format -->
  <define name="xyzPattern">
    <!-- title line -->
    <optional>
      <element name="title">
        <data type="string"/>
      </element>
    </optional>
    <!-- atom lines -->
    <oneOrMore>
      <ref name="atomLine"/>
    </oneOrMore>
  </define>
```

```
  <!-- atom line -->
  <define name="atomLine">
    <element name="atom">
      <attribute name="serial">
        <data type="int"/>
      </attribute>
      <attribute name="name">
        <data type="string"/>
      </attribute>
      <attribute name="x">
        <data type="float"/>
      </attribute>
      <attribute name="y">
        <data type="float"/>
      </attribute>
      <attribute name="z">
        <data type="float"/>
      </attribute>
      <attribute name="type">
        <data type="int"/>
      </attribute>
      <element name="connected.atoms">
        <list>
          <oneOrMore>
            <data type="int"/>
          </oneOrMore>
        </list>
      </element>
    </element>
  </define>
</grammar>
```

Figure 2 shows a sample XYZ file. Figure 2(a) is a fragment of arachidonic acid described in the XYZ format and Figure 2(b) shows the corresponding BioSimML representation of the data. Figure 3 is the same fragment described in the PDB format and the BioSimML representation. Figure 4 shows the mapping of atom names and atom types in this fragment between the CHARMM and TINKER notations originally specified in the CHARMM RTF (residue topology file) and the TINKER PRM (parameter) file with the BioSimML description of this mapping.

```
22  CEL  21.813000  -2.387000  23.707000  130   20   23   24
23  HEL  22.393000  -2.526000  24.631000  120   22
24  CL2  20.742000  -1.379000  23.761000  125   22   27   25   26
25  HL2  20.660000  -1.065000  24.823000  118   24
26  HL2  19.774000  -1.923000  23.750000  118   24
27  CEL  20.974000  -0.109000  22.938000  130   24   29   28
28  HEL  21.166000  -0.428000  21.903000  120   27
```

(a) The XYZ format

```
<xyz>
   <atom serial="22" name="CEL" x="21.813" y="-2.387" z="23.707" type="130">
         <connected.atoms>20   23   24</connected.atoms></atom>
   <atom serial="23" name="HEL" x="22.393" y="-2.526" z="24.631" type="120">
         <connected.atoms>22</connected.atoms></atom>
   <atom serial="24" name="CL2" x="20.742" y="-1.379" z="23.761" type="125">
         <connected.atoms>22   27   25   26</connected.atoms></atom>
   <atom serial="25" name="HL2" x="20.660" y="-1.065" z="24.823" type="118">
         <connected.atoms>24</connected.atoms></atom>
   <atom serial="26" name="HL2" x="19.774" y="-1.923" z="23.750" type="118">
         <connected.atoms>24</connected.atoms></atom>
   <atom serial="27" name="CEL" x="20.974" y="-0.109" z="22.938" type="130">
         <connected.atoms>24   29   28</connected.atoms></atom>
   <atom serial="28" name="HEL" x="21.166" y="-0.428" z="21.903" type="120">
         <connected.atoms>27</connected.atoms></atom>
</xyz>
```

(b) BioSimML representation of the XYZ format

Figure 2: The arachidonic acid described in the XYZ format and the BioSimML representation

In the XYZ format in Figure 2(a), the data fields from left to right are atom serial number, atom name, Cartesian coordinates (`x`, `y`, and `z`), atom type number, and a list of atoms connected to

the current atom. The PDB format in Figure 3(a) is a strictly column-oriented representation of macromolecular structure. It specifies the data fields including the identical record name `ATOM`, atom serial number, atom name, residue name, chain identifier, residue sequence number, Cartesian coordinates (`x`, `y`, and `z`), occupancy, and temperature factor.

```
ATOM  22    C9  ACD A  2     21.588 -1.411 24.227 1.00 0.00
ATOM  23   H91  ACD A  2     21.964 -1.636 25.236 1.00 0.00
ATOM  24   C10  ACD A  2     20.550 -0.327 24.232 1.00 0.00
ATOM  25  H101  ACD A  2     20.073 -0.271 25.234 1.00 0.00
ATOM  26  H102  ACD A  2     19.708 -0.627 23.573 1.00 0.00
ATOM  27   C11  ACD A  2     21.180  0.925 23.772 1.00 0.00
ATOM  28  H111  ACD A  2     21.610  0.800 22.767 1.00 0.00
```

(a) The PDB format

```
<pdb>
  <atom serial="22" atom_name="C9"    res_name="ACD" chain_id="A" res_seq="2"
        x="21.588" y="-1.411" z="24.227" occupancy="1.00" tempFactor="0.00"/>
  <atom serial="23" atom_name="H91"  res_name="ACD" chain_id="A" res_seq="2"
        x="21.964" y="-1.636" z="25.236" occupancy="1.00" tempFactor="0.00"/>
  <atom serial="24" atom_name="C10"  res_name="ACD" chain_id="A" res_seq="2"
        x="20.550" y="-0.327" z="24.232" occupancy="1.00" tempFactor="0.00"/>
  <atom serial="25" atom_name="H101" res_name="ACD" chain_id="A" res_seq="2"
        x="20.073" y="-0.271" z="25.234" occupancy="1.00" tempFactor="0.00"/>
  <atom serial="26" atom_name="H102" res_name="ACD" chain_id="A" res_seq="2"
        x="19.708" y="-0.627" z="23.573" occupancy="1.00" tempFactor="0.00"/>
  <atom serial="27" atom_name="C11"  res_name="ACD" chain_id="A" res_seq="2"
        x="21.180" y="0.925"  z="23.772" occupancy="1.00" tempFactor="0.00"/>
  <atom serial="28" atom_name="H111" res_name="ACD" chain_id="A" res_seq="2"
        x="21.610" y="0.800" z="22.767" occupancy="1.00" tempFactor="0.00"/>
</pdb>
```

(b) BioSimML representation of the PDB format

Figure 3: The arachidonic acid described in the PDB format and the BioSimML representation

```
RESI ACD       0.00

ATOM   C9  CEL1  -0.15
ATOM  H91  HEL1   0.15
ATOM  C10  CTL2  -0.18
ATOM H101  HAL2   0.09
ATOM H102  HAL2   0.09
ATOM  C11  CEL1  -0.15
ATOM H111  HEL1   0.15
```

(a) CHARMM RTF file

```
##############################################################
##  CHARMM Atom Class to TINKER Atom Class Number           ##
##                                                          ##
##   66 HL     70 HEL1   74 CTL3   78 OSL                   ##
##   67 HAL1   71 CL     75 CTL5   79 OBL                   ##
##   68 HAL2   72 CTL1   76 CEL1   80 O2L                   ##
##   69 HAL3   73 CTL2   77 NTL    81 PL     82 OCL        ##
##                                                          ##
##############################################################
atom   118   68   HL2   "Methylene Hydrogen"  1    1.008   1
atom   120   70   HEL   "-CH=CR2 Hydrogen"    1    1.008   1
atom   125   73   CL2   "Methylene Carbon"    6   12.011   4
atom   130   76   CEL   "-CH=CR2 Carbon"      6   12.011   3
```

(b) TINKER PRM file

```
<atom.mappings source.format="charmm" target.format="tinker">
   <map>
      <source name="CEL1 number="76" attr.name="charge" attr.value="-0.15" />
      <target name  ="CEL" number="130" attr.name="charge" attr.value="-0.15" />
   </map>
   <map>
      <source name="HEL1" number="70" attr.name="charge" attr.value="0.15" />
      <target name  ="HEL" number="120" attr.name="charge" attr.value="0.15" />
   </map>
</atom.mappings>
```

(c) BioSimML description of the mapping

Figure 4: The mapping of atom names and types between the CHARMM and TINKER notations

In Figure 4(a), the CHARMM RTF file contains the data fields as atom name (e.g., C9), atom class (e.g., CEL1), and the charge of atom (e.g., -0.15). The TINKER PRM file in Figure 4(b) defines the mapping of CHARMM atom class to TINKER atom class number in the header, followed by the definition of atom entries. By referring to these two files, a CHARMM atom name can be mapped to the corresponding TINKER atom name. For example, the first atom in the CHARMM RTF file is a carbon C9 with the atom class CEL1. According to the mapping data in the header of the TINKER PRM file, the CHARMM atom class CEL1 is mapped to the TINKER atom class number 76. Searching the atom entries below the header, the corresponding TINKER atom name for class number 76 can be determined as CEL with the TINKER atom type number 130 as defined in the fourth entry. This is the process for converting a CHARMM atom name (e.g., C9) in the PDB file in Figure 3(a) to the TINKER atom name (e.g., CEL) in the XYZ file in Figure 2(a). The BioSimML description in Figure 4(c) shows the mapping

of the first two atoms in the CHARMM RFT file to the TINKER atom name and atom type number. All the BioSimML representations in Figure 2 to Figure 4 will be used for the data conversion in Section 4.

BioSimML, as a general data representation, provides a solution to the automatic data conversions in biomolecular modelling. With the support of XML-based data processing tools, BioSimML can be used to readily implement data conversions between different biomolecular data formats as shown in Figure 5. A data representation such as a PDB file or an XYZ file can be captured into a BioSimML document object on which data conversion operations can be applied to modify its contents. Finally, the modified contents of the BioSimML document object can be sent as output to a data file in another format (e.g., XYZ or PDB).

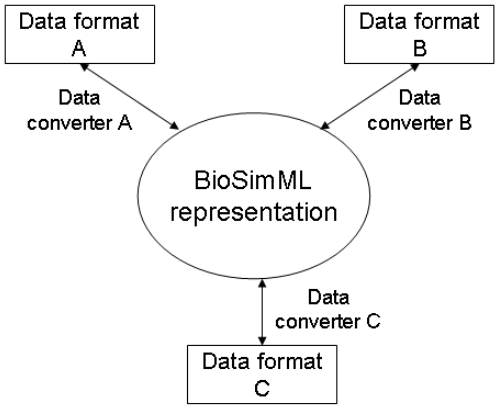

Figure 5: Data conversion between different data formats based on the BioSimML representation

We have developed a toolkit to implement the BioSimML-based data processing and conversion. The toolkit includes parsers to read in a PDB or XYZ file, project the data into a BioSimML document object and write out a BioSimML document object into a PDB or XYZ file. The toolkit also provides a collection of programs to perform data conversions in our multi-level biomolecular simulations. The toolkit is implemented in Java and JDOM [25] (a Java API for processing XML data). Table 1 shows the programs of the toolkit useful to the MD to QM/MM data conversion.

The toolkit is extensible. When a new data format is introduced, a new data converter will be developed for the conversion between the new data format and the BioSimML representation. Therefore, data conversion can be achieved between the new format and existing data formats captured by BioSimML.

4 Data Conversion

In our research, QM/MM modelling is used to study the mechanisms of membrane bound enzyme reactions in cellular environments. As a molecular model built on the atomic and subatomic descriptions of biological system, a QM/MM simulation is extremely computationally expensive that prohibits a long-timescale simulation to be realistic. To break through this limitation, the more efficient molecular dynamics is employed to create representative enzyme conformations in advance using long-timescale MD simulations. The conformation of an enzyme is formed by the interactions between amino acids of the enzyme. Thereafter, QM/MM

Program	Function
PDBXMLHandler	converts between the PDB format and the BioSimML representation.
XYZXMLHandler	converts between the XYZ format and the BioSimML representation.
MDPXMLHandler	converts between the GROMACS mdp file and the BioSimML representation.
AtomMapHandler	builds a BioSimML representation to capture the mapping of atom name and type between CHARMM RTF and TINKER PRM.
XYZModifier	modifies a BioSimML representation of TINKER XYZ file.

Table 1: The programs of the BioSimML toolkit

simulations can be conducted based on the selected conformations to simulate the chemical reactions of enzyme including chemical bond breaking and making between the enzyme and a cell membrane.

CHARMM is used for the MD simulations and then TINKER for the QM/MM simulations. CHARMM produces the data of enzyme conformation in the PDB format. To input the data to TINKER, the PDB format needs to be converted to the XYZ format. The conformation data contains the specification of atomic structures of the enzyme and cell membrane. The TINKER package provides the pdbxyz program to convert a PDB file into an XYZ file. The pdbxyz program can correctly convert the PDB representation of a protein (enzyme). However, the program usually fails to convert non-protein molecules. For example, phospholipids are the structural components of a cell membrane. Arachidonic acid is an omega-6 fatty acid present in phospholipids. Unfortunately the pdbxyz program cannot correctly convert the PDB description of arachidonic acid shown in Figure 3(a) to the corresponding XYZ format as shown in Figure 2(a). Instead, it generates an incorrect XYZ representation as shown in Figure 6.

```
22  C9   21.813000  -2.387000  23.707000   0  20  23  24
23  H91  22.393000  -2.526000  24.631000   0  22
24  C10  20.742000  -1.379000  23.761000   0  22  27
25  101  20.660000  -1.065000  24.823000   0
26  102  19.774000  -1.923000  23.750000   0
27  C11  20.974000  -0.109000  22.938000   0  24  29
28  111  21.166000  -0.428000  21.903000   0
```

Figure 6: The incorrect XYZ file of arachidonic acid generated by the pdbxyz **program**

Comparing Figure 6 with Figure 2(a), we notice that all atom names and type numbers are wrong and all lists of connected atoms except the first list are incomplete. The incorrect atom names and types are caused by the different notations of CHARMM and TINKER on atom

names and the diverse data fields defined in the PDB and the XYZ format. The CHARMM RTF file and the TINKER PRM file in Figure 4(a) and (b) show that the CHARMM atom name C9 should be mapped to the TINKER atom name CEL with the atom type number 130. Due to the fact that the PDB format allows up to four characters to denote an atom name but the XYZ format only allows 3 characters, the `pdbxyz` program truncates a 4-charater atom name such as the hydrogen H101 and H102 in the PDB file down to 3 characters such as 101 and 102 in the XYZ file. To build the lists of connected atoms, moreover, the `pdbxyz` program uses standard protein connectivity that can determine the connected atoms in a protein. For non-protein molecules, however, `pdbxyz` assumes the atom connectivity based on interatomic distances. This assumption cannot generate correct lists of connected atoms in the arachidonic acid as shown in Figure 6.

To cope with this problem, the PDB to XYZ conversion can be rectified by means of the BioSimML representation. To amend the XYZ file, we need to reference the CHARMM atom names in the original PDB file in Figure 3(a) and the mapping of atom names and types defined in the CHARMM RTF file and the TINKER PRM file in Figure 4(a) and (b). Figure 7 demonstrates the workflow of data correction for an XYZ file. The workflow represents an automated data correction process which is implemented based on the BioSimML representation and the toolkit. The entire workflow will be executed each time when an XYZ file needs to be amended. Firstly, the incorrect XYZ file is projected into a BioSimML document object in the format as shown in Figure 2(b). The source PDB file is also projected into a BioSimML document object as shown in Figure 3(b). Meanwhile, two BioSimML document objects of atom name and type mapping are built to describe the mapping of CHARMM atom name to atom class and the mapping of CHARMM atom class to TINKER atom name and atom type number in the form as shown in Figure 4(c). The data correction is performed on the BioSimML document object of the incorrect XYZ file. To rectify an atom name, e.g. 101, the original CHARMM atom name H101 will be extracted from the BioSimML document object of the PDB file by referencing the same atom serial number 25. The CHARMM atom name H101 is then used as a key to retrieve the atom class HAL2 from the BioSimML document object of CHARMM atom name to atom class mapping. The CHARMM atom class HAL2 is further used to retrieve the correct TINKER atom name HL2 and type number 118 from the BioSimML document object of CHARMM atom class to TINKER atom name and type number mapping. In this manner, all atom names and type numbers of the arachidonic acid can be rectified in the BioSimML document object of the XYZ file. Finally, the lists of connected atoms are rectified in this BioSimML document object based on the chemical structure of arachidonic acid, i.e., all carbons are chained in sequence and each hydrogen is bonded to the immediately leading carbon atom in the XYZ file. For example, HEL is connected to CEL while HL2 is connected to CL2 as shown in Figure 2(a). When the data correction has completed, the modified BioSimML document object is output to a new XYZ file.

We adopt the method to rectify the incorrect XYZ file generated by the `pdbxyz` program rather than directly producing an XYZ file from the original PDB file, because this method has been in use by our project collaborators in their QM/MM simulations.

The example above only shows a segment of the data file used in a QM/MM simulation. A full data file contains hundreds of arachidonic acids and other non-protein molecules. Any attempt to correct the XYZ file by hand will be extremely inefficient and error-prone. BioSimML with the associated toolkit provides an effective means to automatically implement the data conversion. In addition to the data conversion presented in this paper, BioSimML has been

used for the data conversion between other biomolecular simulation packages such as between NAMD [26] and CHARMM. BioSimML has also been used to support the integration of multi-level biomolecular simulations on a grid [21, 22].

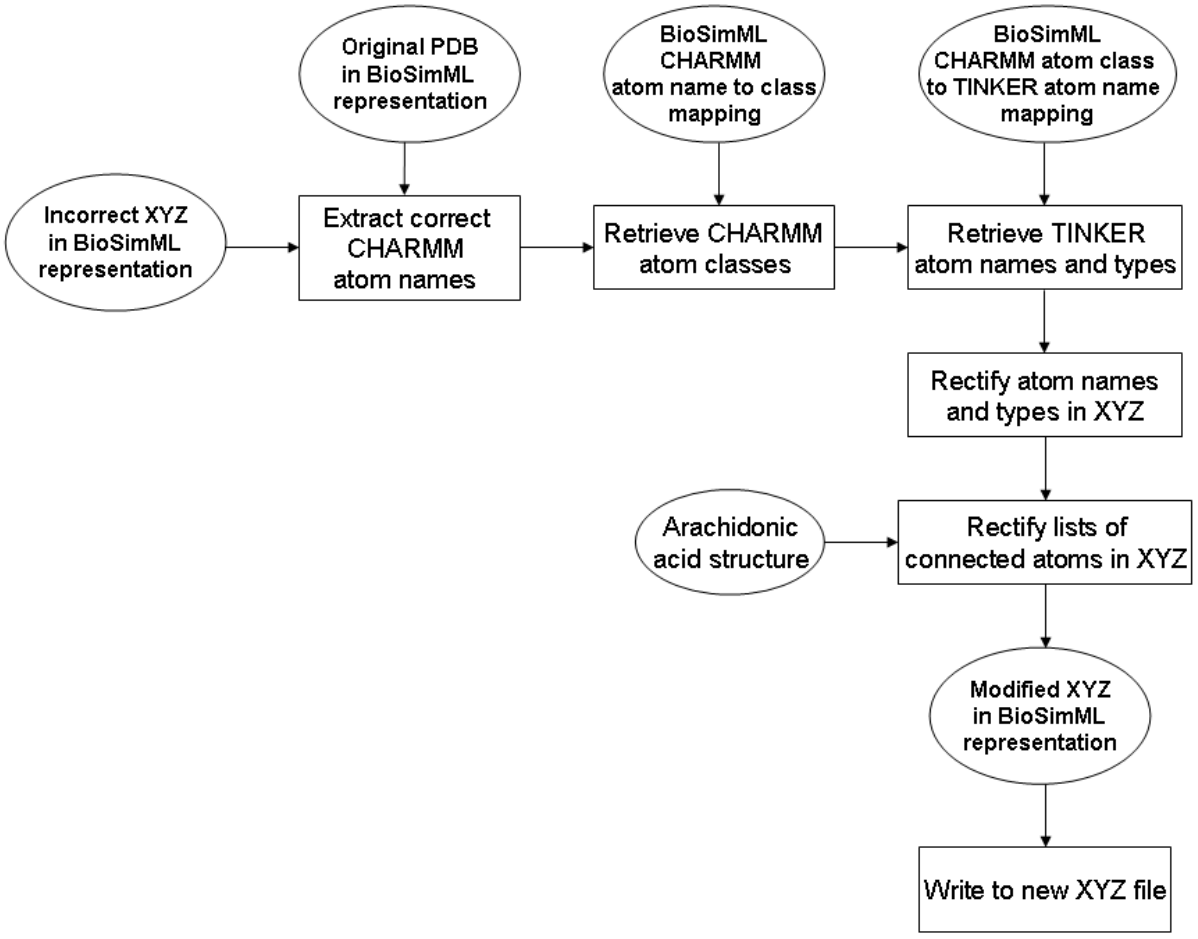

Figure 7: The workflow for correcting the XYZ file

5 Conclusions

As an XML-based data representation, BioSimML has proved a useful and efficient tool to facilitate the data interoperability between different biomolecular models, simulators, and data formats. We will continue the development of this representation to enhance its capability for representing a wide range of data in the domain of biomolecular modelling.

The current version of BioSimML defines individual constructs for capturing molecular structures originally described in the PDB and XYZ formats. To enhance the data interoperability of BioSimML, we plan to develop a unified BioSimML representation for molecular structure which combines major attributes from different data formats. Comparing the PDB and XYZ formats, we can observe that the two formats have common attributes such as atom serial number, atom name, and Cartesian coordinates. On the other hand, each format has unique attributes such as the residue name and residue sequence number in the PDB format and the

atom type number and a list of connected atoms in the XYZ format. Different attributes are required by individual biomolecular simulators. We will design a comprehensive XML construct to capture the major attributes of atoms from different formats. BioSimML will be extensible to incorporate new attributes. Additionally, there are essential systematic data in biomolecular modelling, including the topology (e.g., bonded and non-bonded interactions of atoms) and the force field (used for the calculation of potential energy between atoms). These data types are important to the data interoperability and to the integration of biomolecular models. We will design BioSimML representations for these data types and develop tools for the data transformation of these data types between different biomolecular models. These developments will hopefully lead to an ontology of biomolecular modelling.

Acknowledgments

The authors thank the research collaborators, Dr Adrian Mulholland and Dr Christo Christov at University of Bristol, UK who provided the data files and conversion requirements of QM/MM simulation.

References

[1] P. Bond and M. Sansom. Insertion and assembly of membrane proteins via simulation. *Journal of the American Chemical Society*, 128:2697–2704, 2006.

[2] B. Roux and K. Schulten. Computational studies of membrane channels. *Structure*, 12:1343–1351, 2004.

[3] P. Fowler, K. Balali-Mood, S. Deol, P. Coveney, and M. Sansom. Monotopic enzymes and lipid bilayers: a comparative study. *Biochemistry*, 46(11):3108–3115, 2007.

[4] Andrew R. Leach. *Molecular modelling: principles and applications*. Prentice Hall, 2 edition, 2001.

[5] A. Mulholland. Modelling enzyme reaction mechanisms, specificity and catalysis. *Drug Discovery Today*, 10:1393–1402, 2005.

[6] M. Karplus and J. McCammon. Molecular dynamics simulations of biomolecules. *Nature Structural Biology*, 9:646–652, 2002.

[7] Charmm. `http://www.charmm.org/info/links.shtml`.

[8] TINKER. `http://dasher.wustl.edu/tinker/`.

[9] Protein Data Bank. `http://www.rcsb.org/`.

[10] F. Achard, G. Vaysseix, and E. Barillot. XML, bioinformatics and data integration. *Bioinformatics*, 17(2):115–125, 2001.

[11] C. Lloyd, M. Halstead, and P. Nielsen. CellML: its future, present and past. *Progress in Biophysics & Molecular Biology*, 85(2–3):433–450, 2004.

[12] M. Hucka, A. Finney, H. Sauro, H. Bolouri, J. Doyle, et al. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19:524–531, 2003.

[13] P. Murray-Rust, H. Rzepa, and M. Wright. Development of chemical markup language (CML) as a system for handling complex chemical content. *New Journal Of Chemistry*, 25:618–634, 2001.

[14] G. Holliday, P. Murray-Rust, and H. Rzepa. Chemical markup, XML, and the world wide web. 6. CMLReact, an XML vocabulary for chemical reactions. *Journal of Chemical Information & Modelling*, 46(1):145–157, 2006.

[15] J. Westbrook, N. Ito, H. Nakamura, K. Henrick, et al. PDBML: the representation of archival macromolecular structure data in XML. *Bioinformatics*, 21(7):988–992, 2005.

[16] PDB Dictionary Resources. `http://mmcif.pdb.org/`.

[17] P. Bourne, H. Berman, B. McMahon, K. Watenpaugh, J. Westbrook, and P. Fitzgerald. Macromolecular crystallographic information file. *Methods in Enzymology*, 277:571–590, 1997.

[18] The CCP Data Model. `http://www.ccpn.ac.uk/ccpn/data-model`.

[19] The Open Babel Package. `http://openbabel.sourceforge.net/`.

[20] Arachidonic acid. `http://www.acnp.org/g4/GN401000059/CH059.html`.

[21] Y. Sun, S. McKeever, K. Balali-Mood, and M. Sansom. A multiscale model for efficient simulation of a membrane bound viral fusion peptide. In *Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine (BIBM 2007)*, pages 294–301. IEEE Computer Society Press, 2007.

[22] Y. Sun, S. McKeever, K. Balali-Mood, and M. Sansom. Integrating multi-level molecular simulations across heterogeneous resources. In *Proceedings of tne 8th IEEE/ACM International Conference on Grid Computing (Grid 2007)*, pages 161–168. IEEE Press, 2007.

[23] E. Lindahl, B. Hess, and D. Spoel. GROMACS 3.0: a package for molecular simulation and trajectory analysis. *Journal of Molecular Modeling*, 7:306–317, 2001.

[24] RELAX NG. `http://www.relaxng.org/`.

[25] JDOM. `http://www.jdom.org/`.

[26] J. Phillips, R. Braun, W. Wang, J. Gumbart, et al. Scalable molecular dynamics with NAMD. *Journal of Computational Chemistry*, 26(16):1781–1802, 2005.