Refactoring the Code for Visualizing Protein Database Information in a 3D Viewer for Software Reusability

Yoo Jin Chun², Seong Il Ham³, San Duk Yang², Arang Rhie² and Hyun Seok Park¹²*

¹Institute of Bioinformatics, Macrogen Inc., Seoul 153-023, Korea, ²Department of Computer Science, Ewha Womans University, Seoul 120-750, Korea, ³Department of Architectural Engineering, Yonsei University, Seoul, 120-749, Korea

Abstract
We have released five Java Application Programming Interface (API) packages for viewing three-dimensional structures of proteins from the Protein Data Bank. To this end, the user interface of an earlier version has been refactored in an object-oriented fashion, in which refactoring is the process of changing a software system to improve its internal structure, without altering the external behavior. Various GUI design and features have been provided conveniently thanks to the Model-View-Control (MVC) model, which is an architectural pattern used in software engineering.

Availability: The source code and API specification can be downloaded from https://sourceforge.net/projects/j3dpsv/.

Keywords: API, Bioinformatics, Protein structure visualization, Refactoring

Introduction
Since a protein structure had been visualized entirely with computers in the mid 1970’s (Beem et al., 1977), numerous programs for visualizing 3D protein structures have been described (Sayle et al., 1995; Humphrey et al., 1996; Bernstein et al., 2000; DeLano et al., 2002; Berman et al., 2002; Westbrook et al., 2003; O’Donoghue et al., 2004; Herraez et al., 2006), and programs such as Jmol (http://www.jmol.org/), RasMol (http://www.umass.edu/microbio/rasmol/), Cn3D (http://www.ncbi.nlm.nih.gov/Structure/CN3D/cn3d.shtml), SRS3D (http://www.srs3d.org/), PyMOL (http://www.pymol.org/), UCSF Chimera (http://www.cgl.ucsf.edu/chimera/), VMD (http://www.ks.uiuc.edu/Research/vmd/), YASARA (http://www.yasara.org/), and BRAGI (http://bragi.gbf.de/) have been distributed. However, as most developers were targeting end-users only, the readability of the source code was generally poor.

We had built a 3D protein visualization module from scratch as part of a large bioinformatics project (Cho et al., 2007). However, the architecture of the initial system was poorly designed such that people outside of the formal developer group had a hard time in understanding, modifying, and even utilizing the module. To remedy the problem, we refactored the entire code interfaces of the initial version recently to enhance software reusability and code readability.

Moreover, we decided to participate in an open source project by releasing five jar packages and the Application Programming Interface² (API) specification in Fig. 1 to the public through the Sourceforge³ open source repository. API differs from an end product in that it is the source code interface that a library provides to support requests for services by other programs. This dissemination may make it possible to save software development time for future developers.

An Exemplary GUI System
Reusable modules and classes reduce implementation time and increase the likelihood of eliminating bugs and localizing code modifications when a change in implementation is required. In the initial version of the interface, all of the classes got tangled with each other and corrupted the concept of object-oriented programming. However, they have been completely redesigned, as shown in Table 1. This table summarizes the recent modifications of our system, and the interfaces for each class are documented, similar to Fig. 2. The refactored version is now composed of 3851 lines, compared with the initial version, which was composed of 2765 lines of code.

By importing the five packages, an exemplary software system called J3dPSV⁴ 1.0, shown in Fig. 3, has been developed for viewing 3D structures of proteins from the Protein Data Bank⁵ for demonstrational purposes. J3dPSV supports visualization of proteins for educational purposes by simulating simple molecular graphics. In addition, J3dPSV interactively displays a molecule on the screen in a variety of color schemes, molecular representations, and animation features. The molecular model can be changed by selecting the list (cartoon tubes, backbone, protein, cylinder, or line) in...
Refactoring the PDB Visualization Code

The Style combo box (by importing threeDModel.jar in Table 1). In addition to the default color scheme, structures can be highlighted either by selecting the appropriate amino acid properties or by selecting secondary structure types in the Select combo box. Ligands or water can be displayed or hidden by selecting the radio buttons between the Style and the Select combo boxes. A rotational animation of the molecule can be played by clicking the play button (▶) on the toolbar.

In complex computer applications that present a large amount of data to the user, a developer often wishes to separate data (model) and user interface (view), so that changes in the graphical user interface will not affect data handling. Indeed, dividing the program’s structural architecture between model and view increases data reorganization without any change in the user interface. The main class of J3dPSV roughly corresponds to the view part, and the five packages in the compressed jar files correspond to the model part in the MVC architecture, respectively.

Table 1. Initial version vs. refactored version

<table>
<thead>
<tr>
<th>Classes</th>
<th>Initial version</th>
<th>Packages</th>
<th>Refactored version</th>
<th>Packages</th>
</tr>
</thead>
<tbody>
<tr>
<td>AtomObject</td>
<td>60</td>
<td>pdbviewer</td>
<td>AtomObject</td>
<td>60</td>
</tr>
<tr>
<td>HelixObject</td>
<td>53</td>
<td></td>
<td>HelixObject</td>
<td>53</td>
</tr>
<tr>
<td>NamedColor</td>
<td>29</td>
<td></td>
<td>NamedColor</td>
<td>33</td>
</tr>
<tr>
<td>Protocol</td>
<td>67</td>
<td></td>
<td>Protocol</td>
<td>53</td>
</tr>
<tr>
<td>PDBOpen</td>
<td>205</td>
<td></td>
<td>PDBOpen</td>
<td>205</td>
</tr>
<tr>
<td>ThreeDModel</td>
<td>835</td>
<td></td>
<td>Protein3DVisualization</td>
<td>1,157</td>
</tr>
<tr>
<td>PDBViewer (main)</td>
<td>664</td>
<td></td>
<td>StyleBar</td>
<td>155</td>
</tr>
<tr>
<td>PickMouse</td>
<td>311</td>
<td>pdbviewer_tool</td>
<td>PickMouse</td>
<td>408</td>
</tr>
<tr>
<td>TransColorModel</td>
<td>430</td>
<td></td>
<td>TransColorModel</td>
<td>470</td>
</tr>
<tr>
<td>TransModel</td>
<td>111</td>
<td></td>
<td>TransModel</td>
<td>118</td>
</tr>
</tbody>
</table>

Total 2,765

Total lines in jar files: 3,471
Lines out of jar files: 380

Total 3,851
Conclusion & Future Direction

Bioinformatics is the field of science in which biology and information technology merge into a single discipline; research in bioinformatics includes method development for storage and retrieval issues of data. According to National Institute of Mental Health, the development and implementation of tools that enable efficient access and management of different types of information is one of the important sub-disciplines within bioinformatics, in addition to the development of new algorithms and statistics.

This paper discusses the implementation issues, Computer technologies such as Web Services6 and refactoring are becoming significant in the field of bioinformatics tool development to improve internal consistency and clarity. By developing an exemplary GUI system, we demonstrated that our refactored packages can be efficiently imported into large-scale bioinformatics applications, for which an integrative analysis and visualization play a vital role. It is noteworthy that J3dPSV provides most of the fundamental features of 3D protein visualization, with only 380 additional lines of code after importing the general purpose packages.

Currently, our packages do not provide a command language interface for selecting parts of the molecule for a special display mode. For more advanced users, we plan to provide APIs of built-in command languages and structural prediction. We will constantly upgrade our APIs and provide additional interfaces with more powerful features.

Acknowledgements

The PDBOpen class in the pdbController.jar package only works with traditional PDB files. Minor modifications could be necessary to make it compatible with recent changes in the PDB file format to address inconsistencies (Henrick et al., 2008).

This work was partially supported by a grant from the NEXT program of the Ministry of Information and Communication of Korea. We would like to thank Young Joo Seol, and Jang Ho Hahn of National Institute for Agricultural Biotechnology, for various communications.

1Other numerous software programs are also distributed through large Open Source software development projects, such as SOURCEFORGE.NET (http://sourceforge.net) and Biojava (http://www.biojava.org) consortium, or by releasing their software in online journals such as Source Code for Biology and Medicine (http://www_scfbm.org/).
2An API is the specific method prescribed by an application program by which a programmer writing an application program can make requests of another application.
3A source code repository and centralized location for software developers to control and manage open source software development.
4The sequence table of J3dPSV is connected to the 3D viewer. Clicking a mouse button while hovering over an appropriate part of the selected molecule will highlight both the 3D image and the sequence of this part of the molecule, and the label above the sequence table will display information about this part. Dragging the sequence table will bring a similar result. As many PDB structures consist of multiple subunits, clicking the Chain ID table allows individual chains in the structure to be viewed or hidden.
5The Protein Data Bank archive maintains the coordinates and related information for more than 47,000 structures, including proteins, nucleic acids, and large macromolecular complexes that
have been determined using X-ray crystallography, NMR, and
electron microscopy techniques.

Web services are frequently just Web APIs that can be accessed
over a network, such as the Internet, and executed on a remote
system hosting the requested services.

References

Beem, K.M., Richardson, D.C., and Rajagopalan, K.V.
(1977). Metal sites of copper-zinc superoxide


Bernstein, H.J. (2000). Recent changes to RasMol, re-
combining the variants, *Trends Biochem. Sci.* 25, 453-
455.

Cho, M.K., et al. (2007). The AB05 NIAB tools workbench
for building automatic biopathway maps for agricultural

DeLano, W.L. (2002). The PyMOL molecular graphics sys-
tem, *DeLano Scientific,* (San Carlos, CA, USA: DeLano
Scientific)

Henrick, K., et al. (2008). Remediation of the protein data


O’Donoghue, S.J., Meyer, J.E., Schafferhans, A., and Fries,
K. (2004). The SRS 3D module: integrating structures, se-
quen ces, and features, *Bioinformatics* 20, 2476-2478.

molecular graphics for all, *Trends Biochem. Sci.* 20, 374-
376.

Westbrook, J., Feng, Z., Chen, L., Yang, H., and Berman,
H.M. (2003). The protein data bank and structural geno-