APPLICATION NOTE Genomics & Informatics Vol. 6(1) 50-53, March 2008

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

Yoo Jin Chun[2], Seong Il Ham[3], San Duk Yang[2], Arang Rhie[2] and Hyun Seok Park[1,2]*

[1]Institute of Bioinformatics, Macrogen Inc., Seoul 153-023, Korea, [2]Department of Computer Science, Ewha Womans University, Seoul 120-750, Korea, [3]Department 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/)[1] 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.

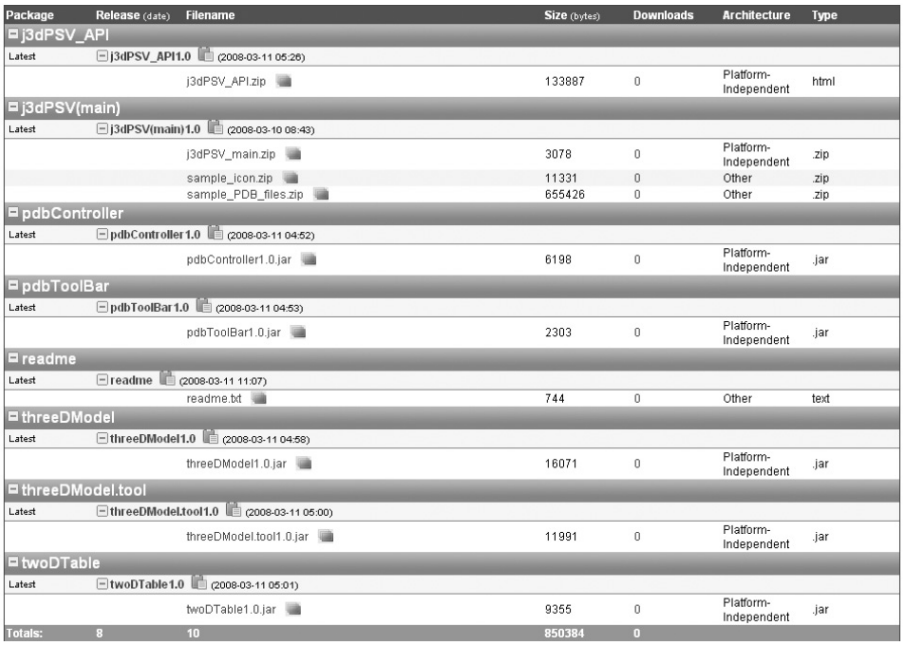
Moreover, we decided to participate in an open source project by releasing five jar packages and the Application Programming Interface[2] (API) specification in Fig. 1 to the public through the Sourceforge[3] open source repository. API differs from an end product in that it is the source code interface 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 J3dPSV[4] 1.0, shown in Fig. 3, has been developed for viewing 3D structures of proteins from the Protein Data Bank[5] 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

*Corresponding author: E-mail neo@ewha.ac.kr
Tel +82-2-3277-2831, Fax +82-2-3277-2306
Accepted 3 March 2008



Package	Release (date)	Filename	Size (bytes)	Downloads	Architecture	Type
j3dPSV_API						
Latest	j3dPSV_API1.0 (2008-03-11 05:26)					
		j3dPSV_API.zip	133887	0	Platform-Independent	html
j3dPSV(main)						
Latest	j3dPSV(main)1.0 (2008-03-10 08:43)					
		j3dPSV_main.zip	3078	0	Platform-Independent	.zip
		sample_icon.zip	11331	0	Other	.zip
		sample_PDB_files.zip	655426	0	Other	.zip
pdbController						
Latest	pdbController1.0 (2008-03-11 04:52)					
		pdbController1.0.jar	6198	0	Platform-Independent	.jar
pdbToolBar						
Latest	pdbToolBar1.0 (2008-03-11 04:53)					
		pdbToolBar1.0.jar	2303	0	Platform-Independent	.jar
readme						
Latest	readme (2008-03-11 11:07)					
		readme.txt	744	0	Other	text
threeDModel						
Latest	threeDModel1.0 (2008-03-11 04:58)					
		threeDModel1.0.jar	16071	0	Platform-Independent	.jar
threeDModel.tool						
Latest	threeDModel.tool1.0 (2008-03-11 05:00)					
		threeDModel.tool1.0.jar	11991	0	Platform-Independent	.jar
twoDTable						
Latest	twoDTable1.0 (2008-03-11 05:01)					
		twoDTable1.0.jar	9355	0	Platform-Independent	.jar
Totals:	8	10	850384	0		

Fig. 1. Jar packages: pdbController to get information from PDB files and set basic variables; pdbToolBar to open files and control 3D images; threeDModel to draw and change view types of 3D images; threeDModel.tool to handle events related to 3D images; twoDTable to handle amino acid sequences

Table 1. Initial version vs. refactored version

Initial version			Refactored version			
Classes	Lines	Packages	Classes	Lines		Packages
AtomObject	60	pdbviewer	AtomObject	60	404	pdbController
HelixObject	53		HelixObject	53		
NamedColor	29		NamedColor	33		
Protocol	67		Protocol	53		
PDBOpen	205		PDBOpen	205		
ThreeDModel	835		Protein3DVisualization	1,157	1,312	threeDModel
PDBViewer (main)	664		StyleBar	155		
			SequenceTable	576	576	twoDTable
			PDBViewer (main)	380	380	*(pdbviewer)*
			PDBToolBar	183	183	pdbToolBar
PickMouse	311	pdbviewer.tool	PickMouse	408	996	threeDModel.tool
TransColorModel	430		TransColorModel	470		
TransModel	111		TransModel	118		
Total	2,765		Total lines in jar files	3,471		Total 3,851
			Lines out of jar files	380		

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.

Constructor Summary

	Protein3DVisualization()

Method Summary

void	aminoTypeView(int aminoViewType)
javax.media.j3d.Canvas3D	getCanvas3D()
void	ligandsView(boolean checkLigandsView)
void	open3D()
void	partiallyView(int viewType, int selectedChain, java.util.Vector checkChain)
void	secondaryStructureView()
void	setViewType(int viewType)
void	tempView()
void	waterView(boolean checkWaterView)

Fig. 2. An exemplary API specification for Protein3DVisualization class.

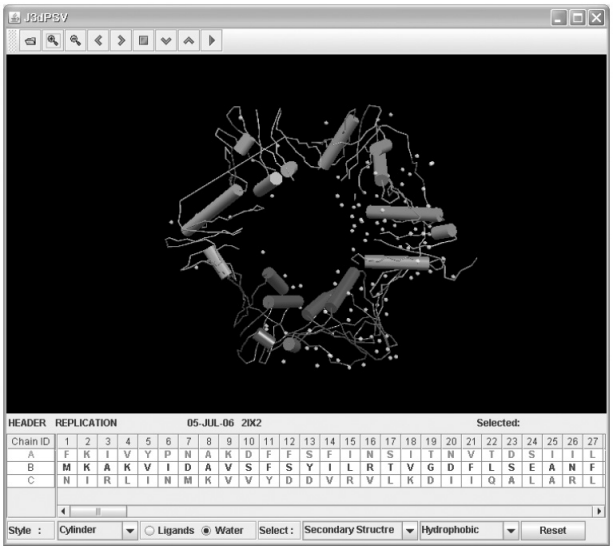


Fig. 3. Screen capture of J3dPSV: three windows are displayed: the main 3D graphics window in the middle (importing threeDModel.jar), the sequence table below it (importing twoDTable.jar), and the menu form and the toolbar on the top (importing pdbToolBar.jar). The 3D viewer and the multiple chain sequence table are connected together.

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 Services[6] 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.

---

[1]Other 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/).

[2]An API is the specific method prescribed by an application program by which a programmer writing an application program can make requests of another application.

[3]A source code repository and centralized location for software developers to control and manage open source software development.

[4]The 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.

[5]The 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.

[6]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 dismutase. *Biochemistry* 16, 1930-1936.

Berman, H.M., Battistuz, T., Bhat, T.N., Bluhm, W.F., Bourne, P.E., Burkhardt, K., Feng, Z., Gilliland, G.L., Iype, L., Jain, S., *et al*. (2002). The protein data bank. *Biopolymers* 22, 2577-2637.

Bernstein, H.J. (2000). Recent changes to RasMol, recombining 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 organisms. *G&I*. 5, 95-97.

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

Henrick, K. *et al*. (2008). Remediation of the protein data bank archive. *Nar*. 36, D426-433.

Herra´ez, A. (2006). Biomolecules in the computer: jmol to the rescue. *Biochem. Educ*. 34, 255-261.

Humphrey, W., Dalke, A., and Schulten, K. (1996). VMD-visual molecular dynamics. *J. Mol. Graphics* 14, 33-38.

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

Sayle, R.A., and Milner-White, E.J. (1995). RASMOL: biomolecular 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 genomics. *Nar*. 31, 489-491.