BIOINFORMATICS

Vol. 20 no. 5 2004, pages 727–734
DOI: 10.1093/bioinformatics/btg476

Java-based application framework for visualization of gene regulatory region annotations

*Hao Sun and Ramana V. Davuluri**

Human Cancer Genetics Program, Comprehensive Cancer Center, Department of Molecular Virology, Immunology and Medical Genetics, The Ohio State University, 420 W 12th Avenue, TMRF 524, Columbus, OH 43210, USA

Received on December 16, 2002; revised on September 5, 2003; accepted on October 14, 2003
Advance Access publication January 29, 2004

ABSTRACT

Motivation: The genome sequences of several organisms are either complete, or being sequenced. Each genome needs to be integrated with various types of annotations, e.g. locations of genes, promoters and other functional elements such as transcriptional regulatory elements. A robust application framework will be useful for developing web-based applications to visualize various genome annotations.

Results: We developed genome data visualization toolkit (GDVTK) as an application framework that consists of a set of data structures and core classes, using Java™ technology. GDVTK is a sound framework for developing web-based applications to present the gene regulatory region annotations in visual form. The current version of GDVTK consists of eight packages and 38 Java classes that are portable, reusable and extensible for plugging in new data sources and models. We implemented GDVTK for visualization of promoter annotations in Mammalian Promoter Database (MPromDb), a web-based gene-regulatory information server.

Availability: GDVTK is available under GNU general public license. Source code and software documentation can be found at the URL http://bioinformatics.med.ohio-state.edu/GDVTK.

Contact: Davuluri-1@medctr.osu.edu

1 INTRODUCTION

The field of molecular biology is producing large amounts of genome data in recent years (Benson *et al*., 2002), and will continue to grow exponentially especially after successful completion of the human (Lander *et al*., 2001) and mouse (Waterston *et al*., 2002) genomes. Complex genomic data is driving the bioinformatics community to develop new approaches to process and present these data. The presentation of genome data and associated annotations in visual forms, such as diagrams and images, can help biologists mine the enormous data in an easy and manageable way.

The bioinformatics community has developed several web-based and stand-alone applications (Rzhetsky *et al*., 1998; Loraine and Helt, 2002; Helt *et al*., 1998; Koike and Rzhetsky, 2000; Wilkinson *et al*., 2002; Siepel *et al*., 2001; Pook *et al*., 1998; Kraemer and Ferrin, 1998) for genome data visualization. A *Caenorhabditis elegans* database (Eeckman and Durbin, 1995), which began as a database to keep track of *C.elegans* strains and information from genetic crosses has been evolved to provide mapping and annotation information of DNA and protein sequences (Kelley, 2000). The Ensembl database project (http://www.ensembl.org) (Hubbard *et al*., 2002), UCSC genome browser (Kent *et al*., 2002; Karolchik *et al*., 2003) (http://genome.ucsc.edu) and NCBI Map Viewer (Wheeler *et al*., 2002) (http://www.ncbi.nlm.nih.gov/mapview) are three leading providers of human and mouse genome annotations and visualization. Recently, a generic genome browser (GBrowse; Stein *et al*., 2002) and a sequence annotation editor (Apollo; Lewis *et al*., 2002) were developed as part of generic model organism system database project (http://www.gmod.org). These genome browsers provide excellent applications to meet different requirements of genome data visualization. However, each application uses its own underlying data model, database management system, server-side software, programming language and has to meet its own functional requirements. A limitation of these applications for building new genome visualization projects, such as visualization of gene regulatory regions and annotations of transcription factor binding sites, is the reusability of the existing source codes.

Genome data visualization toolkit (GDVTK) is an application framework that provides a set of Java APIs (http://bioinformatics.med.ohio-state.edu/GDVTK/docs/index.html) to develop web-based applications for gene regulatory region

*To whom corrrespondence should be addressed.

visualization. The core of the GDVTK framework is a collection of several flexible packages based on J2EE technologies, such as Java Server Pages (JSPs) and Java Servlets. GDVTK provides support for multi-tier application architectures and any standard data access technology including Enterprise Java Beans, and JDBC for the application's database access layer. GDVTK can help build platform independent, reusable, modular, extensible and scalable web-based applications for genome data visualization, based on published standards and proven design patterns on J2EE platform. Rather than implementing these services from scratch, application developers can find it more efficient to use these standard implementations.

2 SYSTEMS AND METHODS

2.1 Concept of genome data visualization and visualization pipeline

Genome data is a subset of biological data that includes the annotations of the genome sequence. The genome data visualization is a process that transfers the genome data into two-dimensional, visually understandable images in which the genomic annotations are represented by different geometric shapes. This process includes several steps that are arranged in the visualization pipeline. According to Lang *et al*. (1995), the database visualization process consists of three fundamental steps (Fig. 1). Data acquisition is the first step that pre-processes the genome data and extracts annotation information from the back-end database. Data-mapping (modeling) is the next step and the core of the visualization process, in which the genome data from the previous step is mapped into different primitive geometrical shapes with appropriate attributes, such as color or opacity. During this step, the visualization pipeline decides which geometric primitive should be generated and what should be assigned to their attributes. Hence the data-mapping step has the biggest influence in the final visual representation of the genome data. The last step of the visualization pipeline is rendering, which generates the final output image by using the geometric primitives from the mapping step.

2.2 Software functional requirements

We designed GDVTK as a framework that can implement web-based applications for genome data visualization with multi-tier architecture based on J2EE technology. The major functional requirements for GDVTK are:

1. GDVTK must provide a mechanism by which a developer can query a database and obtain the object that represents the database data table. Since the genome data is generally stored in heterogeneous database systems, the implementation should be generic to different database systems. This requirement corresponds to the data acquisition step of the visualization pipeline.

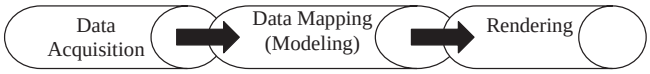

Fig. 1. Pipeline of genome data visualization.

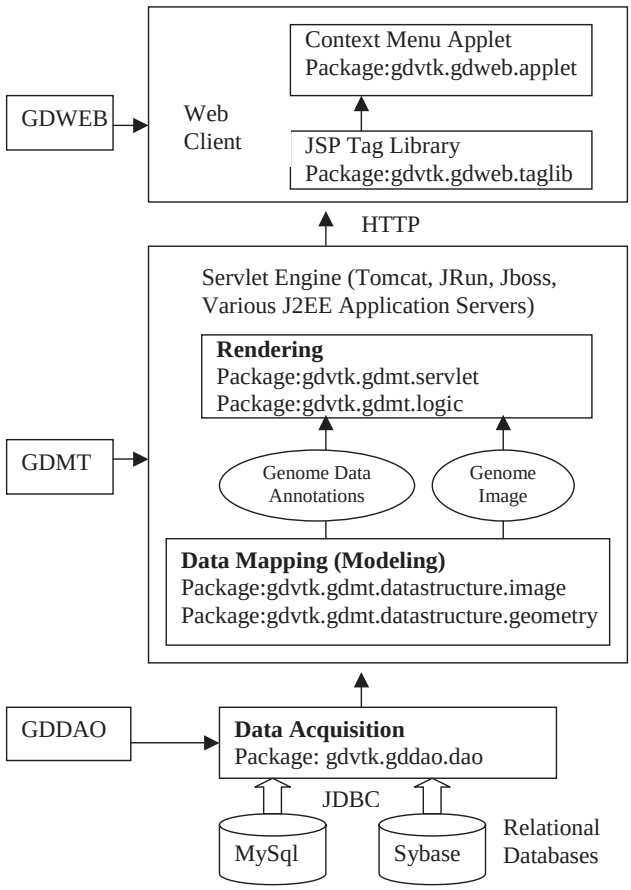

Fig. 2. System architecture for GDVTK.

2. It must support the data-mapping step and provide implementation of data structures that represent core primitive geometries and mapping methods. It must also provide mechanisms to associate the attributes with the primitive geometries in order to get a meaningful genome data presentation.
3. The GDVTK framework must provide support for implementation of the user interface for easy navigation and extraction of the data and associated annotations of interest.

2.3 System architecture

GDVTK (Fig. 2) is a three-tier application framework with relational databases (e.g. MySql and Sybase) that stores genome data in a table-based structure in the back-end. Three modules, Genome Data Database Access Object module (GDDAO), Genome Data Middle Tier module (GDMT) and

Table 1. A list of genome data and GDVTK classes that were used during the data mapping step, and the corresponding visual representations. In this table, exons are presented with black square boxes; genes/mRNAs with black square boxes connected by lines. Arrows are used to provide the strand orientation of the gene

Genome data	GDVTK classes	Visual representations
Exons	SequenceFragment	
First exon	FirstExonMark, SequenceFragment	
Genes/mRNAs	mRNAConnectors, SequenceFragment	
CpG island	SequenceFragment	
Protein binding Site	SequenceFragment	

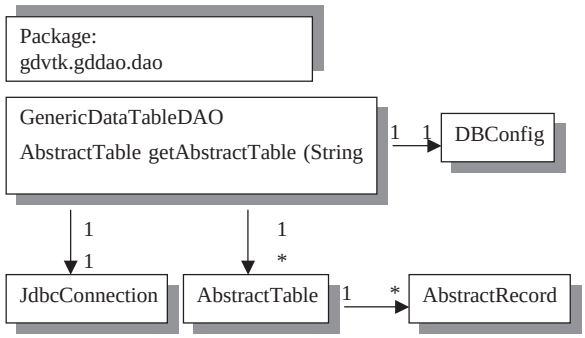

Fig. 3. The class diagram of GDVTK package gdvtk.gddao.dao: 'solid line with an arrow' denotes composition relationship between classes. Each class involved in a composition may specify a multiplicity. A multiplicity is a number that specifies the number of objects of the class that are involved in the relationship. The symbol '*' indicates that the number of objects is unlimited.

Genome Data Web module (GDWEB), are implemented to support each tier. The implementation of the visualization pipeline is embedded in each GDVTK module. The GDDAO module starts the pipeline by acquiring the genome data from the relational database system. This data acquisition step preprocesses the data into a generic and standard format for the following step. The GDMT module then maps, or models, and renders the acquired genome data into their respective primitive geometries (Table 1). It also assigns the appropriate attributes, such as pre-defined colors and direction to the geometries, and generates two data objects; genome data annotation and genome image objects. The rendering step renders the genome image object on the physical device to generate the image (e.g. image with GIF format on the computer hard disk). The rendered image and genome annotation object are then sent to the GDWEB module via HTTP protocol. The GDWEB module then transfers the genome data annotation object with a JSP tag library and constructs a contextual menu for the rendered image with transferred genome annotation data. Finally, it provides the users with a visually understandable image with interactive navigation support.

3 IMPLEMENTATION

3.1 GDDAO module

In GDDAO module (Fig. 2), we implement a package, called gdvtk.gddao.dao, to support the data acquisition step in visualization pipeline. The major goal of this package is to provide a generic database access layer in order to fit different database schemas. To enhance the portability and reusability of GDVTK, the gdvtk.gddao.dao package (Fig. 3) provides GenericDataTableDAO class to access different database systems by using DBConfig objects. We use AbstractTable class as a neutral interface to represent the data objects in database tables. We also provide an access function getAbstractTable() to get the AbstractTable data object. This design helps to isolate the database access operations from other modules in GDVTK, hence any change of the database schema will not affect the codes in other modules, provided the developer uses the AbstractTable object to access the database systems.

To enhance the scalability, GDVTK provides two options: For small to medium size database systems, JdbcConnection class provides an easy and expedient way to access the database systems, while Enterprise JavaBeans (EJB) is the option for large database systems. Since EJBs are implemented under the J2EE specification, the J2EE platform and container may provide a wide range of services, such as data object caching and JDBC connection pooling, to enhance the scalability.

3.2 GDMT module

Implementation of data modeling is the second and the most critical step of data visualization procedure. We design a representative data structure and Java class for each genome data type, and implement the basic operations to draw different geometrical shapes and colors in the Java classes. In the GDMT module (Fig. 2), we provide two key packages: gdvtk.gdmt.datastructure.geometry and gdvtk.gdmt.datastructure.image. Package gdvtk.gdmt.datastructure.geometry implements the data-mapping (modeling) step in the visualization pipeline. It provides the implementation of a set of data structures that represent different types of genome data. The genome data types, corresponding GDVTK classes, and visual representations are shown in Table 1. The class diagram of this package is shown in Figure 4.

The package gdvtk.gdmt.datastructure.geometry (Fig. 4) has a base class called PrimitiveGeometry and various derived types (subclasses): FirstExonMark, mRNAConnectors, Ruler, SequenceFragment, etc. Inheritance is perhaps the most appropriate method in the design of this package, since inheritance has several advantages. Polymorphism (dynamic binding) was implemented by inheriting from base class PrimitiveGeometry and overriding the abstract draw method.

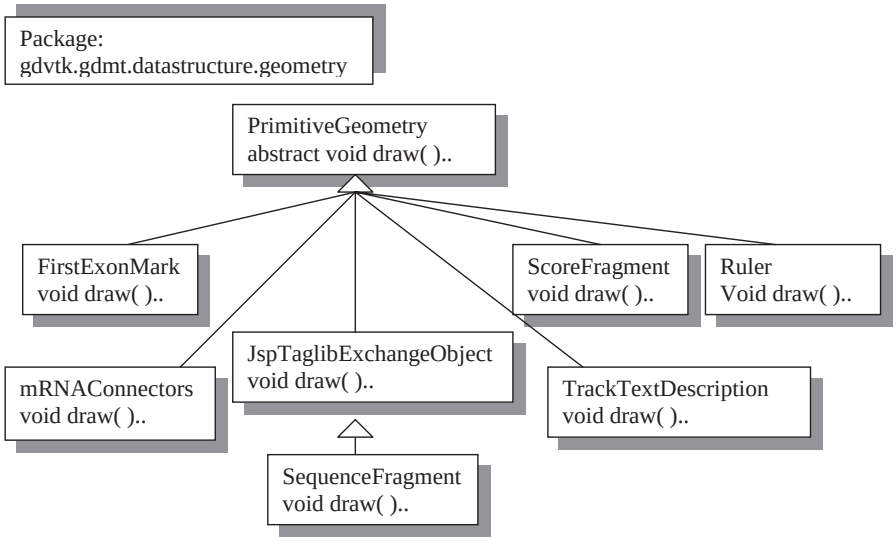

Fig. 4. The class diagram for GDVTK package gdvtk.gdmt.datastructure.geometry: Inheritance is denoted with an open triangle pointing towards the superclass and a solid line connecting superclass and subclass. The inheritance models 'is–a' relationship between classes, as shown in this class diagrams.

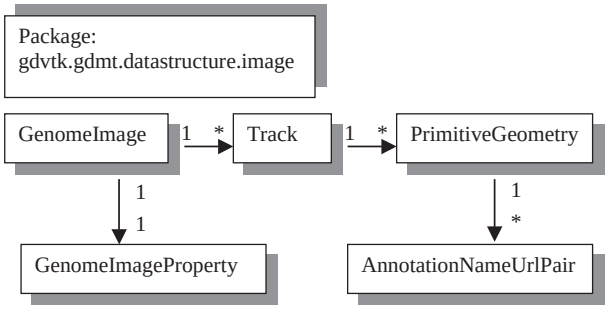

Fig. 5. The class diagram for GDVTK package gdvtk.gdmt.datastructure.image.

This isolates type specific details of PrimitiveGeometry objects from other packages in GDVTK, which further helps the rendering process so that it only needs to access the base class methods, such as the draw method in the PrimiviteGeometry class. This design is very critical for the extensibility of GDVTK to plug in other data structures and models. Genome data are large collections of sequence data and annotations that accompany each assembly. Thus far, there are several data types such as mRNA, gene predictions, cross-species homologies, etc. with certainly many more data types in the future. The isolation of type specific information helps the rendering package (Fig. 2) to represent PrimitiveGeometry objects without knowing the exact data type of genome data. This design helps to add as many new classes that are required, by extending from the PrimitiveGeometry class for new genome data types during the mapping step of visualization pipeline.

Package gdvtk.gdmt.datastructure.image provides the implementation of the viewing transformation step of visualization. We use the class GenomeImage (Fig. 5) to represent the virtual image generated for genome data visualization. This image includes several tracks (Track Class) and each track includes several primitive geometries. Each PrimiviteGeometry contains a set of AnnotationNameUrlPair storing the genome annotation data associated with each PrimiviteGeometry object. We use class GenomeImageProperty to characterize the geometric properties of the virtual image.

3.3 GDWEB module

Web-based applications for genome data visualization require dynamic image generation and mechanisms for easy navigation. A typical scenario for a user to navigate the visualized genome data through a web browser is: the user rolls over the mouse on the genome data of interest so that a context menu pops up to provide a set of menu items representing the associated annotation information. The user can then use different menu items to retrieve the corresponding data and annotation information. GDVTK implements this scenario by rendering the image on the server side, while on the client side, a JSP is used to render the HTML code necessary to download a Java applet, which can then download the server-side generated image and the associated annotations by setting appropriate initialization parameters for the applet. Since this method uses a server-side generated image without communicating with the database directly, the speed and scalability are improved considerably. GDVTK supports this approach by providing APIs by implementing two packages gdvtk.gdweb.taglib and gdvtk.gdweb.applet in GDWEB.

The package gdvtk.gdweb.taglib provides a custom JSP tag library that defines a set of tags to render the HTML code based on the genome data from GDMT module to specify the initial parameters for classes in gdvtk.gdweb.applet package. The tags in this package look like HTML (or XML) tags embedded in a JSP page. These tags have a special meaning to a JSP engine at translation time, and enable application functionality to be invoked without the need to write Java code in JSP page. The package gdvtk.gdweb.applet implements a context menu, which is initiated with the parameters set up by HTML code generated by GDVTK JSP tags. It supports the user to navigate the visualized image.

4 GDVTK IN PRACTICE

Genome data visualization toolkit is being developed as an application framework to provide the necessary support for developing web-based genome data visualization applications. We have successfully implemented GDVTK in developing different web-based genome data visualization applications; First Exon Genome Browser (Davuluri *et al*., 2003a; http://bioinformatics.med.ohio-state.edu/FEGB), Mammalian Promoter Database (MPromDb; Sun *et al*., 2003; http://bioinformatics.med.ohio-state.edu/MPromDb) and Arabidopsis *cis*-regulatory element database (Davuluri *et al*., 2003b; http://arabidopsis.med.ohio-state.edu/AtcisDB).

We explain the implementation of GDVTK in the development of MPromDb (http://bioinformatics.med.ohio-state.edu/MPromDb), an information resource of mammalian promoters with associated annotations, such as first exon, transcription factor binding sites and CpG islands. Users can search this database with Gene Symbol, GenBank Accession Number, Unigene ID, LocusLink ID, and Transcription Factor Name (Binding Protein Name). MPromDb also presents the genome data in a visualized form. The Web Figure 1 in GDVTK demo page (http://bioinformatics.med.ohio-state.edu/GDVTK/MPromDbDemo.jsp) shows a virtual image generated for visualization of MPromDb record that contains annotation information for a pair of homologous genes (human BAX and mouse Bax). The image contains a series of horizontal tracks, each displaying a particular type of annotation. For example, the promoter annotation track provides information about the transcription factor (TF) binding sties (small colored square boxes), and transcription start site (TSS) (small vertical line connected with a small horizontal line with arrow indicating the gene orientation) and first exon (rectangular colored box). It also provides a context menu for the user to display the annotation information for TF binding sites (e.g. P53 binding site with the genomic position, and the DNA binding sequence). CpG scores (Davuluri *et al*., 2001; CpG score is the maximum of CpG% of all sliding windows of length 201 bp) at 50 bp intervals were plotted, along with a threshold line of value six to categorize whether the promoter sequence is CpG related or not.

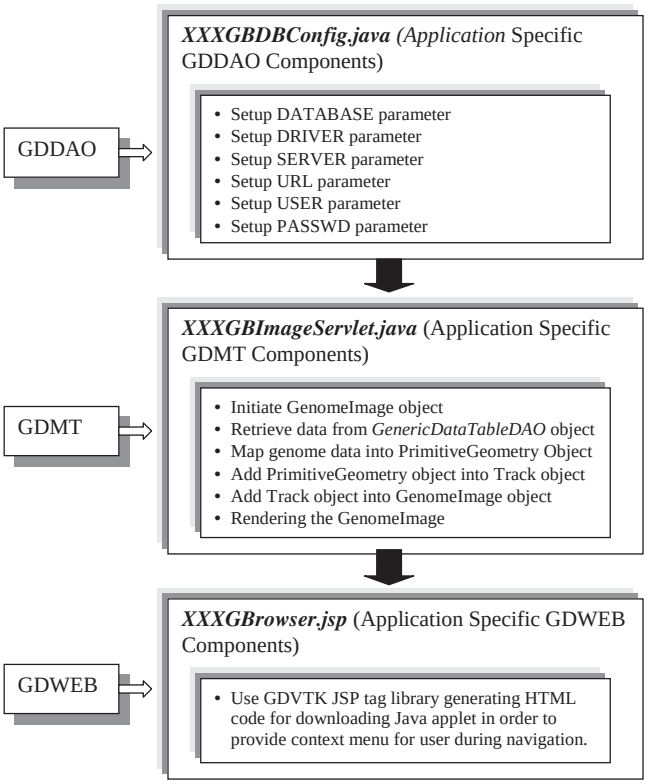

Fig. 6. Web-based application development template.

We have used all the three modules (GDDAO, GDMT and GDWEB) of GDVTK (Fig. 2) in building MPromDb, and the following steps demonstrate the implementation. The user can download the source code from GDVTK webpage.

(1) *Retrieve data from database*

The data of MPromDb is stored in MySql relational database system. In this step, we implement a class MPromDbDBConfig, by inheriting from DBConfig class of gdvtk.gddao.dao package in GDDAO module, to set up the MPromDb specific database connection parameters (Fig. 6). We also implement MPromDbAdaptor class by making use of GenericDataTableDAO and DBConfig classes provided in gdvtk.gddao.dao package (Fig. 3) to implement the customer queries based on different search criteria initiated by the users. GDMT and GDWEB modules can use the MPromDbADaptor class as a database portal to retrieve the genome data from the database.

(2) *MPromDb genome data modeling*

The basic genome data types in MPromDb are promoters, first exons and the associated annotations, such as TSS, TF binding sites and CpG score (Table 1). GDVTK can model all these genome data types with different classes provided in gdvtk.gdmt.datastructure.image (Fig. 5) package of GDMT module. We specifically used the classes SequenceFragment, FirstExonMark, ScoreFragment, and ruler (Fig. 4) to model

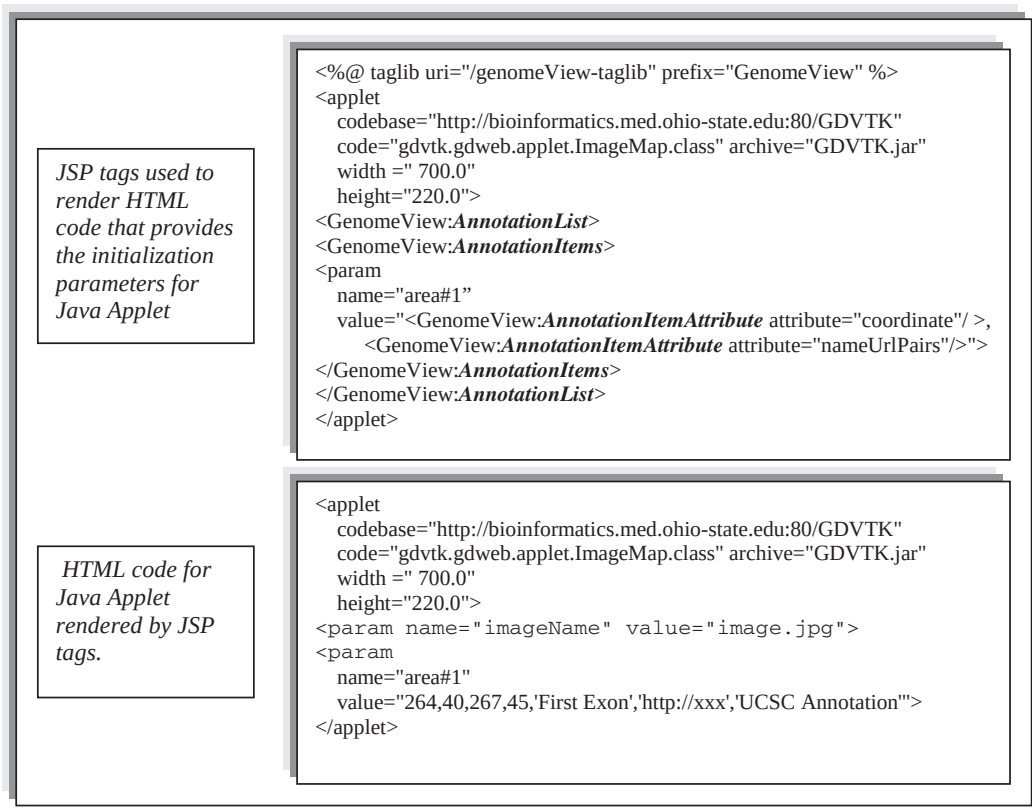

Fig. 7. The JSP tag library used to render HTML code and the rendered HTML code for Java applet.

the data types 'exon', 'TSS', 'CpG score' and 'Genomic Coordinates' (Table 1), respectively. SequenceFragment and FirstExonMark were combined to model 'first exon' data type. Further, we have used the utility classes, such as GenomeImage, Track etc. (Fig. 5), to glue different data structures. The data structure GenomeImage represents the virtual image of genome data annotation (Web Fig. 1). This genome image has three different types of horizontal Tracks; ruler, promoter annotation and CpG Score. Finally, these data structures were used to generate a virtual image with Render class in the package: gdvtk.gdmt.logic, and this image was passed to the client side JSP for presentation.

(3) *Web presentation of the GenomeImage object*

We have used two packages gdvtk.gdweb.applet and gdvtk.gdweb.taglib (Fig. 2) for implementation of the web side presentation of MPromDb data. We implement client side JSP pages to load the virtual image generated by the server side, and then add the annotations to this image with the help of the Java Applet and JSP customer tag library (Fig. 7). As shown in Figure 7, the package gdvtk.gdweb.taglib provides a custom JSP tag library that defines a set of tags, such as, GenomeView, to render the HTML code during run time. By using the GenomeView tag, the annotation information associated with genome image was retrieved from the server side, and the corresponding HTML codes were generated with the initialization parameters of the Java Applet. The applet implemented in the package: gdvtk.gdweb.applet can then make use of these parameters to generate the context menu. The context menu supports the user to navigate the visualized image (Web Fig. 1).

5 DISCUSSION AND CONCLUSION

The development of GDVTK was inspired by the open source Bioinformatics software such as BioJava (Pocock *et al*., 2000) and BioPerl (Stajich *et al*., 2002). It is primarily developed for applications that involve the visualization of gene regulatory regions, such as promoters. GDVTK is being developed using JAVA technology, because of its 'write once, run anywhere' concept and popularity in today's information technology. GDVTK is platform independent, since it is implemented with pure Java and does not need any external libraries to compile. It can be used to build web-based genome data visualization applications on any platform with J2EE support. The web-based application is more portable by using the JDBC and EJB as a database access layer. GDVTK is reusable, because it is designed as a library and provides a set of Java APIs for developers to build different web-based applications. The

modularity of GDVTK is shown in the design of its architecture by distributing different tasks needed for visualization of genome data into different tiers and modules. The architecture of GDVTK is also easy to adapt to meet new and ever-changing requirements. For example, the type specific information of gdvtk.gdmt.datastructure package is isolated from other packages of GDVTK; hence, this package can be extended to include more genome data types in the future.

By design, GDVTK is a complementary to Apollo (Lewis *et al*., 2002), the genome annotation editor of the GMOD project. Unlike GDVTK, which was designed as a library, Apollo is a stand-alone Java application that runs on the end-user's machine. GDVTK is also complementary to GBrowse (Stein *et al*., 2002). The main difference between GDVTK and GBrowse is their use of different platforms to build web-based applications. GBrowse uses Perl with common gateway interface (CGI) technology. CGI is a Web standard for accessing external programs and to integrate databases with Web servers. The drawback of CGI is that it handles web request by creating an additional process on the server machine. Instead, GDVTK uses threads to handle the request by Java Servlet, and hence the web-application built on GDVTK can save the CPU time and memory. This is an important advantage of web-based application that deals with gigabytes of genome data. While there are many advantages for use of GDVTK, one caveat is the high learning curve associated with it, specifically due to the complexities of J2EE. Future directions involve the implementation of additional data structures for genome data-mapping.

ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewers for their constructive suggestions, which greatly helped improve this manuscript and GDVTK. We also thank Twyla for the careful reading of the manuscript, Saranyan, Zhaohui and Anurag for implementing and preparing some of the web pages and figures, and Huating and Sandya for support in biological questions.

REFERENCES

Benson,D.A., Karsch-Mizrachi,I., Lipman,D.J., Ostell,J., Rapp,B.A. and Wheeler,D.L. (2002) *GenBank Nucleic Acids Res.*, **20**, 17–20.

Davuluri,R.V., Grosse,I. and Zhang,M.Q. (2001) Computational identification of first exons and promoters in the human genome. *Nat. Genet.*, **29**, 412–417.

Davuluri,R.V., Grosse,I. and Zhang,M.Q. (2003a) Annotation of promoters and first exons in the human genome. *Genome Res.*, in press.

Davuluri,R.V., Sun,H., Palaniswamy,S.K., Matthews,N., Molina,C., Kurtz,M. and Grotewold,E. (2003b) AGRIS: Arabidopsis gene regulatory information server, an information resource of Arabidopsis cis-regulatory elements and transcription factors. (2003) *BMC Bioinformatics*, **4**, 25.

Eeckman,F.H. and Durbin,R. (1995) ACeDB and macace. *Meth. Cell Biol.*, **48**, 583–605.

Helt,G.A., Lewis,S., Loranie,A.E. and Rubin,G.M. (1998) BioViews: Java-based tools for genomic data visualization. *Genome Res.*, **8**, 291–305.

Hubbard,T., Barker,D., Birney,E., Cameron,G., Chen,Y., Clark,L., Cox,T., Cuff,J., Curwen,V., Down,T., *et al*. (2002) The Ensembl genome database project. *Nucleic Acids Res.*, **30**, 38–41.

Karolchik,D., Baertsch,R., Diekhans,M., Diekhans,T.S., Furey,A., Hinrichs,Y.T., Lu,K.M., Roskin,M., Schwartz,C.W., Sugnet,D.J., Thomas,R.J., Weber,D., Haussler and Kent,W.J. (2003) The UCSC Genome Browser Database. *Nucleic Acids Res.*, **31**, 51–54.

Kelley,S. (2000) Getting started with Acedb. *Brief. Bioinformat.*, **1**, 131–137.

Kent,W.J., Sugnet,C.W., Furey,T.S., Roskin,K.M., Pringle,T.H., Zahler,A.M. and Haussler,D. (2002) The human genome browser at UCSC. *Genome Res.*, **12**, 996–1006.

Koike,T. and Rzhetsky,A. (2000) A graphic editor for analyzing signal-transduction pathways. *Gene*, **259**, 235–244.

Kraemere,T. and Ferrin,T.E. (1998) Molecules to maps: tools for visualization and interaction in support of computational biology. *Bioinformatics*, **14**, 764–771.

Lang,U., Grinstein,G. and Bergeron,R.D. (1995) Visualization related metadata. *Database Issues for Data Visualization: IEEE Visualization '95 workshop*, Atlanta, GA (*LectureNotes in Computer Science*, vol. 1183). Springer-Verlag, Heidelberg, Germany, pp. 26–34.

Lander,E.S. Linton,L.M., Birren,B., Nusbaum,C., Zody,M.C., Baldwin,J., Devon,K., Dewar,K., Doyle,M., FitzHugh,W. *et al*. (2001) Initial sequencing and analysis of the human genome. *Nature* **409**, 860–921.

Lewis,S.E., Searle,S.M., Harris,N., Gileson,M., Lyer,V., Richter,J., Wiel,C., Bayraktaroglir,L., Birney,E., Gosley,M.A. *et al*. (2002) Apollo: a sequence annotation editor. *Genome Biol.*, RESEARCH0082-2.

Loraine,A.E. and Helt,G.A. (2002) Visualizing the genome: techniques for presenting human genome data and annotation. *BMC Bioinformatics*, **3**, 1–8.

Pocock,M., Down,T. and Hubbard,T. (2000) BioJava: open source components for bioinformatics. A*CM SIGBIO Newsletter*, **20**, 10–12.

Pook,S., Vaysseix,G. and Barillot,E. (1998) Zomit: biological data visualization and browsing. *Bioinformatics*, **14**, 807–814.

Rzhetsky,A., Kalachikov,S., Ye,X.L., Zhang,P. and Russo,J.J. (1998) Tools for visualization and integration of intermediate sequencing results in large disease gene discovery projects. *Gene*, **208**, 31–35.

Siepel,A., Farmer,A., Tolopko,A., Zhuang,M., Mendes,P., Beavis,W. and Sobral,B. (2001) ISYS: a decentralized, component-based approach to the integration of heterogeneous bioinformatics resources. *Bioinformatics*, **17**, 83–94.

Stajich,J.E., Block,D., Boulez,K., Brenner,S.E., Chervitz,S.A., Dagdigian,C., Fuellen,G., Gilbert,J.G., Korf,I., Lapp,H. *et al*. (2002) The BioPerl toolkit: Perl modules for the life sciences. *Genome Res.*, **12**, 1611–1618.

Stein,D.L., Mungall,C., Shu,S.Q. *et al*. (2002) The Generic Genome Browser: a building block for a model organism system database. *Genome Res.*, **12**, 1599–1610.

Sun,H., Palaniswamy,S.K. and Davuluri,R.V. (2003) MPromDb: Mammalian Promoter Database – an information resource

of mammalian gene regulatory regions with annotation of CpG-islands and experimentally known cis-regulatory elements. (Submitted).

Waterston,R.H., Lindblad-Tok,K., Birney,E., Rogers,J., Ahril,J.F., Agarwal,P., Agarwala,R., Ainscough,R., Alexandersson,M., An,P. *et al*. (2002) Initial sequencing and comparative analysis of the mouse genome. *Nature*, **420**, 520–562.

Wheeler,D.L., Church,D.M., Lash,A.E., Leipe,D.D., Madden,T.L., Pontius,J.U., Schuler,G.D., Schriml,L.M., Tatusova,T.A., Wagner,L. and Rapp,B.A. (2002) Database resources of the National Center for Biotechnology Information: 2002 update. *Nucleic Acids Res.*, **30**, 13–16.

Wilkinson,M.D., Block,D. and Grosby,W.L. (2002) Genquire: genome annotation browser/editor. *Bioinformatics*, **18**, 1398–1399.