# BioViews: Java-Based Tools for Genomic Data Visualization

**Gregg A. Helt, Suzanna Lewis, Ann E. Loraine, and Gerald M. Rubin**[1]

Berkeley *Drosophila* Genome Project (BDGP), Department of Molecular and Cell Biology, University of California Berkeley, Berkeley, California 94720-3200 USA

Visualization tools for bioinformatics ideally should provide universal access to the most current data in an interactive and intuitive graphical user interface. Since the introduction of Java, a language designed for distributed programming over the Web, the technology now exists to build a genomic data visualization tool that meets these requirements. Using Java we have developed a prototype genome browser applet (BioViews) that incorporates a three-level graphical view of genomic data: a physical map, an annotated sequence map, and a DNA sequence display. Annotated biological features are displayed on the physical and sequence-based maps, and the different views are interconnected. The applet is linked to several databases and can retrieve features and display hyperlinked textual data on selected features. In addition to browsing genomic data, different types of analyses can be performed interactively and the results of these analyses visualized alongside prior annotations. Our genome browser is built on top of extensible, reusable graphic components specifically designed for bioinformatics. Other groups can (and do) reuse this work in various ways. Genome centers can reuse large parts of the genome browser with minor modifications, bioinformatics groups working on sequence analysis can reuse components to build front ends for analysis programs, and biology laboratories can reuse components to publish results as dynamic Web documents.

Biology is an intensely visual science. Most biological knowledge does not exist in the form of mathematical equations (as in physics) or as algorithms (as in computer science), but rather is summarized visually as diagrams, illustrations, three-dimensional reconstructions, and other types of graphic formats. This is understandable in a historical context, considering the origins of the life sciences in descriptive anatomy, botany, and other disciplines, which describe and categorize the natural world. And although molecular biology, the newest biological discipline, is largely concerned with the manipulation of materials that cannot be seen with the naked eye, much of molecular biology involves a type of descriptive anatomy in which the scale has changed dramatically, to the level of cells and molecules.

For the most part, biological processes are also summarized visually. Indeed, one of the most important overall results of computational biology may be the realization that biological systems are so complex that there are almost always significant exceptions to any rule-based algorithmic explanation of ''isolated'' biological phenomena. A calculus of biology will almost certainly have to take into account whole systems, and therefore, is not likely to appear any time soon. Thus for the forseeable future, visual descriptions, rather than equations or algorithms, will likely remain the most powerful method of organizing much of our biological knowledge. Given the prominence of visualization in biology, it is surprising that scientific visualization, the computer-aided display of scientific data, is a somewhat neglected area of research in bioinformatics.

The field of genomics, particularly, is in need of data visualization methods. Genomic DNA sequence data are emerging at an ever-increasing rate. This oncoming deluge of data threatens to overwhelm current visualization systems for providing public access to relevant information that links the DNA sequence to features of biological interest. Finding solutions to the problem of visualizing these data is the prime motivation for the work described in this paper.

Until now there have been three main choices for genome data distribution and presentation. One is the ''distribute everything'' approach. The database software, the user interface, and the data itself are bundled as a complete, integrated system and then delivered to the user, usually in a multimedia

[1]Corresponding author.
E-MAIL gerry@fruitfly.berkeley.edu; FAX (510) 643-9947.

CD format. Well-known examples of this method include the Entrez browser from NCBI (Schuler et al. 1996) and the ACeDB system (Dunham et al. 1994). The primary advantage to the bundled software approach is that it can include an intuitive and interactive user interface. Because there is no transmission of data across a network, the interface is constrained only by the skills of the original programmer. But this approach suffers from the problem of rapid obsolescence. The data package is essentially a snapshot of the data as it exists at the time of distribution, and thus is rendered obsolete almost immediately. Users can update their data periodically over the Internet, but this process is time consuming and error prone, and in practice few users of such systems do this on a regular basis. Another drawback is that these software and data packages require installation on local computer systems and therefore, are not universally accessible.

At the other end of the spectrum is the ''distribute nothing'' approach. Data are delivered only when requested, via an Internet connection such as FTP or gopher. More sophisticated versions of this approach deliver HTML-formated data to a Web browser, which essentially acts as a generic client. WWW-Entrez (Schuler et al. 1996) and the Moulon ACeDB server (G. Decoux, unpubl., http://probe.nalusda.gov/aboutmoulon.html) are good examples of this approach. The main benefits of this approach are that it presents a current view of the data and offers ubiquitous access, as most biologists have access to the World Wide Web. The main disadvantage is that the interface between the biologist and the data is limited to the possibilities offered by HTML and GIF images, with only rudimentary interactivity by way of hyperlinks and image maps. Therefore, these interfaces tend to be much less intuitive and interactive than the local applications they are meant to replace. Helper applications that can be called from the browser can alleviate these problems (e.g., three-dimensional structure viewers), but these helper applications require downloading, installation, and reconfiguration of the Web browser, and therefore, undermine the main advantage of this approach, which is universal access.

The middle ground between these two approaches exploits the classic client–server model. Data are delivered to the user only when requested but are viewed using client software that is distributed to the user. Net Entrez (Schuler et al. 1996) and XGRAIL (Uberbacher and Mural 1991) are examples of this approach. The advantages of this approach are that it can provide a current view of the data, combined with the potential for intuitive, interactive displays using software at the client site. Although the data are transmitted over a network, once downloaded the data can be viewed interactively and analyzed through the local user interface. The disadvantage to this approach is that it cannot be transparently ubiquitous, because to access the data the user must first download and install the client software.

The bioinformatics group at the Berkeley *Drosophila* Genome Project (BDGP) initially took the distribute everything route, releasing in collaboration with FlyBase the Encyclopaedia of *Drosophila* (EofD) CD-ROM, which contained a highly modified form of ACeDB to store and view the data (FlyBase 1995; C. Harmon, S. Lewis, and G.M. Rubin, unpubl.). Realizing that the CD-ROM format limits the availability of new data, more recently the BDGP has also experimented with a distribute nothing approach, implementing EofD-WWW, a Web-based interface for the Encyclopaedia of *Drosophila* available at the BDGP Web site (C. Harmon, S. Lewis, and G.M. Rubin, unpubl.). Although allowing ubiquitous access, displays that are interactive on the CD-ROM become exercises in frustration over the Web as the user clicks to move onto a graphics display and waits for the server to create a new GIF and send it to the Web browser running on the client machine.

However, recent technological developments have presented a novel solution to the problem of distributing genome sequence data to the larger community of biologists. A new class of languages have been developed that are designed specifically for distributed, interactive programming over the World Wide Web. Java is currently the most prominent of these network programming languages (Gosling and Arnold 1996). In this paper we describe an intuitive, platform-independent, Java-based Genome browser prototype (BioViews) developed for the BDGP. This Genome browser allows biologists to survey and analyze genome sequence information via the Internet. The BioViews visualization software runs as an applet in any Java-enabled Web browser and thus serves as an interactive front end visitors can use to browse the information stored in BDGP databases.

The BioViews Genome browser is built on top of software components built specifically for bioinformatics visualization and designed to allow for their reuse by other bioinformatics researchers. The BioViews components, or widgets, feature a well-defined application programming interface (API) that allows other programmers to reuse the components in their own applications without needing to know the details of how they are implemented. The BioViews widgets also incorporate concepts from the field of data visualization, such as semantic zooming, a concept that is particularly important in genomics, in which users need to view the same data at different levels of detail. This paper provides an overview of the BioViews *Drosophila* Genome browser, illustrates a number of data visualization concepts with examples from BioViews, and gives an overview of our component-based approach.

## RESULTS

### BioViews Genome Browser Design—A Description

BioViews combines a component-based approach with the power of implementing in Java, the most prominent of the new network languages. Java allows users to run complete applications called applets using Web browsers such as Netscape's Navigator or Microsoft's Internet Explorer. In the following section we describe briefly the appearance and behavior of the Genome browser to provide some context for the subsequent design discussion. For a better appreciation of the interface than can be given in prose, the BDGP Genome browser applet can be accessed over the Internet (http://fruitfly.berkeley.edu) via any Java-capable Web browser.

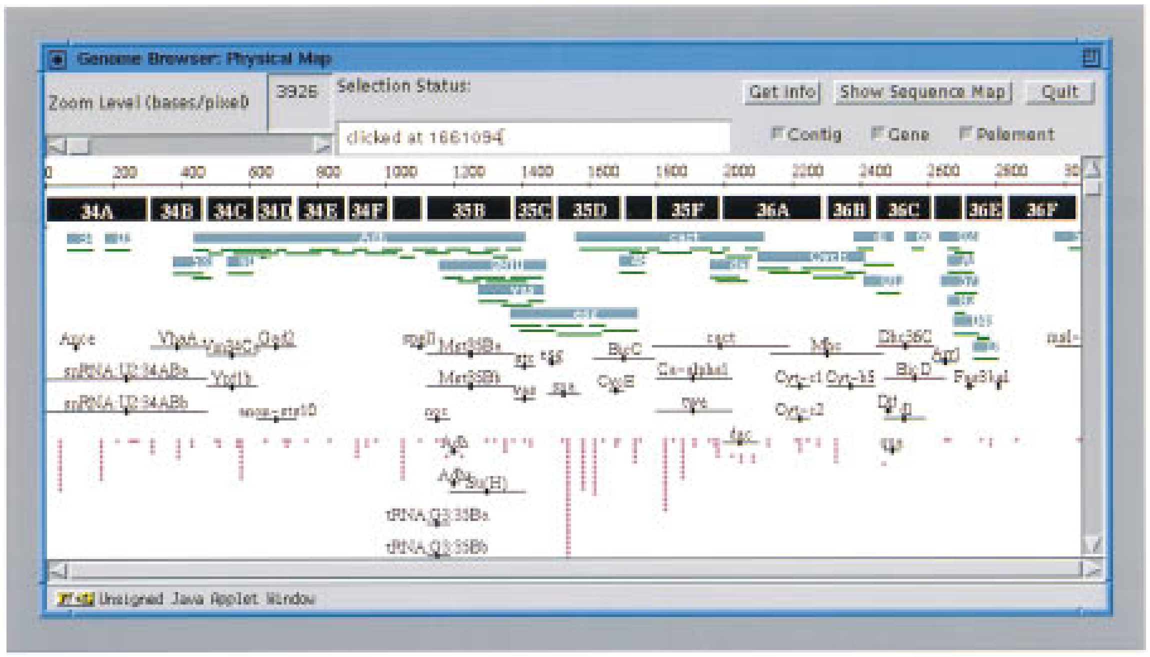
When the Genome browser prototype is first accessed, it creates a top-level window showing the physical map of a 3-Mb region (34A–36F) of *Drosophila melanogaster* genomic DNA (Fig. 1). This region is being used as a test bed for both computational and biological genome-scale analyses at the BDGP. Although the *Drosophila* Genome browser is accessed through a Web browser, the applet appears in a window outside the Web browser itself. Because Java applets have the same capabilities as standard applications in other languages, they can open windows on the user's virtual desktop. The ability to display data in a window outside the browser window has proven very important to biologists at the BDGP who have used BioViews, as the ideal layout for visualization of genomic information seldom matches the document-oriented, page-like format of the standard Web browser window.

The BioViews Genome browser displays two types of maps: a physical map and a sequence-based map. Each type of map contains information showing relative locations of P-element insertions, genes, and other important features, each of which is represented as graphical objects we have termed glyphs. The image in Figure 1 is a ''screen shot'' from the Physical Map viewer showing a mapping of bacteriophage P1 genome DNA clones onto the *Drosophila* cytogenetic map. This mapping unites the P1-based map with the classic cytological map of the *Drosophila* genome (LeFevre 1976). The classic map is based on the pattern of light and dark bands observable in stained preparations of polytene chromosomes from *Drosophila* larvae. This pattern is visible in the light microscope and has been used since the 1930s to map genes and chromsome aberrations in *Drosophila*. The P1 clones were assigned to positions on the cytogenetic map by in situ hybridization to polytene chromosomes (Hartl et al. 1994; BDGP, unpubl.), and positions and sizes of P1 clones relative to each other were assigned by STS content mapping (Kimmerly et al. 1996). Chromosomal bands themselves range from 5 to 200 kb in length, and based on the relative amount of DNA in each band and estimates of the overall *Drosophila* genome size, lengths (kilobase of sequence) have been assigned to each band (Sorsa 1988). The Physical Map viewer shows at the top a scale based on these estimates, which add up to ~3 Mb over the bands 34A1–36F11. At the lowest resolution view (Fig. 1A), chromosomal divisions, each comprised of several bands, are shown. At higher levels of resolution, individual bands appear (Fig. 1B).

The relative positions for each annotation shown on the map are delivered to the applet by the server. However, the BioViews map component is responsible for calculating where to place the features perpendicular to the scale so as to avoid overlap with other features. The glyphs (individual graphic elements) representing contigs, P1 clones, genes, and P-element insertion lines are all selectable and highlighted when selected. Whole regions can also be selected by clicking and dragging a rectangle around the region (''rubberbanding''). Moreover, selection of some features may also highlight others. For example, selecting a contig also highlights all the P1 clones that are contained in that contig. This can make sense in situations where one graphic object is in some way related to another in a whole-part relationship (i.e., the relationship between contigs and their constituent P1 clones). This illustrates an important design feature of the BioViews Genome browser. Graphical objects may have a type of parent–child relationship to other graphic objects that mirrors the relationship of the represented annotations to each other.

Selecting sequenced P1 clones and pressing the Show Sequence Map button invokes the Genome browser's second tier display, a higher resolution sequence -based map viewer that displays annotations

A



B

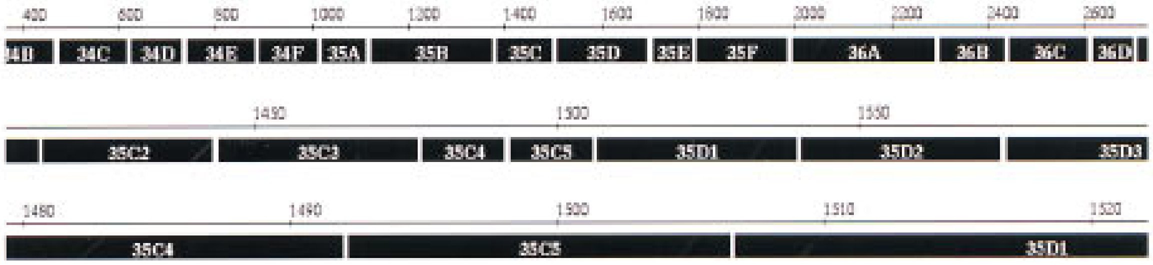


Figure 1 Physical and genetic map viewer. The physical map of a 3-Mb region of *Drosophila* genomic DNA is displayed within the physical map widget of the Genome Browser. DNA sequence length (kb) is shown at the top of the display. Chromosomal divisions and chromosome bands are shown underneath as labeled boxes. Below the chromosomal bands, the P1 contigs that map to the region are displayed as labeled blue bars, and the P1 clones that make up each contig are shown as smaller green bars underneath their respective contigs. Sequenced P1s are shown in light green. The Physical Map viewer calculates where to place contigs on the map by using the cytogenetic map position of their constituent P1 clones. The length of the contig is estimated from the number of P1s in the contig and the average P1 size, and individual P1s are drawn with a length proportional to the number of STSs contained within the P1 clone. In the current display, only the minimal tiling path of P1s per contig is shown as all the P1 clones for each contig would add much more data to the display but contribute little to the information conveyed. In addition to the contigs, all the genes that have been mapped genetically to the region are displayed as labeled spans, based on their genetic mapping to a range of chromosome bands; these data were obtained from FlyBase (Flybase 1995). Lethal P-element transposon insertions, which have been mapped cytogenetically to this region, are also shown as pink squares at the bottom of the display (Spradling et al. 1995). From this high level view it is immediately apparent that the distribution of mapped P-element insertions is very uneven, which is a recognized biological phenomena where some genomic regions have proven to be ''hot spots'' for P-element insertion. (*A*) Low-resolution view. The user has invoked the Physical Map viewer, with the resolution of the display set to the lowest level of detail. At this level of zoom, chromosomal divisions, each comprised of several bands, are shown as labeled black rectangles just below the DNA sequence length scale. (*B*) Higher-resolution views. Here just the Physical Map viewer chromosomal band display is shown. The user has used the zoom slider at the top left of the display to show successively higher resolution views of the region shown in *A*. Note that the chromosome divisions (labeled 35C, etc.) shown in *A* have resolved into their constituent bands (labeled 35C5, etc.) and that finer scale markings have appeared in the kilobase scale.

retrieved from the BDGP database, as shown in Figure 2. Annotations shown on the sequence-based map represent results obtained from computational analyses and database similarity searches.

One issue in the design of the Genome browser was whether the physical and sequence-based maps should appear in separate displays, as both present similar annotations. We decided that the two maps should remain separate, because they operate under different coordinate systems. That is, the physical map represents approximate sizes of features placed along a rough coordinate system derived from uncertain sizing of chromosome bands. On the other hand, the sequence-based map has a very definite coordinate system. The base pairs of the actual sequence, and most of the annotations, map precisely to absolute locations on the sequence. Eventually when the whole sequence of a large physical region is known, it will be possible to recalculate the physical map scale based on the more accurate sequence coordinate system. But until larger contiguous regions are entirely sequenced, it makes more sense to place the two map types in separate displays.

## DNA View

The user can inspect the nucleotide sequence of features selected in the sequence map viewer by requesting the third level of the Genome Browser, a DNA sequence viewer that displays the DNA sequence of features selected in the annotated sequence map (Fig. 3). The DNA view window shows the DNA bases as their letter codes tiled in rows across and down the window. If the mouse is moved

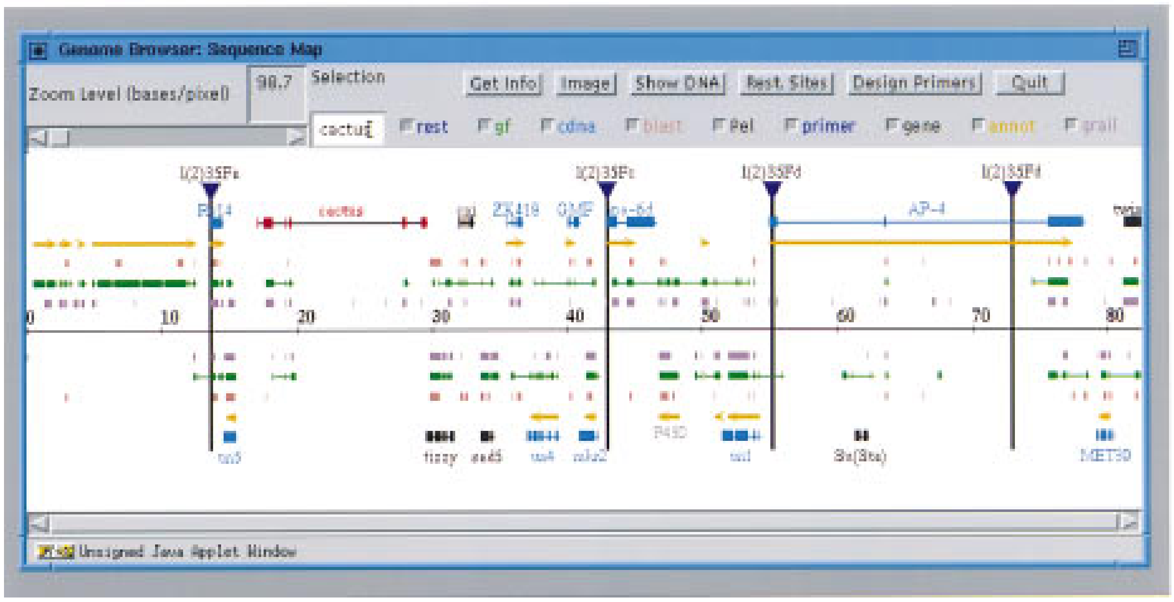
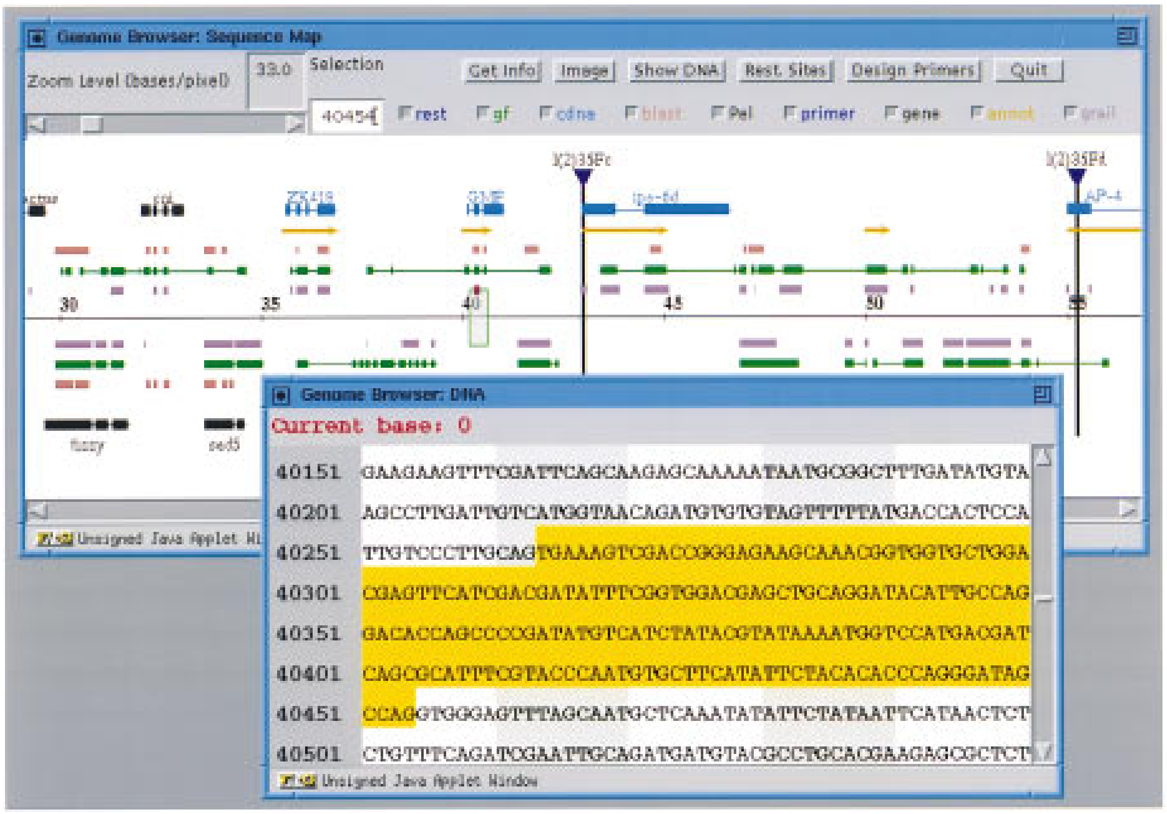


Figure 2 Annotated sequence map. The sequence map widget is shown above at the lowest level of resolution (i.e., the zoom control is set all the way to the left). The P1 clone displayed here is a gene-dense 83-kb clone (GenBank accession no. L49408) that the BDGP is using as a testbed for computational and experimental sequence analysis methods. The *cactus* locus has been selected. The sequence map displays the length of the DNA shown in a kilobase scale along the horizontal axis in the center of the display. Many of the features have directionality (i.e., they are present on either the forward or reverse strand). Therefore, features on the forward strand are displayed above the axis, and features on the reverse strand are shown below the axis. Vertical lines represent P-element insertions that inactivate essential genes and that have been mapped to the exact nucleotide of the insertion are also shown (Spradling et al. 1995). GenBank entries for known genes in the region are shown in black labeled with the gene name. The intron/exon structure of known genes are represented as black boxes (exons) connected by black lines (introns). Similarly, gene structures determined by comparison with cDNAs sequenced by the BDGP are shown in blue. Putative exons identified by gene prediction programs are shown as purple boxes (*Drosophila* GRAIL) or as green boxes connected by lines representing introns (Genefinder). The results from TBLASTX homology searches are shown in maroon. Gold arrows depict annotations added by human curation; these correspond to previously uncharacterized genes predicted by biological data and computational analyses. The vast majority of these predictions were later confirmed by isolation of corresponding cDNAs from a *Drosophila* cDNA library (L. Hong, D. Harvey, and G.M. Rubin, unpubl.).



```
Current base: 0
40151  GAAGAAGTTTCGATTCAGCAAGAGCAAAAATAATGCGGCTTTGATATGTA
40201  AGCCTTGATTGTCATGGTAACAGATGTGTGTAGTTTTTATGACCACTCCA
40251  TTGTCCCTTGCAGTGAAAGTCGACCGGGAGAAGCAAACGGTGGTGCTGGA
40301  CGAGTTCATCGACGATATTTCGGTGGACGAGCTGCAGGATACATTGCCAG
40351  GACACCAGCCCCGATATGTCATCTATACGTATAAAATGGTCCATGACGAT
40401  CAGCGCATTTCGTACCCAATGTGCTTCATATTCTACACACCCAGGGATAG
40451  CCAGGTGGGAGTTTAGCAATGCTCAAATATATTCTATAATTCATAACTCT
40501  CTGTTTCAGATCGAATTGCAGATGATGTACGCCTGCACGAAGAGCGCTCT
```

Figure 3 DNA view. The user has increased the level of zoom in the sequence map display so that a 30-kb, rather than 80-kb, region of the P1 is shown. The user has then selected a short region over the 40-kb mark on the central horizontal axis of the sequence map display, and has pressed the Show DNA button at the top of the browser window. Both the sequence data and the code for the sequence display viewer are dynamically loaded only as needed. Once the DNA view appears, a ''shadow'' representing its span along the sequence map appears as a gray box outlined in green on the horizontal axis. The user has selected an exon (predicted by *Drosophila* GRAIL) within the sequence map, resulting in this feature being highlighted in red in the sequence view and in yellow in the DNA view. The DNA view can be resized, and the DNA viewer deals with resizing intelligently by ensuring that rows are always displayed as multiples of ten. To further provide positional cues, two slightly different background colors are used for adjacent 10-base columns.

over the DNA view, the numbered position of the base under the cursor is reported in red in the top left corner. Stretches of DNA can be selected and highlighted in the manner typical of word processors, by clicking and dragging through the bases.

When the sequence map is brought up by selecting a P1 in the physical map, the annotated features of the P1 are loaded from the server, but the actual sequence DNA is not. Loading the sequence data is delayed because the annotated map is often sufficient, and there is no need to see the actual DNA sequence. This spares users forced to endure a slow connection from downloading unneeded data. If detailed view of the DNA is desired, then the DNA sequence can be retrieved from the server by pushing the Get DNA button. The DNA sequence is then sent to the client machine, and a new type of window is opened (i.e., the DNA View shown in Figure 3).

## Data Visualization Principles and Techniques Used in BioViews

### *Responsiveness*

One important aspect of any graphical user interface (GUI) is how interactive and responsive the interface feels to users. This includes not only the types of interactions between users and the software, but also the perceived response time to those interactions. The genome browser supports typical interactions such as feature selection and scrolling of the display. Most features are selectable and respond by becoming highlighted. Whole regions can also be selected by rubberbanding.

We have tried to ensure that the Genome browser responds to user input in as close to real-time as possible. For example, not only can the user scroll the display along the axis of the physical map and sequence map, but the resolution of the map can also be changed in real-time with another scroll bar; therefore, the user can zoom in and out of the maps interactively. This ability to do real-time interactive scaling may not seem critical at first, but it becomes very important when biologists are trying to understand biological features and phenomena that need to be evaluated at widely different levels of detail. For example, mutations may be caused by single base pair changes or by megabase-scale deletions. In the case of a large deletion, the effects of such a change would be most easily evaluated using the physical map or sequence-based map viewer. The consequences of smaller mutations, however, would be best viewed using the browser's highest resolution view, the DNA sequence display. By providing a scalable view of the *Drosophila* genome, the Browser will make it easier to conceptualize and predict the gross effects of different kinds of mutations.

### *User Customization*

Another important consideration in GUI design has to do with how easily individual users can customize the interface to suit their particular purpose and preferences. As a first step toward enhancing the user customization capability of BioViews, different annotation classes (database similarities, P-element insertions, etc.) can be toggled on and off as groups. This feature allows users to focus on annotations they consider important without being distracted by information irrelevant to their particular needs. Although this type of option is useful now, it will become essential in the future. We anticipate adding results from a number of other types of analyses to the underlying database, and viewing all of these at once would be overwhelming.

### *Graphical Hyperlinks*

Since the advent of the Web, hyperlinks have become the most important and popular cross-window navigation tool. For HTML-based systems, these take the form of textual or image hyperlinks to URLs, or in more sophisticated systems, mapping of specific areas of images to different URLs. We believe it is critical to maintain this capacity for arbitrary hyperlinks in graphics-based data navigation as well. In the genome browser we have incorporated this idea by giving each map feature loaded from the server a unique tag that can be used to retrieve associated information from the server. This tag allows, for example, the selection of a cDNA to retrieve an expression pattern image (Fig. 4A), or the selection of a gene in the sequence map to retrieve an HTML-formated GenBank record (Fig. 4B). This graphical hyperlinking is currently very similar to HTML hyperlinks in that the tag is used to build a URL, from which the text or image is loaded.

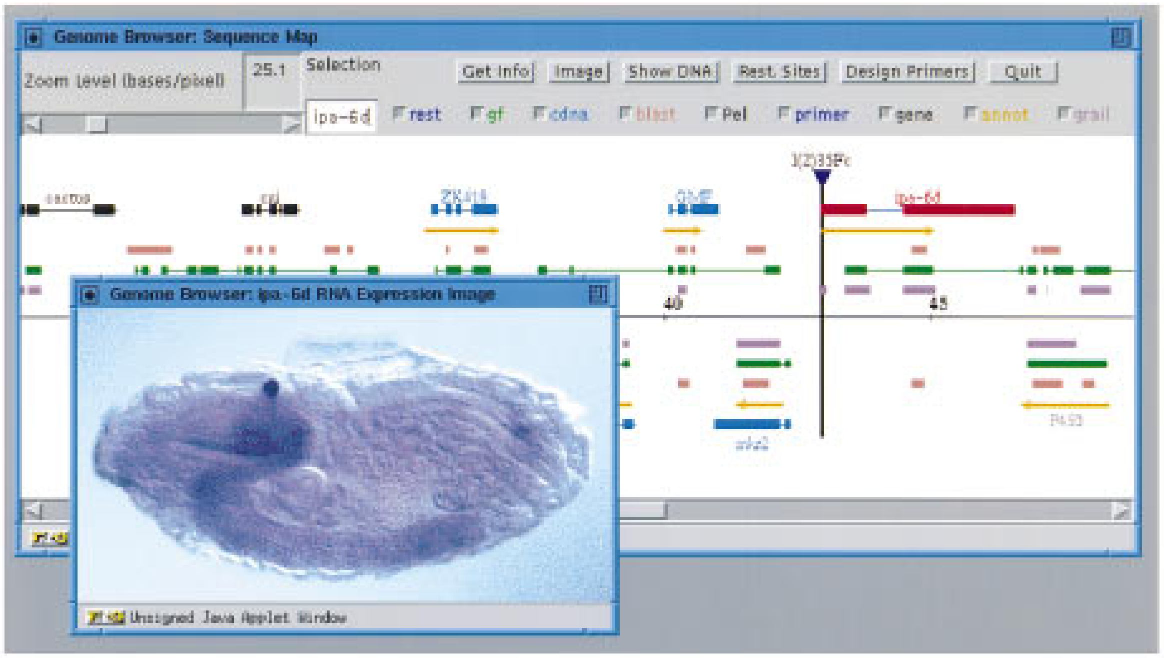
### *Maintaining Context*

Another data visualization principle is that it is important to maintain visual context when the user is examining a particular feature. To illustrate this, consider the physical map. Because of the way the features are laid out perpendicular to the horizontal map axis so as to avoid overlap, some of these features may be placed in the virtual space below the viewable window. Therefore, in addition to being able to scroll the view along the horizontal axis of the physical map, the display can be scrolled vertically to bring such features into view. However, rather than scroll the entire display, only the features (contigs, P1 clones, genes, and P-element insertions) scroll vertically—the map axis and the chromosome bands remain in place. In this way, more features can be processed visually, but their positions relative to the display axis, that is, their estimated position and cytological localization, remain in view.

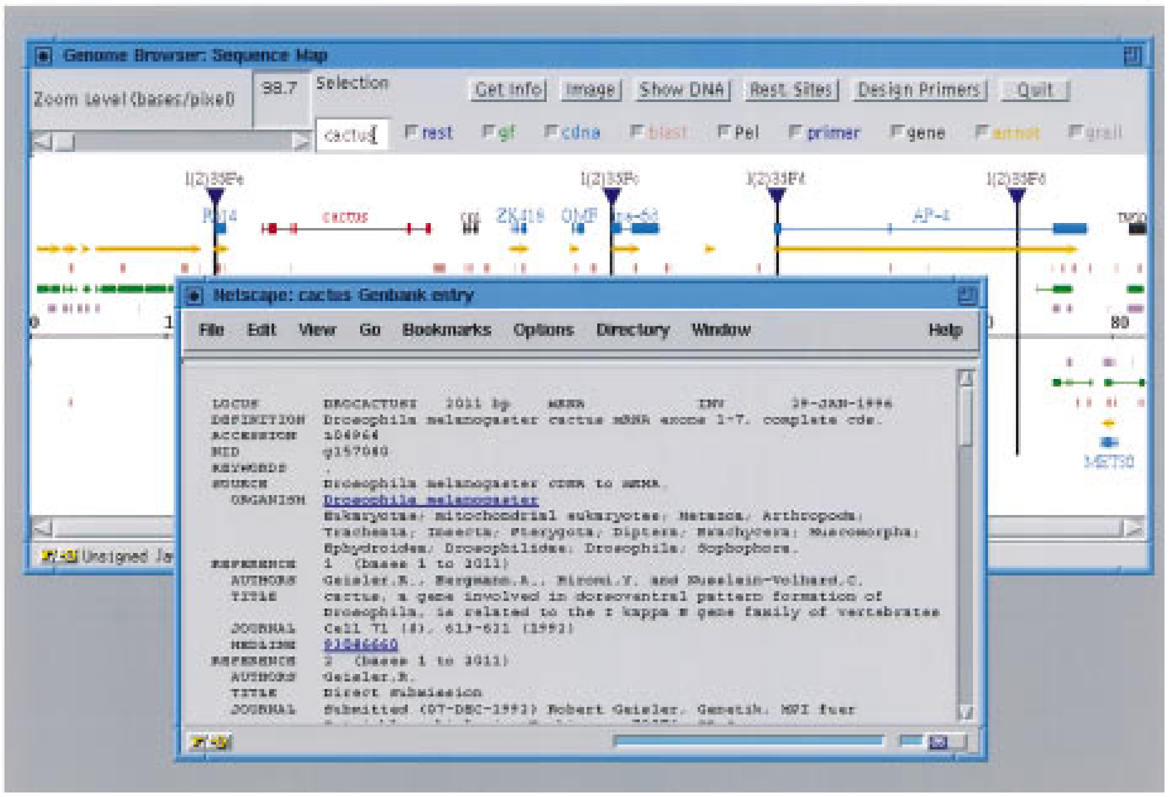
Another example of maintaining context is the relationship between the sequence map view and the DNA base view. One issue is why these should be considered as two views in the first place. As mentioned earlier, there is a separation between the physical map and the sequence map because they are based on incompatible coordinate systems. However, the DNA view and the sequence map use exactly the same coordinate system (i.e., the base pair positions relative to the start of the sequence) and therefore, to inspect the DNA sequence, the user could simply continue zooming into the sequence map until individual base pairs become visible as letters. The fundamental reason for adding a third level to the browser is to permit inspection of the sequence at the highest level of detail while still maintaining the sense of context that the lower resolution Sequence Map viewer provides.

An alternative solution would have been to create two or more horizontal sequence maps, each showing a range of resolutions. But the length of sequence visible at the resolution at which bases can

A



B



```
LOCUS       DROCACTUSI   2011 bp    mRNA            INV       29-JAN-1996
DEFINITION  Drosophila melanogaster cactus mRNA exons 1-7, complete cds.
ACCESSION   L04964
NID         g157040
KEYWORDS    .
SOURCE      Drosophila melanogaster cDNA to mRNA.
  ORGANISM  Drosophila melanogaster
            Eukaryotae; mitochondrial eukaryotes; Metazoa; Arthropoda;
            Tracheata; Insecta; Pterygota; Diptera; Brachycera; Muscomorpha;
            Ephydroidea; Drosophilidae; Drosophila; Sophophora.
REFERENCE   1  (bases 1 to 2011)
  AUTHORS   Geisler,R., Bergmann,A., Hiromi,Y. and Nusslein-Volhard,C.
  TITLE     cactus, a gene involved in dorsoventral pattern formation of
            Drosophila, is related to the I kappa B gene family of vertebrates
  JOURNAL   Cell 71 (4), 613-621 (1992)
  MEDLINE   93046660
REFERENCE   2  (bases 1 to 2011)
  AUTHORS   Geisler,R.
  TITLE     Direct Submission
  JOURNAL   Submitted (07-DEC-1992) Robert Geisler, Genetik, MPI fuer
```

Figure 4 Hyperlinks. (*A*) In situ hybridization pattern of *Ipa-6d* in an early *Drosophila* embryo. The user has selected gene *Ipa-6d* in the Sequence Map viewer and has pressed the Image button at the top of the browser window. This selection invokes a window showing the embryonic expression pattern of the *Ipa-6d* gene as determined by RNA in situ hybridization. (*B*) GenBank entry for *cactus* accessed through the Sequence browser. The user has selected the *cactus* locus, and has pressed the Get Info button at the top of the Sequence Map viewer display. The *cactus* GenBank record with HTML hyperlinks is returned in a new separate browser window (the lower box). The lower window has the full capabilities of typical browser windows in that the user is able to follow the hyperlinks shown in blue.

be represented as individual letters on the sequence map is relatively low, typically ~100 bases (1000 pixels/10 pixels per base). On the other hand, the DNA view can show many more bases because the sequence wraps around to fit the window. However, large numbers of features are hard to visualize with this type of display, because the vertical axis is then being used both to separate features and to wrap the sequence. Therefore, the DNA view does not show all the features displayed in the sequence map, but instead highlights the bases covered by the currently selected feature in the sequence map.

To indicate the relationship between the sequence map and the DNA view, a ''shadow'' of the DNA view is shown along the axis of the sequence map. The shadow appears as a transparent gray box outlined in green (see Fig. 3). As the DNA view is scrolled, the shadow also moves along the sequence map. Also, double-clicking along the axis of the sequence map moves the shadow to the clicked location and scrolls the DNA view to the appropriate position. Figure 3 also shows how selecting a feature in the sequence map highlights the spanned bases in the DNA view. An exon predicted by *Drosophila* GRAIL has been selected, the DNA view allows one to see that the flanking intron borders contain the expected 5′ AG and 3′ GT splice site consensus sequences. This shadowing of one view within another represents a simple form of the data visualization technique known as data lenses (Bier et al. 1993). The general principle of visually maintaining context around the area of interest has also been described as focus + context (Robinson and Flores 1997).

### *Semantic Zooming*

The Genome browser relies heavily on ''semantic zooming,'' a concept from the field of data visualization that refers to how the graphical representation of data can change in nonscalar and nongeometric ways as the display resolution changes (Bederson and Hollan 1994). As a specialization of this concept, the depth of data that is shown can depend on the scale of the view. Semantic zooming is a powerful concept when considering the structure of the data presented by genome databases. Users need to examine the structure of a genomic region at many different levels of resolution. At the lowest level of resolution, the map widget can display the order and length of genes as they appear in a large (megabase size) section of genomic DNA. At the highest level of resolution, the user can inspect the nucleotide sequence of individual genes.

An example of semantic zooming in the Genome browser is in visualizing the chromosomal banding pattern in the physical map viewer. Displaying all the smallest bands in a 3-Mb region would result in a very dense and confusing display. Instead, at low resolution only the major chromosome divisions are shown. As the resolution increases, the chromosomal divisions disappear and are replaced by their constituent bands. The display of band labels is also affected by the resolution, therefore labels are only displayed if they can fit within the band (see Fig. 1B). A similar form of semantic zooming is used for the map axis, with the continual appearance or removal of finer tick marks and labels on the axis as the resolution of the map is increased or decreased.

Other forms of semantic zooming in the maps are more subtle. For example, most features will always be displayed as at least a pixel-wide span, even if the coordinate span of a feature at the map's displayed resolution corresponds to a screen representation of less than a pixel. Without this type of semantic zooming, at low resolution smaller features would simply disappear. Most of the features scale geometrically along the axis, but the representation of P elements, which are single point insertions, always remain the same size. Also, although most of the features themselves expand geometrically along the axis, their labels stay the same size. The labels also can ''float'' along their associated feature, to ensure that they stay visible as the view's resolution increases. If the labels did not have this ability to float to a new position in the display, they would be centered at the feature's center, which would often force the labels offscreen even when their corresponding feature is still visible in the window. Floating labels are also helpful when viewing sequences at high resolution, as some features will span a far larger region than is visible in the current view. Yet another type of semantic zooming is the appearance of the DNA base pairs along the axis when the sequence map is zoomed to the highest resolution.

### *Integration of Analysis Tools*

In addition to being able to browse genomic data using a graphical front end, it is becoming increasingly important for biologists to apply different kinds of computational analyses to the annotated genomic data. The BDGP Genome browser combines visualization and analysis tools in a single application, thereby enhancing the usefulness of both, as their results can be compared alongside previously determined annotations, revealing relationships that would otherwise be difficult to discern.

The first such tools we have added are for restriction site mapping (Fig. 5B) and PCR primer–pair design (Fig. 5A). The restriction site analysis executes as Java code on the client, allowing instantaneous response as the user selects different restriction sites that are displayed as features along the axis of the sequence map. Unlike restriction mapping, the algorithms for primer design are executed on the server, the code on the client providing only the user interface. For complex computations, like primer design, keeping the code on the server has a number of advantages. Performance does not have to rely on the client's local processing power, the potentially large amount of code does not have to be downloaded to the client, and existing non-Java implementations can be used. For the *Drosophila* Genome browser, we used Primer 0.5 (Lincoln et al. 1991) within a Perl5 CGI wrapper on the server. Parameters including the selected sequence region are passed from the Java user interface to the server in the form of a CGI–URL. Results are returned through this URL and are parsed and displayed along the axis of the sequence map.

Both the analysis code for restriction mapping and the GUI code for primer selection are loaded dynamically. This means that the code is not loaded from the server until the user requests the analysis. Dynamic loading allows the number of analysis tools added to the browser to increase indefinitely without adding significantly to the initial download time for the applet, thus avoiding ''software bloat.'' Because most users will only use a few of the analysis tools, transfer time is not wasted loading unnecessary code.

## BioViews Components and Reuse

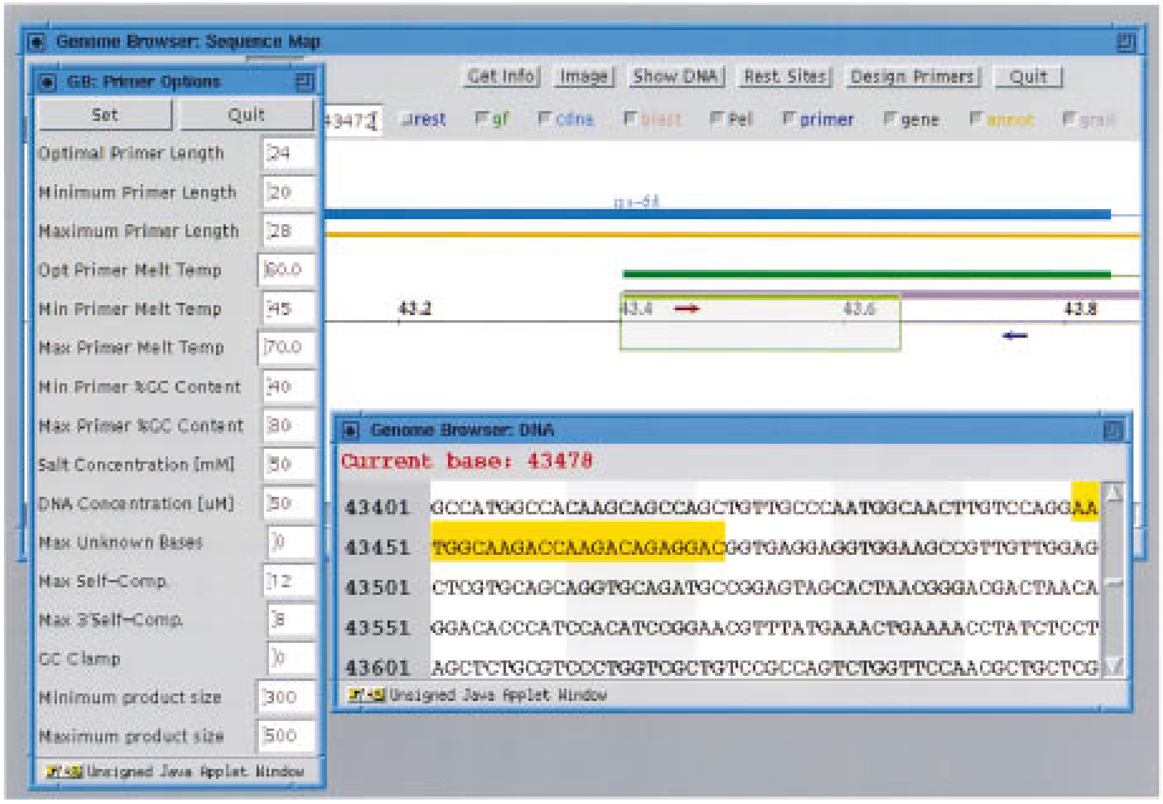
Many bioinformatics researchers and genome centers besides the BDGP face a similar need for data visualization tools. Therefore, one of our goals in designing the *Drosophila* Genome browser and its constituent BioViews components was to allow parts of the software to be reused as desired by others. In recent years this component approach has gained a number of advocates in the field of bioinformatics (Goodman et al. 1995).

The *Drosophila* Genome browser applet uses two main BioViews components, a map widget for displaying features along a linear map (which is used for both the physical map and annotated sequence views) and a DNA display widget for showing nucleotide sequences at base-pair resolution. At a high level of abstraction these BioViews components are conceptually similar to the map and sequence widgets of bioTk (Searls 1995) and bioTkPerl (Helt 1997).

Although there were previous efforts to create reusable tools for bioinformatics data visualization, the development of bioTk by David Searls and colleagues introduced two new concepts. First, the fundamental unit of reuse should be a high-level graphical component or widget, rather than a framework (as in the Genetic Data Environment; Smith et al. 1994), an application (as in AceDB; Dunham et al. 1994), or a graphics library (as in Vibrant; Schuler et al. 1996). Second, these widgets should be concerned only with visualizing the data, and that any semantic meaning represented by the graphics should be handled by the application using the widget, not the widget itself. BioTk was implemented using the Tcl language and Tcl's built-in Tk graphics architecture, and is composed of widgets designed to encapsulate functionality common to many high-level bioinformatics displays. The bioTk widget set includes a map widget that supports layout of graphical objects along a common coordinate system, a sequence widget that shows a scrolling window of sequence residues and a chromosome widget for displaying chromosomal banding patterns. These widgets are manipulated via an API composed of global Tcl procedures that are invoked from an application and passed the widget as an argument.

Figure 5 Analysis tools. (*A*) Primer selection in the Genome browser. The user has asked the browser to design PCR primers suitable for amplification of the region selected within the Sequence Map window. The results are shown above. In the Sequence Map window, sense and antisense primers are displayed as maroon and blue arrows, respectively. In the Sequence Display window (bottom right), the sequence for the sense primer recommended by the primer selection program is highlighted in yellow. (*B*) Restriction mapping in the Sequence browser. The user has pressed the Rest. Sites button at the top of the browser window. This action has activated a new browser window (shown at the left) that permits selection of restriction enzymes. The user has selected three: AclI, AhaIII, and EcoRI. Sites for these enzymes are shown in light blue, magenta, and green, respectively. Note that these sites in the original map window are marked along the linear map, and that they are also highlighted in yellow within the Show DNA window.

A

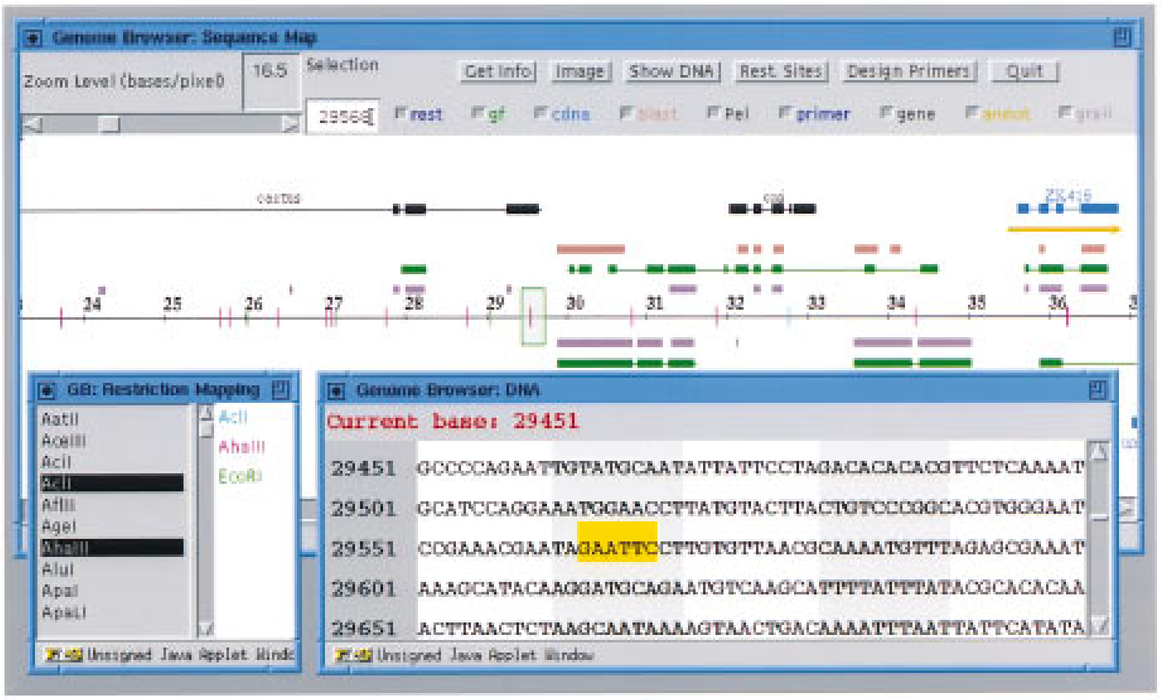


| | |
|---|---|
| Optimal Primer Length | 24 |
| Minimum Primer Length | 20 |
| Maximum Primer Length | 28 |
| Opt Primer Melt Temp | 60.0 |
| Min Primer Melt Temp | 45 |
| Max Primer Melt Temp | 70.0 |
| Min Primer %GC Content | 40 |
| Max Primer %GC Content | 80 |
| Salt Concentration [mM] | 50 |
| DNA Concentration [uM] | 50 |
| Max Unknown Bases | 0 |
| Max Self-Comp. | 12 |
| Max 3'Self-Comp. | 8 |
| GC Clamp | 0 |
| Minimum product size | 300 |
| Maximum product size | 500 |

```
Current base: 43478

43401  GCCATGGCCACAAGCAGCCAGCTGTTGCCCAATGGCAACTTGTCCAGGAA
43451  TGGCAAGACCAAGACAGAGGACGGTGAGGAGGTGGAAGCCGTTGTTGGAG
43501  CTCGTGCAGCAGGTGCAGATGCCGGAGTAGCACTAACGGGACGACTAACA
43551  GGACACCCATCCACATCGGAACGTTTATGAAACTGAAAACCTATCTCCT
43601  AGCTCTGCGTCCCTGGTCGCTGTCCGCCAGTCTGGTTCCAAGGCTGCTCG
```

B



```
AatII
AccIII
AciI
AclI
AflII
AgeI
AhaIII
AluI
ApaI
ApaLI
```

```
AclI
AhaIII
EcoRI
```

```
Current base: 29451

29451  GCCCCAGAATTGTATGCAATATTATTCCTAGACACACACGTTCTCAAAAT
29501  GCATCCAGGAAATGGAACCTTATGTACTTACTGTCCCGGCACGTGGGAAT
29551  CCGAAACGAATAGAATTCCTTGTGTTAACGCAAAATGTTTAGAGCGAAAT
29601  AAAGCATACAAGGATGCAGAATGTCAAGCATTTTATTTATACGCACACAA
29651  ACTTAACTCTAAGCAATAAAAGTAACTGACAAAATTTAATTATTCATATA
```

Figure 5 (*See facing page for legend*.)

Conceptually the BioViews components were heavily influenced by bioTk and bioTkPerl. Like the bioTk components, the principal unit of reuse is a graphical component, and these components are only concerned with visualization rather than semantics. The components are designed to be reused to represent biological data in terms of ''domain graphics'' (Felciano et al. 1997): graphical representations of knowledge that are widely accepted and understood within a specific scientific domain. Unlike bioTk, BioViews implements these concepts within an object-oriented system rather than through global procedures as is done in bioTk. This difference reflects the underlying difference between Java, which is object oriented, and Tcl/Tk, which is procedure based. Therefore, the BioViews widgets are implemented as classes that can be instantiated as objects, and the widgets are reused through an API composed of methods that can be called on these widget objects. For example, the map widget API includes methods for manipulating annotations along a linear map, including adding, removing, selecting, hiding, labeling, and positioning annotations, and specifiying the annotation's shape and color. The API also includes methods for manipulating the entire map, such as scaling and scrolling.

We envision three distinct levels of reuse for the BioViews software: the whole Genome browser, the BioViews components, and the underlying graphics architecture. Like the bioTk widgets, the BioViews widgets are intended to be reused by other bioinformatics researchers who may incorporate them into their own applets and applications. At a higher level of reuse, the whole Genome browser can be reused with minor modifications to visualize data from other genome projects, or any large sequencing project. In this case, the genome browser can be considered as a large display component for a distributed system. Finally, at a very low level, the graphics architecture underlying the map widget can be reused by component developers to create new types of widgets. To our knowledge, at least six other groups have built applications using the BioViews components; a summary of this work will be reported elsewhere.

As discussed previously, the primary advantages of implementing the ''componentry'' concept in Java rather than in another language are that Java programs are, in theory, platform independent, and because of the inclusion of a Java interpreter in the major Web browsers, a Web-based Java front end can allow nearly universal access to biological data. We have found that Java's promise of platform independence has been realized to a large extent, with some minor differences between the different implementations of the Java virtual machine on different platforms. One limitation of Java is that the Java graphics architecture is very primitive, and thus a large part of our efforts have centered on implementing support for graphics operations that more mature graphics systems, such as Tcl/Tk, already provide. Printing from applets has also been a problem, but the latest release of Java (JDK 1.1) includes support for printing from applets downloaded from the Web. Download times for large applets have also been a concern, but new applet compression formats have alleviated this problem, and ''push'' technology, such as Marimba's Castanet channels, offers intelligent local caching of applet code. The most significant limitation of using Java over the Web has been that browser-implemented security restrictions limit the ability of users to store and retrieve data locally running applets from the Web. However, this problem is addressed in more recent browsers that allow ''trusted'' applets (registered with a central authority) to read and write local data on client machines.

## DISCUSSION

### Relevance to *Drosophila* Research

Like most other genome centers, the BDGP has decided that the best way to manage initial data distribution is to publish the information on our local Web site. BioViews, our Java-based front end, is steadily being revised to accommodate new information and new needs in genomic research. The BDGP intends to use an updated version of BioViews to give the *Drosophila* research community access to our rapidly expanding store of *Drosophila* sequence data, and is committed to using future versions of BioViews and other Java-based bioinformatics components as the public front end to *Drosophila* genomic data. The BioViews software is available upon request.

### Relevance to Computational Biology

Computational biology is a thriving field, and the exponential growth rates of both computing power and available sequence data is driving creation of many new methods for computational analysis of molecular biology data. Unfortunately, few biologists will benefit from these computational innovations unless the results from these analyses are made available over the Internet in an intuitive display.

And even if the designers of an algorithm produce the necessary code for displaying their results, what most biologists will see are static images via the Web. Few biologists will go through the trouble of installing new software to view interactively the results of new analysis techniques. And few computational biologists will go through the trouble of becoming experts in distributed data visualization to make their results accessible. We believe that biologists and computer scientists should work together to test each other's predictions and results, and the Web provides the perfect medium for such collaborations. BioViews and similar reusable component-based software tools (for example, see Felciano et al. 1997) for distributed data visualization offer a solution to this dilemma. These tools will provide a mechanism that not only allows bioinformatics researchers to easily customize graphical views of their results for their own research, but will allow anyone else to view their results in an interactive fashion across the Web.

### Relevance to Genome Informatics

Most bioinformatics researchers at genome centers would choose to avoid duplication of effort if software solutions already existed to meet their needs. But problems that on the surface appear very similar often end up differing in details that necessitate slightly different solutions. The choices for the genome researcher are either to develop their own solutions from scratch or to adopt someone else's existing solution and live within its limitations (Goodman et al. 1995). The component software approach has been proposed as a solution to this barrier against software sharing. But with a few exceptions (Searls 1995), efforts to introduce components into bioinformatics have concentrated on standards for data models and data exchange formats. Unfortunately, there is already a long and contentious history of competing standards in these two areas, complicated by large amounts of legacy code. In contrast, the field of distributed data visualization is quite new, and it may still be possible to establish widely accepted standards. Toward this end the BDGP participates in the BioWidget Consortium, a group of developers working to define standards for Java-based components for bioinformatics data visualization. The BioWidget Web site can be found at: http://goodman.jax.org/projects/biowidgets/consortium.

### Relevance to the Biosciences

Obviously the larger community of biologists benefits from the introduction of universally accessible interactive genome browsers, as amply demonstrated by current HTML-based Web interfaces that already offer some interactivity. Distributed computing and the Internet also provide an opportunity to establish community access to a wealth of previously unavailable data by making the networked computer an attractive supplement, or even an alternative, to laboratory notebooks for recording routine annotations of experimental results.

We believe these first attempts by our group and others to create graphical front ends based on network languages are just the beginning of a move toward truly distributed bioinformatics. Combining component-based software design with Java's ability to dynamically load code, and to invoke methods on remote code, leads to a future where components from anywhere on the network interact as needed. Instead of having a single local graphical client connected to a single remote server, the graphical front end will be the visible portion of a collection of interacting local and remote components. Eventually, as the distinction between the network and the personal computer blurs, so too will the distinction between the researcher's personal data space and the public databases. We are developing prototypes to explore these future possibilities.

## METHODS

### Java

Java is a new language recently being promoted as a standard for creating interactive content on the Web (Gosling and Arnold 1996). A Java-capable Web browser runs a virtual machine, which in turn runs applets, small applications that are loaded from a server by being embedded in HTML pages. Java solves the problem of distributing code, for as long as the browser itself can run Java, no code needs to be distributed. Instead it is loaded on the fly from the server (and cached for that session by the browser). We chose Java over other network languages because of its wide availability.

### Annotations and Sequence Data

Sequence data were taken from DS02740, an ~83-kb P1 clone of *Drosophila* genomic DNA (GenBank accession no. L49408) that was sequenced at the BDGP. All data used in the figures were taken from the BDGP database, but may have originated elsewhere. For the physical map view, data for contigs and P1 clones is taken from the BDGP mapping project (Kimmerly et al. 1996). Chromosomal banding data was taken from Sorsa (1988). Genetic mapping of genes are from FlyBase (1995). P-element insertions were mapped precisely as part of a BDGP large-scale genome disruption project (Spradling et al. 1995; J. Rehm, A. Beaton, and G.M. Rubin, unpubl.). cDNAs were isolated from an embryonic cDNA library, sequenced, and analyzed for expression patterns at the BDGP (L. Hong, D. Harvey, and G.M. Rubin, unpubl.)

DNA sequence data was analyzed using homology searching and two different gene prediction systems optimized for use in analyzing *Drosophila* DNA. These were *Drosophila* GRAIL (Xu et al. 1995; Yu and Uberbacher 1996) and *Drosophila* Genefinder (Genefinder: P. Green and L. Hillier, unpubl., *Drosophila* tables: G. Helt, unpubl.) Genefinder parses multiple genes from large sequences and results from this analysis system are shown as gene models showing the intro–exon structure of predicted genes. GRAIL, however, parses exactly one gene from a given sequence. Therefore, we have incorporated the GRAIL results at the stage in the analysis just before parsing a gene model, when all potential exons are scored individually, and have shown these individual exons rather than the single predicted gene. A tblastx search (Altschul et al. 1994) of GenBank, in which the query sequence is translated in all six possible reading frames and then compared to a six-frame translation of every sequence in the nucleotide database, was used to search for homologies to previously reported sequences. BLASTX was used to check the sequence for similarities to an up-to-date, nonredundant protein database and to find known *Drosophila* genes in the region (Altschul et al. 1990). BLASTX results were filtered through HSPcrunch (Sonnhammer and Durbin 1994).

## ACKNOWLEDGMENTS

We are grateful to Michael Ashburner, Nomi Harris, Sima Misra, and other members of the BDGP for their critical comments on the manuscript. The RNA in situ image shown in Figure 4B was provided by T. Serrano. This work was supported by grant P50-HG00750 from the National Human Genome Research Institute. G.M.R. is an Investigator of the Howard Hughes Medical Institute.

The publication costs of this article were defrayed in part by payment of page charges. This article must therefore be hereby marked "advertisement" in accordance with 18 USC section 1734 solely to indicate this fact.

## REFERENCES

Altschul, S., W. Gish, W. Miller, E. Myers, and D. Lipman. 1990. Basic local alignment search tool. *J. Mol. Biol.* 215: 403–410.

Altschul, S.F., M.S. Boguski, W. Gish, and J.C. Wootton. 1994. Issues in searching molecular sequence databases. *Nature Genet.* 6: 119–129.

Bederson, B. and J. Hollan. 1994. Pad++: A zooming graphical interface for exploring alternate interface physics. *User Interface Software and Technology.* ACM Press, New York, NY.

Bier, E.A., M.C. Stone, K. Pier, W. Buxton, and T.D. DeRose. 1993. Toolglass and magic lenses: The see-through interface. In *Proceedings of SIGGRAPH '93, Computer Graphics Annual Conference Series,* pp. 73–80. ACM Press, New York, NY.

Dunham, I., R. Durbin, J. Thierry-Mieg, and D.R. Bentley. 1994. Physical mapping projects and ACeDB. In *Guide to human genome computing* (ed. M.J. Bishop), pp. 110–158. Harcourt Brace, New York, NY.

Felciano, R.M., R.O. Chen, and R.B. Altman. 1997. RNA secondary structure as a reusable interface to biological information resources. *Gene* 190: GC59–70.

LeFevre, G. 1976. A photographic representation and interpretation of the polytene chromosomes of *Drosophila melanogaster* salivary glands. In *The genetics and biology of* Drosophila (ed. M. Ashburner and E. Novitski), pp. 31–66. Vol. 1A. Harcourt Brace, New York, NY.

Flybase. 1995. Flybase—the *Drosophila* database. Available from the flybase.bio.indiana.edu network server and Gopher site and at the URL http://morgan.harvard.edu *Nucleic Acids Res.* 24: 53–56.

Goodman, N., S. Rozen, and L. Stein. 1995. The importance of standards and componentry in meeting the genome informatics challenges of the next five years. Second meeting on the interconnection of molecular biology databases, Cambridge, UK, http://www.ai.sri.com/people/pkarp/mimbd/95/abstracts/goodman.html.

Gosling, J. and K. Arnold. 1996. *The Java programming language.* Addison-Wesley, Reading, MA.

Hartl, D.L., D.I. Nurminsky, R.W. Jones, and E.R. Lozovskaya. 1994. Genome structure and evolution in *Drosophila:* Applications of the framework P1 map. *Proc. Natl. Acad. Sci.* 91: 6824–6829.

Helt, G.A. 1997. "*Data visualization and gene discovery in* Drosophila melanogaster." PhD. thesis. University of California at Berkeley, CA.

Kimmerly, W., K. Stultz, S. Lewis, V. Lustre, R. Romero, J. Benke, D. Sun, G. Shirley, C. Martin, and M. Palazzolo. 1996. A P1-based physical map of the *Drosophila* euchromatic genome. *Genome Res.* 6: 414–430.

Lincoln, S., M. Daly, and E. Lander. 1991. PRIMER 0.5. http://www.genome.wi.mit.edu/ftp/distribution/software/primer

Robinson, A.J. and T.P. Flores. 1997. Novel techniques for visualizing biological information. In *Conference on intelligent systems for molecular biology.* Halkidiki, Greece. pp. 241–249. AAAI Press, Menlo Park, CA.

Schuler, G., J. Epstein, H. Ohkawa, and J.A. Kans. 1996. Entrez: A hypertext retrieval tool for molecular biology. *Methods Enzymol.* 266: 141–162. (WWW-Entrez: http://www3.ncbi.nlm.nih.gov/Entrez)

Searls, D. 1995. bioTK: Componentry for genome informatics graphical user interfaces. *Gene* 163: GCI–16.

Smith, S.W., R. Overbeek, C.R. Woese, W. Gilbert, and P.M. Gillevet. 1994. The genetic data environment: An expandable GUI for multiple sequence analysis. *Comp. Appl. Biosci.* 10: 671–675.

Sonnhammer, E. and R. Durbin. 1994. An expert system for processing sequence homology data. In *Conference on Intelligent Systems for Molecular Biology* Palo Alto, CA. pp. 363–368. AAAI Press, Menlo Park, CA.

Sorsa, V. 1988. *Chromosome maps of* Drosophila. CRC Press, Inc, Boca Raton, FL.

Spradling, A., D. Stern, I. Kiss, J. Roote, T. Laverty, and G.M. Rubin. 1995. Gene disruptions using P transposable elements: An integral component of the *Drosophila* genome project. *Proc. Natl. Acad. Sci.* 92: 10824–10830.

Uberbacher, E. and R. Mural. 1991. Locating protein-coding regions in human DNA sequences by a multiple sensor-neural network approach. *Proc. Natl. Acad. Sci.* 88: 11261–11265.

Xu, Y. and E. Uberbacher. 1996. Gene prediction by pattern recognition and homology search. In *Conference on Intelligent Systems for Molecular Biology,* Cambridge, U.K. pp. 333–344. AAAI Press, Menlo Park, CA.

Xu, Y., G. Helt, J.R. Einstein, G.M. Rubin, and E. Uberbacher. 1995. *Drosophila* GRAIL: An intelligent system for gene recognition in *Drosophila* DNA sequences. In *Symposium on Intelligence in Neural and Biological Systems,* pp. 128–135. IEEE Computer Society Press, Los Alamitos, CA.

*Received September 30, 1997; accepted in revised form January 20, 1998.*