Development Approach and Architecture of GenSAS: the Genome Sequence Annotation Server

T. Lee[1], I. Cho[2], C. Peace[1], S. Jung[1], P. Zheng[1], and D. Main[1]
[1]Horticulture and Landscape Architecture, Washington State University, Pullman, WA, U.S.A.
[2]Computer Science, Saginaw Valley State University, Saginaw, MI, U.S.A.

Abstract - *Advances in DNA sequencing technology have significantly reduced the costs associated with sequencing an organism's genome. However, the operating costs of hardware, software, and labor to analyze the sequence data are still too high for most users to process in house. Henceforth, most of the current bioinformatics applications used by bench scientists will be accessible through a Web environment. This paper presents GenSAS, the **Gen**ome **S**equence **A**nnotation **S**erver, a JavaScript-based framework of gene prediction and comparative sequence similarity applications for structural and functional sequence annotation. Among other web-based genome annotation pipelines, GenSAS is unique in that it offers a one-stop website with a single graphical interface for running multiple structural and functional annotation tools, visualization and manual curation of genome. We present its functionality, the technology used in implementing each functionality, and software architecture of the overall implementation.*

Keywords: genomics tool, genome, sequencing, annotation, architecture

1 Introduction

An important component of specialized genomic databases that serve a specific community is to provide useful online tools for researchers to conduct web-based sequence analysis. These web-based tools can include BLAST (Basic Local Alignment Search Tool) [1] and FASTA [2] servers for pair wise comparison of clade-specific datasets, sequence assembly tools to assemble EST transcripts and microsatellite detection and primer identification tools. With the advances of sequencing technology, more and more clade-specific databases have started to store and display whole genome sequences with automatic gene annotation data using graphic viewers such as GBrowse [3]. Automatic gene annotation often needs to be refined by further analysis. There are many gene prediction algorithms and pipelines available to help in gene identification, but there are no online tools that easily allow biologists to readily combine the evidence from several gene prediction tools and create curated gene models within the same graphic interface. We have implemented a flexible online tool called GenSAS (Genome Sequence Annotation Server, www.bioinfo.wsu.edu/gensas) for genome sequence annotation that can assist researchers in identifying genes in genomic sequences for the Rosaceae family. GenSAS is implemented in a modular way to allow it to be easily used with other genome annotation projects.

Tool development is one of the key areas in bioinformatics research, along with the analysis and interpretation of genome data. A tool can be developed from the ground up and fine tuned for specific applications, but such a tool often ends up only being used by its developers rather than being offered for wider community use. In many cases, development of web-based systems has been ad hoc, lacking systematic approach, quality control and assurance procedures [4]. Such problems are inherent in most software development, but made worse in a Web environment due to the rapid growth of the Web, high demand for web-based applications, shorter time-to-market requirements, and relatively short history of the Web (less than 20 years). While not an exception for bioinformatics tool development, it is a challenge for bioinformatics tool developers to follow a disciplined engineering approach so that tools remain usable and stable against deviation from original assumptions about their optimal working condition. This paper does not attempt to provide a solution to all the problems mentioned. Instead, it shows one success story in bioinformatics tools development (GenSAS) and the approach taken to make the tool widely useful by researchers. GenSAS was developed with the following objectives:

- To develop a computational pipeline incorporating multiple genome sequence annotation tools.
- To develop a visualization tool to display the output from annotation tools graphically.
- To develop intuitive web-based user interfaces to facilitate curation by biologists

We first examine, in section 2, the progress of Web application development and Web architecture in recent years. In section 3, general and specific activities involved in the gene annotation process are presented. In section 4, we introduce the implementation details for GenSAS and software components that GenSAS supports. After this, related work is discussed. This paper concludes with a summary and future work in section 6.

2 Progress in Web Application Development and Web Architecture

When the Web was created in early 1990s, most websites were simple and served static content. Most emphasis was on content layouts and overall look, and easy maneuvering of the site. Little programming was required and no rigorous software engineering were required to build such websites. The growth of online resources soon made it necessary to implement search engines on the Web and process the user provided input from the Web browser. CGI was the first mechanism that allowed Web clients to execute programs on a Web server and receive their output [5]. The further growth of the Internet and World Wide Web led to full blown software applications available on the Web, rapid uncontrolled growth, hastily written code and a lack of Web standards. All this contributed to what is known as the Web Crisis. The wide use of Web applications from all over the world made them only more vulnerable to failure. To remedy the crisis and support the development of quality Web applications, the field of *Web Engineering* emerged to provide scientific, engineering and management principles and disciplined and systematic approaches to the successful development, deployment and maintenance of high quality web-based systems and applications [4]. Web engineering shares some of its principles with traditional software engineering, but it also has unique requirements: shorter development time, content-oriented development, greater importance of visual look and feel, and a more diverse user demographics.

A key area in software engineering is Software Architecture which is defined as "The software architecture of a program or computing system is the structure of the system, which comprises software components, the externally visible properties of those components, and the relationship among them [6]." There are many different architectural styles used in software applications, and it is important to decide which architecture is the best fit for the software development. The layered (or multi-tiered) architecture has many benefits: interoperability, flexibility, maintainability, and reusability, to name a few. The Communication Network protocol is an example of layered architecture. Structured Web applications also reveal multi layers where the presentation, the application processing, and the data management are logically separate processes, as shown in Figure 1. We will look at how GenSAS fits in this architecture in section 4.

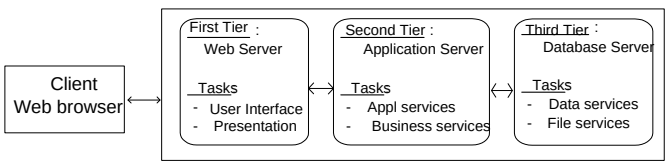

Figure 1. Multi-Tier Web Architecture

3 Genome Annotation Process

After a genome has been sequenced and assembled, the process of genome annotation starts. The purpose of genome annotation is to understand the content of the genome through locating genes and other sequence features and determining gene putative function. The annotation process can be categorized into manual and automated annotations [7], and structural and functional annotations [8, 9].

While manual annotation tends to deliver higher quality results over automated annotation, it is time consuming and expensive process, particularly impractical for large-scale whole genome sequence data. In contrast, automated annotation is a relatively inexpensive and fast process, but the output is less reliable, typically ranging from 30-70% accuracy for predicting a relatively small sample of known genes [10]. Structural annotation focuses on identifying the genomic elements on a sequence. Genomic elements include regulatory motifs, repetitive sequences, gene structure and Open Reading Frames (ORFs). Gene identification (or prediction) tools are based on statistics (*ab initio*) or sequence similarity based methods. Because each approach has its own strengths and weaknesses, it is common for gene identification tools of both types (a hybrid approach) to be used in gene annotation. Statistics based methods do not use extra information for gene prediction. Instead, they identify genomic features based on statistical patterns inside and outside of gene regions as well as patterns typical of the gene boundaries. GENSCAN is one of the most widely used statistics-based gene prediction software for human and vertebrates [11]. Other statistics based tools in wide use are FGENESH [12], GlimmerM [13], and GeneMark [14]. They use algorithms based on Markov models [15] and dynamic programming [12].

Systems have also been developed which integrate the results from several gene prediction tools and the evidence from cDNA/ESTs and protein alignments. JIGSAW, formally known as Combiner [17] is a gene prediction system that utilizes multiple sources of evidence to predict gene structure. A weight is assigned to each evidence source, and gene predictions are based on a weighted voting scheme, yielding the best consensus predictions.

Sequence similarity-based gene prediction methods are typically more reliable than statistics-based methods as experimental data are used to predict the genes. The target genome is searched for similar regions in existing sequences such as ESTs from the same species of known gene models from closely related species. The rationale behind this methodology is that homologous sequences from closely related organisms typically share evolutionarily conserved or common functions. However, sequence similarity-based tools are useful only if existing sequence data are available. For example, genes that are expressed at low levels or expressed in certain cell types, developmental stages, or growth conditions may not be adequately represented. To remedy the shortcomings of each approach, systems have been developed

to combine and integrate the results from several gene prediction tools, as in GenomeScan [16].

Functional annotation is the process of attaching biological information to the genomic elements identified during structural annotation. Such information includes, but is not limited to, biochemical function, biological function, physiological function, and Gene Ontology (GO) terms. A general approach for functional characterization of unknown genes is to infer protein functions based on significant sequence similarity to annotated proteins in sequence databases. Typically, a sequence of a gene with unknown function is compared against public databases such as Swiss-Prot [18], TrEMBL [19] or NCBI [20] using the BLAST sequence similarity algorithm.

Each of the tools mentioned have their own attributes, for instance, certain annotation tools are more robust for certain species than others, having originally been developed for those species. Generally, statistics-based methods find genes with a full-length CDS (coding sequence) but they perform poorly on finding genes with partial CDS which can be annotated more correctly with sequence similarity-based methods. Thus, quality of the results generated from each tool is not regarded as equal; some results are more reliable than others, and the result varies in different circumstances. Therefore, it is unwise to rely on only one source of evidence but rather best to combine different types of results to draw conclusions.

Often computational annotation programs generate results in text formats. Thus, several visualization tools have been developed to display the text file data graphically so that researchers can view and interpret the results more easily. The Generic Genome Browser (GBrowse) [3] is one of the most widely used products developed through the Generic Model Organism Database (GMOD) project (http://gmod.org). As it is a web-based application, annotation data can be easily shared with other researchers. However, it does not allow users to dynamically edit the annotated genomic elements.

In summary, to effectively apply structural and functional annotations, researchers are required to understand the different attributes of annotations tools, and how to specify proper parameters for their genome of interest. Also, the need for visualization through setup and management of genome viewers can be overwhelming for some researchers. Researchers typically use several annotation tools and obtain results for DNA sequences of interest in text format. In some cases, researchers must wait for the result by email when the process is computationally intensive or the sequence is very large. Then, they need to convert the text result to a format specifically required by a specific genome viewer using scripting languages like Perl. Finally, researchers analyze and compare the annotation data from different sources of evidences on the viewing tracks provided by the program. As such, researchers have to go through many time-consuming steps before reaching the final analysis steps, and currently no one-stop website exists for researchers to access several gene prediction tools and have the integrated and optimized results returned to them for further analysis.

4 GenSAS

GenSAS was developed to help researchers perform structural and functional gene annotations and provides visualization curation tools. The focus of the design was on usability and effectiveness for biologists, efficient maintenance and decreased cost for IT administrators and developers. Being a web-based tool rather than a standalone tool frees users from expending effort in installation, configuration, and upgrade. The tool was made simple to use by providing all gene annotation tasks in one Web interface.

Figure 2 shows the Web front end of GenSAS. The front page consists of five panels; User Information, Sequence Information, Task Information, Retrieve Saved Data, and Task Queue. These panels are designed to assist researchers to create various tasks intuitively and efficiently. To create a task in GenSAS, users upload a genomic sequence and select one or more annotation tools. Then, the newly created task is appended to the Task Queue panel. GenSAS currently supports nine annotation tools as listed in the Tool Information panel, but more will be added in the future.

GenSAS allows researchers to save four different types of results: Sequence, Task, Output and SVG. These results can be later retrieved to reduce redundancy in the task creation process. Clicking on the third column icon (SVG button) on the Task Queue panel will generate a report page. The reporting page in Figure 3 displays the results from nine tools and a custom track for notes as a curator manually evaluates the annotations. Genomic features from the results of gene prediction programs are colored distinctively and types of these features are identified in the legend table. The reporting page allows users to zoom in and out or set the zoom ratio, and scroll left and right to examine the genomic features in the desired location of the DNA sequence.

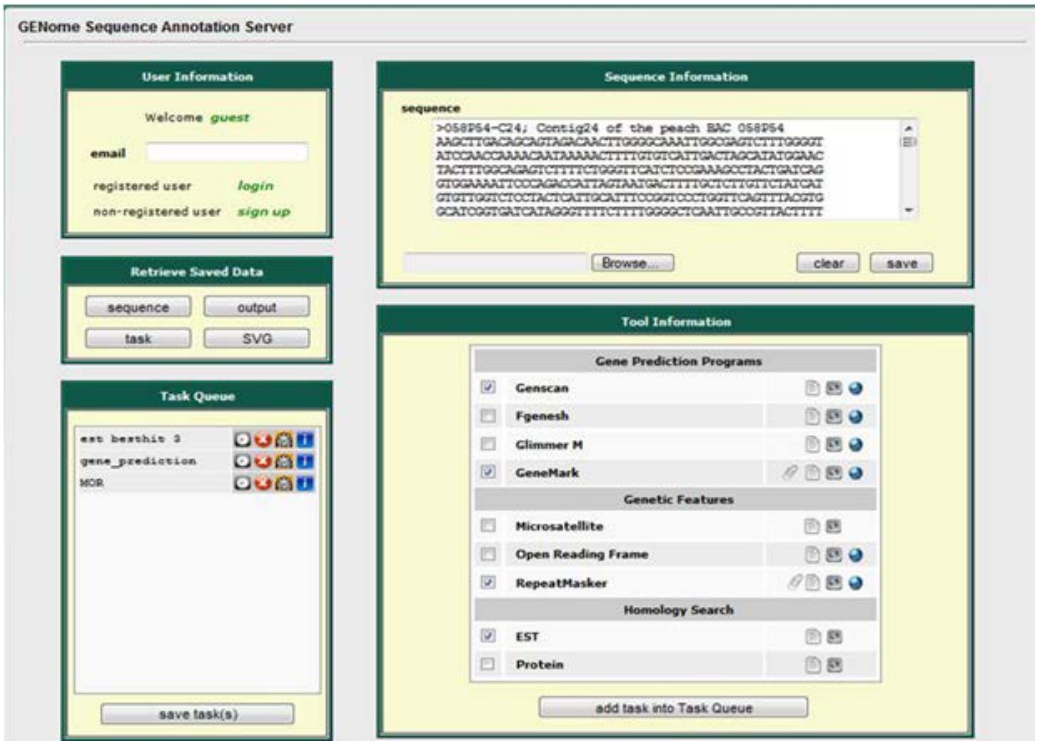

Figure 2: GenSAS Front Page

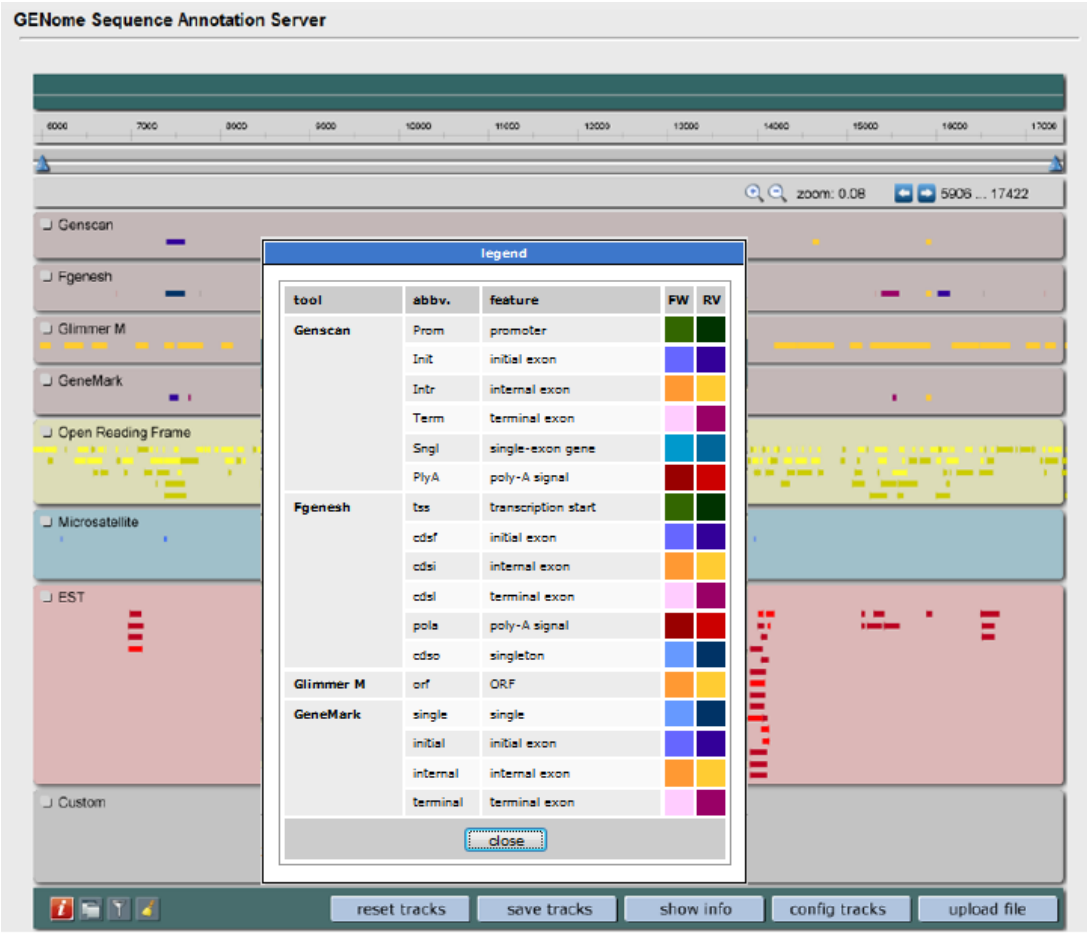

Figure 3. GenSAS Reporting Page with legend

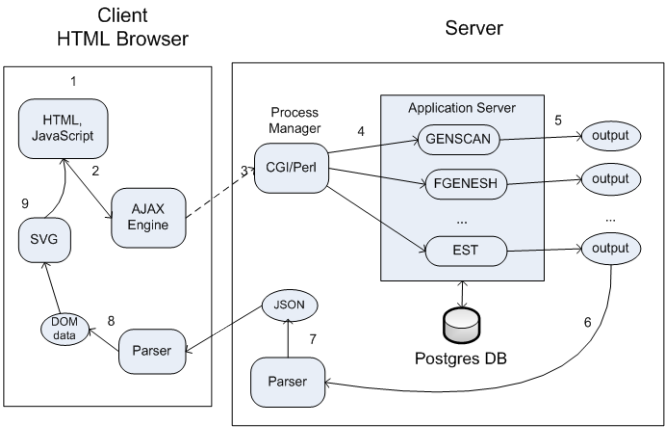

Figure 4. Overall structure of GenSAS

Figure 4 shows the overall structure of GenSAS. The software architecture conforms to the three-tier architecture covered in section 2. The rounded rectangles represent running programs or processes, and the oval shapes represent data. The typical interactions between the client and server in gene annotation are:

1. The user interacts with the GenSAS front page and creates tasks.
2. The user sends annotation job to the server (by clicking the SVG button) and waits for the result.
3. Asynchronous job requests are created and sent to the Web server running CGI/Perl (the dotted line indicates asynchronous communication).
4. CGI/Perl engine parses the input and calls exec to execute UNIX commands to run the corresponding annotation server.
5. The annotation tools execute the command and generate output in their own text formats.
6. The parser reads the text data and converts to JSON data.
7. The JSON data is sent back to the client machine.
8. The parser on the client side converts the JSON data to DOM data.
9. JavaScript works with SVG to render the data to be displayed on the Web browser.

The core technologies used in GenSAS are JavaScript, JSON, AJAX, SVG, PostgreSQL database, and CGI/Perl, all open-source and freely available tools. We now look at the details of each technology incorporated in GenSAS.

JavaScript is a scripting language commonly used in the development of client-side dynamic Web applications. It is embedded directly into HTML pages and can be readily used in most Web browsers without any further installation or configuration. It is most commonly used in creating dynamic contents into an HTML page. GenSAS uses JavaScript in various ways. It is used to manipulate HTML elements on a webpage via Document Object Model (DOM, http://www.w3.org/DOM). DOM is the primary data structure by which a Web browser represents an HTML page. It provides methods and properties to retrieve, modify, update, and delete elements of a HTML document. For example, JavaScript allows users to reorder annotation tracks by drag & drop. JavaScript is also used for handling events. When an event takes place from mouse click or drag & drop, one or more corresponding JavaScript functions are called and executed to properly handle the event on the client machine. Instead of waiting for the server to respond to the user actions and reloading the entire browser page, JavaScript allows users to interact with the browser without disruption.

JSON (JavaScript Object Notation) is a common data exchange formant which provides a structure to data like XML (http://www.json.org). It is in human readable text format and easy for machines to parse and generate. It is a native data format for JavaScript and less complicated than XML to work with for most modern programming languages. Also, JSON data can be easily transmitted between client and server machines over the network. The format of the results from each annotation tool varies and these results need to be converted into one standard format before sent to the Client Web browser. Its portability makes JSON well suited for web-based applications that intensively use JavaScript. JSON data can be easily converted to DOM format so that JavaScript can work with SVG to render the data to be displayed on the Web browser. With these reasons JSON is used as the standard text format in GenSAS; outputs from annotation tools as well as some of data types that users can save in their account in the database are formatted in JSON.

AJAX (Asynchronous JavaScript and XML) is a Web development technique that is used to create interactive Web applications on the client-side (http://www.ajax.org). AJAX is not a single technology but a combined technology of HTML, CSS, DOM, XML, and JavaScript. Traditionally, once a client request is sent to a server, the client has to wait for the response from the server without being able to do further work on the Web browser. When the response reaches to the client, the whole webpage needs to be refreshed to display the result, which results in disruption of user attention. Using AJAX, the client accesses the server asynchronously in the background without waiting for the response. Once the response arrives on the client machine, the Web browser displays the result without refreshing the whole page. AJAX is frequently used in GenSAS. All access requests to the database server are carried out in the background with AJAX. AJAX is also used to perform parallel processing on annotation tools on multi-processor server. Parallel processing (or concurrent processing on single processor server) is crucial for this system as the processing time of annotation tools varies significantly. It avoids having to wait for the completion of a large process before viewing the results of other smaller processes.

SVG (Scalable Vector Graphics) is a vector graphics file format and Web development language based on XML (http://www.w3.org/TR/SVG). Raster graphics, sometimes

called bitmap, is based on pixels and it represents an image as an array of pixels. Some genome browsers like GBrowse use raster graphics and generate image files to be sent and displayed on the client Web browser. The size of raster graphics files are relatively large compared to vector graphics and it degrades the performance of GBrowse. Also, when zoomed in, the images lose the quality with jagged lines, while vector graphics easily scale up without degrading the quality. As images for genomic features are needed to be displayed distinctively when scaled up, vector graphics is well suitable for the annotation server. Most modern Web browsers support and render SVG markup either natively (in Firefox and Safari) or with plug-ins (in Internet Explorer 8) to view SVG images correctly on Web browser. Internet Explorer 9 will natively support and render SVG. SVG is the main force behind the reporting page. Together with JavaScript, interactive graphical Web applications can be efficiently developed. All graphic features on the page are drawn with SVG images either statically or dynamically. For example, when one of exon images is clicked, it triggers the script that pops up the dialog window which shows the information about the exon such as orientation, frame and, start and stop locations. SVG graphics is also used to create GUIs such as buttons and a slider with two thumbs on the reporting page. In general, GUIs created by HTML tags are relatively plain and simple, however, the appearances of these HTML GUI components vary based on types of platforms and browsers; the looks of the buttons on the same webpage viewed by Safari and IE, for example, become different. SVG graphics allows for developers to create any shape and color, and the appearances of these GUIs will not change across browsers or platforms. Because SVG is written in XML, SVG content can be easily manipulated from JavaScript with DOM API in GenSAS.

PostgreSQL is one of the most popular relational database management systems publically and freely available (http://www.postgresql.org). It is used as a database server residing in the background of the annotation server system. It manages information about user accounts as well as their data. The Perl script has a module called Database Interface (DBI). DBI offers the standard database interface, which is capable of conducting primitive database functions on various types of database systems. DBI allows the database server to efficiently perform database operations online.

CGI/Perl The Common Gateway Interface (CGI) is a mechanism that allows Web clients to execute programs on a Web server and to receive their output. CGI applications are often used to produce HTML pages on the fly and process the input from an HTML form [5]. While many programming languages like C/C++, Java, Visual Basic, and Perl can be used to implement CGI, Perl is most often used to write CGI scripts for Web servers due to its long history of usage in UNIX systems and its strength in text manipulation. It is optimized for scanning arbitrary text files, extracting information from those text files, and printing reports based on that information. A project called BioPerl is supported by Open Bioinformatics Foundation (http://www.open-bio.org), which further strengthens its popularity. In GenSAS, Perl script is used to build CGI pages and access the database server to manage users' data. In addition, Perl is used to execute annotation tools installed on the server using the exec functions. With this function together with the AJAX described above, various annotation tools can be simultaneously executed on the annotation server to perform parallel processing.

5 Related Work

JIGSAW [17] integrates weighted outputs from multiple gene prediction tools to predict genes. Ergatis [21] enables workflow creation with multiple bioinformatics tools to perform automated gene annotations and comparative analysis. However, these tools do not provide a graphic viewer for further annotation. MAKER [22] is a genome annotation pipeline that produces annotation results that can be viewed by GMOD browsers like GBrowse. DNA subway (http://dnasubway.iplantcollaborative.org) allows users to use multiple gene prediction tools, edit the gene model using Apollo [23] and view the results in GBrowse. It is the most similar tools to GenSAS, but GenSAS has its own graphic viewer that allows users to edit and view the results in the same window.

6 Conclusion and Future Work

GenSAS has been developed in close cooperation with biologist users. Interacting with users helped identify problems and issues in currently used gene annotation tools, and has brought forth new ideas for GenSAS features. Rather than developing from the ground up, GenSAS was developed with proven technologies and well supported standards-based tools. By conforming to the industry standard three-tier Web architecture, GenSAS can be easily managed and updated for future needs. The most important issues identified and put to work in GenSAS were the ease of use, prompt response, and effectiveness for the biologist user

Ease of Use: GenSAS incorporates several different annotation tools together with available customized experimental data such as cDNA, ESTs and proteins, to provide researchers with faster processing and access to the various types of generated evidence without ever leaving the GenSAS browser page. User management through accounts is supported and users can store output results which can be later retrieved for further analysis.

Prompt Response Time: AJAX allows easy implementation of concurrent and parallel processing on Web applications. GenSAS allows users to run multiple gene annotation tools, and by using asynchronous communication mechanisms in AJAX, the result will show up on the Web browser as soon as the corresponding annotation tool finishes its job. Also, AJAX allows users to continuously interact with the browser without having to wait for the server.

Effectiveness: The Web front end is very compact and the five panels of windows are well laid out for users to easily navigate. Any users with nominal experience of using gene annotation tools will be readily able to use GenSAS quite effectively. Different shapes and colors of icons with tooltip support further help users with easy navigation of the tool. Graphic features on a track can be customized with different colors, and they can be saved to the custom track for further evidence gathering.

Since its inception, GenSAS has been constantly improved and many issues have been suggested to further enhance its capability. One notable feature under development is support of multiple tracks for the same annotation tool run with different parameters. Allowing drag-and-drop for copying features onto the custom track is also being considered. GenSAS supports private and group user accounts for users to save and allow file sharing among group members, similar to UNIX file sharing, but more an advanced and versatile user account management system is desired. Utilizing a content management system like Drupal is one possibility. To further improve the performance, the annotation processes can be sent to high performance clusters or grids. We look forward to implementing these and additional features in future GenSAS versions.

Acknowledgements

We acknowledge the Department of Horticulture and Landscape Architecture of Washington State University for the support of this work. The future development of GenSAS will be supported through USDA NIFA Award #2011-67009-30030.

7 References

[1] Altschul S.F., Gish W., Miller W., Myers E.W., and Lipmanl, D. J. (1990) Basic Local Alignment Search Tool. *J. Mol. Biol.* 215, 403-410.

[2] Pearson W.R. (1995) Comparison of methods for searching protein sequence databases. *Protein Sci.* 4: 1145-1160.

[3] Stein L.D., Mungall C., Shu S., Caudy M., Mangone M., Day A., Nickerson E., Stajich J.E., Harris T.W., Arva A., and Lewis S. (2002) The Generic Genome Browser: A building block for a model organism system database. *Genome Res.* 12: 1599-1610

[4] S. Hansen S. Murugesan, Y. Deshpande and A. Ginige. Web engineering: A new discipline for development of web-based systems. In Proceedings of the First ICSE Workshop en Web Engineering, 1999.

[5] Deep J and Holfelder P. Developing CGI Applications with Perl. Wiley 1996.

[6] Clements P, Bass L and Kazman R. Software Architecture in Practice. Addison Wesley, 1998.

[7] Collins F.S., Morgan M., and Patrinos A. (2003) The Human Genome Project: lessons from large-scale biology. *Science*, 300, 286–290.

[8] Head-Gordon, T.; Wooley, J. C. "Computational challenges in structural and functional genomics," *IBM Systems Journal* , vol.40, no.2, pp.265-296, 2001

[9] Bright L, Burgess S, Chowdhary B, Swiderski C, and McCarthy M. *BMC Bioinformatics* 2009, 10**:**S8

[10] Flicek P. (2007) Gene prediction: compare and CONTRAST. *Genome Biol.*; 8(12):233

[11] Burge C, Karlin S. Prediction of complete gene structures in human genomic DNA. *J Mol Biol* 1997, 268:78-94.

[12] Solovyev V.V., Salamov A.A., and Lawrence C.B. (1995) Identification of human gene structure using linear discriminant functions and dynamic programming. *Proc. Int. Conf. Intel.l Syst. Mol. Biol.*;3:367-75.

[13] Pertea M. and Salzberg S.L. (2002) Using GlimmerM to find genes in eukaryotic genomes. *Curr Protoc Bioinformatics.* Nov;Chapter 4:Unit 4.4.

[14] Borodovsky M. and Mcininch J. (1993) GenMark: parallel gene. recognition for both DNA strands. *Comput. & Chem.*, 17, 123–133.

[15] Salzberg S., Delcher A., Kasif S., and White O. (1998) Microbial gene identification using interpolated Markov models. *Nucleic Acids Res.* 26:2 544-548.

[16] Yeh R.F., Lim L.P., and Burge C.B. (2001) Computational inference of homologous gene structures in the human genome. *Genome Res.* 11: 803-816.

[17] Allen J.E. and Salzberg S.L. (2005) JIGSAW: integration of multiple sources of evidence for gene prediction. *Bioinformatics*. Sep 15;21(18):3596-603.

[18] Gasteiger E., Jung E., and Bairoch A. (2001) SWISS-PROT: Connecting Biomolecular Knowledge via a Protein Database. *Mol. Biol.*;3 (3): 47-55.

[19] Bairoch A. and Apweiler R. (2000) The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000". *Nucleic Acids Res*. 28: 45–48.

[20] Maglott D, Ostell J, Pruitt KD, Tatusova T. (2005) Entrez Gene: gene-centered information at NCBI. *Nucleic Acids Res.* 33:D54-8.

[21] Hemmerich, C. et al. (2010) An Ergatis-based prokaryotic genome annotation web server. *Bioinformatics* 26, 1122–1124.

[22] Cantarel, B. et al. (2008) MAKER: An easy-to-use annotation pipeline designed for emerging model organism genomes. *Genome Res*. 18: 188-196.

[23] Lewis S.E. et al. (2002) Apollo: a sequence annotation editor. *Genome Biol*. 3(12):RESEARCH0082.