Implementing an Interactive Web-Based DAS Client

Bernat Gel and Xavier Messeguer

Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya (UPC)

Summary. The Distributed Annotation System (DAS) allows clients to access many disperse genome and protein annotation sources in a coordinate manner. Here we present DASGenExp, a web based DAS client for interactive visualisation and exploration of genome based annotations inspired in the Google Maps user interface. The client is easy to use and intuitive and integrates some unique functions not found in other DAS clients: interactivity, multiple genomes at the same time, arbitrary zoom windows,... DASGenExp can be freely accessed at `http://gralggen.lsi.upc.edu/recerca/DASgenexp/`.

Keywords: DAS, visualisation, genome, browser.

1 Introduction

In recent years, genomic data comprising genome sequences and its annotations have been growing at fast pace. Every year more organisms are sequenced and studied and more data is generated. It is important to note that most of this data is completely free and available on the internet.

As more research groups publish their data and a greater number of specific databases appear it becomes more difficult to work with the available data and even find it. Some of the biggest databases offer data visualisation tools for the user to explore its content but those visualisation systems usually fail to integrate data from different sources. Additionally small groups are setting up project related databases which are hardly maintained and so updates in the genomic assemblies can draw data from two different sources incompatible.

In an attempt to address those issues the Distributed Annotation System (DAS) was proposed. By adding a simple interface to the already existing databases a new abstraction layer was created which converted the diverse and different databases into a unique global distributed database.

Here we present DASGenExp, a web-based visualisation tool designed to interactively explore the bast amount of genomic data available in this emerging distributed database.

J.M. Corchado et al. (Eds.): IWPACBB 2008, ASC 49, pp. 83–91, 2009.
springerlink.com

2 The Distributed Annotation System

The Distributed Annotation System (DAS) [1] was developed to provide an integrated interface to the annotation data from diverse sources. It allows clients to access DAS data sources from different institutions and geographically dispersed using a simple standard protocol and to fetch coherent data from them.

The DAS protocol is based on HTTP and XML and very network friendly. To request information, the client makes an HTTP request encoding all needed parameters in the URL and gets an XML file with the needed data. There are few different URLs to encode different types of requests and the parameters are very consistent between them. This simplicity makes it easy to implement in different languages and, actually, client libraries exist at least for Perl and Java. DAS servers, or sources as they are usually referred, are somewhat more complex but free Perl and Java implementations are also available.

A DAS source must be referenced to a DAS Reference Server so it's positional annotations, the main class of available annotations, are correctly referenced. This is even more important since different versions of the reference sequences exist due to updated reference genomic assembly or new advances in the genome sequenciation. A DAS Reference Server is a special DAS source able to answer to some additional requests, mainly the `sequence` command which is used to retrieve the reference sequence itself. The reference sequences in a DAS source are known as entry points and usually correspond to chromosomes – for genome based sources – and to proteins – in protein based servers –.

There are more than four hundred DAS servers currently active, some of them supported by big institutions as EBI. Additionally, a central repository have been proposed[4] where DAS sources can be registered. Those registered sources can be browsed or searched by clients. Different freely available DAS client have been developed either desktop or web-based. Clients are usually specialised either in genome or protein visualisation. In this section we will refer only to genome based clients.

Both UCSC[2] and Ensembl[3] have added DAS support to their existing web-based genomic browsers. However, DAS integration is somewhat difficult and may not be completely clear how data coming from DAS sources can be added to the browsers. In addition, both browsers suffer from lack of interactivity since the whole representation have to be reloaded in order to move the viewer to a different part of the sequence. Those browsers, however, are very powerful and a reference in genomic browsing nowadays.

3 DASGenExp

DASGenExp is a web-based genome oriented DAS client. It offers an easy to use user interface partially inspired by Google Maps usability and provides some new functionality and options not found in other clients along with a high level of interactivity.

The viewer can integrate genome oriented annotations from any DAS server and draw them in a graphical representation of the genome along with the

reference sequence. This graphical representation can be moved by dragging it with the mouse and zoomed in and out. Zoom can be changed both via a slider or the mouse wheel and ranges from the closest base pair view to a chromosome wide one. Extended information about a feature pops up when the mouse is over it. Double clicking on any point of the genome a new window is created viewing the same region but centred at the clicked base and with th zoom to its closest level.

More than one window, or track container, can be present on the page at the same time. It is possible, for example, to have different zoom levels of the same region synchronized at the same time. It is possible, too, to have two viewer windows displaying two different independent regions from the same or different genomes.

Short Example

A simple example session with DASGenExp could be like that. First of all, open the page `http://gralggen.lsi.upc.edu/recerca/DASgenexp/` in the browser (preferably Firefox). When the page is completely loaded, select 'Homo sapiens - NCBI36' in the first combobox in order to explore the human genome based on the NCBI assembly release 36. A request is then made to get the chromosomes for the selected organism.

Select a chromosome from the second combobox, for example '2', and the sequence of the chromosome will appear. The `Tracks` menu will be enabled. Add two tracks from the karyotype by selecting `Tracks` → `ensembl NCBI36 karyotype` → `band:gneg` and `Tracks` → `ensembl NCBI36 karyotype` → `band:gpos50`. The data for those two tracks will be loaded and a blue box will appear. Drag the view slightly to the right with the mouse to reveal the beginning of the blue box where the name of the feature can be seen: `p25.3`.

Zoom out using the mouse scroll wheel until most of the chromosome can be seen. Drag the image to center it around the '100Mb' tick and then zoom in using either the mouse or the slider on the right up to half the maximum zoom. Then add another track selecting `Tracks` → `ensembl NCBI36 transcript` → `exon:coding`.

Add a zoomed-out view using `Window` → `Window Zoom`. A new window will appear with the same data. If we drag the main window, the linked zoom window will move accordingly.

The final state of the application will be similar to Figure 1.

4 Implementation

DASGenExp is a client-server application with the client being a javascript web application and the server a collection of Perl CGI scripts.

The client runs inside the web browser which acts as the application container. It controls data representation and user interaction. The communication between the client and the server is done via AJAX technology but using JSON

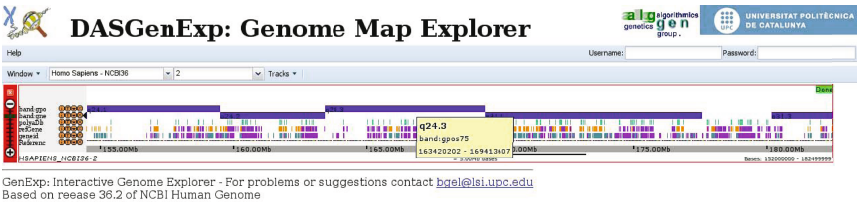

Fig. 1. Screenshot of DASGenExp with some data added

as transport instead of XML as classical AJAX. JSON, JavaScript Object Notation, is somewhat less expressive than XML but its much lightweight and less verbose and so it's very well suited for scenarios where large amounts of very simple data have to be transmitted.

The server subsystem is the one responsible of fetching the actual data from thediversee DAS sources, process it and send it back on clients request. Communication between the server and the sources is made using the DAS protocol (Figure 2).

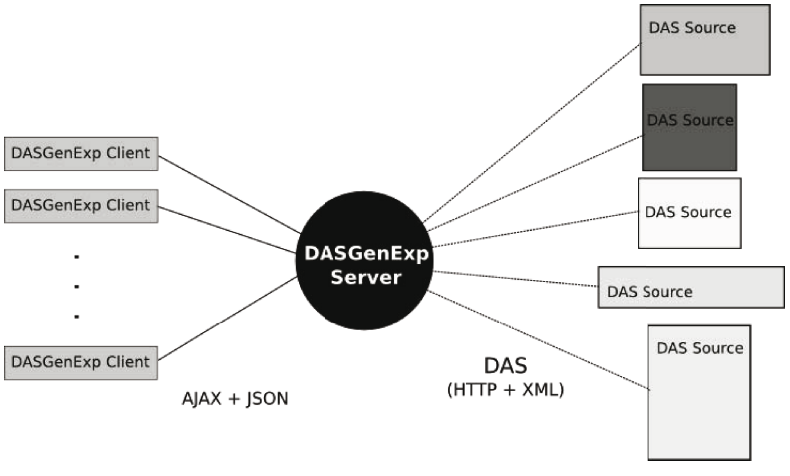

Fig. 2. Scheme of DASGenexp system

4.1 The Client

The most important and complex part of DASGenExp is the web-based client. Entirely written in javascript, the client runs inside the internet browser and is the main responsible of the data visualisation and interaction. It uses AJAX technology to asynchronously request information from the server without ever recharging the entire webpage. In fact, the DASGenExp application is made of only one and fairly simple HTML file coupled with a collection of JavaScript files containing all the application logic.

The client uses two javascript libraries. The Prototype library [5] is used as a base javascript library providing useful and commonly used functionalities such as DOM manipulation and AJAX handling. It also provides shorthands for some often used functions resulting not only in faster and less error-prone programming but also in shorter and more legible code. Another very useful functionality Prototype offers since it's last version is a complete system to manage custom made events in exactly the same way the default events are handled. This event system is very powerful and gives the opportunity to use event based communication between different parts of the system which is very convenient in a GUI based application and even more when there are asynchronous processes involved such as AJAX requests.

The second library used, EXT [6], offers a full range of very interactive and desktop-looking widgets and a modular object-oriented framework to handle them. It is the library used to create and control the most of the graphical user interface: the menus, the toolbars, dialogs, child windows.

However, there is a very important part of the GUI which is not rendered using any library but by direct manipulation of the underlying HTML file: the sliding ribbon where the genome representation takes place. This kind of hand made manipulation is fundamental in order to get the most from the browser and it's very important to offer good response times and interactivity to the user.

The client has been build as a modular system with three modules with very clear functionality: the main GUI, the data proxy and the track-container modules. The modular design has important advantages when extending or adapting the system and allows for easy implementation of new functionalities.

Main GUI Module

The first module is the one in charge of the main window and GUI. It renders the main and track-container specific toolbars as well as dialogs and messages. It also controls the size of the screen and notifies the non-EXT parts of the application when a resizing is necessary via the Prototype custom-event system. The main GUI is what can be considered as the main module of the application since it creates and controls the others and is the main part interacting with the user. Most of the active user interface is build with EXT and its very convenient code based GUI creation functions. This module is also the one controlling the creation and destruction of track-containers and its positioning and sizing.

Data-Proxy Module

The second module is a data-proxy. It receives the data requests from the other client modules and request it from the server in a suitable way. It encapsulates and handles all AJAX communication between client and server and can be replaced or extended if other data sources or protocols should be used. This module is responsible of another fundamental feature of a system intended to manage and visualise large amounts of data: caching and data sharing.

The data-proxy stores all genomic data in the client and serves it to the different track-containers when required. The existence of a unique data-proxy ensures that the data used by different tracks in different track-containers is never requested more than once to the server reducing the server load, the client memory footprint and the network bandwith used.

Petitions to the data-proxy are made via its external interface but direct access to the underlying data structures is granted to the track-container modules in order to reduce the overhead of function calls in heavy used data. The notification of completed requests is made via the custom events system provided by Prototype.

Track-Container Module

The track-container is the module creating and handling all graphical representation of the genomic features and sequence. It handles the data requests to the data-proxy module as well as the user interaction with the data such as movements, zooms and addition or deletions of tracks.

Each track-container in the application is actually a long page loaded inside an HTML iframe. Each of these pages has its own copy of the track-container module running and controlling the specific genomic view. They are all linked via event observation of the main GUI module or the shared data-proxy module.

Two different kinds of tracks can be drawn in a track container:

Feature based tracks. This kind of track is rendered by creating a single and unique DIV HTML element for each feature in the track, and then attaching the necessary listeners to it. This respresentation allows for very fast response when dragging the genome and specially when zooming, since no new data have to be downloaded. The feature based tracks allows in client control of the representaction and easy customisation such feature colour changing, etc. Since in feature based tracks each feature has to be added individually to the DOM and the browser has to render them one by one, the exact representation of each feature is heavily optimised. Despite the optimisations, this representation is not suitable for very dense tracks with more than about ten or fifteen thousand features per entry point. This limit also varies depending on the client hardware and the browser used.

Image tracks. This second kind of track is used in dense tracks such as SNPs and basically any track with more than ten thousand features per entry point. In this case, the track is represented as a set of contiguous images representing each of them a small part of the genome at a given zoom level. In this case, the heavy work of finding the features present in a genomic region and its graphical representation is done by the server. Usually, the server not only creates the image itself but also an associated HTML map which is send to the browser alongside with a link to the image file in the JSON response to the AJAX request. This map is used to give to the images almost the same level of interactivity as the HTML based features – tooltip, extended information, useful links... Both the image and the associated map is cached

in the server in order to reduce the server load and the response time. Most current browsers are also configured to save a cached copy of images used in web pages and thus, when revisiting a region previously examined, most of the data will be already cached in the client, reducing latency and incrementing responsiveness. This kind of tracks is used to represent the underlaying sequence, which can be viewed at most close zoom levels.

The structure of the track-container as a long sliding ribbon is one of the keys to achieve speed and response. Simple small movements translate into movements of the whole ribbon reducing the repositioning computations to be made by the browser. When zooming, however, the tracks are completely redrawn from scratch for feature based tracks and new images are requested from the server for image based tracks.

The client has been developed and thoroughly tested using Firefox2, however it also been tested under Opera and Safari and it works correctly. The only browser where the client has problems is Internet Explorer and it's because of some quirks in its JavaScript handling.

4.2 The Server

The DASGenExp server subsystem is a collection of Perl scripts accessed via CGI interface. The main purpose of those scripts if fetching the data from the DAS sources and serving it to the client. However they also handle the application configuration and perform a very important task of caching.

Most of the interaction between the client and the server is done using the Bio::Das::Lite[7] Perl module. This module encapsulates the DAS queries in Perl functions getting and returning Perl objects. By using it all the hassle of parsing the XML – and eventually adapting to updates on the diverse DAS DTDs – is avoided.

When a client request the data for a track, the server connects to the DAS source and asks for all features of the requested type on the requested entry point. When the response is received, it is processed, transformed to JSON and send to the client. A copy of that data is also stored in the server cache and will be returned the next time the track is requested for that entry point. That scheme is used only in non-dense tracks – approximately les than ten thousand features per entry point – and thus responses can be fast.

In order to respond to an image request the server uses the cached data stored in a series of files corresponding to 10 kbases for sequence data and 10 Mbases for features. The images are created using the GD library and it's Perl bindings. The use of small files gives usually fast image creation and response despite in some cases the actual data have to be previously fetched from the actual DAS source. When generating images at a close enough zoom level – less than 200Kbp per 1000px – an associated HTML map is generated to associate feature information with the image. Both the images and maps are cached to avoid unnecessary server work.

The application configuration is made on the server by editing some simple text files and no access to any database is needed. Since it only requires an HTTP

server, a Perl interpreter and some freely available Perl packages, the custom install of DASGenExp is simple and straightforward. This option, however, is still not available since the application is still in development state.

5 Conclusions

We have developed DASGenExp, a web-based genome oriented DAS client easy to use and highly interactive. Our client gives to the user the opportunity to explore integrated data coming from different DAS sources using an intuitive GUI. It additionally offers some newer functionalities as showing multiple genomic locations at the same time, or arbitrary number of additional synchronized zoom windows. It is important to note that despite all the information it offers the user interface is very responsive specially when moving around the genome.

DASGenExp is currently available at `http://gralggen.lsi.upc.edu/recerca/DASgenexp/` and can be freely accessed.

6 Future Work

At the moment the server is running in a single Sun machine (Sun Fire 280R Server, 2xUltraSPARC III 1.2GHz, 4GB of RAM) and this is the cause of most of the lag that can be observed when downloading data and specially image based data. We plan to add more computational resources and we expect to get a big improvement in speed.

In the client part we are working on the possibility of the user dynamically adding new DAS sources to the sources list since the current list it is not exhaustive and there can even exist user specific sources.

Acknowledgements

This project is supported by research project LogicTools TIN2004-03382 and by a predoctoral FI grant from the Comissionat per a Universitats i Recerca del Departament d'Innovació, Universitats i Empresa of the Generalitat de Catalunya and the Euopean Social Fund.

References

1. Dowell, R.D., Jokerst, R.M., Day, A., Eddy, S.R., Stein, L.: The Distributed Annotation System. BMC Bioinformatics 2, 7 (2001)
2. Kent, W.J., Sugnet, C.W., Furey, T.S., Roskin, K.M., Pringle, T.H., Zahler, A.M., Haussler, D.: The Human Genome Browser at UCSC. Genome Research 12(6), 996–1006 (2002)
3. Stalker, J., Gibbins, B., Meidl, P., Smith, J., Spooner, W., Hotz, H.R., Cox, A.V.: The Ensembl Web Site: Mechanics of a Genome Browser. Genome Research 14(5), 951–955 (2004)

4. Prlic, A., Down, T.A., Kulesha, E., Finn, R.D., Kahari, A., Hubbard, T.J.P.: Integrating sequence and structural biology with DAS. BMC Bioinformatics 8, 333 (2007)
5. Prototype JavaScript framework: Easy Ajax and DOM manipulation for dynamic web applications (May 2008), http://www.prototypejs.org/
6. Ext - A foundation you can build (May 2008), http://extjs.com/
7. Bio::Das::Lite - Perl extension for the DAS (HTTP+XML) Protocol (May 2008), http://biodas.org/, http://search.cpan.org/~rpettett/Bio-Das-Lite-1.054/lib/Bio/Das/Lite.pm