Implementation

Qrator has been implemented as a lightweight, web-based application that connects to a series of Representational State Transfer (REST) based web services that are backed by a PostgreSQL database. The web application can be accessed at http://glycomics.ccrc.uga.edu/qrator/. Guest users may browse structures, and view the status page, along with the canonical trees, but they cannot review or curate structures. However, those interested may install a local copy of Qrator, giving them the ability to curate their own structures. The source code and installation instructions are available at https://code.google.com/p/qrator/. Qrator can be configured to interact with our Ontology Web API (still in development), or to function in standalone mode, meaning that it will not be able to add structures to an instance of the GlycO ontology. Even without this capability, curated structures are always available for download from Qrator's status page.

User Interface

The user interface is implemented in JavaScript, and relies heavily on the jQuery framework (http://www.jquery.com/). Many interface elements and CSS styling also come from the Twitter Bootstrap framework (http://getbootstrap.com/). Asynchronous calls to REST-based web services implemented as Java servlets supply it with data. These calls are made using the AJAX (Asynchronous JavaScript And XML) collection of web techniques. After glycan structures are retrieved from the server side, they are rendered on the client side using D3.js (http://d3js.org/), and further enhanced with supplementary information such as residue and link labels that are displayed on mouse-over. Annotations, references, and provenance for structures are also retrieved when necessary, and rendered in a second pane alongside the list of structures.

Data is encoded in JSON for transmission between the web services and the web interface. JSON offers a more concise data exchange format than XML, and integrates well with a JavaScript user interface. Similar to XML schema, a JSON schema validation method is available in case there is a need, but Qrator does not utilize it at this time.

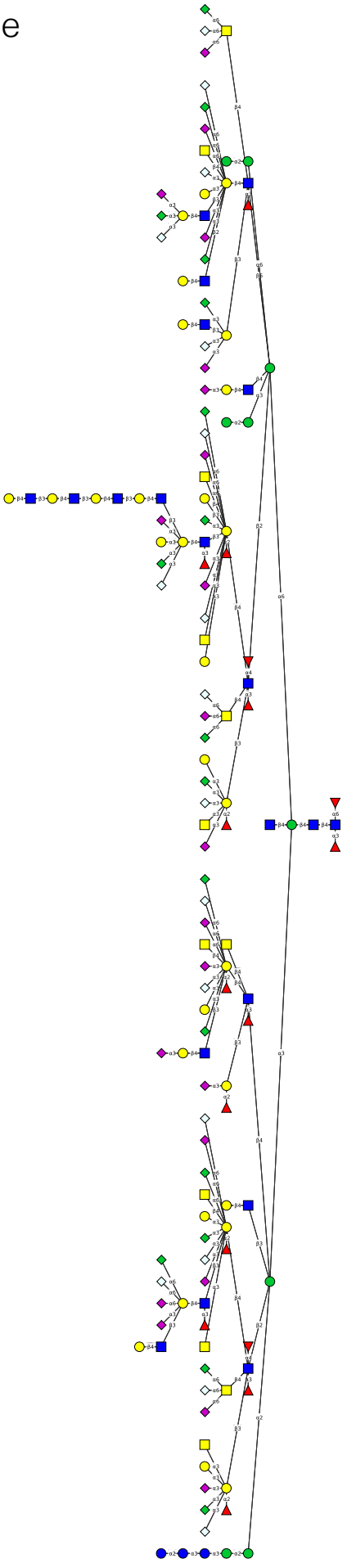
Web Services

The service layer has been divided into multiple REST-based web services based on their functionality. Creation, edit, and search functions for most types of objects within Qrator are contained in their own individual services, and all are available by querying them through a suitable Java web server. For example, a user may want to create more references for a structure, edit an annotation for a structure, or search over structures they have uploaded. Individual services are provided for structures, literature and database references, annotations, user accounts, and administration functions for updating GlycO or downloading structures. Currently, we deploy Qrator on a JBoss Application Server (http://jbossas.jboss.org), but we have also tested the application on an Apache Tomcat web server (http://tomcat.apache.org).

Data Storage

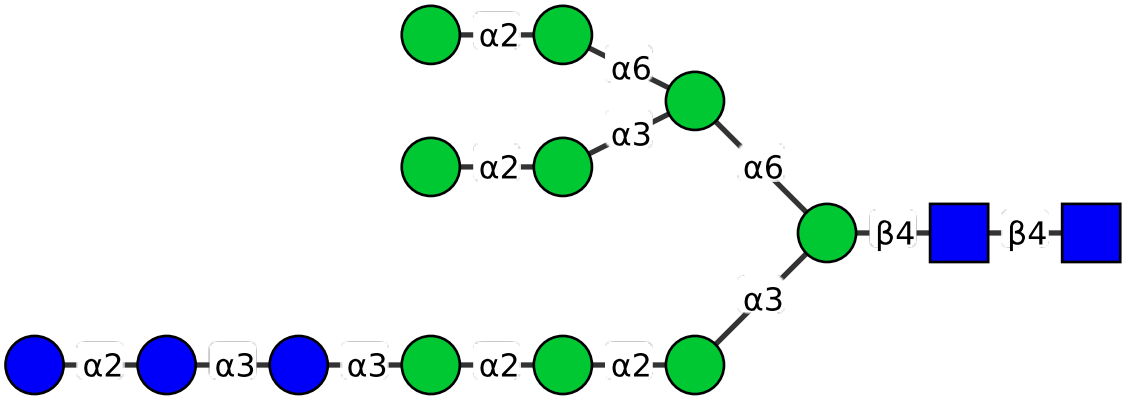
Qrator utilizes a PostgreSQL database for keeping track of submitted structures in different phases of review, along with references, annotations, and provenance information for these structures. We also keep track of which canonical trees are available in GlycO, and sub-classes of structures within those trees. Furthermore, we retain copies of all canonical trees in the database to compare structures against during the structure-matching step. The GlycO ontology itself is maintained by a separate application, and is updated via web service invocations by Qrator when necessary.

N-glycan Canonical Tree

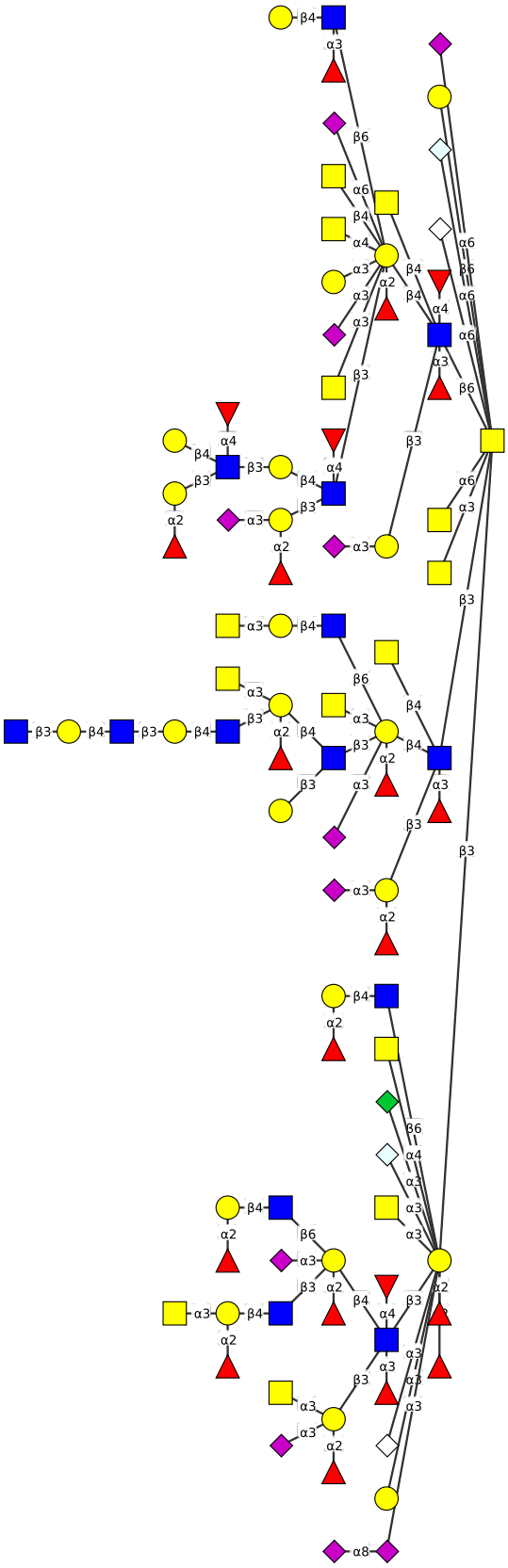


Fucose
Galactose
GalNAc
GlcNAc
NeuAc
NeuGc
Neu
KDN

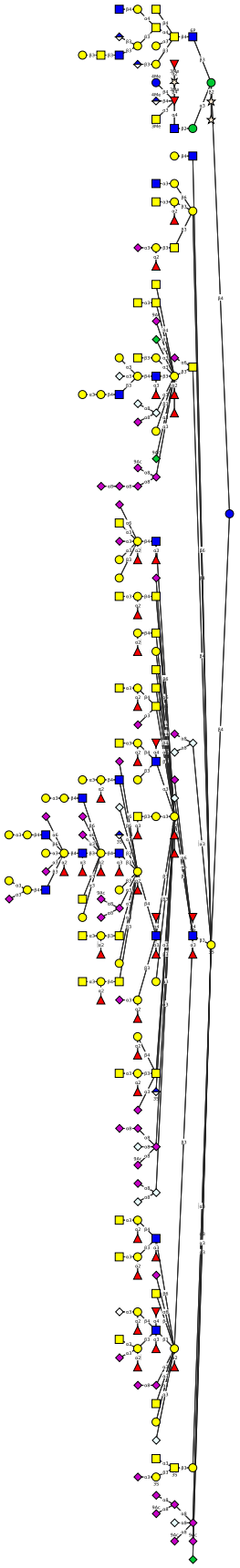
N-glycan Lipid-linked Precursor Canonical Tree



GalNAc Initiated O-glycan Canonical Tree



Glc-Initiated Glycosphingolipid Canonical Tree



Gal-Initiated Glycosphingolipid Canonical Tree