High Level Design Document

# Gene Therapy Discovery Service

# 1. High Level Architecture

The diagram below illustrates the overall layout of the processes and services involved in delivering gene therapy analysis for clinicians. In my explanation of the architecture I will attempt to discuss both how this will look in prototypical form and also a more robust and production worthy design.

This document will provide an explanation in a Model-View-Controller software architectural pattern (see diagram 1.1). This pattern is useful in segregating the tasks occupied by the model (data sources, i.e. databases, data warehouses, other types of data repositories), the view or presentation layer or the user interface (UI) and the scripts that provide the maintenance of the model’s records and enhanced rules layer for extracting and comparing sequence data in the presence of therapy outcome and patient data.

* 1. Diagram for the Model-View-Controller (MVC) Pattern

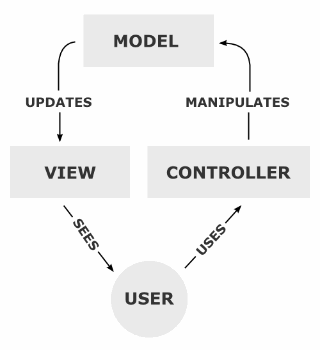


Diagram 1.2 illustrates the flow of data through the system and the services and processes involved. All of these elements are explained in more detail in sections 2 through 5.

1.2 Architectural Diagram of BigOnc

# 

### 2. External Sources

The data contained within the model layer will be obtained with permission from remote resources such at the Sequence Ontology (SO), Broad-Novartis CCLE, NCI60, The Cancer Genome Atlas, etc.

#### 2.1 Prototype phase

Data collected in the prototype phase will be downloaded by hand and uploaded using conventional bulk uploading tools into a PostGreSQL database. These data will be stored in a schema called staging and the table structure will be representational of the layout of the upload file. This will provide a clear picture of the disparate formats in which these data are organized. It will also provide some insight into how to organize these data in a more integrated and normalized table structure. Finally it will make it easy too implement auditing scripts which will insure the integrity and accuracy of the uploads vis-à-vis the source files

#### 2.2 Production

In production this will require a more robust and built out mechanism for adding data resources, detecting changes to those resources, systematically transferring the data. Finally the data will be uploaded to the appropriate tables using either bulk loaders or custom scripts, if necessary.

This will involve scheduling jobs and possibly scripts built around the database to insert and/or update data.

### 3. Staging Database

The project description concentrates on issues of performance, data integrity and security for some of the data to be held in the model part of this pattern. PostGreSQL has been chosen as the staging and normalizing process for this purpose. This product is open source. This means that at least initially during development and testing the instance is free. More importantly though is the capability within this database product to modify it’s capabilities via a rich Application Programmer Interface (API). APIs allow the user to modify the functionality of a product to customize it for internal needs. It is also a standard Relational Database Management System (RDBMS), which is essential for maintaining data integrity and security through the data delivery process.

The staging database will contain three schemas. The first will be a staging schema, which is explained in section 2 - External Data Sources. The sequence and therapies schema will be a public schema and will contain the data from the staging schema in a normalized and moderated format. Finally a patient schema will separate out patient data since it will require the greatest level of security relative to the other two schemas. It will be a private schema and data made available from this schema will be either sanitized and/or encrypted prior to delivery to any other resource.

#### 3.1 Prototype Phase

The database will be stored on a single node for simplicity during this phase. Primary keys, indexes, foreign key constraints will be in place during this phase. Some database triggers and stored procedures will be present in this phase, which will allow for the movement of data amongst schemas.

#### 3.2 Production Phase

In production, enhanced security measures will be put into place where appropriate to insure compliance with Health Industry privacy protocols.

In addition, PostGreSQL provide the ability to create customized indexes schemes that might be useful to boost performance during sequence searches.

Archiving policies might also be implemented if necessary. This will preserve the history of changes to the database over time, again for auditing purposes.

### 4. The Query Database

To provide the level of performance required for timely analysis of strategies for gene therapy, it will be necessary to use an alternative to the standard OLTP pattern used in most traditional RDBMS like PostGreSQL. The database chosen for this role is an Apache project called Cassandra. This is a database structure optimized for extracting data from a repository (querying as opposed to other DML type statements like insert, update or delete). Structurally it is a hybrid between an RDBMS and a key/value pair storage mechanism like LDAP.

#### 4.1 Prototype Phase

In the prototype phase, Cassandra will be stored on a single node for simplicity. This phase will provide insight into how to set up the grouping of data in a manor appropriate for the production instance. It will also allow us to gauge the scalability and better estimate the cost of adding hardware in production to support a multi-node distribution.

The data exchange between the PostGreSQL server and Cassandra will be simple scripts.

#### 4.2 Production Phase

In production this database will be installed over multiple nodes based on our experience during prototyping. New nodes can be added as required to allow for scalability. In addition, host machines optimized for the architecture of a database like Cassandra attached to arrays of solid state disk drives could provide additional performance gains and increased reliability

The data exchange process will be more real-time through the use of a replication process similar to either a full blown change data capture process or a customized messaging system process to improve performance and robustness.

### 5 Clinician’s Interface

Finally the view or user interface of this design will be provided via representational state transfer (RESTful) web services. This is the are of design we will require the most collaboration with our clients in the process of requirements gathering with regards to the typically types of questions and analysis they expect from the system and the most effective presentation for the output.

#### 5.1 Prototype Phase

In the prototype phase web services may be written in a higher level language to speed the delivery of a workable mockup.

#### 5.2 Production Phase

This production version of these services will most likely require an implementation by a lower level programming language for performance purposes like Java EE.