# Introduction

The purpose of this document is to define the high-level vision for the next major release of caArray – version 2.5.0. It focuses on the capabilities needed by the users of caArray, and the features planned in 2.5.0 to address those needs.

The scope of caArray 2.5.0 is being driven by user community needs to support new assay types especially those spawned by emerging technologies like next generation sequencing, the desire to scale to higher volumes of data, current limitations of the upload/import process, the user community’s desire to better manage user roles and collaboration groups, the NCI CBIIT infrastructure requirements for Grid security, and the need to stay current with the technologies used by caArray.

# Problem Statement

Emerging technologies like next generation sequencing are now becoming widely used, in the user community that caArray seeks to serve, e.g., the TCGA project. As investigators move from microarray experiments to such newer types of experiments, there is a need to enhance caArray in order to be able to store these experiments.

As the number of experiments in caArray grows, there is a critical need for the application to be able to scale to support large volumes of data. E.g., the TCGA project is now generating next generation sequencing data at the rate of 10 TeraBytes per month. (While a significant portion of this TCGA data may be stored in external repositories like the NCBI SRA, caArray still needs to be able to support increasing amounts of data.) The architecture for data storage, import and download must be examined and updated in order to support large data volumes.

There are current limitations in the user experience during upload, import and download of data to/from caArray, and there is a need to make it fast and easy to perform these tasks without a lot of manual effort to “chunk” large data sets.

Feedback from the user community has revealed that there is a need to be able to support different types of user roles, each of which has a different level of access to data and/or features in the application. A successful solution would allow appropriate levels of access to users based on their role, and would make it easy for these roles to be configured. In addition, there are known problems with visibility of files to collaboration groups, which need to be addressed.

Current invokers of the caArray programmatic Grid APIs (e.g., analysis applications like geWorkbench and GenePattern) are restricted to publically available data, and do not allow access to protected data. This is because the

Grid APIs do not support Grid security. In order to meet the NCI CBIIT infrastructure requirements, the caArray Grid APIs and application will need to support Grid user login, and also allow migration of existing user accounts to Grid accounts.

Finally, several of the technologies used by caArray are becoming dated, and there is an urgent need to upgrade them. Key technologies being targeted for 2.5.0 are Jboss 5.1, Java 6, MySQL 5.1 and BDA 1.7.

# Product Overview

## Needs and Features

|  |  |  |
| --- | --- | --- |
| **Need** | **Priority** | **Features** |
| Support next generation sequencing assays | *High* | \* Support the new MAGE-TAB v1.1 SDRF columns (Assay Type and Technology Type) that allow specification of non-microarray assay types.  \* Support specification of fastq and BAM/SAM files, possibly as links to accessioned files in an external repository like the NCBI Short Read Archive. |
| Support storage of large volumes of data | *High* | \* Allow storage of data files on the file system instead of the database.  \* Support cloud storage of data files, allowing a local installer to configure the same.  \* Prototype storage of parsed data on the file system (using NetCDF or similar). Also prototype storage of parsed data in a Postgres database. Compare the two options on performance and transaction limits. A migration strategy for existing data will be needed. |
| Make it easy to upload, import and download large sets of data | *High* | \* Implement an Upload/Download manager that eliminates the 2GB upload limit and allows resumable downloads and transparent compression.  \* [See parsed data storage options above for ways to eliminate the current 4GB import limit imposed by MySQL.]  \* Add better Import Queue management, including the ability to see the position of your import in the queue and the ability to cancel an import. |
| Support a variety of user roles and privileges | *Medium* | \* Gather requirements around what user roles and privileges are needed, and what portions of the application features/data need to be restricted based on role. |
| Fix problems with collaboration group access to files | *Medium* | \* The fix to Gforge #14630 would ensure that collaborators with the appropriate privileges would have the right access to uploaded-but-not-yet-imported files. |
| Support Grid credentials and login access through the Grid API | *Medium* | \* Design a Grid security enabled application and API. Components would include migration of current user accounts to Grid accounts using CGMM, allowing local installers to configure their preference for Grid security versus local security, and implementing Grid authentication in the remote Java (EJB) API. |
| Upgrade the tech stack | *High* | \* Upgrade the application server to Jboss 5.1  \* Upgrade to Java 6  \* Upgrade to MySQL 5.1  \* Upgrade to BDA 1.7 |