
GDPC: The Genomic Diversity and Phenotype Connection

Middleware for Genomic Diversity and Phenotypic Data

Terry M. Casstevens¹ and Edward S. Buckler²

¹Departments of Statistics and Genetics, North Carolina State University, Raleigh, NC 27695-7566, ²USDA-ARS, Department of Genetics, North Carolina State University, Raleigh, NC 27695-7614

This Revision: February 26, 2003

Updated: January 8, 2003

Created: October 18, 2002

Abstract

Databases and software analysis tools that integrate information on genomic diversity with phenotypic data provide an opportunity to help bridge genomics and plant breeding. "The Genomic Diversity and Phenotype Connection," GDPC, provides a middleware interface that gives access to data on genomic diversity (SNPs, RFLP, AFLP, etc.) and phenotypic data collected in field, genetic, or physiological experiments.

The goal of this project is to simplify access to that data (i.e. loci, taxa, experiments, phenotypes, etc.) by creating a middle layer between the software analysis tools and the data itself. This middleware separates the tools from the data format, allowing developers to concentrate on the tools' purpose. Tools request data from the GDPC interface in a standardized format. Since the middleware manages the interface between the tools and the data format, the tools, once designed, are able to work with any number of databases. GDPC also provides a way for the database developers to write connections for their database, allowing any given database to work automatically with tools supporting the GDPC interface. This would allow tools to take advantage of new databases (with supporting connections) without additional software development. Finally, interactions with the databases are separated from analysis and presentation, allowing software tools to work with larger sets of genomic and phenotypic data.

The early release version of GDPC is available at <http://www.maizegenetics.net>. We are looking for community input on how this interface can best serve the plant community needs. Please feel free

to email us at terry_casstevens@hotmail.com or buckler@statgen.ncsu.edu.

Introduction

The motivation for GDPC stems from the increasingly large genotypic and phenotypic datasets that are becoming available in plant biology. Millions of phenotypic and genotypic data points are now being collected, but most of the analysis tools still rely on numerous, different flat file formats. It appears that the public and private sectors need a way for general analysis tools to work with multiple databases directly. Once a standard interface is available, then developers of analysis tools would be more likely to build database-aware tools. This would facilitate the combined mining of molecular genomic data with phenotypic data. GDPC integrates genomic and phenotypic data by defining the relevant elements and relationships between these elements. GDPC also provides a way to access this data from any number of data sources.

Genomic Diversity and Phenotypic Data Life Cycle

There are currently many research projects that collect genomic diversity and phenotypic data. This data often remains on individual's desktop PCs or in private databases even after publication. Ideally, this data would be made public. That way, the data would be available to other researchers for reanalysis. GDPC makes this a more attractive option, since it facilitates access of the data. The flow charts (fig. 1) show the typical data life cycle and the proposed data life cycle.

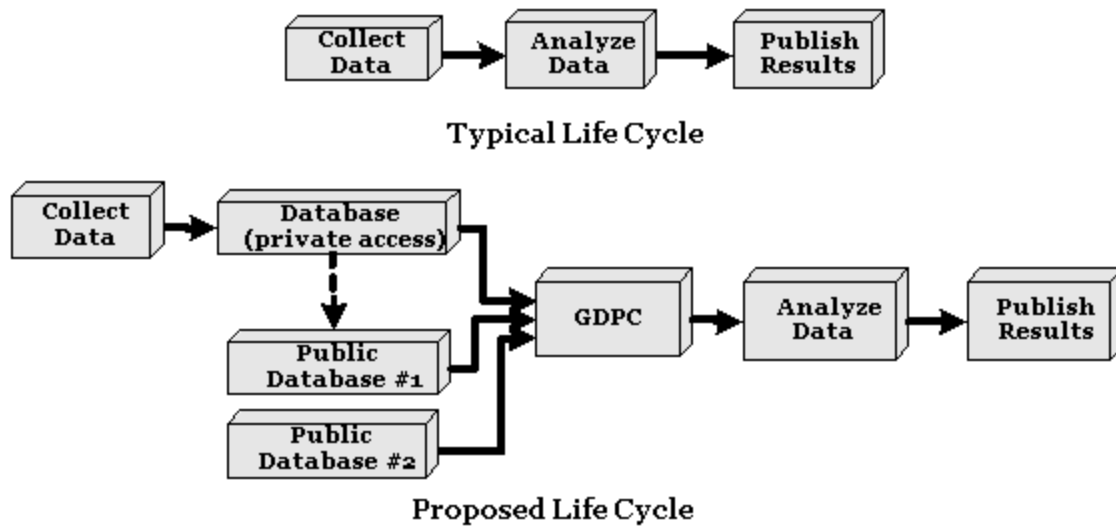


Figure 1: Genomic Diversity and Phenotypic Data Life Cycle

Using GDPC

For any software analysis tool requiring genomic and phenotypic information, data must be acquired from a data source, usually a database. GDPC standardizes the way in which these tools obtain the required information. More specifically, GDPC has a Java interface that acts as a gateway to one or more data sources. The tools request data from this interface and, in turn, receive the combined relevant information from all the managed data sources. Tool developers working with GDPC would not need to know the specifics of the data sources. In addition, if other data sources were later adapted to support GDPC, the

tools would automatically be able to access those additional data sources.

Integrating Genomic and Phenotypic Data

The diagram (fig. 2) illustrates the various genomic and phenotypic data elements accessible through the GDPC Gateway. The lines connecting the data elements show their relationship, such that the element next to the diamond contains the element pointed to by the arrow. The notation, “1..n”, represents a “one to many” relationship, and its absence indicates a “one to one” relationship. For example, one or more taxa are components of a taxon group.

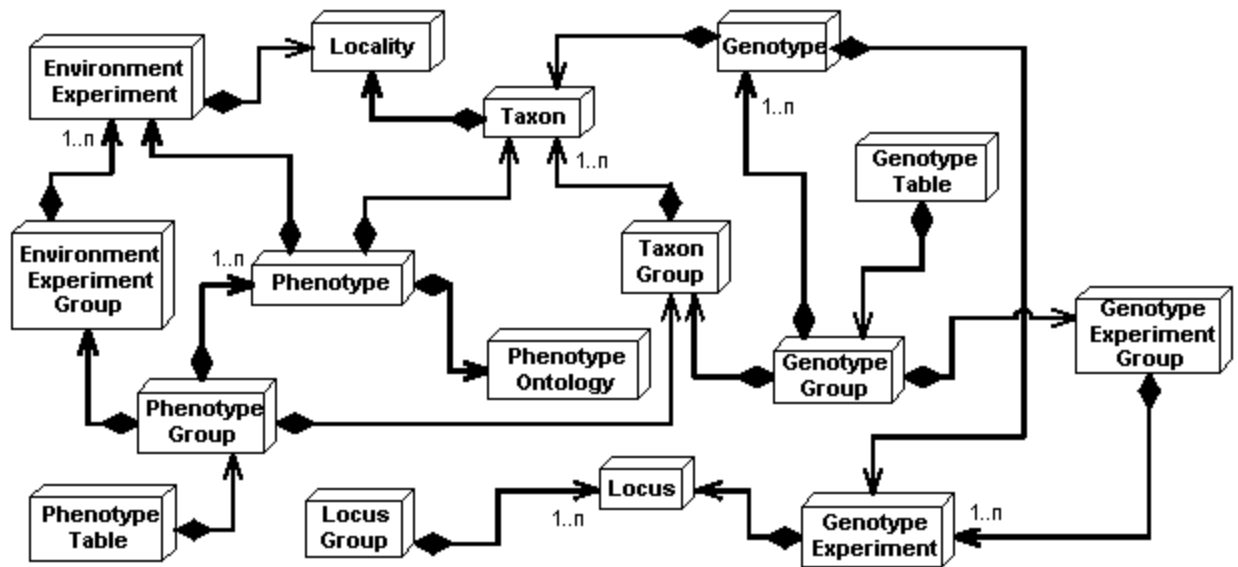


Figure 2: Genomic and Phenotypic Data Elements

- **Environment Experiment:** This defines when and where the phenotypic evaluation was carried out.
- **Environment Experiment Group:** For example, all environment experiments harvested during a particular range of dates.
- **Genotype Experiment:** This defines how the sequence, SNP, SSR, etc. data was collected.
- **Genotype Experiment Group:** For example, all genotype experiments performed on a particular locus.
- **Genotype:** The genotype for a particular taxon and genotype experiment.
- **Genotype Group:** Group of genotypes
- **Genotype Table:** Table representation of a genotype group.
- **Locality:** Description of a particular geographic location.
- **Locus:** A specific location on a chromosome. (e.g. gene, etc.)
- **Locus Group:** For example, all loci located on chromosome 3.
- **Phenotype:** The phenotype for a particular taxon, environment experiment, and phenotype ontology.
- **Phenotype Group:** Group of phenotypes.
- **Phenotype Ontology:** Defines the physical trait of an organism.
- **Phenotype Table:** Table representation of a phenotype group.
- **Taxon:** Defines the source of the germplasm used in the experiment.
- **Taxon Group:** For example, all inbred tax.

Data Sources

The diagram below (fig. 3) shows how software analysis tools can use multiple data sources via the gateway provided by GDPC. All the information is requested from the GDPC gateway, and the gateway handles the aggregation of the data. The

gateway queries each data source individually and combines the data, which it then returns to the tool. The GDPC gateway queries the data sources via a special connection designed specifically for each supported data source. These connections know the specifics of their data source. For example, a connection supporting the Panzea database knows

how to construct the appropriate database query (SQL) commands to retrieve the requested data. It also knows how to map the database's fields and tables to the Java representations of the genomic and phenotypic data elements (see fig. 2). The

connections do not need to know anything about the tools requesting the data. These tools can therefore operate with any existing or future connection.

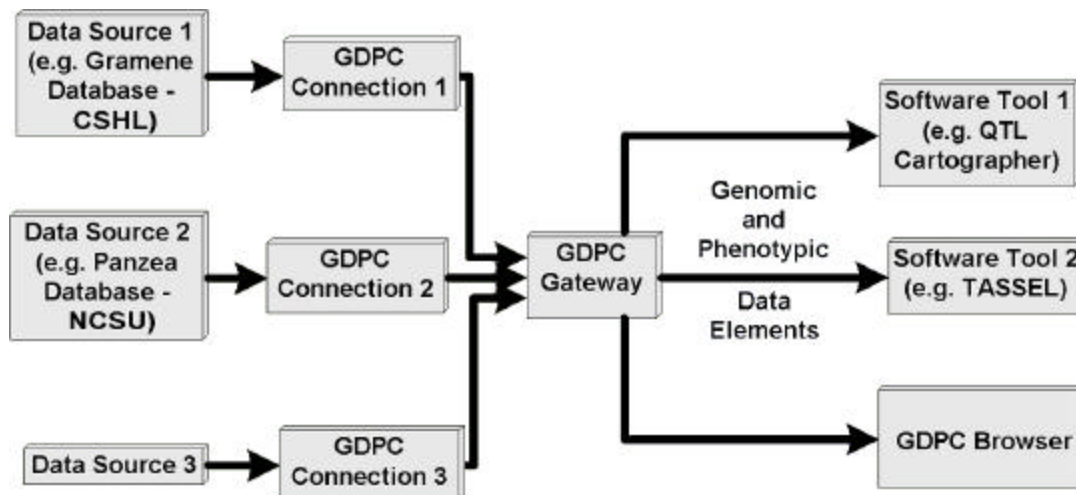


Figure 3: Example of GDBC managing multiple data sources

GDBC Gateway

As explained in the previous section, software analysis tools make all requests for genomic and phenotypic data via the GDBC gateway. The table (fig. 4) below lists the methods and return values that the gateway supports. A few of these methods, specifically `addDBConnection`, `getDBConnections`, and `removeDBConnection`, manage the connections described above (see fig. 3). These methods allow the user to decide which data

sources will be used when information is requested. The `getDistinctProperties` method retrieves distinct values for the given properties (e.g. locus chromosome number, environment experiment plant date, etc.). These values can then define the filters, which are used to request data with specific characteristics. The remaining methods retrieve the genomic and phenotypic data (see fig. 2).

Interface <code>gov.usda.gdpc.DBGateway</code> Method Summary	
Return Value	Method
boolean	<code>addDBConnection</code> (DBConnection connection) Adds a database connection to this gateway. Any connection added via this method will be queried by the gateway when users request data.
DBConnection[]	<code>getDBConnections</code> () Returns the list of database connections that this gateway is managing.
Distinct Property Values	<code>getDistinctProperties</code> (Property[] properties) Returns list(s) of distinct property values for the given properties.

Environment Experiment Group	getEnvironmentExperimentGroup (EnvironmentExperimentFilter filter) Gets an environment experiment group based on the given environment experiment filter.
Genotype Experiment Group	getGenotypeExperimentGroup (GenotypeExperimentFilter filter) Gets a genotype experiment group based on the given genotype experiment filter.
Genotype Group	getGenotypeGroup (GenotypeExperimentGroup experimentGroup, TaxonGroup taxonGroup) Gets the genotypes for all combinations of genotype experiments and taxa defined by the respective groups.
Locality Group	getLocalityGroup(LocalityFilter filter) Gets a locality group based on the given locality filter.
Locus Group	getLocusGroup (LocusFilter filter) Gets a locus group based on the given locus filter.
Phenotype Ontology Group	getPhenotypeOntologyGroup(PhenotypeOntologyFilter filter) Get a phenotype ontology group based on the given pheotype ontology filter.
Phenotype Group	getPhenotypeGroup (EnvironmentExperimentGroup experimentGroup, TaxonGroup taxonGroup, PhenotypeOntologyGroup ontologyGroup) Gets the phenotypes for all combinations of environment experiments, taxa, and phenotype ontologys defined by the respective groups.
Taxon Group	getTaxonGroup (TaxonFilter filter) Gets a taxon group based on the given taxon filter.
boolean	removeDBConnection (DBConnection connection) Removes the specified database connection from this gateway.

Figure 4: List of methods supported by the GDPC gateway

Code Sample

The following code sample shows how to create a database connection, add it to the GDPC Gateway, and retrieve the sequences for taxon B73 and locus adh1. This code would then print out the entire sequence.

```

/*
 * This class demonstrates the use of GDPC:
 * The Genomic Diversity and Phenotype Connection
 */

package gov.usda.gdpc.panzea;

import gov.usda.gdpc.*;

public class TestGDPCGateway {

    /** The GDPC gateway */
    private final DBGateway myDBGateway = DefaultDBGateway.getInstance();

    /** Creates a new instance of TestGDPCGateway */
    public TestGDPCGateway() {
    }

    /** main */
    public static void main(String[] args) {

        // create test object

```

```

TestGDPCGateway test = new TestGDPCGateway();

// make data source connections to gateway
test.makeConnections();

// Get sequences for taxon B73 and locus adh1
GenotypeTable sequences = test.getSequence("B73", "adh1");

// Print out table
System.out.print(sequences.toString());
}

/** This creates and adds a database connection
 * (Panzea database) to the GDPC gateway. */
public void makeConnections() {

    // oracle jdbc driver
    // database url of the form jdbc:subprotocol:subname
    // database user on whose behalf the connection is being made
    // the user's password
    DBConnection connection = new PanzeaDBConnection(
        "oracle.jdbc.driver.OracleDriver",
        "Panzea database",
        "oracle.jdbc.driver.OracleDriver",
        "jdbc:oracle:thin:@gaut.statgen.ncsu.edu:1521:panzea",
        "????", "????");

    // add connection to gateway
    myDBGateway.addDBConnection(connection);

}

/** Returns the sequences for specified taxon and locus. */
public GenotypeTable getSequence(String taxon, String locus) {

    // Create taxon filter specifying taxon's name and retrieve group using filter.
    TaxonFilter taxonFilter = new TaxonFilter();
    taxonFilter.addValue(new FilterSingleValue(TaxonProperty.NAME, taxon));
    TaxonGroup taxonGroup = myDBGateway.getTaxonGroup(taxonFilter);

    // Create genotype experiment filter specifying its name and
    // retrieve genotype experiment group using that filter.
    GenotypeExperimentFilter expFilter = new GenotypeExperimentFilter();
    expFilter.addValue(new FilterSingleValue(GenotypeExperimentProperty.LOCUS, locus));
    GenotypeExperimentGroup expGroup = myDBGateway.getGenotypeExperimentGroup(expFilter);

    // Get genotype group
    GenotypeGroup group = myDBGateway.getGenotypeGroup(expGroup, taxonGroup);

    // create the genotype table
    GenotypeTable table = GenotypeTable.getInstance(group);

    // Return table.
    return table;
}
}
}

```

Summary

The use of GDPC has three primary advantages over conventional software design. First, it integrates genomic and phenotypic data to promote the development of software tools that analyze that information. Secondly, it supports the acquisition

of data from multiple data sources allowing analysis of larger, more complete, data sets. Thirdly, analysis tools require no additional software development to use new data sources. The implementation of GDPC will integrate and simplify access to genomic and phenotypic data.