PubSearch and PubFetch: A Simple Management System for Semiautomated Retrieval and Annotation of Biological Information from the Literature

Database curators and biology researchers must keep track of the literature concerning their genes of interest. Such investigators have an interest in obtaining and using more sophisticated tools for this purpose than spreadsheets and laboratory notebooks. PubSearch and PubFetch comprise a literature curation system that integrates and stores literature and gene information into a single relational database. The PubSearch system provides curators with a central Web application interface to support querying and editing publication articles, genes, and keywords such as the Gene Ontology (GO) terms. It also facilitates annotating genes with keywords and article references, and allows controlled access to protected PDF documents. The PubFetch system supports PubSearch by providing a general interface to search and retrieve publications from online literature sources. An overview of the PubSearch workflow is shown in Figure 9.7.1.

In this unit, a set of protocols is provided for populating and using PubSearch and PubFetch. Basic Protocol 1 describes, in a step-by-step fashion, how to populate articles, genes, keywords, and annotations in standard format into the database. The Alternate Protocol is a procedure for populating articles using a Web interface, GO terms from the GO database, and annotations in a tab-delimited format. Basic Protocol 2 describes how to index the articles for full-text searching. Basic Protocol 3 shows how to use PubSearch to search for genes, articles, keywords, and annotations using a Web browser. Basic Protocol 4 describes ways to update and add data one item at a time using the Web browser. Basic Protocol 5 describes how to annotate genes using GO and other controlled

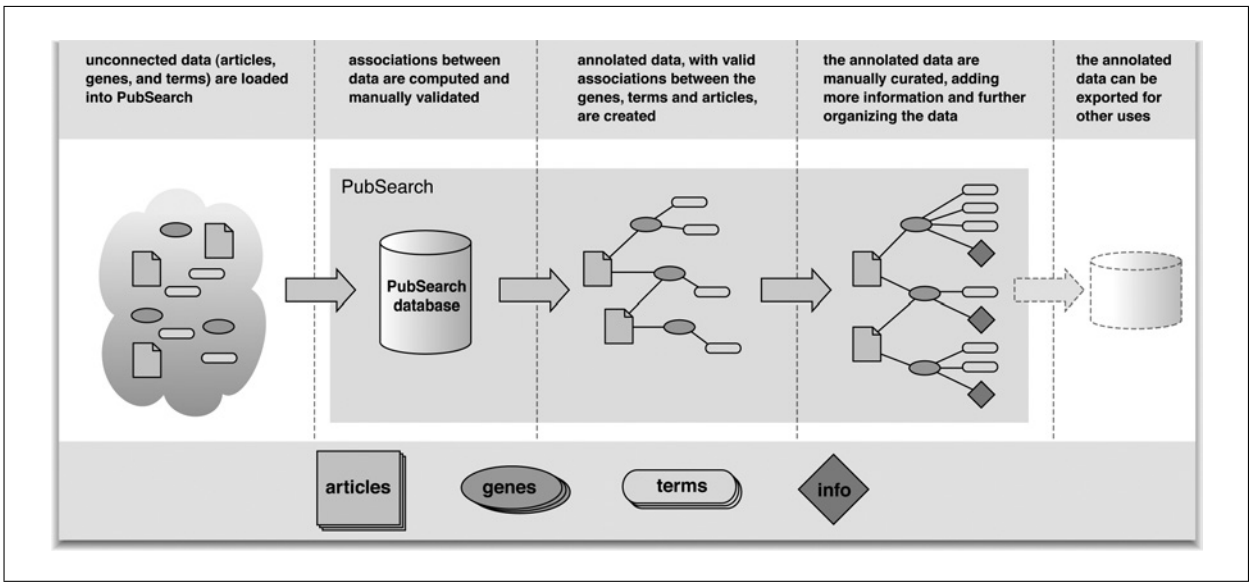

Figure 9.7.1 An overview of the PubSearch workflow illustrating how published articles, genes, and key biological terms are brought together and integrated within PubSearch. Manual review and annotation of these data creates an annotated database of literature, genes, and related information that can be used within PubSearch alone or exported to other applications.

Contributed by Danny Yoo, Iris Xu, Vijay Narayanasamy, Tanya Z. Berardini, Simon Twigger, and Seung Yon Rhee

Current Protocols in Bioinformatics (2006) 9.7.1-9.7.27

Table 9.7.1 Guide to Conventions Used for Naming Directories

Name	Explanation
`${TMPDIR}`	A temporary scratch directory
`${WebAPP}`	The Web application directory where the servlet engine looks for installed Web applications. Apache Tomcat uses a variation of `jakarta-tomcat/Webapps`.
`${PUBHOME}`	The root directory where PubSearch will be installed. This will be `WebAPP/PubSearch` for typical installations.
`${DOMAINNAME}`	A placeholder for the domain name of the PubSearch hosting machine
`${PASSWORD}`	A placeholder for the database password used to access the "pubdb" database in MySQL
`${TOMCATBIN}`	Stands for the binary directory for Apache Tomcat. Typically, this is `apache-tomcat-[version]/bin`.

vocabulary terms. Basic Protocol 6 shows how to generate and load GO annotations from proteins that have been annotated with InterPro (*http://www.ebi.ac.uk/interpro/*) domains. The unit also provides support protocols for installing PubSearch and PubFetch. Support Protocol 1 describes how to install PubSearch; Support Protocol 2 describes how to install and run PubFetch as a stand-alone software application. Finally, the Commentary provides background information and related resources and includes information on troubleshooting potential problems and future directions of software development. More information about this project, including schema and software documentation, can be found online at *http://www.pubsearch.org*.

Conventions for naming directories that are referred to in this unit are given in Table 9.7.1. Unix conventions (also see *APPENDIX 1C*) for navigating through directories will be assumed. Unix commands will be prefixed with > to indicate the shell prompt. When commands are to be sent to the MySQL console, these commands will be prefixed with the prompt `mysql>`.

BASIC PROTOCOL 1

POPULATING PubSearch

In this protocol, an instance of the PubSearch curation system will be populated. A description of how to load articles, genes, and keywords in XML format, and annotations in GO annotation format, is provided. In addition, the protocol describes how to generate indices of the articles using the gene and keyword names. Upon completion of this protocol, a functional installation of the system will have genes, articles, keywords, and gene annotations that can be edited and queried from the Web interface.

Necessary Resources

Hardware

PubSearch has been tested on the following systems:
Intel Xeon, 866 MHz, 2 CPUs (512 Mb RAM)
Apple PowerBook, 1 GHz (1 Gb RAM)
Dedicated hard drive space required for indexing full text

Software

PubSearch has been tested on the following operating systems:
Red Hat Enterprise Linux 3
Mac OS 10.3
PubSearch has not yet been tested on the Windows platform.

Installation of the following list of programs is a prerequisite for installing and running PubSearch:

Java JDK 1.4 or higher (*http://java.sun.com/j2se/1.4/*)
Any Java Servlet platform, such as Apache's Tomcat (*http://jakarta.apache.org/tomcat/*)
MySQL 4 (*http://mysql.com*)
In order to have transactional support, MySQL should be configured to support the INNODB table type. INNODB is described online at *http://dev.mysql.com/doc/mysql/en/innodb-overview.html*
Perl 5.8 (*http://www.cpan.org/src/README.html*)
Python 2.3 (*http://www.python.org/2.3/*)
Perl and Python are used as scripting languages to administer many of the subsystems, including cron jobs and other maintenance

The following are software requirements for performing this protocol:

GO-DB-PERL and GO-PERL Perl bindings for loading GO terms. These bindings are used to process data files that have been released by the Gene Ontology. GO-PERL and GO-DB-PERL are available as part of the standard set of development tools from the Gene Ontology's SourceForge repository at *http://sourceforge.net/projects/geneontology*.

The "go-dev" download package linked from the SourceForge page contains both Perl modules, and instructions on installing them are included in the package. PubSearch has been tested against the `go-dev-20040609-amigo2.0` release, and that version is strongly recommended.

XPDF tools from *http://www.foolabs.com/xpdf/download.html*. The current version at the time of writing is `xpdf-3.00pl3-linux.tar.gz`. XPDF is a separate set of tools to parse PDF files. XPDF includes the "pdftotext" utility, which is used to extract full text from a PDF file for searching and indexing. The XPDF source and binary distributions include instructions on how to install the XPDF toolset.

Files

A sample dataset is provided on the PubSearch Web site for demonstration purposes. The archived demonstration database can be downloaded from *http://pubsearch.org/releases/pubsearch-database-newest.sql.bz2* Note that the file is compressed (using bzip2) to conserve space.

1. Install PubSearch and PubFetch as described in Support Protocols 1 and 2.
2. Download the sample dataset from the PubSearch Web site at *http://pubsearch.org/releases/pubsearch-database-newest.sql.bz2* using the following command:

```
>wget http://pubsearch.org/releases/pubsearch-database-newest.sql.bz2
```

3. Load the dataset into the database using the command:

```
>bzcat pubsearch-database-newest.sql.bz2 | mysql pubdb
```

Bulk load articles from XML

As an alternative to bulk loading using PubFetch, articles can be added in bulk through the command-line interface. The bulk input format is a subset of the document type definition (DTD) file `pubmed_020114.dtd` (*http://www.ncbi.nlm.nih.gov/entrez/query/DTD/pubmed_060101.dtd*) used by PubMed as part of the PubMedArticleSet (*http://eutils.ncbi.nlm.nih.gov/entrez/query/DTD/index.html*). This subset includes the following

attributes: PMID, MedlinePgn, Volume, Issue, MedlineID, ArticleTitle, AbstractText, AbstractNumber, ISSN, PubSourceId, PubSourceName, and PublicationType. Note that the DTD from PubMed may change in the future. The authors will update PubSearch to work with the latest version of the DTD. Users are advised to check for the latest version of the DTD when installing PubSearch.

4. Construct a data file. This file must conform to the DTD definition described above. A sample set of articles in the PubSearch Article XML format is included in `${PUBHOME}/data/test/sample_pubmed_articles.dtd`. NCBI PubMed queries can also produce appropriate XML output for the bulk loader.

   The XML parser that is bundled with PubSearch is nonvalidating, because the selected PubMed subset that is used in the bulk article loader is itself not compliant with the PubMed DTD. In future releases of this software, the bulk article loader will define its own custom DTD format and use a validating parser for greater safety.

5. Run article importing script on the data file. Assuming that the input file is called ARTICLE.XML, go into the PubSearch home directory:

   ```
   >cd ${PUBHOME}
   ```

   Execute the article bulk loading command:

   ```
   >bin/bulk_load_articles.pl -pubsources ARTICLE.XML
   ```

   The `-pubsources` flag tells the loader to add new Publication Sources such as Journals, as necessary. If the flag is not given, then articles that refer to a nonexistent pub source will be ignored. After the command is executed, the set of articles will be entered into the PubSearch database, along with any necessary publication journals as PubSource entries.

Bulk load Term Ontologies with XML

PubSearch provides two ways of loading the Gene Ontology terms into its internal databases: loading straight from an XML data file, or from a MySQL dump of the GO database. Both methods are documented below. GO provides two types of XML dump for the ontology. PubSearch provides a loader for the GO-RDF format defined at *http://www.godatabase.org/dev/database/archive/latest/go_yyyymm-rdf.dtd.gz*. To get the latest DTD, replace "*yyyymm*" with the year and month, e.g., "*....go_200601-rdf.dtd.gz*."

6. Download the GO RDF XML file at *http://www.godatabase.org/dev/database/archive/latest/go_200503-termdb.obo-xml.gz*, using the following command:

   ```
   >wget
   http://www.godatabase.org/dev/database/archive/
   latest/go_yyyymm-termdb.obo-xml.gz
   ```

   For example, "*...go_200601-termdb.obo-xml.gz*."

7. Decompress the file:

   ```
   >gunzip go_yyyymm-termdb.obo-xml.gz
   ```

   When the year and month are substituted, this will generate a file named `go_200503-termdb.obo-xml` *in the same directory.*

8. Run the bulk term loader over the decompressed file:

```
>$ {PUBHOME}/bin/bulk_load_terms.pl go_yyyymm-
termdb.obo-xml
```

This should load all the terms and the term-to-term ontology structure into the PubSearch database.

9. Restart PubSearch by restarting the servlet container. The system keeps a cached view of the ontology graph that is updated every hour. A restart forces the system to refresh its view of the ontology:

```
>cd ${TOMCATBIN}

>bin/shutdown.sh

>bin/startup.sh
```

Bulk load genes from xml

The PubSearch software provides a simple data format for bulk loading gene objects into the system. This format is defined in:

```
${PUBHOME}/data/dtds/bulk_gene.dtd
```

10. Prepare a GeneXML file in the format described in the DTD. An example file in this format can be found in:

```
${PUBHOME}/data/test/one_gene.xml
```

Once a file has been prepared in this format, it can be run through the provided `bulk_gene_loader.pl script`.

11. Load the data file into PubSearch. For example, to load the `one_gene.xml` file:

```
>${PUBHOME}/bin/bulk_load_genes.pl

>${PUBHOME}/data/test/one_gene.xml
```

Bulk load GO gene annotations

There are currently two types of file formats for loading annotations: user-submission annotation file format, which is used by TAIR (*UNIT 1.11*), and GO annotation file format, which is used by all databases contributing GO annotations to the GO Web site. More information about the user-submission file format is found online at *http://arabidopsis.org/info/functional_annotation.submission.jsp*. More information about the GO annotation file format can be found online at *http://www.geneontology.org/GO.annotation.shtml#file*.

12. Download a GO gene association file from the GO Web site (*http://www.geneontology.org/GO.current.annotations.shtml*). For example, to retrieve *Arabidopsis thaliana* annotation data from TAIR, download the file from *ftp://ftp.geneontology.org/pub/go/gene-associations/gene_association.tair.gz*.

13. Save file in the `${PUBHOME}/maint/tigrannotation/data` directory and unzip that file by executing the following commands:

```
>cd ${PUBHOME}/maint/tigrannotation/data

>wget ftp://ftp.geneontology.org/pub/go/gene-
associations/gene_association.tigr_Athaliana.gz
```

```
>gunzip gene_association.tigr_Athaliana.gz
```

This generates a file called `gene_association.tigr_Athaliana` *in the same directory.*

14. Run `generateAnnotationFromTigrFile.pl` on this new file to load GO annotation file by executing the following commands from the `${PUBHOME}` directory:

```
>cd maint/tigrannotation

>perl generateAnnotationFromTigrFile.pl -D
database_name
```

Generating hits

Hits are associations between terms and articles that can be generated by exact-term matching. The program `generate_hits.pl` will do a bulk search for database terms within the titles and abstracts of all articles.

15. Execute `generate_hits.pl`:

```
>${PUBHOME}/bin/generate_hits.pl
```

This step may take several minutes, depending on how many terms exist in the database.

SUPPORT PROTOCOL 1

INSTALLING PubSearch

The PubSearch Web application is one of the main components of the literature curation system. This protocol describes how to install the software on a clean machine, configure its connection to a relational database, and add initial users to the system.

Necessary Resources

Hardware

PubSearch has been tested on the following systems:

Intel Xeon, 866 MHz, 2 CPUs (512 Mb RAM)
Apple PowerBook, 1 GHz (1 Gb RAM)

Dedicated hard drive space required for indexing full text

Software

PubSearch has been tested on the following operating systems:

Red Hat Enterprise Linux 3
Mac OS 10.3

PubSearch has not yet been tested on the Windows platform.

Installation of the following list of programs is a prerequisite for installing and running PubSearch:

Java JDK 1.4 or higher (*http://java.sun.com/j2se/1.4/*)
Any Java Servlet platform, such as Apache's Tomcat (*http://jakarta.apache.org/tomcat/*)
MySQL 4 (*http://mysql.com*)
In order to have transactional support, MySQL should be configured to support the INNODB table type. INNODB is described online at *http://dev.mysql.com/doc/mysql/en/innodb-overview.html*
Perl 5.8 (*http://www.cpan.org/src/README.html*)
Python 2.3 (*http://www.python.org/2.3/*)
Perl and Python are used as scripting languages to administer many of the subsystems, including cron jobs and other maintenance

The following are software requirements for performing this protocol:

GO-DB-PERL and GO-PERL Perl bindings for loading GO terms. These bindings are used to process data files that have been released by the Gene Ontology. GO-PERL and GO-DB-PERL are available as part of the standard set of development tools from the Gene Ontology's SourceForge repository at *http://sourceforge.net/projects/geneontology*.

The "go-dev" download package linked from the SourceForge page contains both Perl modules, and instructions on installing them are included in the package. PubSearch has been tested against the `go-dev-20040609-amigo2.0` release, and that version is strongly recommended.

XPDF tools from *http://www.foolabs.com/xpdf/download.html*. The current version at the time of writing is `xpdf-3.00pl3-linux.tar.gz`. XPDF is a separate set of tools to parse PDF files. XPDF includes the "pdftotext" utility, which is used to extract full text from a PDF file for searching and indexing. The XPDF source and binary distributions include instructions on how to install the XPDF toolset.

1. Download the binary distribution of PubSearch from *http://pubsearch.org/releases/pubsearch-newest.tar.gz* into `${TMPDIR}`. Most versions of UNIX have a command called `wget` that can be used to retrieve the contents of Web URLs from a shell prompt:

   ```
   >cd ${TMPDIR}

   >wget http://pubsearch.org/releases/pubsearch-newest.
   tar.gz
   ```

   See Table 9.7.1 for explanations of the directory names used here.

2. Untar this file from within the `${WebAPP}` directory:

   ```
   >cd ${WebAPP}

   >tar xzvf ${TMPDIR}/PubSearch-newest.tar.gz
   ```

   A new subdirectory called `PubSearch` will be produced underneath the `${WebAPP}` directory.

3. Change the current working directory to `${WebAPP}/pubsearch`:

   ```
   >cd pubsearch
   ```

4. Initialize the PubSearch Database. Use MySQL's administrative tool `mysqladmin` to create a new database called `pubdb`:

   ```
   >mysqladmin createdb pubdb
   ```

   An empty database called `pubdb` will be created.

5. Load schema structure into the pubdb database. This structure is defined in the files `schema.mysql` and `schema-support.mysql`:

   ```
   >mysql pubdb < data/schema.mysql

   >mysql pubdb < data/schema-support.mysql
   ```

6. Create a separate MySQL user account for PubSearch. It is strongly recommended that a separate MySQL database user be used to connect to the database, with a separate password. This can be done through the MySQL console:

```
>mysql pubdb

mysql> grant all on pubdb.* to pubuser@${DOMAINNAME}
identified by "${PASSWORD}";;

mysql> exit
```

7. Configure PubSearch's global preference file. The PubSearch Web application maintains its configuration settings in the following file:

```
${PUBHOME}/Web-INF/classes/pub/config/program.
properties
```

   This file must be edited so that the system knows what resources it can use (e.g., database settings, index directories, PDF repositories):

```
>emacs ${PUBHOME}/Web-INF/classes/pub/config/program.
properties
```

   The most relevant of the properties are:

```
pub.database_username

pub.database_password

pub.database_connection_string

pub.aux_data_dir
```

   which should be adjusted to appropriate values. The first three property values define the values necessary to connect to the MySQL database. The last value, `pub.aux_data_dir`*, defines an auxiliary data directory that is used to store indices for the full-text search engine as well as temporary scratch space. Full-text indices typically take up 30% of the full text.*

8. Test the Database Connection. To verify that the `program.properties` file has been successfully modified, execute the following command from within the ${PUBHOME} directory:

```
>bin/test_database_connection.pl
```

   If the PubSearch system can successfully connect to the `pubdb` *database, then the message* `Database connection looks good` *should be displayed. Otherwise, correct the* `program.properties` *file and repeat this step until PubSearch can connect to the database.*

9. Notify the Servlet Engine of the PubSearch Web application. Apache Tomcat rescans the Web application directory WebAPP on startup. If Tomcat is already running, shut it down, and then start it again.

```
>${TOMCATBIN}/shutdown.sh

>${TOMCATBIN}/startup.sh
```

   Otherwise, start Tomcat:

```
>${TOMCATBIN}/startup.sh
```

The PubSearch Web application should be running at this point. Under default settings, the page will show up under the URL *http://${DOMAINNAME}:8080/pubsearch/*.

10. View PubSearch on a Web browser to see that the application is active.

Adding curators to the system

Only curator users are allowed to make changes to PubSearch. Curators can be added through a command-line interface. Also, if a PDF repository has been constructed, curators have access to those protected links.

11. Execute the `add_curator.pl` program. Change directory to `${PUBHOME}` and run the `command bin/add_curator.pl`.

```
>cd ${PUBHOME}

>bin/add_curator.pl
```

Prompts from the program will ask for username and initial password.

Adding regular users to the system

Regular users are allowed to inspect and query the PubSearch system, but are not allowed to make changes or to view PDF files. Regular users can be added through a command line interface, similarly to curators. The command `add_user.pl` is used.

12. Execute the `add_user.pl` program. Change directory to `${PUBHOME}` and run the `command bin/add_user.pl`:

```
>cd ${PUBHOME}

>bin/add_user.pl
```

Prompts from the program will ask for username and initial password.

Setting up administrative cron jobs for periodic maintenance

On Unix systems, a cron job can be initialized to perform regular tasks on the PubSearch system. Such tasks might include running the full text indexing, generating hits, and exporting bulk output out of PubSearch.

An example cronable script is included in `${PUBHOME}/maint/cron/pub_daily.sh`.

13. To schedule `pub_daily.sh` on a regular basis, execute:

```
>crontab -e
```

This will execute the crontab editor. Add the following entry into the crontab:

```
0 0 * * * ${PUBHOME}/maint/cron/pub_daily.sh
```

which will schedule the execution of `pub_daily.sh` every midnight.

14. Create a cronjob entry for each maintenance script desired.

INSTALLING PubFetch FOR USE OUTSIDE OF PubSearch

SUPPORT PROTOCOL 2

PubFetch is available as Java Archive (JAR) library that can be used by any other Java application. PubFetch currently contains adaptors to allow the standardized retrieval of literature references from PubMed at NCBI (National Center for Biotechnology Information) via the eUtils interface and from Agricola (National Agricultural Library) via their Machine Readable Cataloging (MARC)–based system. PubFetch is part of the standard

PubSearch release, so no further action is required to use PubFetch with PubSearch. The protocol below is employed to obtain and use PubFetch as an independent software library.

Necessary Resources

Hardware

Any computer that runs Java and has an Internet connection

Software

Java SDK available at *http:www.java.sun.com*

Xerces XML Parser (*http://xml.apache.org/*): `xercesImpl.jar` can be found in the `lib` folder of PubFetch releases

MARC4J (*http://marc4j.tigris.org/*): provides an easy to use Application Programming Interface (API) for working with MARC records in Java (`marc4j.jar` can be found in the `lib` folder of PubFetch releases

Log4j (*http://logging.apache.org/log4j/*): a logging package for Java (`log4j-1.2.8.jar`) can be found in the `lib` folder of PubFetch releases). The logging behavior can be controlled by editing the configuration file (see below)

Apache Ant: a common build utility for Java

Files

Log4J configuration file (`log_configuration.properties` can be found in the `data` folder of the release)

Entrez Journal List file containing journals in PubMed and the molecular biology databases (`J_Entrez.txt.gz` can be found in the `data` folder of the release)

1. Download the latest PubFetch binary or source files from GMOD project (Generic Model Organism Database) SourceForge site (*http://sourceforge.net/project/showfiles.php?group_id=27707*).

2. Unarchive the files using appropriate software for the operating system. For example use WinZip on the Windows operating system. On Unix and Macintosh OS X operating systems use the command:

   ```
   >tar -zxvf file-name
   ```

3. To use PubFetch as an API for fetching records from Agricola and/or PubMed in Java applications, it is first necessary to install the `pubfetch.jar` final in `CLASSPATH`. `PubFetch.jar` is provided as part of the binary release or can be built from scratch from the source code release by running `ant` with the `jar` target:

   ```
   >cd ${PUBFETCH}

   >ant jar
   ```

4. Add other essential `jar` files to `CLASSPATH` such as XML Parser (`xercesImpl.jar`), MARC4J (`marc4j.jar`) and Log4J (`log4j-1.2.8.jar`). These `jar` files can be found in the `lib` folder of the release. They can also be downloaded from the appropriate software application's Web site.

5. Once installed, PubFetch provides the following features to Java clients:

   Common format: A common output format is implemented so that downstream applications can easily use the retrieved literature. PubFetch retrieves articles in

MEDLINE Display Format, which is also one of the standard formats used by the GMOD (Generic Model Organism Database) project. PubFetch converts MARC Record Format to MEDLINE Display Format, in the case of Agricola, by replacing MARC tags with corresponding MEDLINE tag. For example MARC tag `245` is MEDLINE tag `TI`, which corresponds to the Title of the article.

Full text URL: PubFetch can return the URL for the full text of each document if the full text link is available in PubMed LinkOut, or PubMed Central, or if a CrossRef/DOI (Digital Object Identifier) is provided. This can be used for the subsequent download of the full text PDF for full text indexing or printing.

Duplicate filtering: When searching multiple databases, the potential exists for records to be present in both databases, resulting in a duplicate record. PubFetch provides a duplicate filtering algorithm based upon common attributes such as the Title, ISSN number, and starting page, which can be used to identify and then remove duplicate records. Cross-references to the duplicated record are maintained, so links can be created to both sources.

Explicit examples for using PubFetch as a stand-alone tool are provided in the `README` *files distributed with the release. These illustrate how to search and retrieve documents from a literature repository and also acquire URLs for full-text articles, where available.*

For more in depth explanations of the PubFetch API, `javadoc` *files are available for the source code. The can be found in the* `htdocs/javadoc` *folder in the binary release, or run* `ant-doc` *in the source release to produce the* `javadoc` *from scratch.*

OTHER WAYS TO POPULATE PubSearch

ALTERNATE PROTOCOL

As an alternative to Basic Protocol 1, this protocol explores other ways of loading data into the PubSearch database. Procedures are described for loading articles from Agricola and Medline using a Web interface, for loading an entire ontology directly from the Gene Ontology database, and for processing a bulk set of annotations from a tab-delimited input file.

Necessary Resources

See Basic Protocol 1 and Support Protocol 1

Bulk loading articles using the Web browser

Users can query online publication databases and load the results into the PubSearch database. This functionality uses the PubFetch software in the background. The user must be logged in to perform this task.

1. From the PubSearch Web interface (see Basic Protocol 1), go to Add Articles in Bulk on the Add toolbar.
2. Select the type of data source: Agricola or PubMed.
3. Enter published date range in the format `YYYY/MM/DD` (e.g., `2005/01/08`) of the articles to be retrieved.
4. Enter the keyword to be used in the input box following Search For; this will limit the search to those articles which have this keyword in their titles or abstracts.
5. Click the FetchToPub button.

After the user clicks the FetchToPub button, the underlying PubFetch application will fetch the articles the user wants from the user-specified data source and add the articles into PubSearch after filtering out the duplicates. If the publication source of the article (e.g., journal) does not exist in the database, the application adds the journal automatically.

6. Confirm/Check the inserted articles. After the software fetches and inserts the articles, a summary page will be displayed with three sections, as follows:

   a. Number of new articles that were added to the database. For each added article, the page has a link to the article detail page, where the user can check the detail of that article and add/modify article information.

   b. Number of new journals that were added. This will also link to the journal detail page, where the user can modify journal information. Journal entries are occasionally duplicated with some existing journals in database due to slight differences in naming conventions in the source data. If this is the case, the user can merge the two entries by "obsoleting" (i.e., by marking the entry obsolete in the database, one is effectively "deleting" it without removing it from the database) the new journal and replacing it with the other journal. This step will also associate any articles that were linked to the old journal with the new one.

   c. Number of articles that were skipped because they were duplicates of existing entries. For the duplicated articles, the article entries in the database and entries from the data source are displayed side by side so the user can compare them and modify/add information to the article entries in database.

Bulk loading term ontologies from a Gene Ontology database

7. Load a Gene Ontology database.

   The Gene Ontology defines an SQL schema for storing terms and the associated relationships. These can be found on the GO Web site at http://geneontology.org. PubSearch contains a set of loading scripts to read the native Gene Ontology databases and to import the terms and ontologies into a local database.

8. Download a suitable ontology dump file. For example, the GO consortium publishes a dump of its term database once a month. Use the following command:

```
>wget
http://archive.godatabase.org/latest/go_yyyymm-termdb-
tables.tar.gz
```

   The string "yyyymm" is to be replaced with the latest year and month in the file name, e.g., for January, 2006, the file name would be `go_200601-termdb-tables.tar.gz`.

9. Restore the MySQL dump into the local database. For the purposes of this guide, assume the database is named `gene_ontology`.

```
>mysqladmin create gene_ontology

>tar xzvf go_yyyymm-termdb-tables.tar.gz

>cd go_yyyymm-termdb-tables

>cat *.sql | mysql gene_ontology

>mysqlimport -L gene_ontology *.txt
```

10. Run Gene Ontology loaders. Once the Gene Ontology MySQL database is created, the following steps will add terms and term-to-term ontology relationships into the PubSearch system.

```
>cd ${PUBHOME}/maint/gene_ontology

>perl add_goterm_to_pubterm.pl

>python load_term2term.py
```

```
>cd ${PUBHOME}

>perl bin/import_go_term_synonyms.pl gene_ontology
```

Loading user-submitted annotations

The bulk annotation loader provides a way to load gene-to-term associations with article references. The loader takes a tab-delimited file as input, and the format that the file reads is documented in `${PUBHOME}/maint/bulk_annotation_loader/format.txt`. An example file is included in `${PUBHOME}/maint/bulk_annotation_loader/sample/example.txt`.

11. To load the annotations, execute the `bulk_load_annotations.pl` script on the data file:

```
>cd ${PUBHOME}/maint/bulk_annotation_loader

>perl bulk_load_annotations.pl sample/example.txt
```

SETTING UP A PDF REPOSITORY FOR FULL-TEXT INDEXING

BASIC PROTOCOL 2

Articles in the PubSearch database often have Adobe Portable Document Format (PDF) files associated with them. These documents can be processed by PubSearch's full-text index system, thereby enabling a powerful full-text search engine. However, access to these PDFs may need to be restricted due to licensing issues. PubSearch provides a rudimentary scheme for restricting PDF access to curators only. The protocol below describes how to set aside a protected PDF repository for PubSearch and schedule regular maintenance of a full-text index for PDF documents.

Necessary Resources

See Basic Protocol 1 and Support Protocol 1

Setting up A PDF repository for full-text indexing

1. Set aside a directory for documents. This directory can be placed in any file system with sufficient storage. Once a directory has been made, PubSearch must be configured to use that directory, using the `program.properties` configuration file described in the PubSearch installation section.
2. Edit the `program.properties` file in the `${PUBHOME}/Web-INF/classes/pub/config` directory. Change the property `pub.pdf_document_base` to the PDF repository directory.
3. Once a PDF repository directory is configured, PDFs can be copied into the repository. The filename of each PDF must be named to match the article id within the PubSearch system.

   In a future revision of the system, a Web interface for adding PDFs will be implemented.

4. Once a PDF has been copied into the PDF repository directory, a FullText URL will be available from the article detail page. Only logged-in curators with the "can_access_pdfs" capability will be allowed to access the URL. At the current time, SQL update statements issued from the MySQL client are required to change this property.

Generating full-text indices

5. Generate text files for article PDFs by executing the following command from the `${PUBHOME}` directory:

```
$cd maint/perl

$perl extract_full_text.pl
```

PubSearch sets up a cron job for this task (see Support Protocol 1).

6. Set up a PDF repository for storing article full text, and specify this resource in `program.properties` file in the `${PUBHOME}` directory. The following illustrates the settings from PubSearch:

```
## Where are the PDFs located on the system?

pub.pdf_document_base = /opt2/pub_documents
```

7. Set aside a directory for storing the indices that the Lucene search engine will generate. The following lines are the configuration from PubSearch's `program.properties` file:

```
## Directory where Pub can store auxiliary data
(indices)

pub.aux_data_dir = /opt2/PubSearch/var
```

8. Add the following lines to the `program.properties` file (see Basic Protocol 1). The settings below indicate to the system which specific content types are used as document collections. Classes are separated from one another by a comma. PubSearch comes with three standard document type index classes listed below:

```
pub.lucene_document_iterators = pub.db.search.
LuceneTermIterator,
pub.db.search.LuceneGeneIterator,
pub.db.search.LuceneArticleIterator
```

9. To generate the indices for full text, execute the following commands under the `${PUBHOME}` directory:

```
>cd ${PUBHOME}

>bin/index_full_text.pl
```

BASIC PROTOCOL 3

USING PubSearch TO SEARCH DATA

Once data have been warehoused into PubSearch, the data can be queried from the Web interface. Procedures are described below for logging into the system as a privileged curator and performing simple and complex queries. The intended users for this protocol are biologists, database curators, computational biologists, bioinformaticians, or bench scientists who need to manage a large amounts of literature and gene data. This protocol uses the following convention for the URL where PubSearch can be accessed: `http://${DOMAINNAME}:8080/pubsearch`.

Necessary Resources

See Basic Protocol 1 and Support Protocol 1

Navigation and home page

The page header displayed on top of every page (Fig. 9.7.2) illustrates all of the functionalities available in PubSearch. In addition to logging in, searching, browsing, and adding data, users can access curation and usage guides, as well as submit bugs. The home page displays a short description of the software, recent changes, and database statistics.

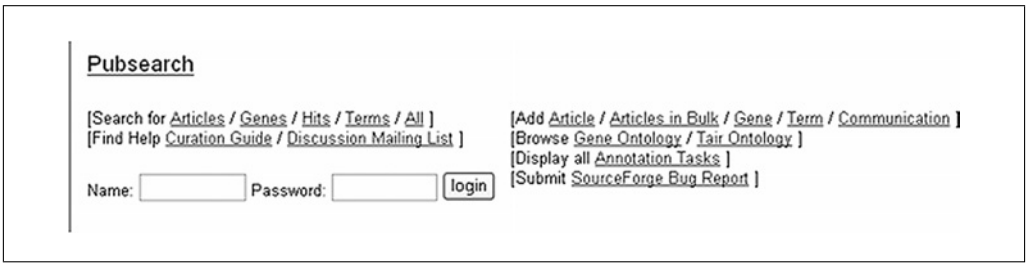

Figure 9.7.2 A screenshot of the Navigation toolbar of PubSearch Web user interface. It lists the different types of user functions, links to usage guide, and text boxes for logging in.

Logging into the database

Users can log in from any page in PubSearch. Logging in is required for updating, inserting, and viewing full-text articles. Searching and viewing results do not require logging in.

1. In the page header, type the user name in the Name input box, and enter the password in the Password input box.

Searching the database

It is not necessary to be logged in to search the database. There are two types of search interfaces available, a simple search of all datatypes, and advanced searches for each of the major datatypes.

Simple searching

2. From the page header, click the All hyperlink in the "Search for" section or go directly to the simple search interface from the URL (*http://${DOMAINNAME}:8080/pubsearch/Search?middle_page=ALL*).

3. Type the search string (both words and phrases are accepted) in the text input box next to Query:, then click the Submit button. For example, try typing "`water channel`" (including the quotes, which searches for the phrase).

   This search uses Lucene's full-text search algorithm. For a complete list of available query string options and formats, refer to http://lucene.apache.org/java/docs/queryparsersyntax.html.

4. Results are ordered in terms of how they score in terms of "density" in the Lucene search engine (frequency of term occurrence per document size; Fig. 9.7.3). In the example query, from TAIR's PubSearch instance, 125 results are retrieved, which include genes, controlled vocabulary terms, and articles. Clicking on the name/title of the gene, term, or article leads to the detail page of the data object.

   The simple search function searches all of the data fields. It is experimental and is not supported.

Advanced searching

There are four types of data—articles, genes, terms, and hits—that can be searched by using more parameters. These data types are listed in the page header. The user interface and usage of the search and result pages for all of these data types are similar. Therefore, only the hit search is described here. "Hit search" is used to find papers that are associated with a gene (or other types of terms) of interest. Both articles and terms can be restricted for finding matches between papers and terms.

5. Go to the Hit search page by clicking on the Hit hyperlink in the Search section of the page header (Fig. 9.7.4).

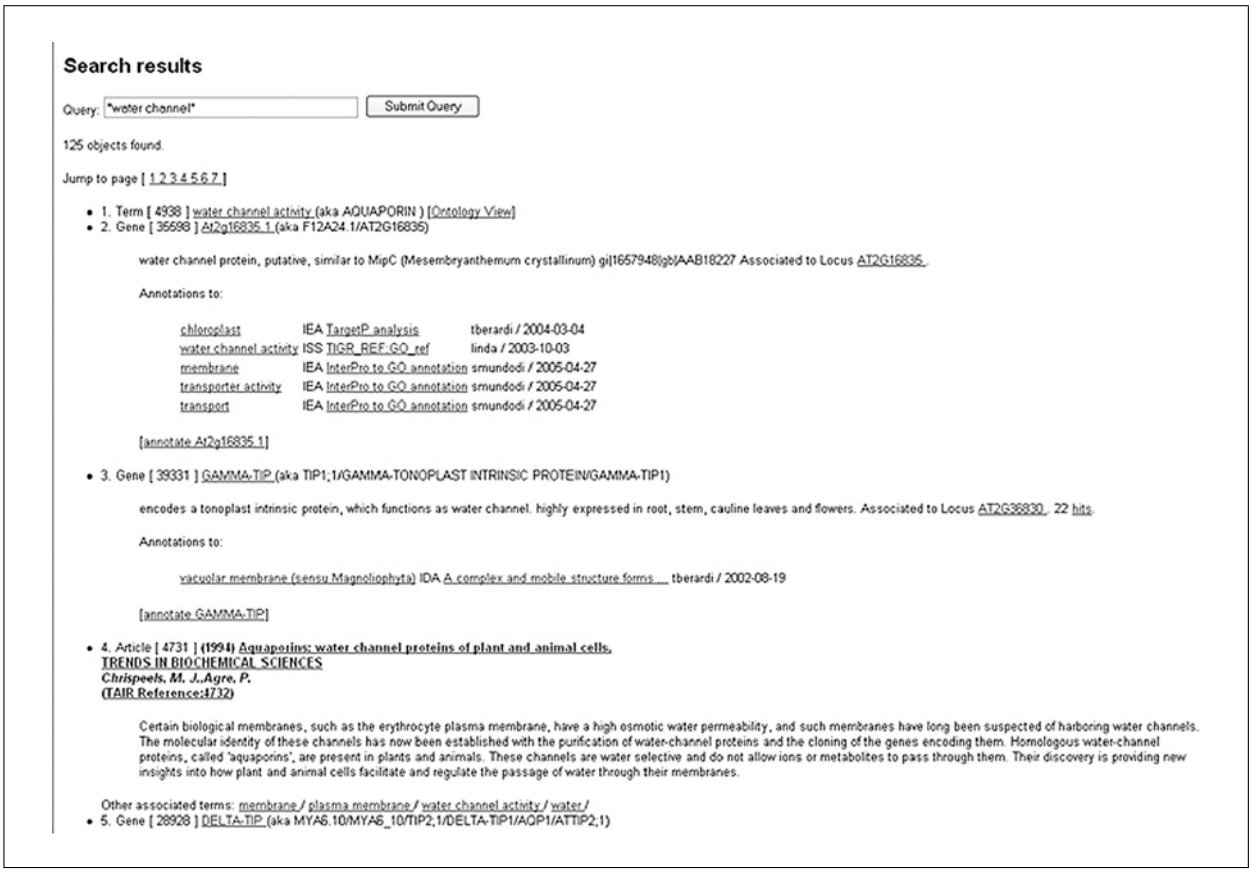

Figure 9.7.3 A screenshot of the Search All function's result page showing the first page of results from a search with `"water channel"` (including quotes) as query string. Results are displayed in the order of "density" of the match, which is a measure of the frequency of the matching string over the length of the entry. Underlined text (shown also in blue on screen) indicates a hyperlink to more information.

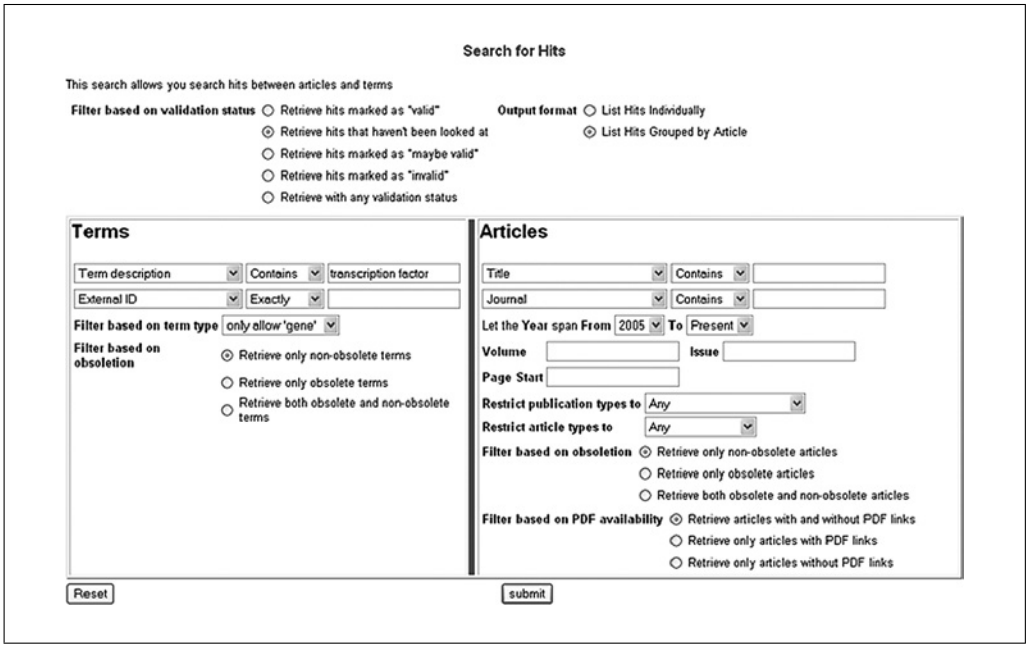

Figure 9.7.4 A screenshot of the Search Hits function. Users can restrict the search by terms (lower left box), articles (lower right box), and validation status of the automated hits between terms and articles (upper left). Options for displaying the results are listed in the upper right corner.

6. For the "Filter based on validation status" parameter, click "Retrieve hits that haven't been looked at" radio button to retrieve hits that have not been validated manually.

   There are three types of parameters to restrict the search: validation status, terms, and articles. All hits are generated automatically by the software, which can be validated by users using the Web browser. Valid hits refer to those that have been validated by a user.

7. To restrict the search by terms, the Term section on the left side of the search page can be used. Users can limit the search by term name, description, ID(s), type, and obsoletion status. As an example, type `transcription factor` (without quotes) in the first text input box, change the drop-down menu to "Term description," and choose the "Contains" option. Leave the "Filter by term type" drop-down menu as the default. This will limit the search to all genes whose description contains the phrase `transcription factor`.

8. To restrict the search by articles, the Article section on the right side of the search page can be used. Users can limit the search by year of publication, title, authors, abstract, journal name, ID(s), publication type (e.g., journal article or book chapter), article type (e.g., research article or review), obsoletion status, and local full-text availability. As an example, limit the year of the publication by selecting "2005" from the "Let the Year span From" parameter. This will limit the search to all articles published in 2005 or later.

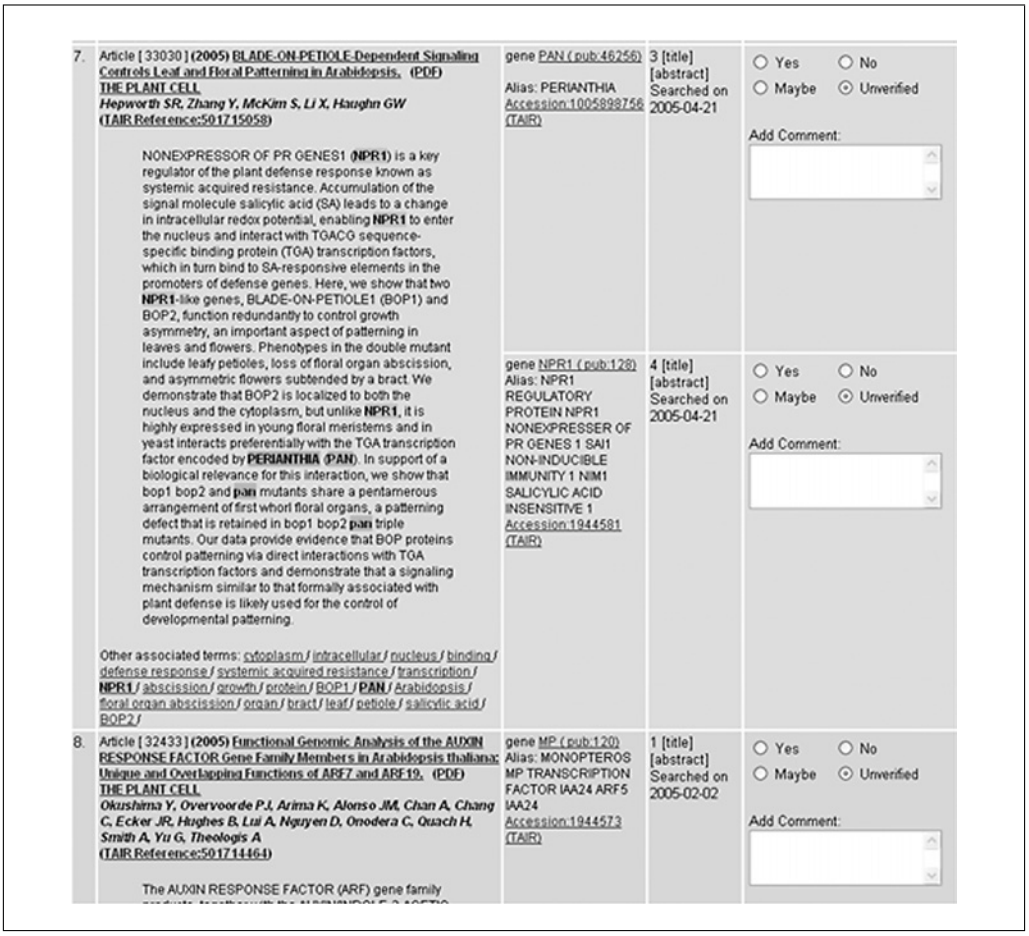

Figure 9.7.5 A screenshot of Search Hits result page. Results are grouped by each article. The first column shows article information, the second column shows matching genes, the third column indicates information about the matching, and the fourth column displays the options for validating the matches between the papers and genes. Underlined text (shown also in blue on screen) indicates hyperlinks to more information.

9. There are two types of output formats: List Hits Individually and List Hits Grouped by Article. The first option lists individual hits ordered by article ID. Choose the second option, and hit the Submit button.

10. The results page (Fig. 9.7.5) shows how many matches (in this case, matching articles) are retrieved. Each row of result has four columns that contain article information, matched term information, details about the match, and a column that allows logged in users to validate the match. Hyperlinks lead to the detail pages of the articles, hits, and other PubSearch objects.

BASIC PROTOCOL 4

USING PubSearch TO ADD AND UPDATE DATA

PubSearch also provides basic interfaces for curating data in its database. All searchable data types can be edited using the Web browser by logged in users. The mechanism for editing is similar for all data types. This protocol describes the updating of existing article data as well as the addition of new articles from the Web interface.

Necessary Resources

See Basic Protocol 1 and Support Protocol 1

Updating article information

1. Go to the "Article search" page (Fig. 9.7.6) by clicking on the Article hyperlink in the "Search for" page header.

2. Search for articles of interest by using the parameters provided on the search page. For example, type `"auxin biosynthesis"` (including quotes) in the text input

Figure 9.7.6 A screenshot of the Article Search form.

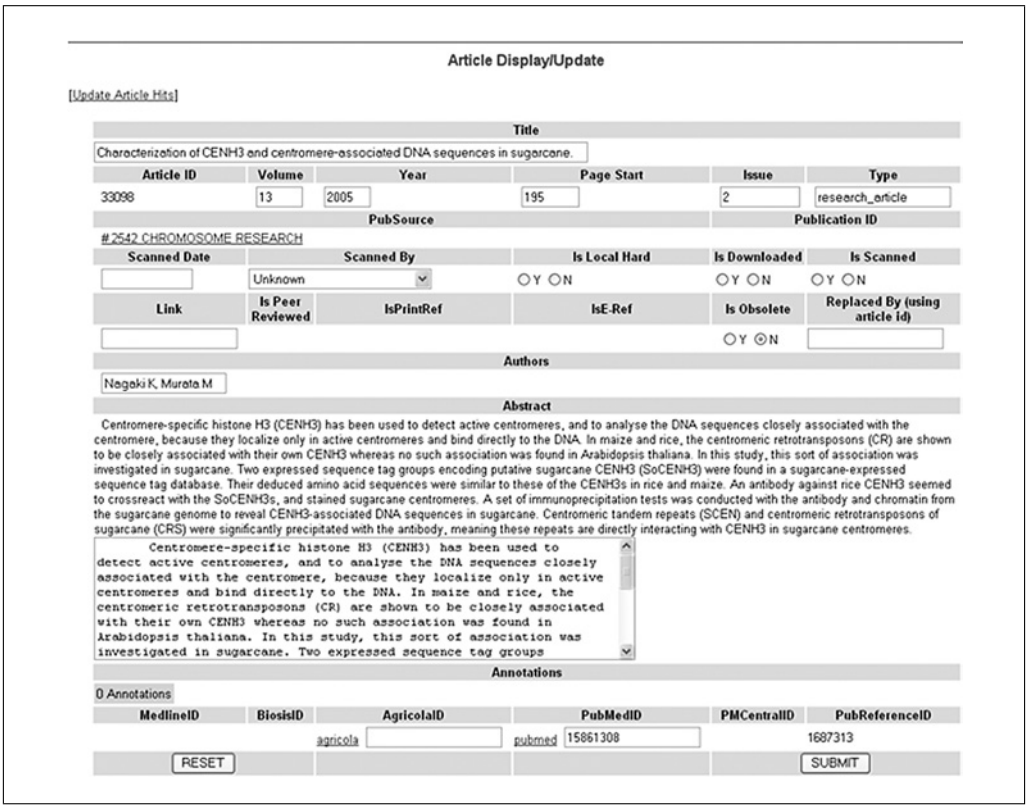

Figure 9.7.7 A screenshot of the Article Detail page. Logged users can update the fields on this page.

box under the Simple section, and change the "Let the Year span From:" drop-down menu to "2003."

The text input box in the Simple section of this page searches the full text of all of the articles in a manner similar to that of Google. If phrases are not enclosed within quotes, individual words will be searched separately.

3. All articles that contain the phrase `"auxin biosynthesis"` in the text, and that were published in 2003 or later, will be retrieved.
4. Results are displayed grouped by articles. For each article, the year, title, journal, authors, and abstract are displayed. In addition, links to the PDF version of the full text, associated terms and genes, and article detail page are provided. To go to the article detail page to edit the information, click on the title.
5. If logged in, a number of fields will be seen that can be updated in the form of text boxes, radio buttons, and drop-down menus (Fig. 9.7.7). Multiple fields can be updated at once. Update the fields as necessary, then click the Submit button.
6. If the publication source information (e.g., periodicals) needs to be updated, click on the PubSource name.
7. Modify the updatable fields as necessary, then click the Submit button.

Adding data individually

In addition to the bulk import of data described in Basic Protocol 1 and the Alternate Protocol, logged-in users can insert new data entries using the Web forms. Currently data types that can be added into the database via these Web forms include articles, publication sources, genes, sequences, alleles, germplasms, terms, hits, and controlled vocabulary annotations. Web forms for adding new data can be found by clicking on the

data object names in the "Add:" toolbar on top of each page (e.g., articles, genes, terms; Fig. 9.7.2). In addition, data that are associated with genes or articles such as alleles, sequences, germplasms, or publication sources can be added from the gene or article update pages. The principle for adding new data objects is the same for all objects. In this chapter, adding articles individually is used as an example.

Adding article information

8. Log in to the database as described in Basic Protocol 3.
9. Click on Article in the "Add:" toolbar to get to the Add Article page (Fig. 9.7.8).
10. There are two ways of adding an article into the database. If the PUBMED ID is known, enter the ID in the text input box. For example, enter `15861308` and click the "Get it" button.
11. If the article does not exist in the database, the article information will be automatically entered into the database and an article update page will be displayed. Check the data and update the information if necessary.

    If the article already exists in the database, the user will be given an error page that includes a link to the existing article entry. Click on the link to check that the correct article is in the database.

12. If the article to be entered into the database does not have PubMed ID, the fields can be filled in manually on the Add Article page (Fig. 9.7.8). Fields marked with an asterisk (*) are required.

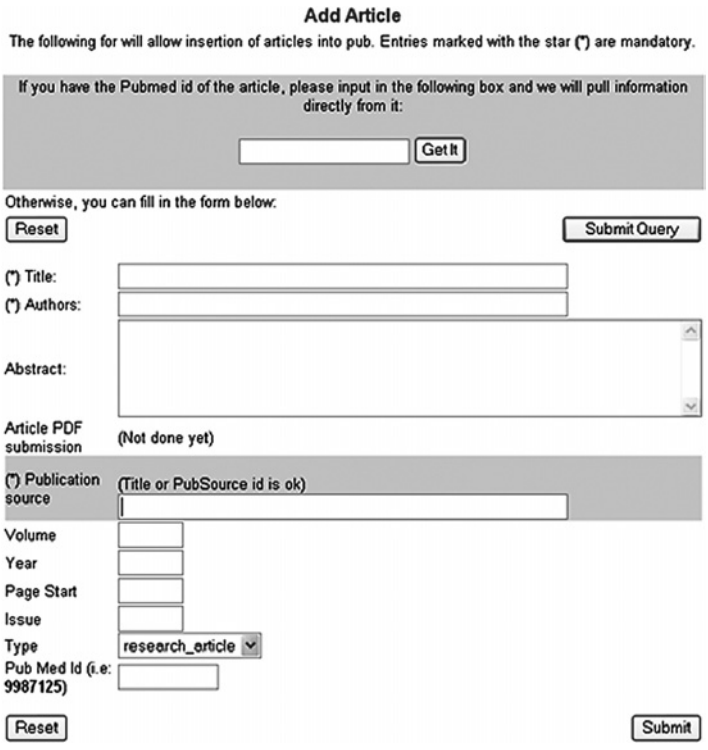

Figure 9.7.8 A screenshot of the Add an Article form. This form allows users to insert an individual article. Entering the PubMed ID will retrieve all the article information from PubMed automatically using the PubFetch software, check for duplicates with articles in PubSearch database, insert the article if it does not yet exist in the database, and allow users to update the retrieved fields if necessary. If the PubMed ID is not known, users can enter the fields of the article.

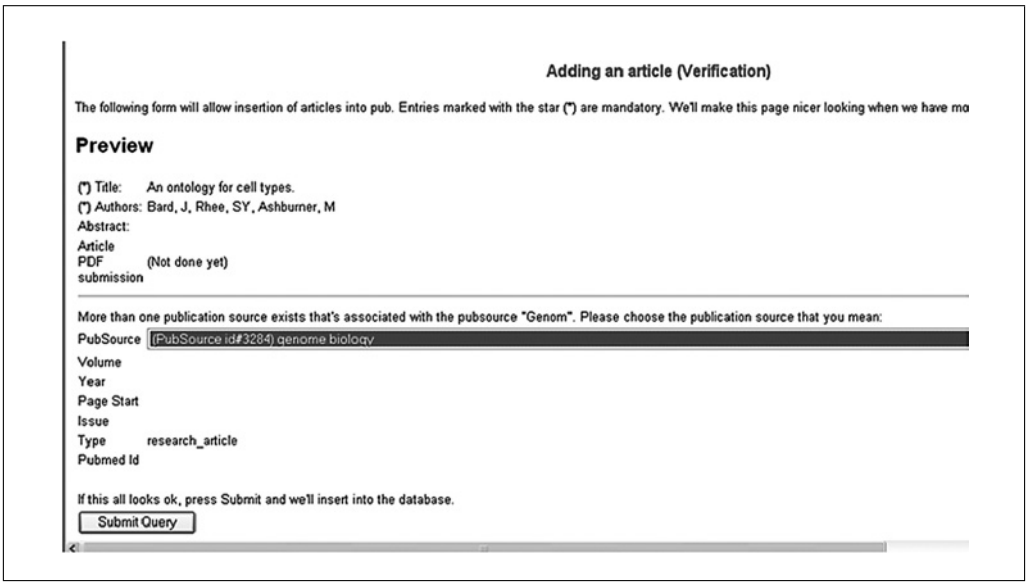

Figure 9.7.9 A screenshot of the Add an Article function's preview page. If the fields of the new article have been entered manually, the preview page allows users to choose the correct publication source using a drop-down menu, or to add a new publication source.

13. If the input string for the publication source (e.g., Journal or Book Series) matches existing publication sources, one will be redirected to a page with a drop-down list of publication sources from which to choose the source (Fig. 9.7.9). Choose the correct journal and, if all other fields are correct, enter Submit. To change any fields, use the Back button of the browser to go back to the previous page to update the fields as necessary.
14. If the input string for the publication source does not exist in the database, the user will be notified so as to be able to go back and update the search parameter or go to a page to add it as a new publication source.
15. Click on the hyperlink to add a new publication source. Add the necessary fields and click the Submit button. This will return the article update page with the article data that has just been inserted into the database. If any of the fields need to be updated, update the necessary information and click the Submit button.

USING PubSearch TO MAKE GENE ONTOLOGY ANNOTATIONS

BASIC PROTOCOL 5

This section describes how to use PubSearch to make associations between genes and Gene Ontology terms using the Web browser. The intended users for this section are database curators. This protocol uses the demo version URL. For one's own version of PubSearch, the base URL will be different. By default, it takes the form: `http://${DOMAINNAME}:8080/PubSearch`.

Users can login from any page in PubSearch (see Basic Protocol 3). Logging in is required for making or updating GO annotations.

Necessary Resources

Hardware

Computer with Internet access

Software

Up-to-date browser such as Netscape 6.X, Internet Explorer 5.X, Safari 1.X

Select the gene to be annotated

1. Go to Gene search page either by clicking on the Genes hyperlink in the "Search for" page header or by going to the URL *http://${DOMAINNAME}:8080/pubsearch/Search?middle_page=genes_exp*.
2. Enter the name of the gene to be annotated; for example, `HST`. Click Submit.
3. From the search results page, either click on the "annotate HST" link at the bottom of the gene entry or click on the gene name and then click the Add Annotations link on the Gene Detail page.

Select a GO term

4. Type in the term to be used in the input box below Term Name. For example, type in `kinase`.
5. Click the Term Search button. This brings up the Term Search page with `kinase` filled in for a "contains" search.
6. Restrict "term type" to the aspect of interested. For example, if doing a function annotation, select "only allow func" from the "Filter based on term type" drop-down menu. Click Submit.
7. From the search results page, select the term that looks the most appropriate for the particular annotation. One can click either on the "term name" or on the Ontology View link.

   Clicking on "term name" opens a "term detail page" with the definition of the term and the term's parentage. Clicking on the Ontology View link opens a term browser and allows one to traverse up and down the structured hierarchy of the GO. In either case, it is possible to click on the button with the GO id to be used for the annotation. Doing so will enter both the term name and the term id into the annotation window in the appropriate slots.

Select a relationship type

8. The relationship type clarifies the gene-to-term relationship. Select the appropriate one for the annotation. For example, when using a GO biological process term, a commonly used relationship type is "involved in."

Select an evidence code

9. From the Type drop-down menu, select the appropriate three-letter evidence code for the annotation.

Select an evidence description

10. Depending on the evidence code selected, a number of evidence descriptions will be displayed in the Description drop-down menu. Select the one that is most appropriate for the annotation.

Select a reference

11. Click on either Article, Communication, Analysis Reference, or Book, depending on what type of reference is appropriate. If Article, Analysis Reference, or Communication are chosen, click the Select button beside the appropriate reference to use it in the annotation. If Book is chosen, select the title of the book from the Book menu, highlight the chapter to use within the Book Chapter menu, and click the Select button.

Enter the completed annotation into the database

12. Click on the Update button. The annotation will appear in the list of completed associations with the user's name and the current date in the "Annotated by" and Date fields.

Updating existing annotations

13. It is possible to update the term name, relationship type, evidence code, or evidence description by making changes and then clicking the Update button. However, if changing from one reference type to another (i.e., Communication to Article), it is necessary to obsolete the old annotation by clicking on the Obsolete "Y" radio button and then creating the new annotation as described above.

Propagating annotations

14. The user may find it desirable to propagate an annotation that has been made for one gene to other genes discussed in the same paper. If the annotations will be identical, except for the gene being annotated, one can use the annotation propagation function that is built into PubSearch.
15. Start on the Gene Detail page of the gene that has the annotation(s) to be propagated.
16. Select the annotations to propagate from the Annotations band by ticking the boxes beside the annotations in question, then click the Propagate button at the top of the Annotations band.
17. Enter the gene name for propagation. If there are several genes with the same base name, for example, ABC1-10, select "contains," type in the base name `ABC`, and click on Search Gene. If there are several genes that do not share a base name, it is possible to upload a file with all of the gene names from the computer desktop using the Browse button.
18. From the search results page, select the genes to which the annotation(s) are to be propagated. One may get multiple matches to the query, so make sure that the correct genes are selected.
19. Click the Propagate button.
20. A list of annotations that will be propagated will come up. Confirm that this is the course of action to take. Click Propagate Annotations.

GENERATING AND LOADING InterProToGo ANNOTATIONS

BASIC PROTOCOL 6

A common source of associations between genes and GO terms is via InterPro protein mappings. This protocol describes how to generate GO annotations from InterPro mappings to proteins and load them to PubSearch database on the fly using a perl script.

Necessary Resources

Hardware

See Support Protocol 1

Software

PubSearch (see Support Protocol 1 and Support Protocol 2)

Files

To generate GO annotations from InterPro mappings to proteins, the authors of this unit use the Interpro2gene mapping file and Interpro2Go mapping file. The Interpro2gene mapping file is a two-column file in which the first column is the gene name and second column contains the InterPro Ids. This file is generated using `InterProScan.pl`, which can be downloaded from *http://www.ebi.ac.uk/interpro*.

The sample file for *Arabidopsis*, `INTERPRO.Arab_R5.txt`, is located in the `maint/interpro2goAnnotation under ${PUBHOME}` directory.

The Interpro2Go file contains a mapping between InterPro domains and corresponding GO terms. It is manually generated and maintained by InterPro curators and is available from the GO Website *http://www.geneontology.org/external2go/interpro2go*.

The Perl script used in this protocol will automatically retrieve this mapping file.

1. To generate and load GO annotations from InterPro annotations, run `addAnnotationFromInterproGo.pl`. This script retrieves the latest Interpro2Go mapping file from the GO Website, generates GO annotations from the Interpro2Go mapping file and the Interpro2Gene mapping file, and loads the converted GO annotations into a PubSearch database. To run the script, execute the following commands from the `${PUBHOME} directory`:

```
>cd maint/interpro2goAnnotation

>perl addAnnotationFromInterproGo.pl -D database_name
-U user_id
```

COMMENTARY

Background Information

The systematic review and curation of scientific literature to extract relevant information is a task faced by every researcher. Broad-based, systematic curation of literature has been transformed by the World Wide Web. Tasks that were impossible when journals and articles were only available in print are now routine. Online literature repositories like PubMed, online journals, and the move toward providing the full text of research articles online have all made the literature more accessible than ever before. However, with this increase in accessibility has come an increase in complexity that is familiar to any user of the World Wide Web. How do you effectively find the literature you are looking for out of the ever-increasing quantities of literature that are not relevant to the task at hand?

Model Organism databases such as TAIR and RGD rely on the primary research literature as one of the main sources of information about an organism's genes and proteins and their functional role in that organism. While a variety of other information is stored in these databases, genes remain the primary focus. PubSearch was developed with the goal of facilitating the curation of literature pertaining to the genes of *Arabidopsis thaliana*. Figure 9.7.10 shows how PubSearch is used at TAIR for facilitating literature curation. In step 1, articles from literature databases such as PubMed, gene data from TAIR, and ontologies from GO and PO (Plant Ontology, *http://www.plantontology.org*) are imported. Next, the articles are indexed with gene data and ontologies. After the automatic associations are made in the database, curators can access the data using a Web browser and perform a number of functions such as editing data, validating the associations between data objects and articles, making controlled vocabulary annotations, and adding missing information (step 3). Finally, curated data are exported to TAIR and other databases such as the GO database (step 4).

The literature identification strategy of PubSearch is to first collect a larger corpus of broadly relevant articles and then narrow it down using specific terms relevant to the task at hand. In the case of PubSearch at TAIR, all articles mentioning *Arabidopsis* published since the previous analysis are identified. These are then searched for "terms" (known keywords of interest such as *Arabidopsis* gene symbols, ontology terms, etc.) to identify papers that have a higher chance of being useful for the curation of data relevant to a gene. The process of screening the articles against the list of relevant terms generates "hits" between an article and a term. The hits are then validated by human review of the full abstract to determine if the article does indeed pertain to the expected gene and has potentially useful information about its function. The combination of automated term matching and the manual validation of the subsequent hits is a relatively quick way to screen large numbers of articles to identify those that should be read in full by a human curator.

PubSearch provides a variety of search interfaces to allow curators to retrieve genes,

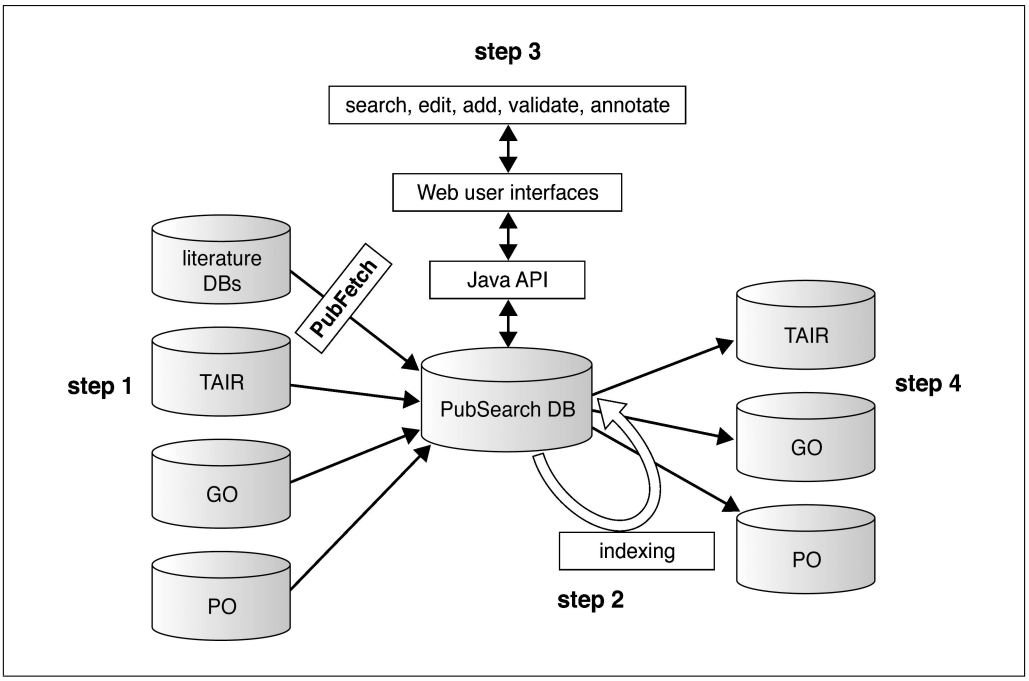

Figure 9.7.10 The PubSearch database is the central component of the PubSearch system. The following operations are performed during PubSearch use. In step 1, the PubSearch database is loaded in batch mode using input from other databases—e.g., articles from literature databases such as PubMed and Agricola using PubFetch software, biological data like gene, allele, and germplasm information from TAIR (an example model organism database), and ontologies from Ontology databases such as Gene Ontology and Plant Ontology. In step 2, the PubSearch database indexes the information by populating the Hit table using the Lucene engine. In step 3, through the Java API and a set of Web user interfaces, curators search, browse, edit, and add data, relying on the indexed data in the database. Finally, in step 4, the edited biological, literature, and annotation data are exported to the TAIR production database and other databases such as Gene Ontology and Plant Ontology.

keywords, articles, and matches between articles and terms using a variety of search parameters. Upon retrieval of relevant information, curators can annotate gene function, cellular location, expression patterns, genomic location, and other attributes by reading the matched articles. The Web user interface for editing annotations is designed to reduce free-text data entry, in order to increase the efficiency of annotation and reduce data-entry errors. The appropriate controlled vocabulary terms can be selected using an integrated ontology browser—a modified version of the AmiGO browser developed by the GO Consortium (The Gene Ontology Consortium 2001)—which allows interactive traversing of structured vocabularies and point-and-click selection of terms. The annotation interface facilitates data entry using pull-down menus or clickable lists that are generated on the fly with the appropriate data for the annotation task at hand. For example, the GO evidence codes (*http://www.geneontology.org/doc/GO.evidence.html*) and evidence descriptions, a controlled vocabulary of experiment types developed at the *Arabidopsis* Information Resource (Rhee et al., 2003), can be selected from pull-down menus.

PubSearch can be used as a stand-alone literature-management tool for biologists. In this case, all that is required in addition to periodic literature downloads using PubFetch is to update the gene and term lists to keep pace with modifications to the various ontologies and identification of new genes. Data import/export tools are provided to upload new vocabularies and updates to the Genes and to export the curated information for use in downstream applications or databases. By default, PubSearch is set up as an internal-use-only Web application and is thus password-protected; a login is required when starting a session. This authentication scheme would also work well for a group of investigators working on the annotation of a gene family or a microarray result set. The login also allows tracking of operations that a user has performed during a session, which can be used to verify consistency of annotation between users.

Alternative approaches

The PubSearch/PubFetch system is unique in that it provides a stand-alone, integrated literature-management and data-curation environment. Literature-curation software developed by other curation groups is typically tightly coupled to the computing environment of the particular curation group, making it difficult if not impossible for others to reuse the software components. As an alternative to a server-based approach, many researchers are familiar with desktop bibliography software such as EndNote and Reference Manager. These provide extensive literature retrieval and searching capabilities; however, they provide no capacity for curating information from this literature. In this scenario, data are often recorded in other desktop applications such as Microsoft Excel, which certainly has advantages for small-scale endeavors; however, this quickly becomes inconvenient for larger-scale annotation efforts. Another open-source text-mining project, Textpresso (Müller et al., 2004), is being developed as a component of the Generic Model Organism Database project. Texpresso uses customized biological concept ontologies to search the full text of an article, providing a sophisticated semantic search algorithm that goes beyond the existing PubSearch term-matching approach. Texpresso currently provides no data-curation functionality, so one might envisage integrating PubSearch with the Textpresso search engine to enable more precise categorization of the literature for subsequent curation.

Future directions

PubSearch is an open-source project, and, as such, all contributions by interested developers are most welcome. Below are listed a number of potential areas for extensions of the software that might provide ideas about how PubSearch could be used in the future.

PubSearch as a framework for the implementation of additional classification algorithms

PubSearch uses a robust but simple term-matching technique to identify relevant articles. This approach will miss relevant articles that do not contain these keywords. In addition to tools such as Texpresso, described above, other machine-learning algorithms exist that could be implemented within the PubSearch environment to identify articles in a more sophisticated fashion. The manual validation of articles in PubSearch divides them into those that have been shown to be relevant for further curation and those that are not relevant for further curation. One could use these datasets to train a Support Vector Machine to recapitulate this classification, with the expectation that it might work better at identifying articles that were lacking the exact keywords but nonetheless had overall content that indicated relevance. Even if these algorithms were not implemented inside PubSearch, the annotated literature corpus that is created through the use of PubSearch would be of great use to natural language processing researchers.

Expansion of PubSearch to allow curation of additional data beyond genes

PubSearch is currently gene-centric, allowing the curation of gene-related information from the literature. PubSearch could be expanded to allow the curation of data for other objects of interest, such as quantitative trait loci, subspecies of an organism (e.g., inbred rat strains, plant ecotypes), or other objects of interest. This would enable PubSearch to become a broader literature curation platform, allowing researchers to integrate a variety of data types with links to the literature, terms and vocabularies, and other data objects.

Further support for desktop applications such as EndNote

PubSearch provides the means to download articles from online literature databases and link them to genes and other biological terms. It would be convenient if relevant articles could be exported in a format compatible with EndNote or Reference Manager, so that articles stored in PubSearch could be easily used as citations in a manuscript.

Troubleshooting

Failure in database connection

If there is a failure in database connection to MySQL, there are two major possibilities: first that MySQL's network support has been turned off, and second that MySQL's permissions are too restrictive. In the first case, the MySQL configuration file `/etc/my.cnf` may contain the directive "skip networking." If this is the case, comment this directive out and restart MySQL. In the second case, the MySQL administrator must grant privileges to allow PubSearch to communicate with the database. The administrator may need to execute step 6 of Support Protocol 1 and inspect the `program.properties` file, to make sure that the granting SQL statement uses the same hostname as the configuration file.

Failure in unarchiving the distribution

If the `tar` utility fails with an error message about directory checksum errors, it is likely that the native `tar` utility on the system does not support long filenames. In this case, it is recommended that GNU `tar` be used to unpack the PubSearch distribution. GNU `tar` can be found online at *http://gnu.org/*.

Failure in logging in

If a user cannot log in, then it is possible that the user has not yet been added to the User table. To show a list of users on the system, execute:

```
>mysql pubdb
mysql> select * from pub_user
```

to verify that the user does exist in the PubSearch database. If not, then the user can be added by using the `bin/add_curator.pl` or `bin/add_user.pl` commands.

If the user does exist in the pub_user table, then it is likely that the password has been entered incorrectly and may need to be updated. There is no command-line utility to update a user's password, but the following SQL command will refresh the database:

```
mysql> update pub_user set
password=PASSWORD("[pass-
word here]") where name=
"[username]"
```


Acknowledgement

The development of PubSearch is supported in part by NHGRI grant number R01HG02728 (SYR, ST), NSF grant number DBI-9978564 (SYR), and NIH grant number HL64541(ST). The authors of this unit wish to thank Barbara Buchanan at NAL for compiling the *Arabidopsis* papers from Agricola and BIOSIS. The authors also thank Julie Tacklind for designing and maintaining the Web site, and are grateful to Suparna Mundodi, Leonore Reiser, Eva Huala, Margarita Garcia-Hernandez, Hartmut Foerster, Katica Ilic, Chris Tissier, Rachael Huntley, Nick Moseyko, and Peifen Zhang for their valuable input in improving the usability of the software. The authors are also grateful to the former members, Bengt Anell, Behzad Mahini, Victor Ruotti, and Lukas Mueller, who were involved in the project during its initial stages, and also thank Doug Becker, Dan MacLean, Chris Wilks, Jon Slenk, Susan Bloomberg for their careful reading of the manuscript.

Literature Cited

The Gene Ontology Consortium. 2001. Creating the gene ontology resource: Design and implementation. *Genome Res.* 11:1425-1433.

Müller, H., Kenny, E.E., and Sternberg, P.W. 2004. Textpresso: An ontology-based information retrieval and extraction system for biological literature. *PLoS Biol* 2:e309.

Rhee, S.Y., Beavis, W., Berardini, T.Z., Chen, G., Dixon, D., Doyle, A., Garcia-Hernandez, M., Huala, E., Lander, G., Montoya, M., Miller, N., Mueller, L.A., Mundodi, S., Reiser, L., Tacklind, J., Weems, D.C., Wu, Y., Xu, I., Yoo, D., Yoon, J., and Zhang, P. 2003. The Arabidopsis Information Resource (TAIR): A model organism database providing a centralized, curated gateway to Arabidopsis biology, research materials and community. *Nucl. Acids Res.* 31:224-228.

Internet Resources

http://sourceforge.net/projects/geneontology

Gene Ontology's SourceForge repository.

http://pubsearch.org

PubSearch homepage.

http://tesuque.stanford.edu:9999/pubdemo

PubSearch demo version.

http://lists.sourceforge.net/lists/listinfo/gmod-pubsearch-dv

PubSearch support mailing list.

http://www.gmod.org

Generic Model Organism Database project home page.

Contributed by Danny Yoo, Iris Xu, Tanya Z. Berardini, and Seung Yon Rhee
Carnegie Institution
Stanford, California

Vijay Narayanasamy and Simon Twigger
Medical College of Wisconsin
Milwaukee, Wisconsin