Bioperl Pipeline

A Workflow Framework for Bioinformatics Analysis

Shawn Hoon
Fugu Informatics Group
Institute of Molecular and Cell Biology

Bioperl Pipeline

- Motivation
- Design and Implementation
- Status and Plans for the Future

Complex

10^5 sequences

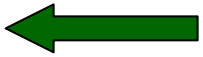
Trees

Alignments

Families etc

Multi-stage analysis

Objects, Pipeline



Find the conserved non-coding sequences between Fugu and Human orthologs and look for known/novel motifs

Basic

What might my sequences code for?

10^3 dna sequences

Single analysis

Flat Files and Scripts

- Each Analysis comes with a set of assumptions that must be considered during interpretation
- Each set of analysis of different parameters can be considered a protocol
  - Ortholog finding
  - CNS detection
  - Protein Family Studies

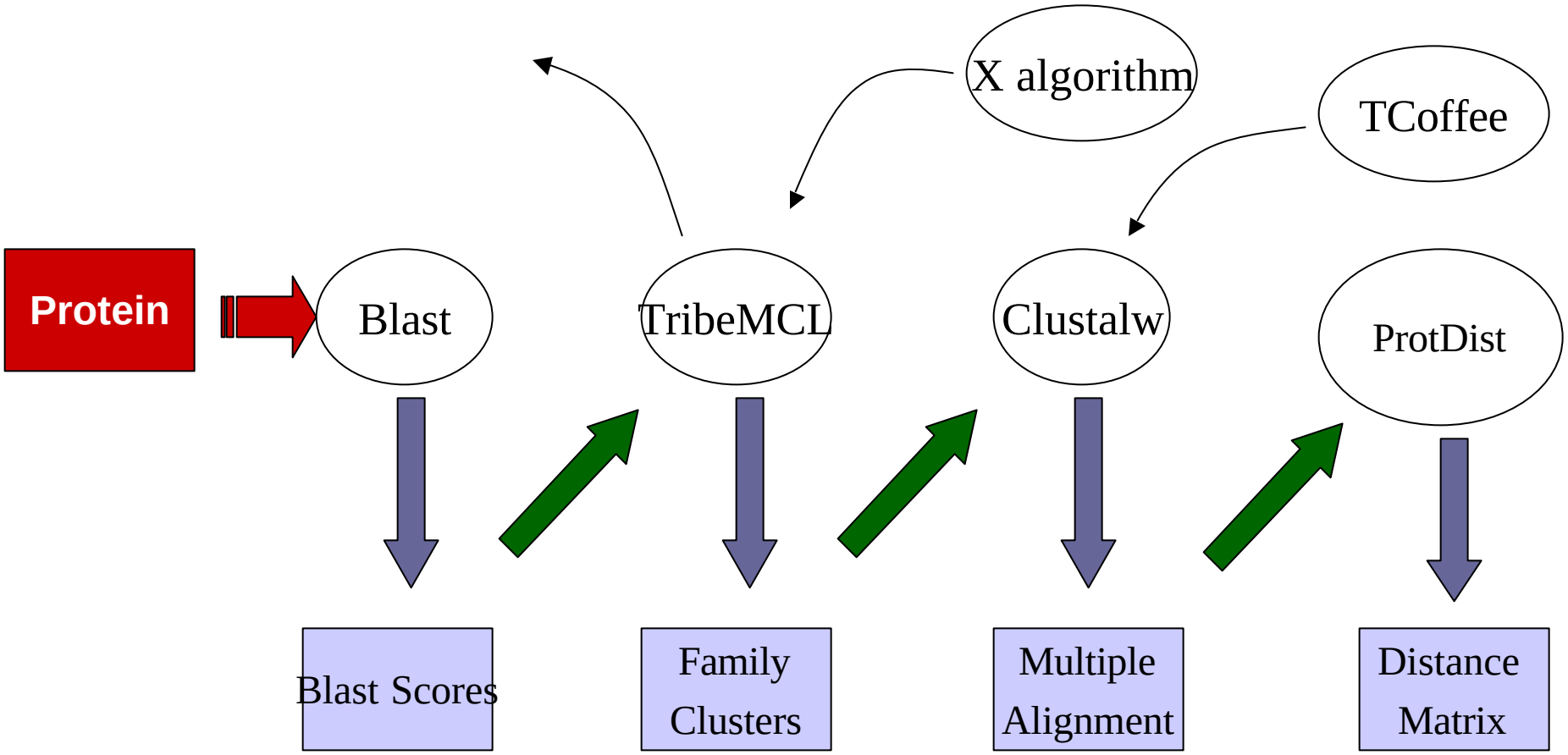
A Possible Protocol

Phylogenetic Analysis:

	Data
Pairwise Blast on a set of proteins	=> Blast Scores
Cluster based on protein scores	=> Families
Multiple Alignment of Family	=> Alignments
Estimate Protein Distance	=> Distance Matrix

Re-run this for every new genome/assembly

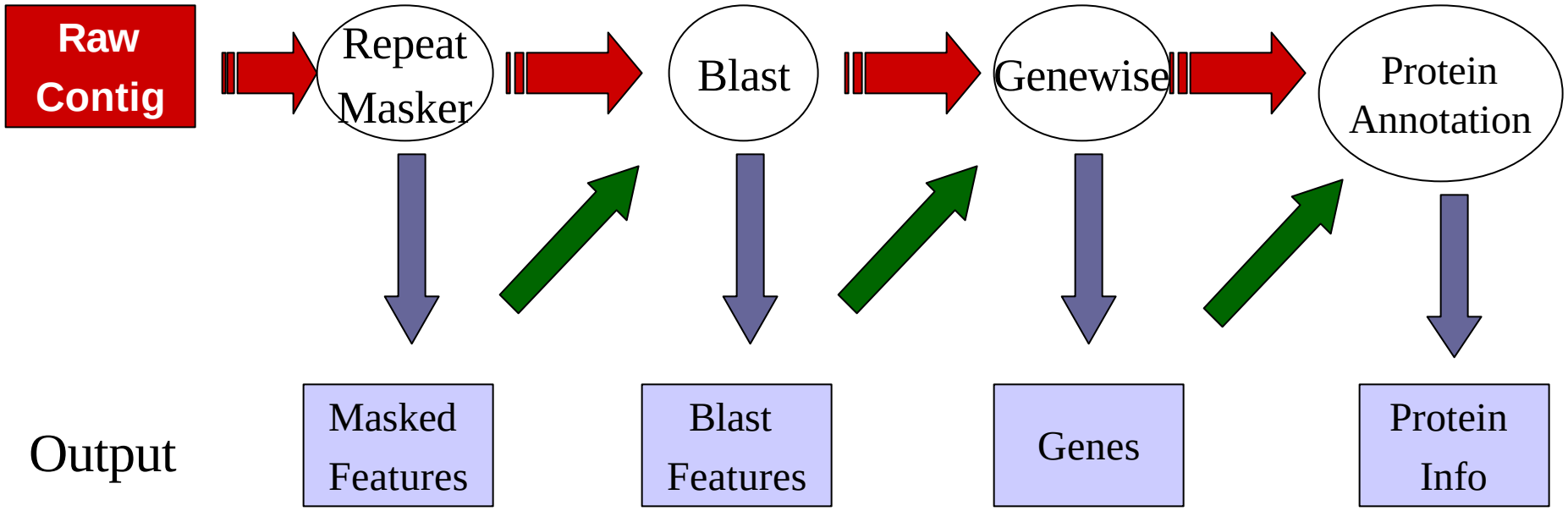
Modularity


X algorithm
TCoffee
Protein
Blast
TribeMCL
Clustalw
ProtDist
Blast Scores
Family Clusters
Multiple Alignment
Distance Matrix

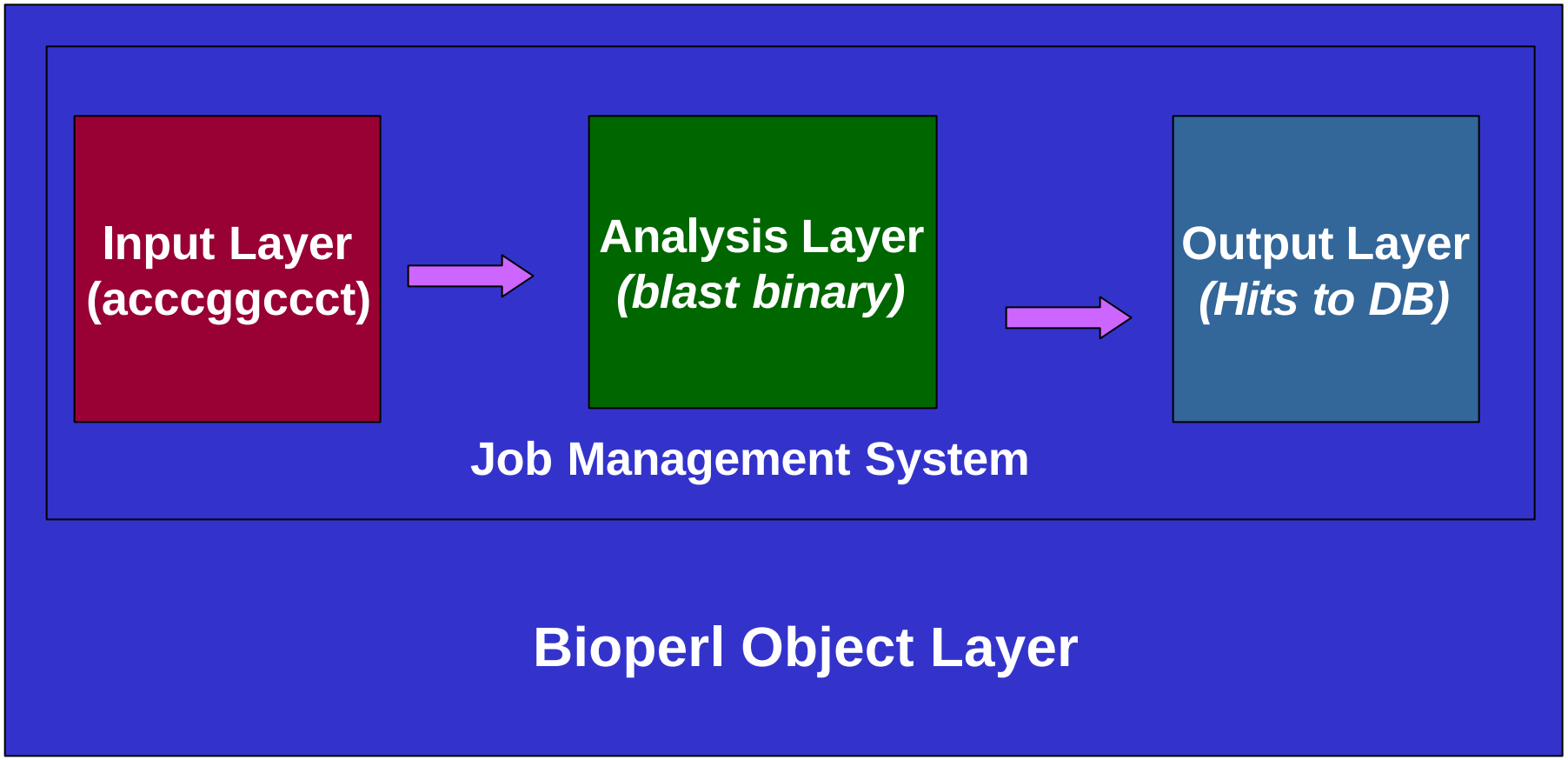
Wish List

- Handle input data from disparate sources
- Modular approach to analysis (plug and play)
- Management of jobs running in parallel over a compute farm
- Output results to format/database of choice for easy interpretation
- Easy to package and reproduce

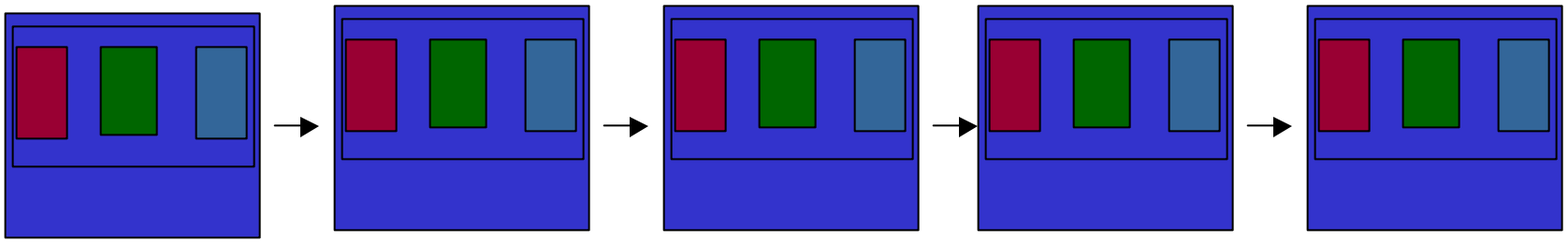
Ensembl Annotation Pipeline System



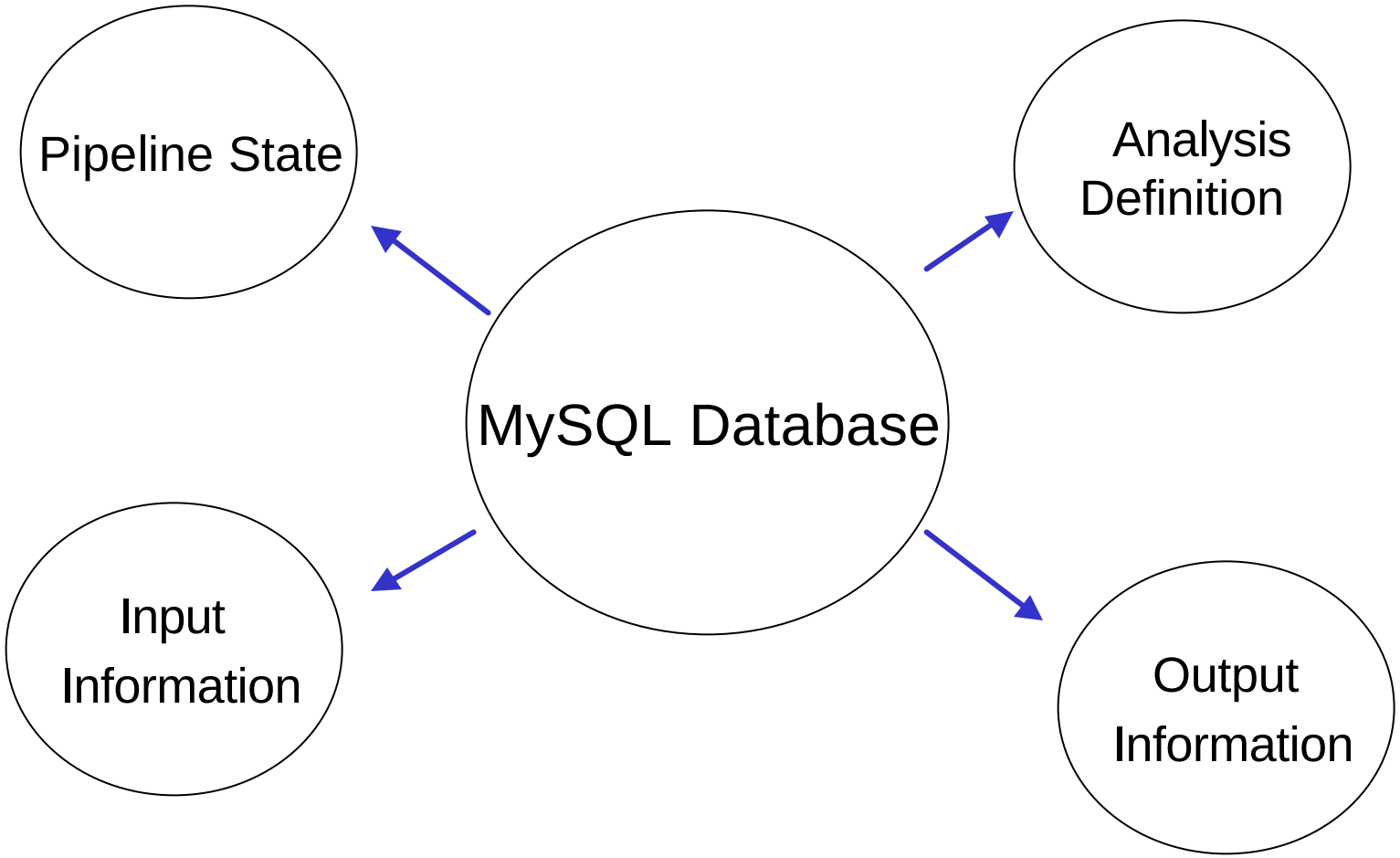
A unit of analysis

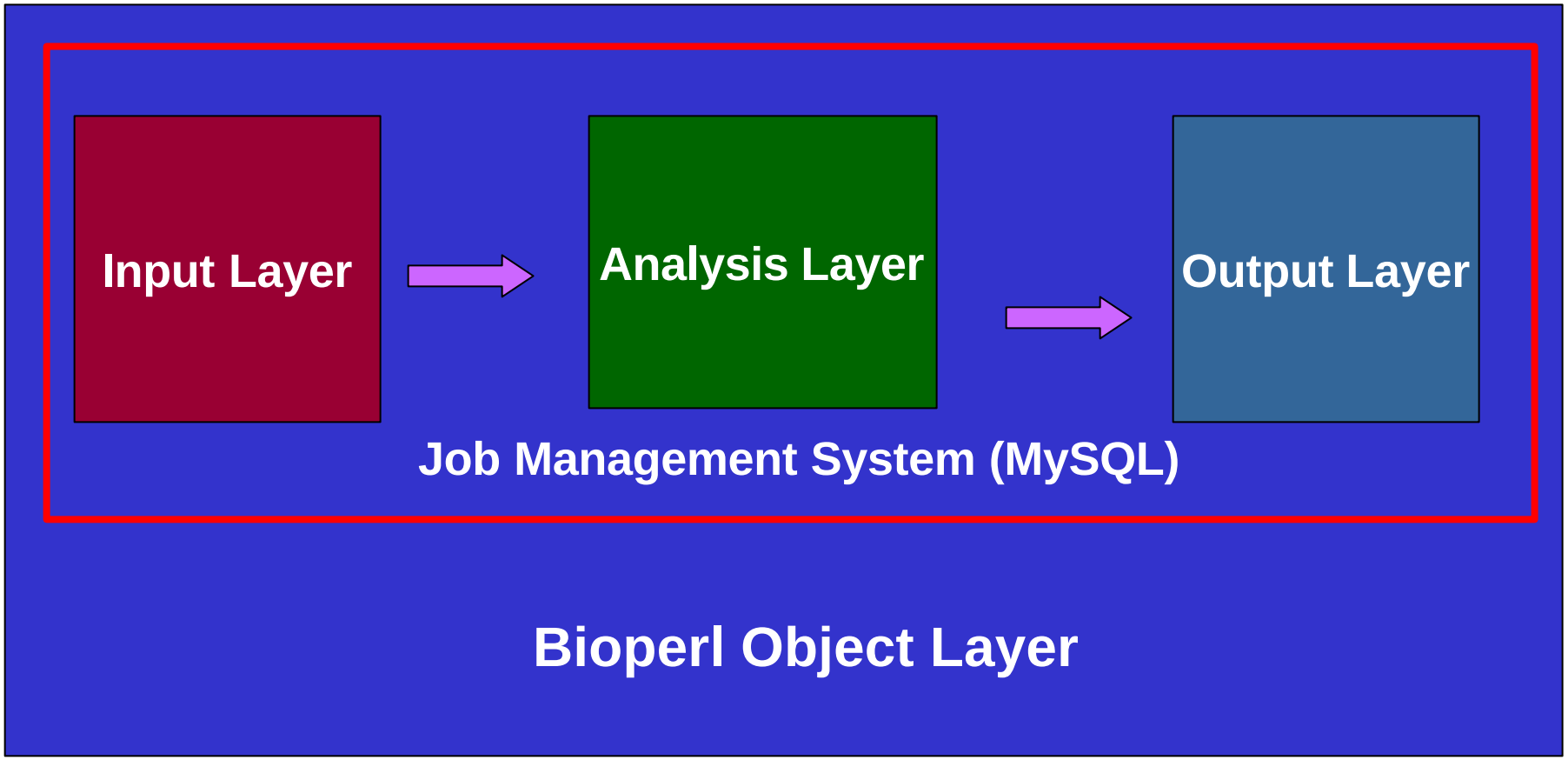


Bioperl Pipeline



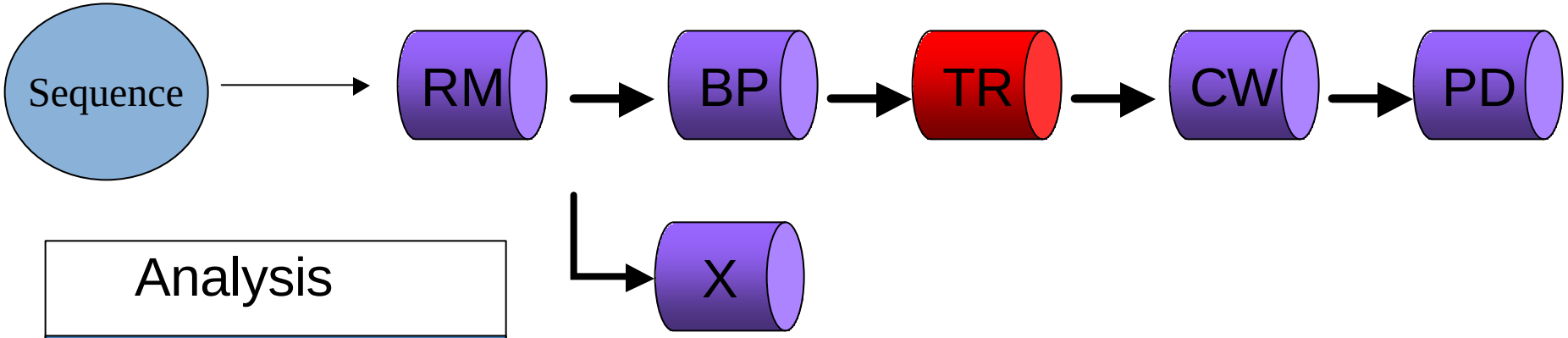
DB-Centric Design




Input Layer
Analysis Layer
Output Layer
Job Management System (MySQL)
Bioperl Object Layer

Job Management System

Jobs are spawned according to Rules



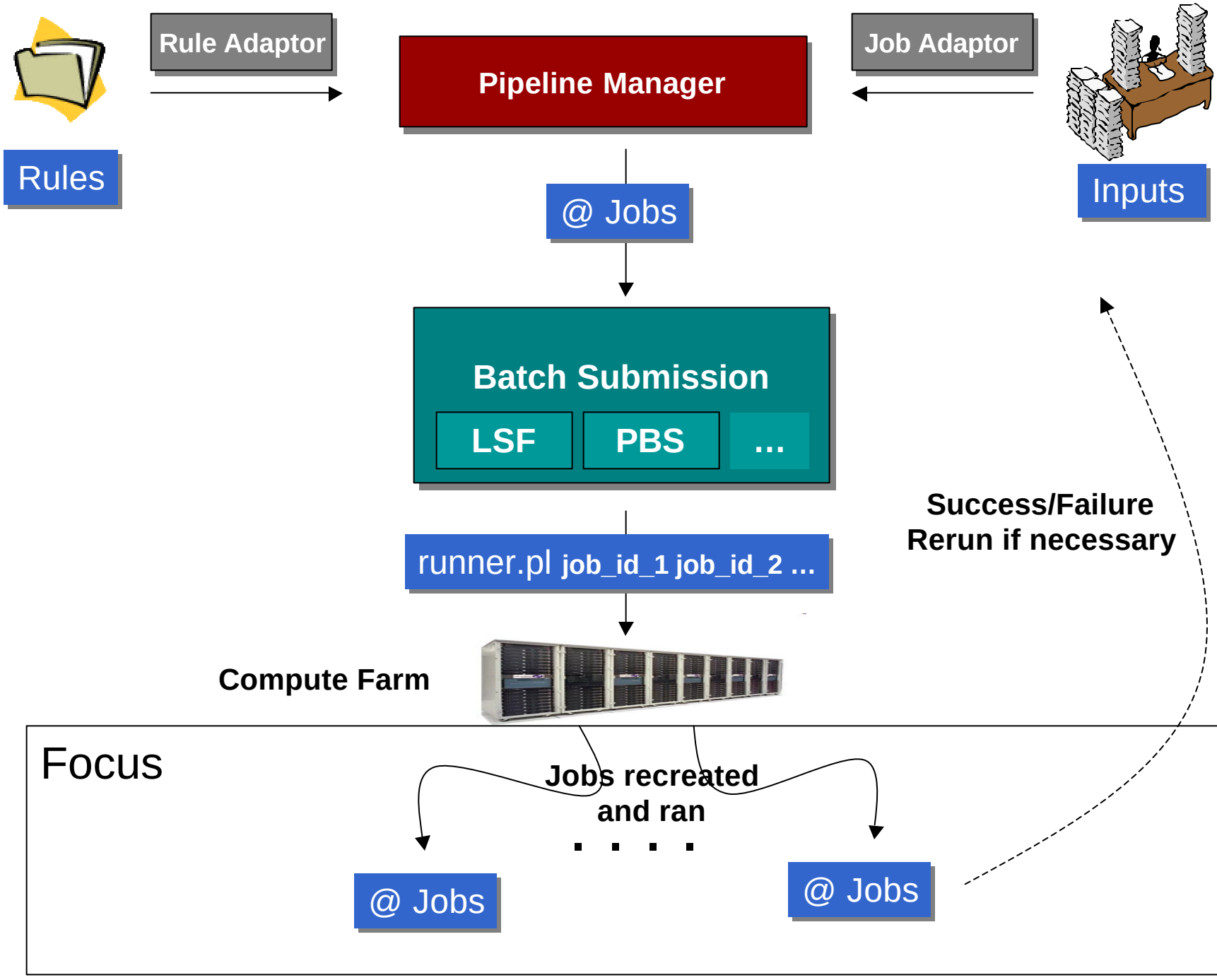
Analysis

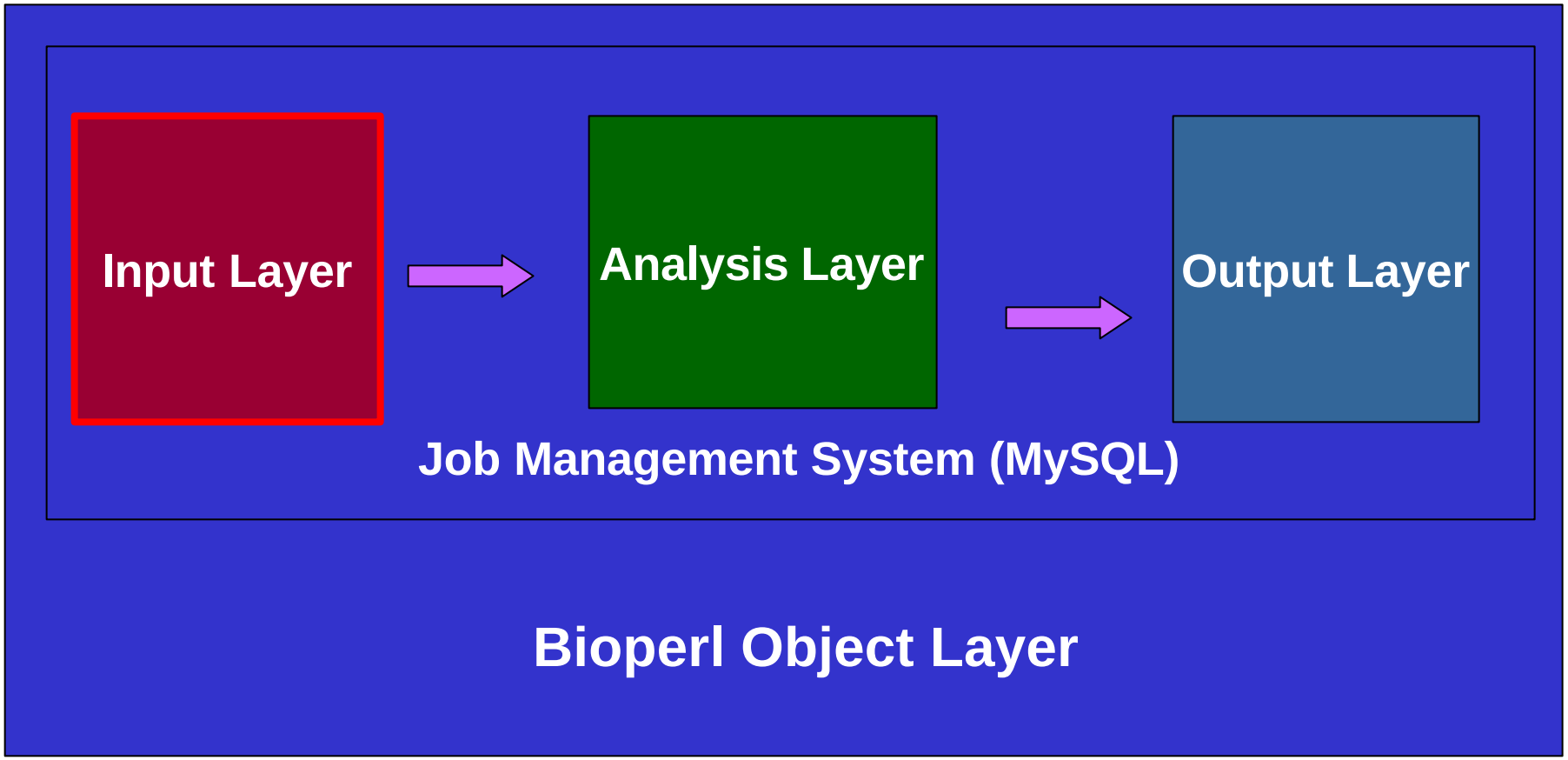
1) RepeatMasker
2) Blastp
3) TribeMCL
4) Clustalw
5) Protdist

6) X

Rule

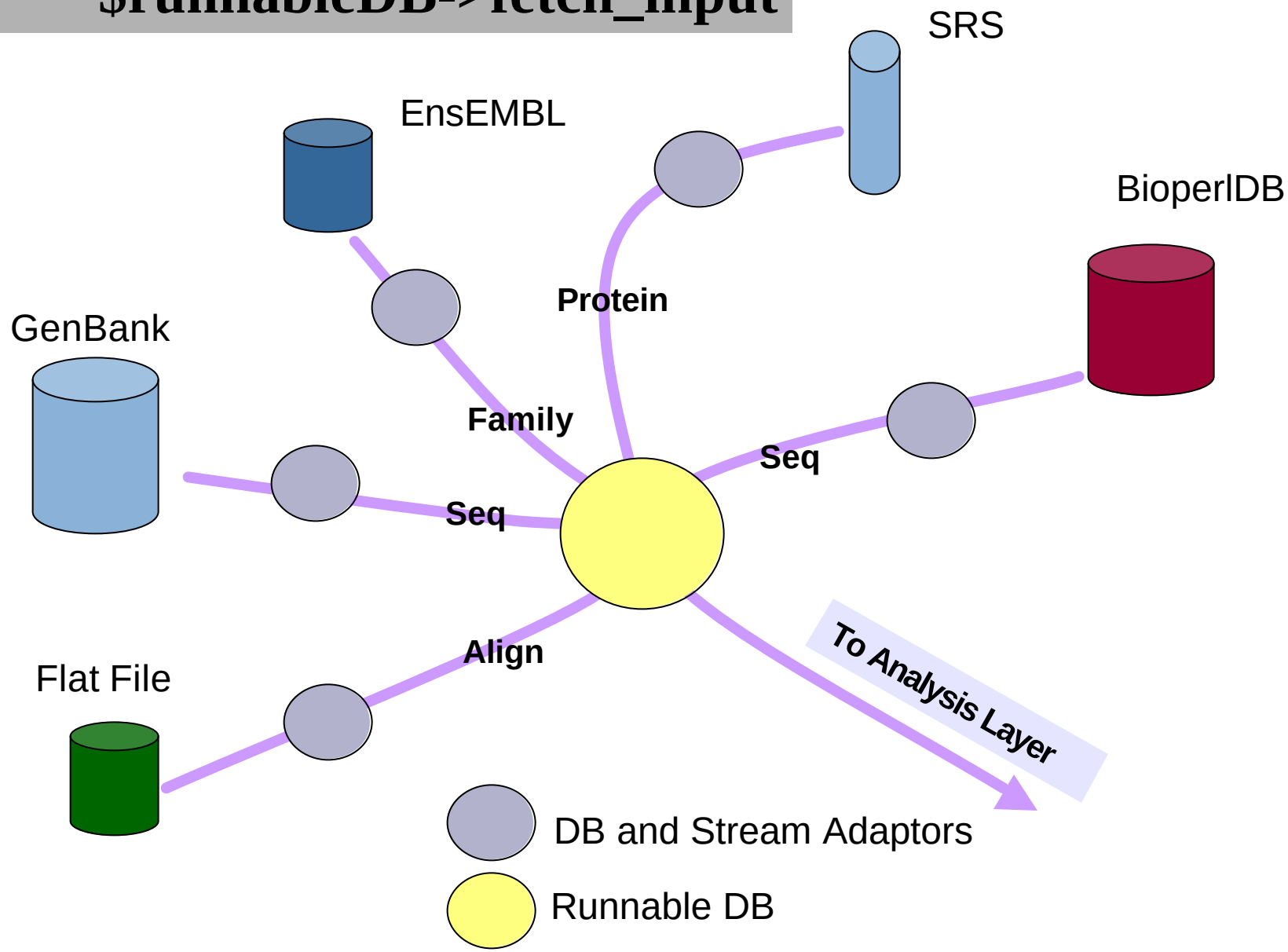
Once 1 is completed, do 2
Wait for all 2 to finish before doing 3
Once 3 is completed do 4
Once 4 is completed do 5
Once 1 is completed do 6

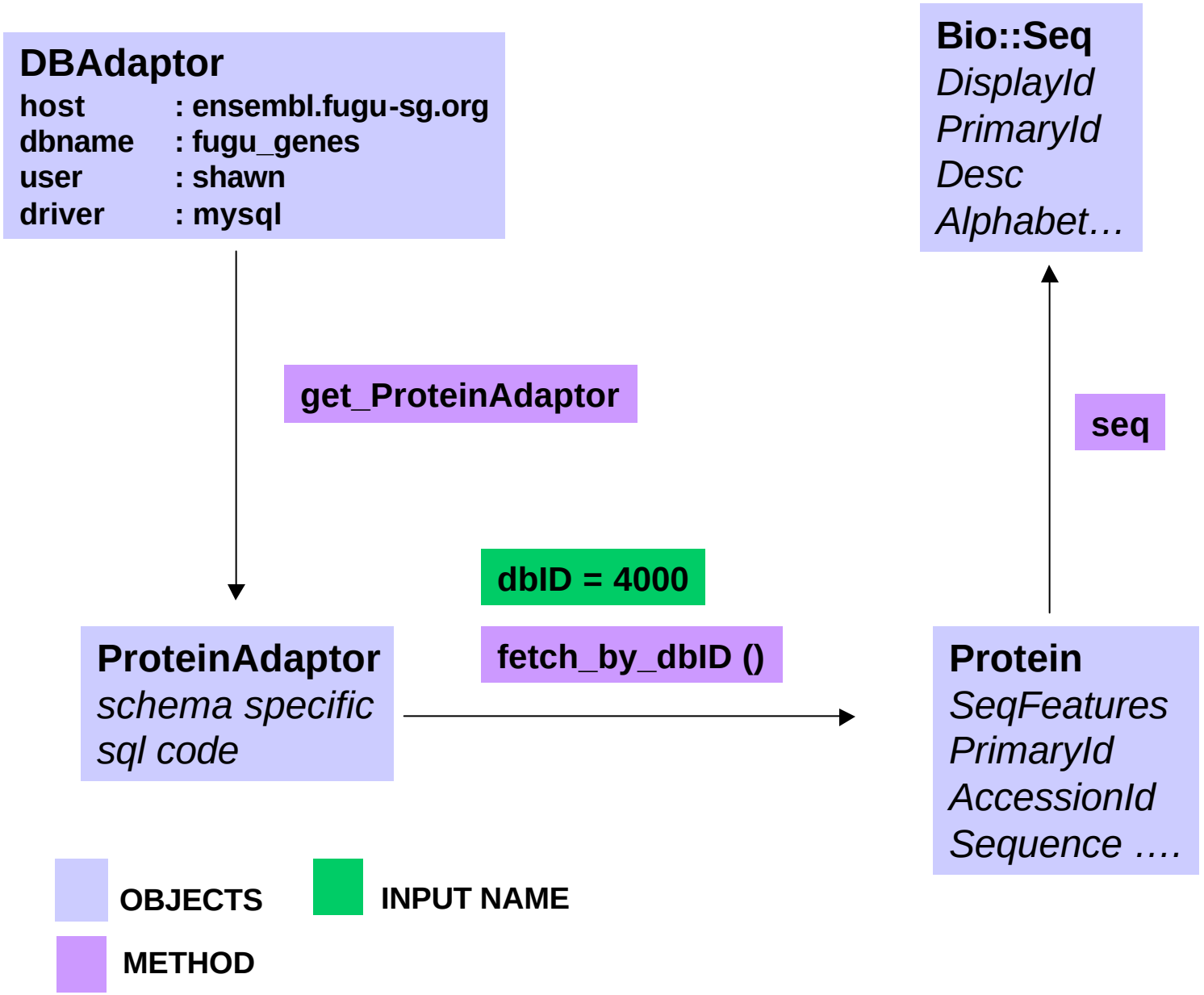

Rule Adaptor
Pipeline Manager
Job Adaptor
Rules
Inputs
@ Jobs
Batch Submission
LSF
PBS
...
Success/Failure
Rerun if necessary
runner.pl job_id_1 job_id_2 ...
Compute Farm
Focus
Jobs recreated
and ran
@ Jobs
@ Jobs

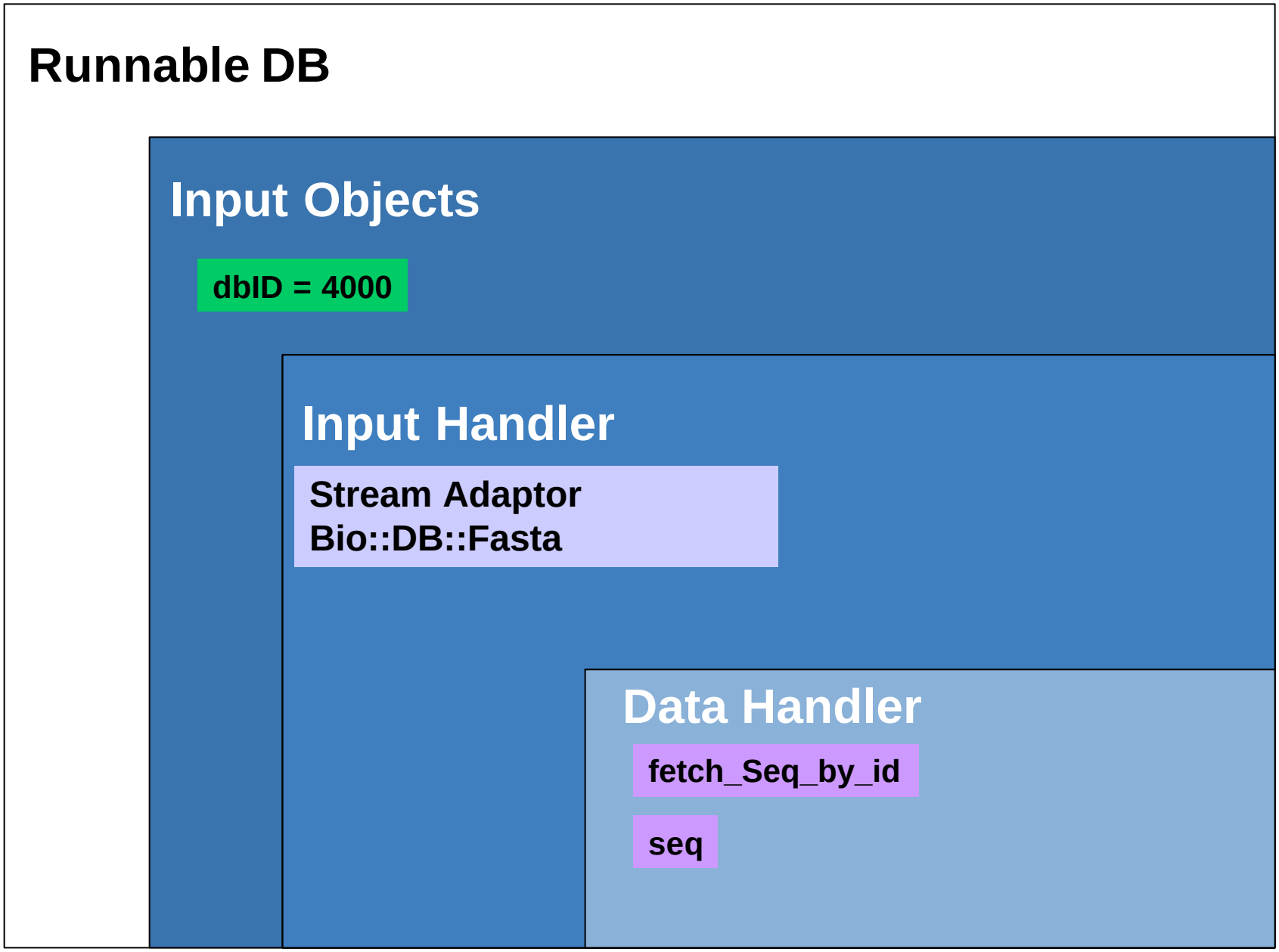

Input Layer
Analysis Layer
Output Layer
Job Management System (MySQL)
Bioperl Object Layer

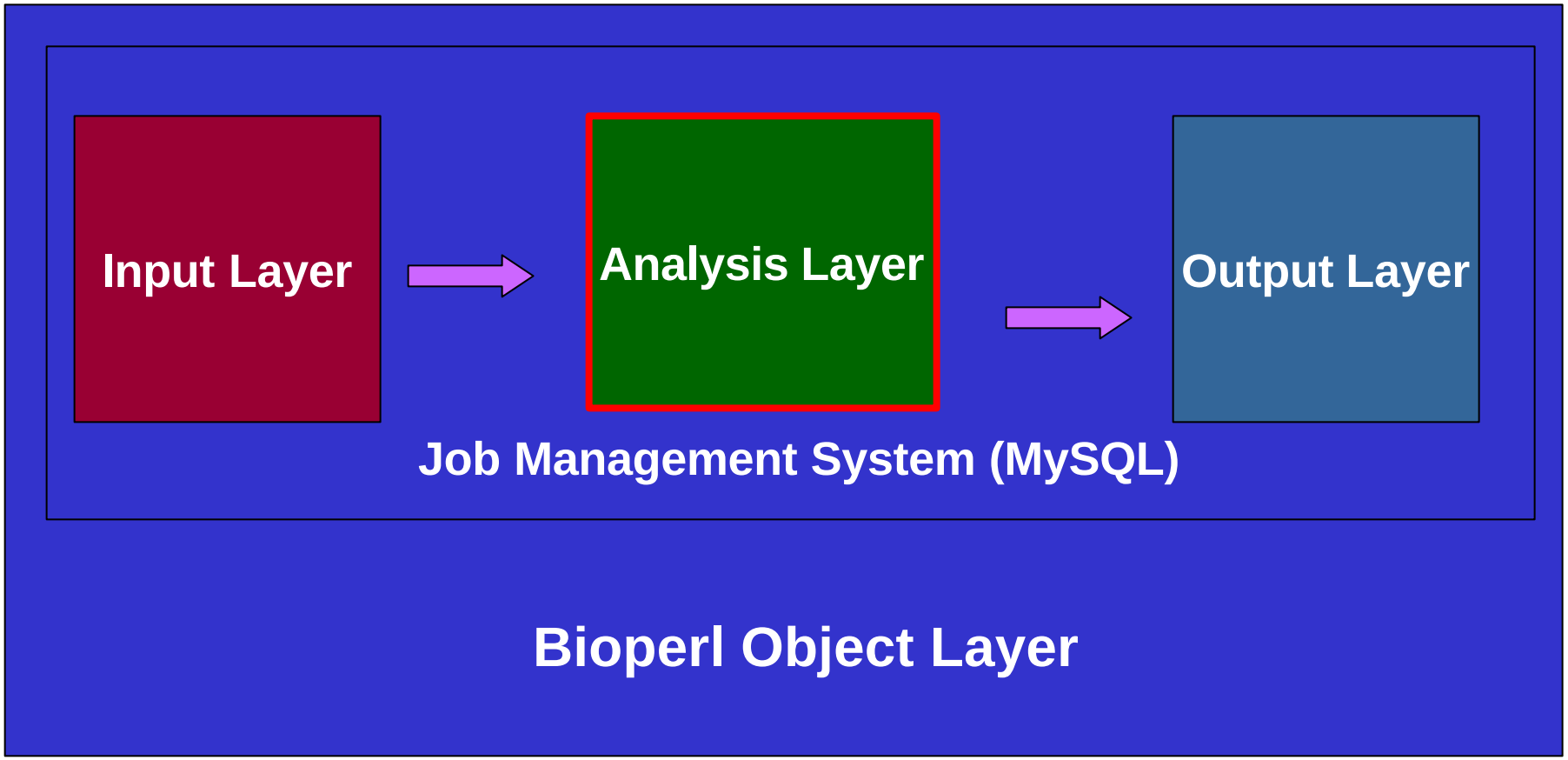
Input Layer

foreach $job {
$runnableDB->fetch_input

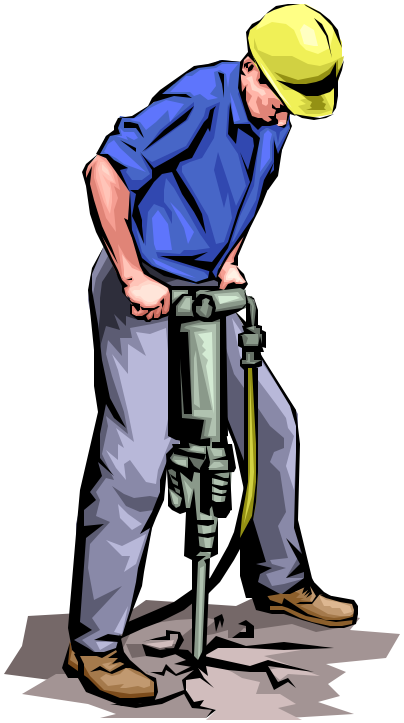



DBAdaptor
host : ensembl.fugu-sg.org
dbname : fugu_genes
user : shawn
driver : mysql
Bio::Seq
DisplayId
PrimaryId
Desc
Alphabet…
get_ProteinAdaptor
seq
dbID = 4000
fetch_by_dbID ()
ProteinAdaptor
schema specific
sql code
Protein
SeqFeatures
PrimaryId
AccessionId
Sequence ….
OBJECTS
INPUT NAME
METHOD

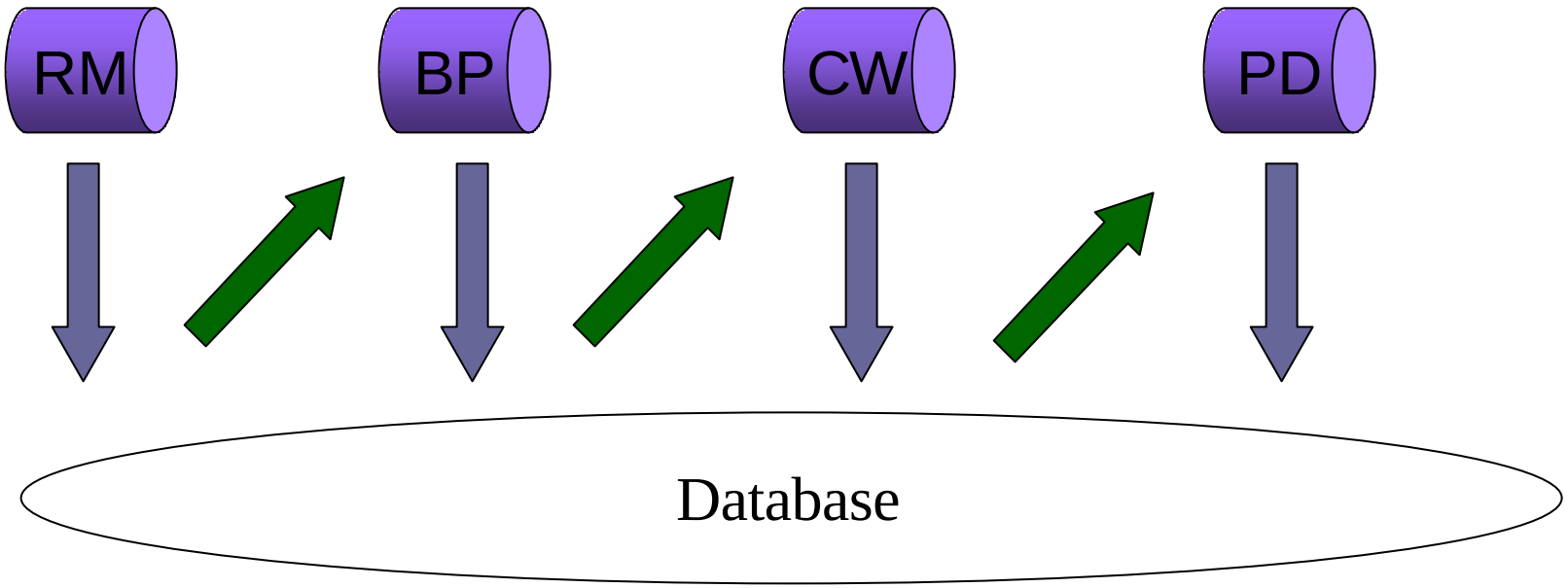

Runnable DB
Input Objects
dbID = 4000
Input Handler
Stream Adaptor
Bio::DB::Fasta
Data Handler
fetch_Seq_by_id
seq


Input Layer
Analysis Layer
Output Layer
Job Management System (MySQL)
Bioperl Object Layer

Analysis Layer



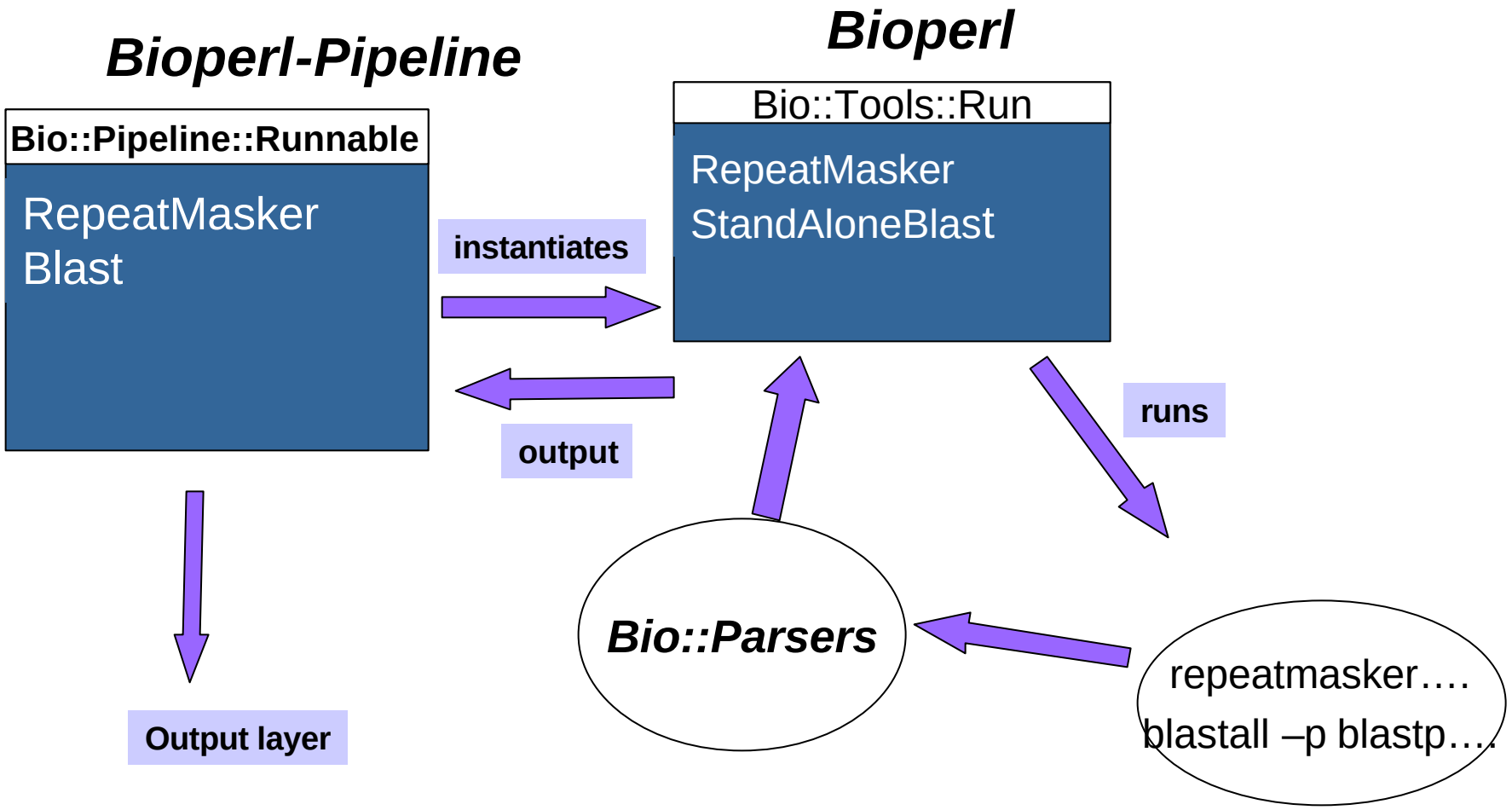
```
foreach $job {
        $runnableDB->fetch_input;
        $runnableDB->run;
```

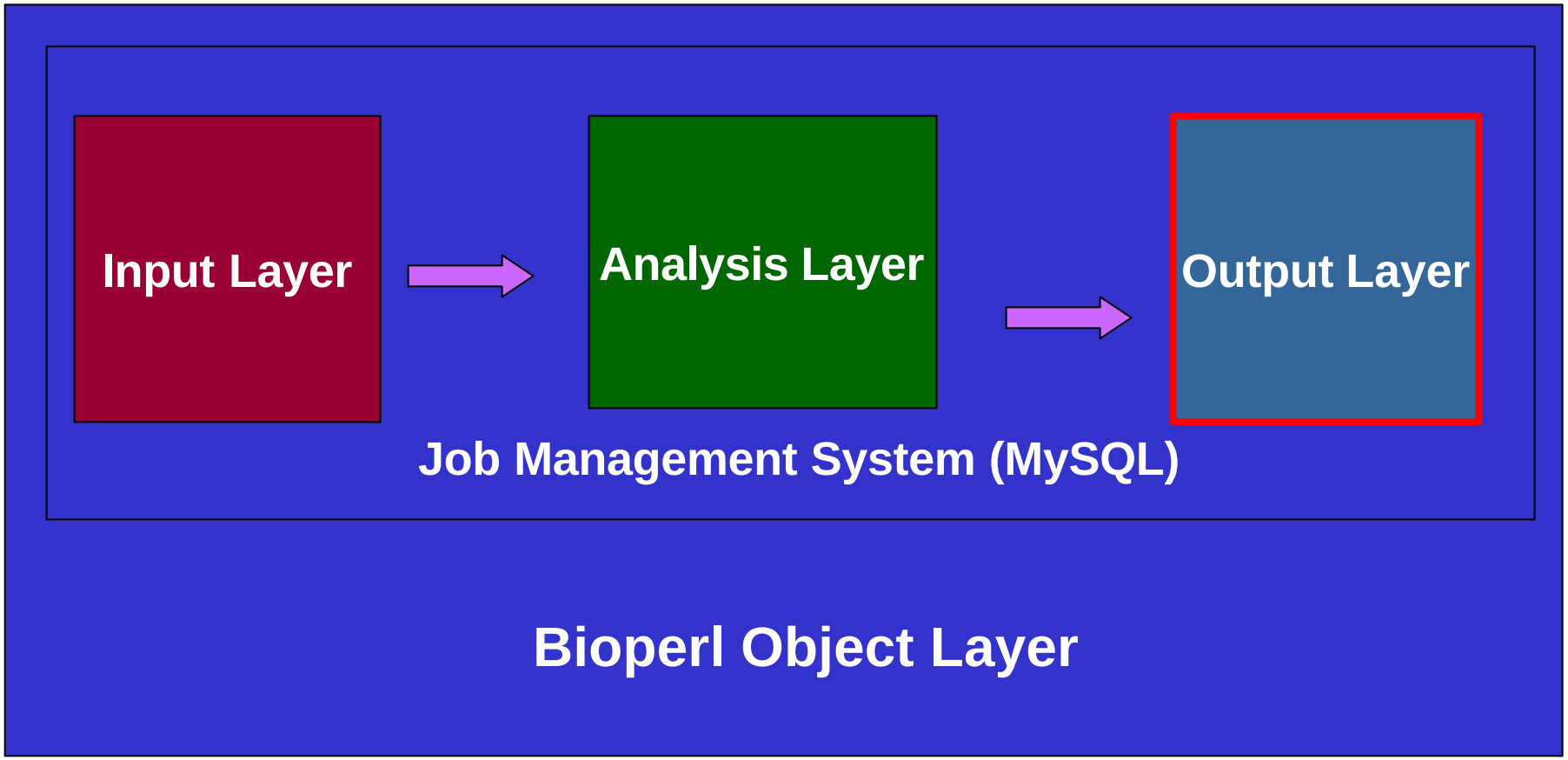


Each Analysis has a separate Runnable

Runnable

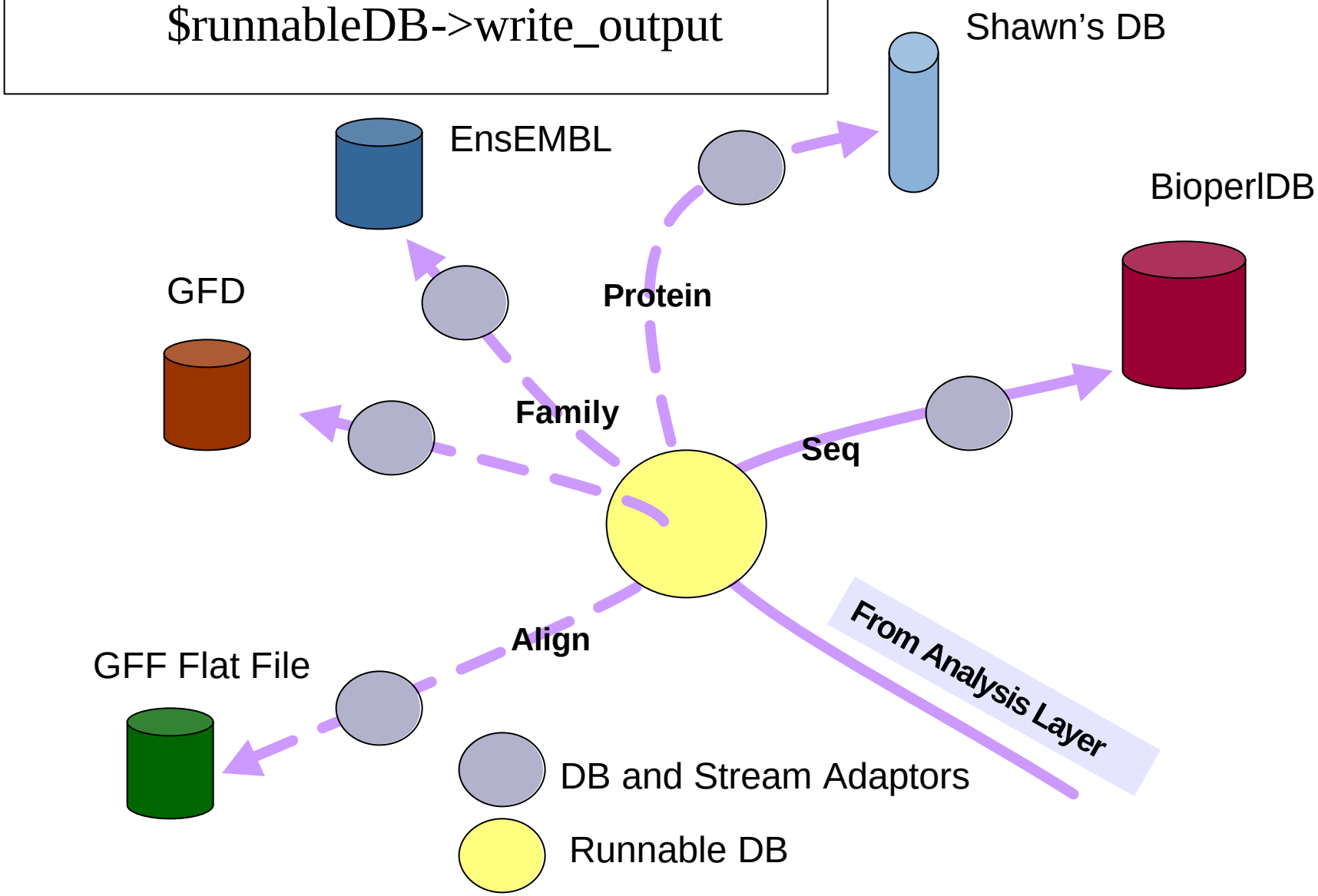
"A wrapper around the wrapper"

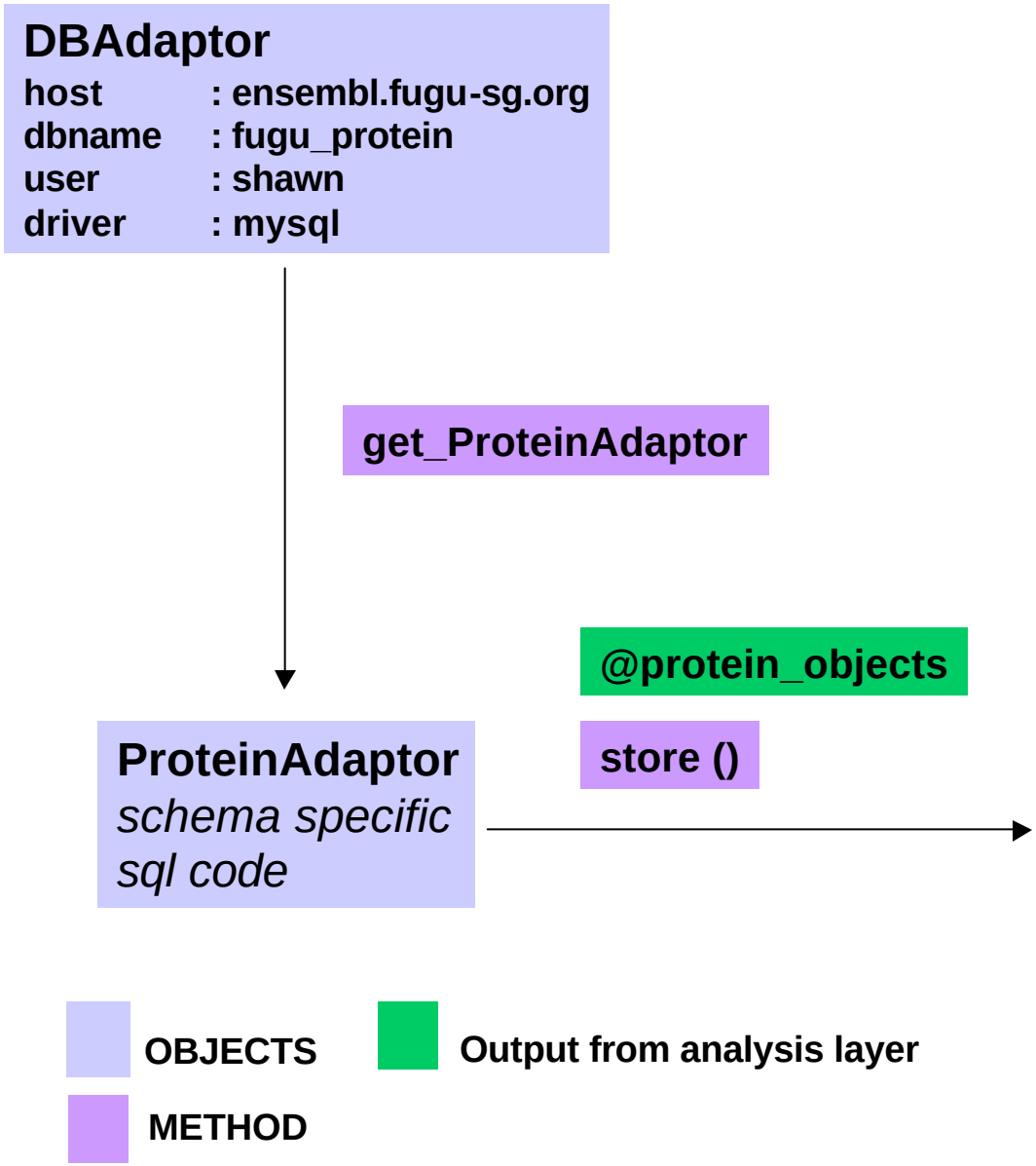


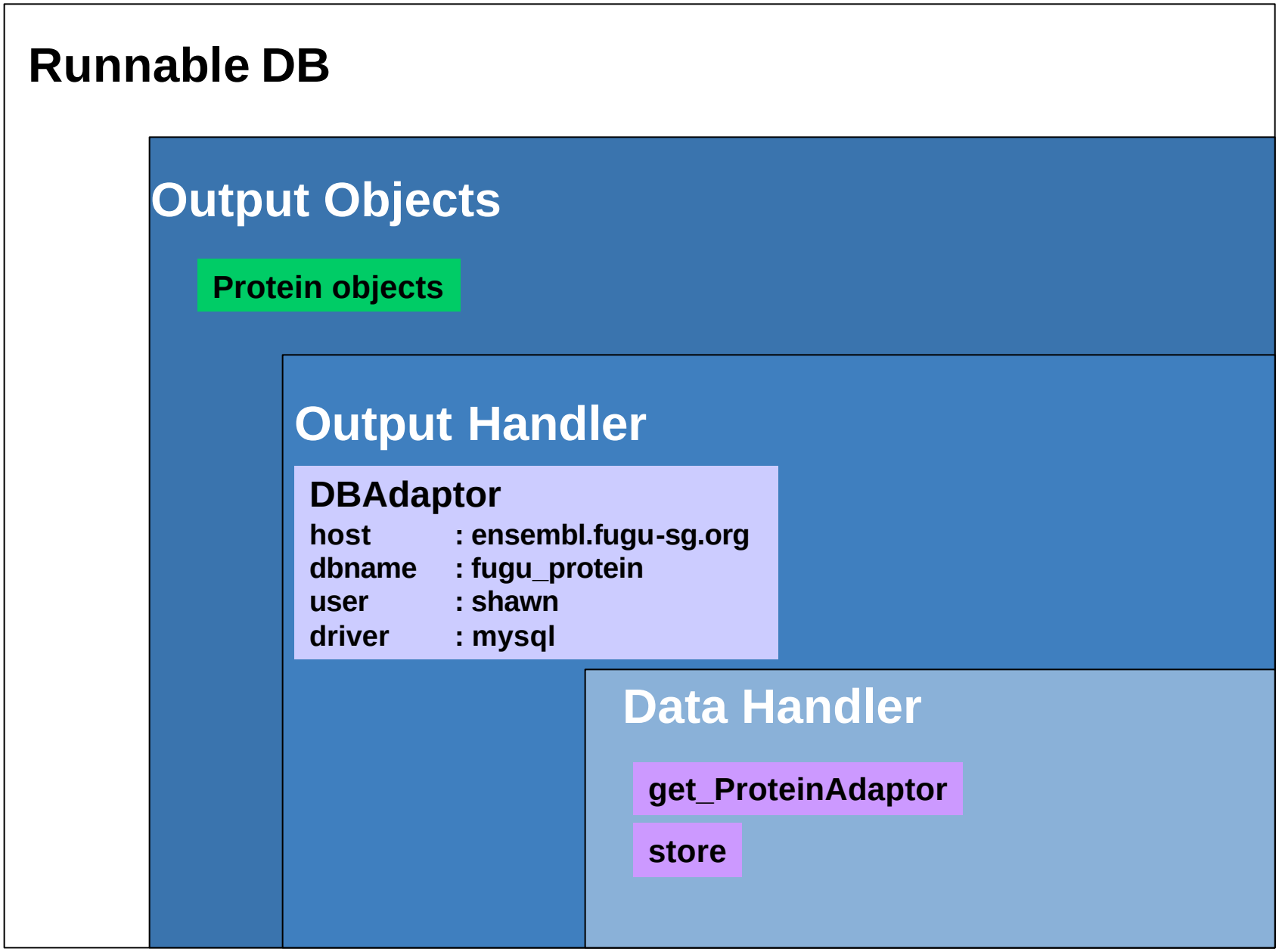

Input Layer
Analysis Layer
Output Layer
Job Management System (MySQL)
Bioperl Object Layer

Output Layer

```
foreach $job {
        $runnableDB->fetch_input;
        $runnableDB->run;
        $runnableDB->write_output
```




DBAdaptor
host : ensembl.fugu-sg.org
dbname : fugu_protein
user : shawn
driver : mysql
get_ProteinAdaptor
@protein_objects
ProteinAdaptor
schema specific
sql code
store ()
OBJECTS
Output from analysis layer
METHOD


Runnable DB
Output Objects
Protein objects
Output Handler
DBAdaptor
host : ensembl.fugu-sg.org
dbname : fugu_protein
user : shawn
driver : mysql
Data Handler
get_ProteinAdaptor
store

Wish List

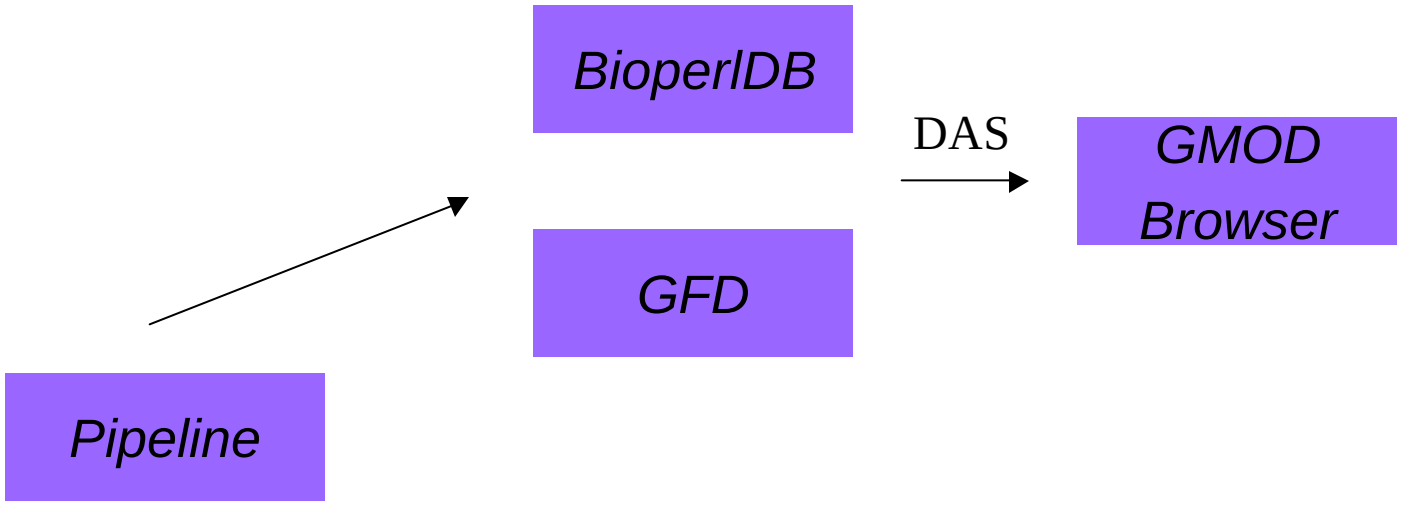
- Handle input data from disparate sources
- Modular approach to analysis (plug and play)
- Management of jobs running in parallel over a compute farm
- Output results to format/database of choice for easy interpretation
- Easy to package and reproduce

Pipeline to go

XML

```
<pipeline_flow_setup>
  <analysis id="1">
    <logic_name>blast</logic_name>
    <runnable>Bio::Pipeline:Runnable::Blast</runnable>
    <dbfile>/data0/family_run_17_6_2002/all_pep</dbfile>
    <program>blastall</program>
    <parameters>-p blastp -e 0.00001</parameters>
    <output_iohandler_id>1</output_iohandler_id>
    <nodegroup_id>1</nodegroup_id>
  </analysis>
<rule>
  <current_analysis_id>1</current_analysis_id>
  <next_analysis_id>2</next_analysis_id>
  <action>WAITFORALL_AND_UPDATE</action>
  </rule>
</pipeline_flow_setup>
```

Generic Feature Database

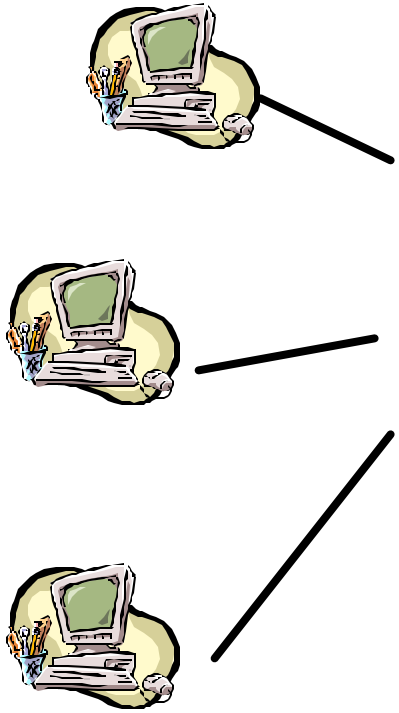


- Flexible Scheme to store Analysis Results
- Complex Biological Objects are aggregation of basic Entities and Relationships

More on Friday …

In the works…

Web-Pipeline



Pipelines Available

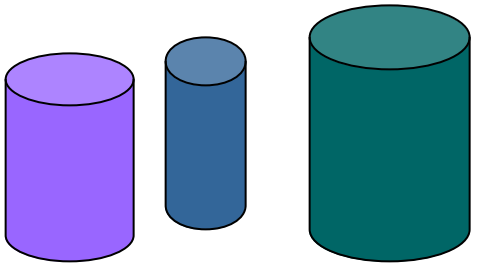
Blast

Family Clustering

Primate

Phylogenetic Analysis

Compute farm



User uploads sequences

Updated mirror of databases

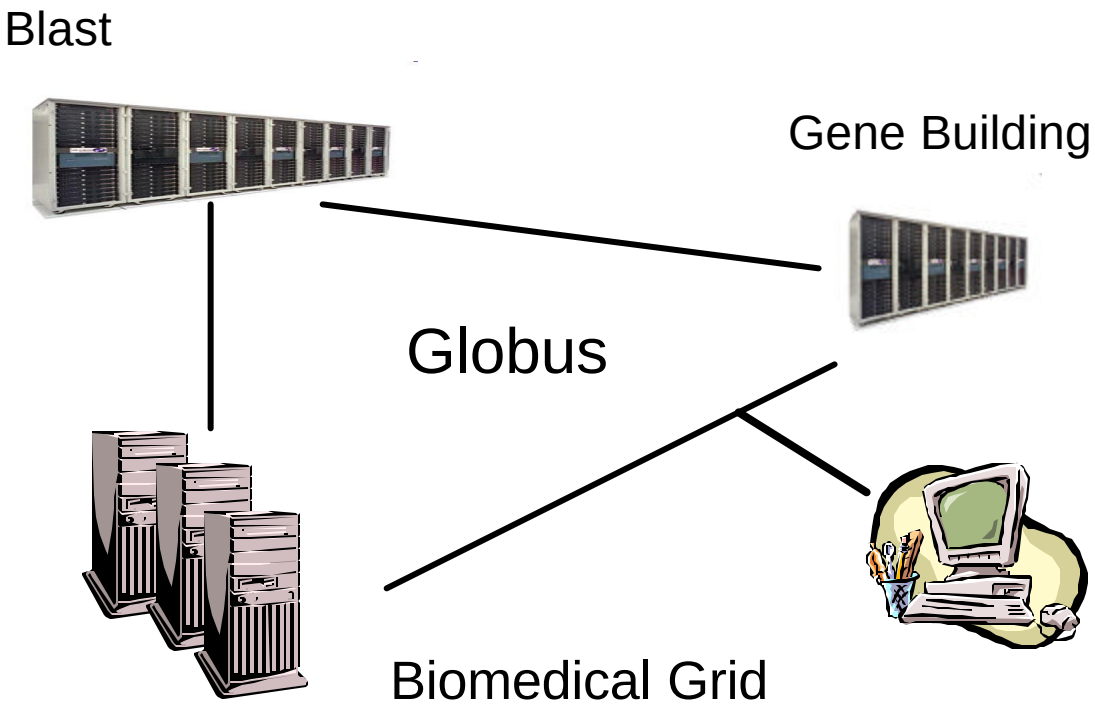
Customized Pipelines

In the works…

Grid-Pipeline

Heterogeneous Environment

Policy Issues



Availability

Developer's Version 0.1

Bioperl CVS

cvs -d :pserver:cvs@cvs.open-bio.org:/home/repository/bioperl checkout bioperl-pipeline

Web site

http://www.fugu-sg.org/bioperl-pipeline

Acknowledgements

Fugul Team

Elia Stupka

Chen Peng

Kiran Kumar Ratnapu

Allison Soo

Frans Verhoef

Shawn Hoon

Xiao Ju Guang

Bala Murugan

Luo Ming

Low Yi Jin

Chris Chong

Alumni

Jer-Ming Chia

Tania Oh