InterMine
Web-Services and Interoperability

InterMine
Open source data warehouse system

intermine.org

Richard Smith, Daniela Butano, Adrian Carr , Sergio Contrino, Hu Fengyuan, , Alex Kalderimis, Mike Lyne, Rachel Lyne, Julie Sullivan, Radek Štěpán, Gos Micklem

Systems Biology Centre, University of Cambridge, Tennis Court Road, Cambridge CB2 1QR, UK
+44 1223 760262
dev@intermine.org

InterMine is:
- An Integrated Data-Warehouse System
- An Optimised Query Engine
- A Graphical Web-Interface
- ***A REST-ful web-service***

Java	Perl	Python	Ruby	JS	
Web-Services					**Web-Application**
Internal API					
ObjectStore					
DataBase					

With the Webservices You Can:

Get Data:
- Fetch paged, type-safe results of arbitrary queries
- Summarise data (averages, counts)
- Use biological formats for features and intervals (GFF3, FASTA, BED) as input and output.
- Human readable (TSV) and machine readable (XML, JSON) output

Modify Stored Data:
- Create, modify, tag and delete lists

Introspect Services:
- Analyse the data-model
- View the structure of template queries
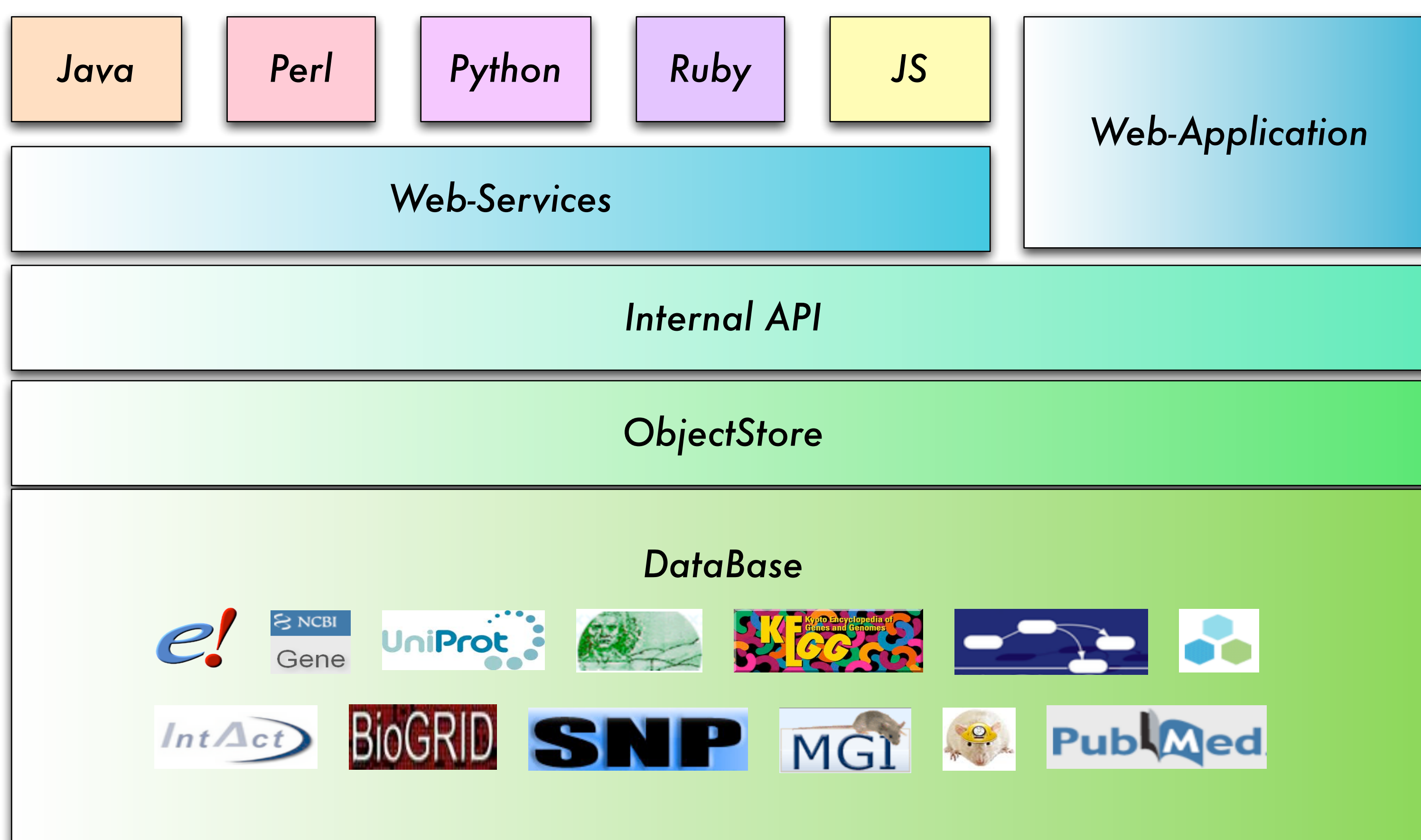- Retrieve a service listing.

view the full list of resources at *intermine.org/wiki/WebService*

We aim to expose all the fuctionality available to the internal API through HTTP based web-services, ultimately with the aim of enabling alternate front-end web applications to be built using them. Work to support this in active development is listed below:

In Development:
- List Analysis Results
- Web-Service powered result-tables
- User Management

Use Cases:

Bio-Informatics

The capability to script query operations allows users to automate complex but frequently used workflows, giving researchers fast, repeatable, parameterized access to the data they are looking for. Bio-informaticians can get straight to the data, without having to worry about parsing it into the correct form first.

A sample application, based on a real-life use case by one of our users:

```python
#!/usr/bin/python
# From a file containing a set of intervals identified as
# regulating gene expression, identify display the 3 regions
# with the highest number of genes

from intermine.webservice import Service
from interminebio import RegionQuery

intervals = open("expression_regions.bed").readlines()

s = Service("www.flymine.org/query", token = "TOKEN")
org = "D. melanogaster"
types = ["Gene"]
lists = [(i, s.create_list(RegionQuery(s, org, types, [i])))
                              for i in intervals]
top3 = sorted(lists, key=lambda x: x[1].size, reverse = True)[:3]

for interval, genes in top3:
    print "%s has %d genes" % (interval.strip(), genes.size)
    for gene in genes:
        loc = gene.chromosomeLocation
        vals = (gene.symbol if gene.symbol else gene.primaryIdentifier,
            gene.chromosome.primaryIdentifier,
            loc.start, loc.end)
        print "%s: %s:%d..%d" % vals
```

[code available at: http://gist.github.com/1331226]

This example shows the use of biological formats for input, the ability to access data in the native idiom of a client library's language without extra parsing of data, and the ability to use the query-engine's data model to do the hard work, leaving you to decide what is important.

Similar facilities exist in Perl, Java and Ruby, with libraries available from major public repositories (cpan.org, pypi.python.org, rubygems.org, intermine.org/lib).

## Site-Maintainers/developers: MODS

The InterMod project funds InterMine to work with Model-Organism database groups to set up and run data-warehouses. The current list of collaborators is:
- RGD (RatMine - ratmine.mcw.edu)
- SGD (YeastMine - yeastmine.yeastgenome.org)
- ZFin (ZebrafishMine)
- MGI (MouseMine)
- Wormbase (WormMine)

Of these groups, SGD and RGD already have publicly released data-warehouses, and FlyMine, a data-warehouse developed with Fly data is produced by the InterMine group in Cambridge.

For the most part, these groups are focussed on curation and community support, with little developer resource to invest in query optimization or data-warehousing. By implementing an InterMine, these groups are able to gain a rich feature-set for comparatively little effort, and most of their configuration is done by domain expert biologists, not specialist software developers.

The InterMine set of webservices allows these groups to bring these advantages back into their own websites. Projects are in place in the MOD community to replace existing search tools, report pages and results displayers with implementations powered by InterMine web-services. The extensive support for JSON in InterMine means consuming these services in web pages is straightforward with the many excellent modern JavaScript tools that are now available. MODs can consume their own, and others' data equally, meaning richer experiences for users.
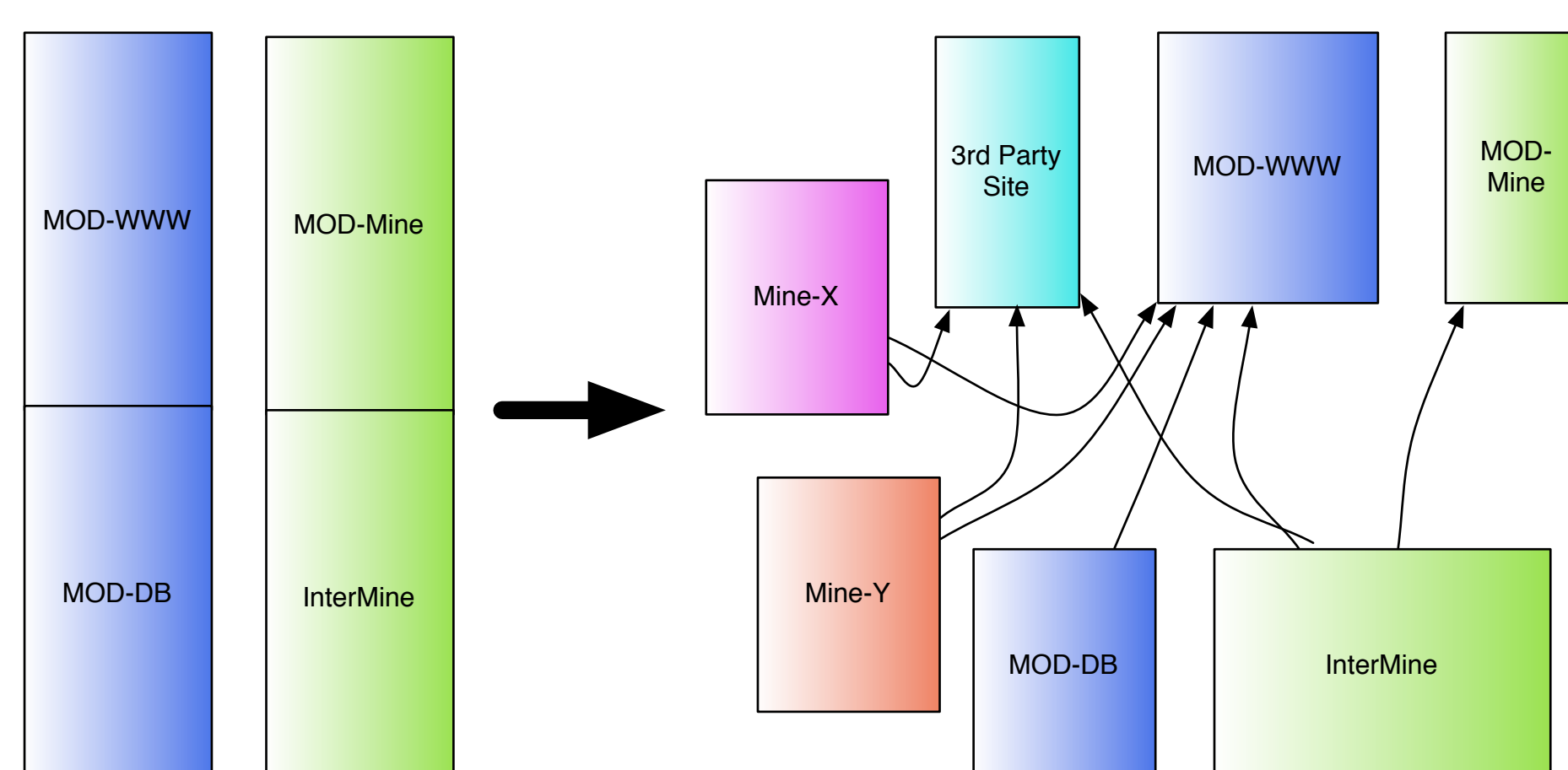
Inter-project Communication: Galaxy

Powerful web-services mean that InterMine instances are able to provide data to other projects, in mutually beneficial exchanges. One excellent example of this has been the InterMine-Galaxy interoperation, which is currently undergoing heavy development.

InterMine provides the ability to analyse data by its relationships to other data. Galaxy provides a workspace for consequence analysis. By allowing the two tools to talk together, these two projects each gain the other's strengths.

InterMine Queries / *Galaxy Workflow*
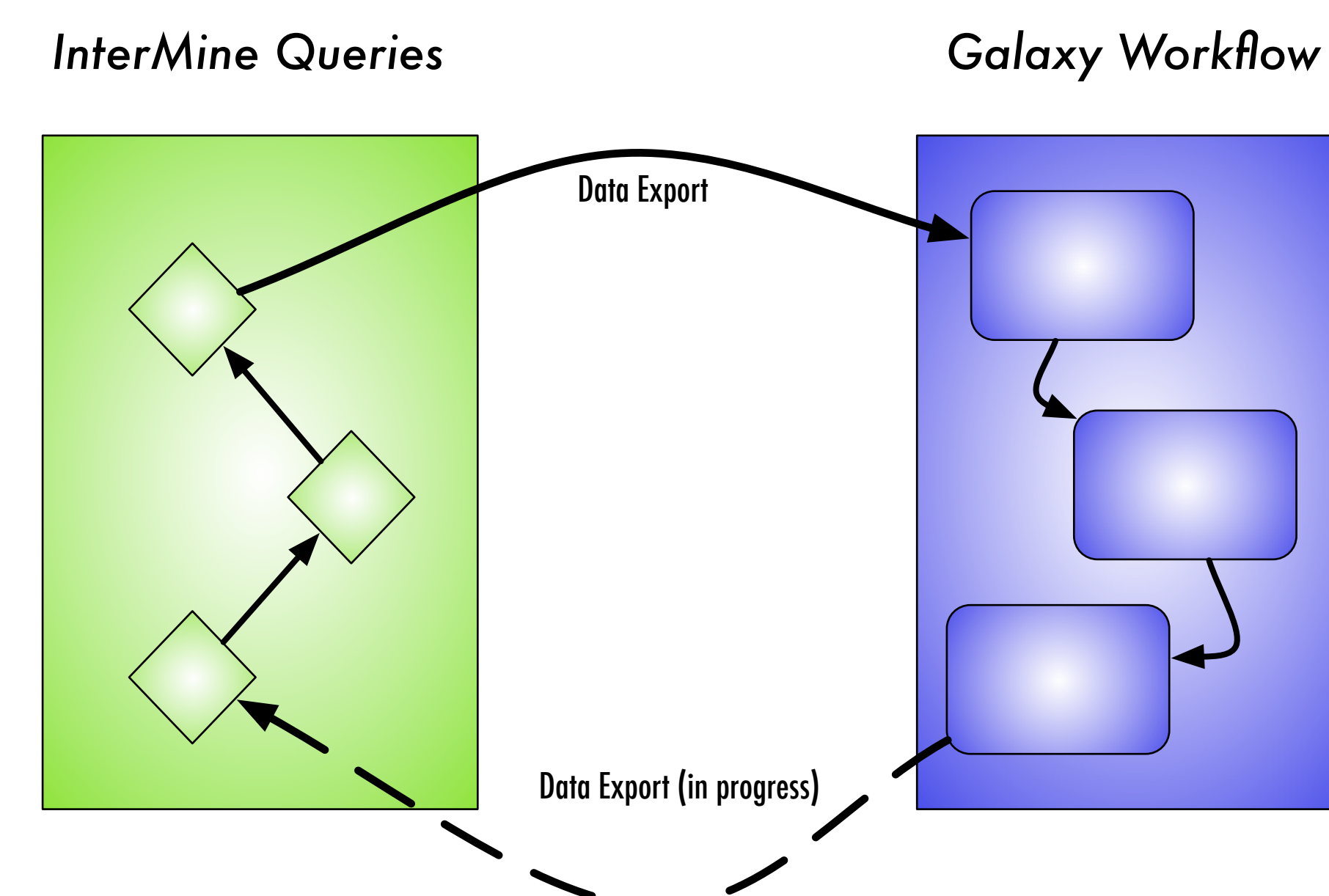
Currently InterMine has data export tools in Galaxy and as part of the core web-appliation that allow data to be imported into Galaxy. New InterMine tools will allow data to flow backwards and forwards between Galaxy, allowing for complex rounds of computational biology combining consequence and relationship analysis. The Galaxy API which is currently under construction will allow InterMine to gain full access to powerful workflow tools, in the same way that InterMine provides other projects with sophisticated query resources.

Visit galaxy at: www.usegalaxy.org