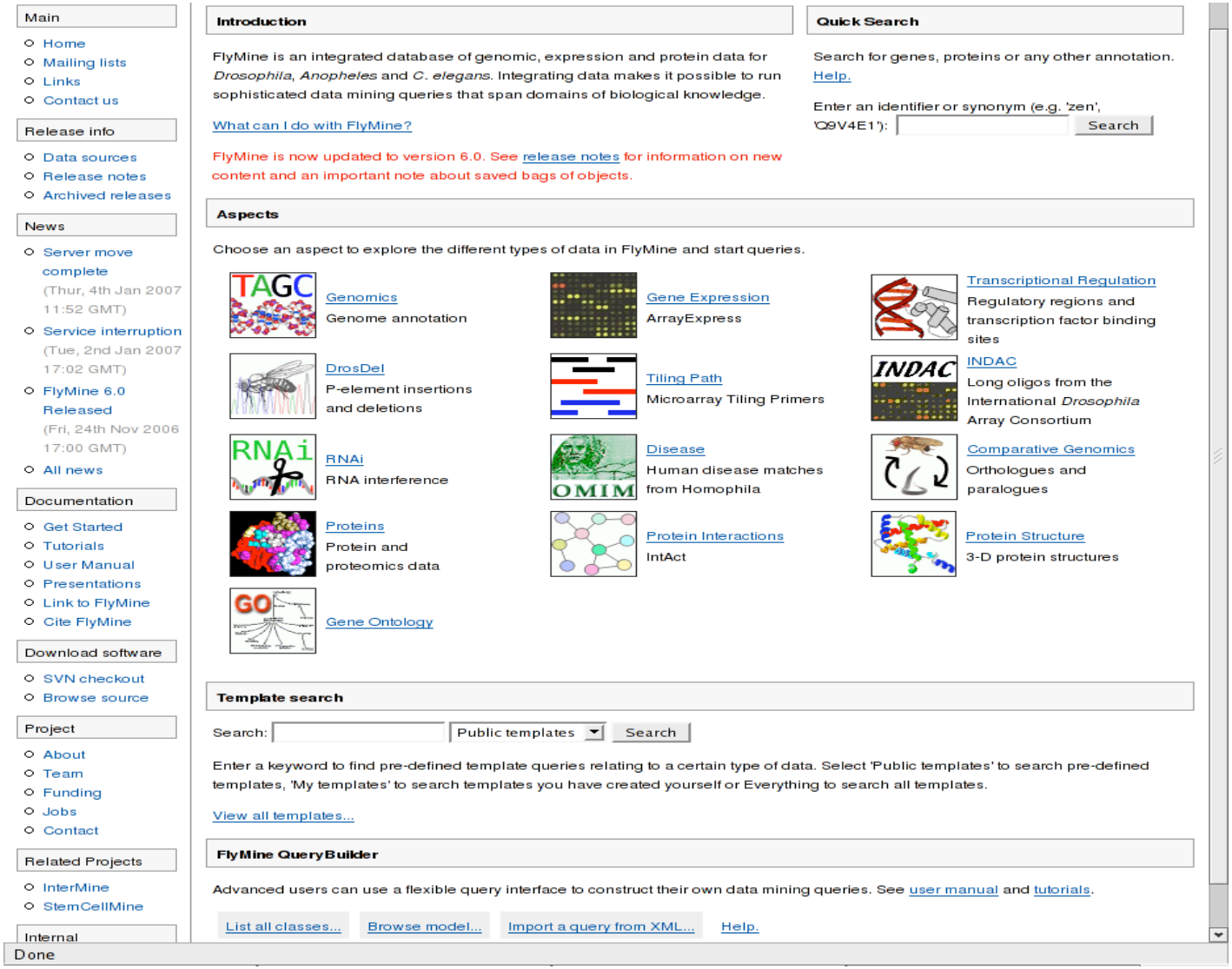
InterMine
Data Warehouse

Gos Micklem

InterMine used to build FlyMine

An integrated database of Drosophila and Anopheles data

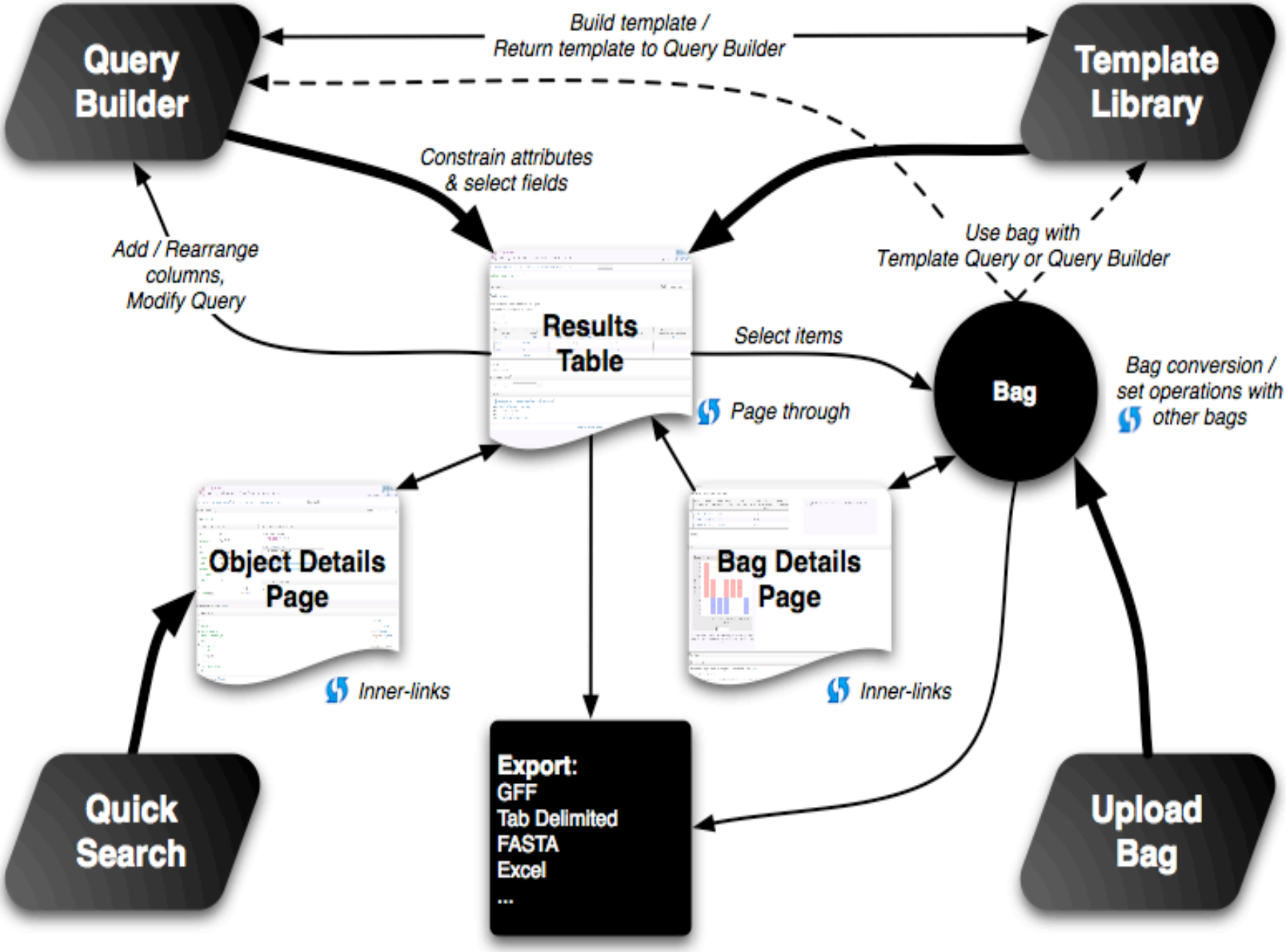


Project Stats

- Team of 7 FTE
  - 5 developers, one sys admin,
  - 1 biologist/ bioinformatician
- Java/ postgreSQL
- SVN: 125,000 lines of code + 57,000 lines of tests
- Started in 2002
- In use by others in Cambridge, Edinburgh, Vienna… + modENCODE DCC if funded
- modENCODE/ Chado

InterMine Guiding Principles

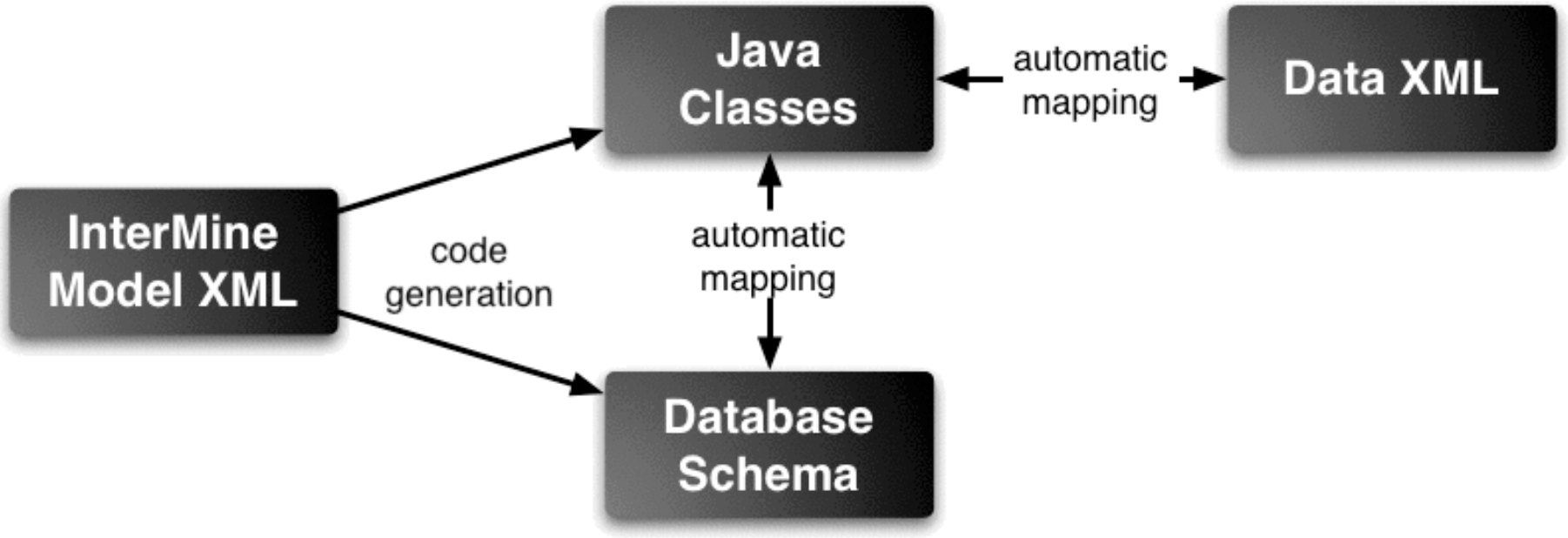
- Open Source
- Generic
- Maximum us of automatic code generation/ testing
- Integrate diverse types of data - standard formats
- Load order independent (fine-grained source priorities)
- Query re-writing: uncouple data model and tuning
- Allow performance tuning of the live database
- Allow domain-specific configuration of the user interface
    - By domain experts not developers
    - Release independently of database
- Scope: small project databases to large scale data warehouses


Query Builder
Build template /
Return template to Query Builder
Template Library
Constrain attributes
& select fields
Add / Rearrange
columns,
Modify Query
Use bag with
Template Query or Query Builder
Results Table
Select items
Page through
Bag
Bag conversion /
set operations with
other bags
Object Details Page
Inner-links
Bag Details Page
Inner-links
Export:
GFF
Tab Delimited
FASTA
Excel
...
Quick Search
Upload Bag

Technical Overview

- Custom Java object/relational system
  - When we started, couldn't select from multiple classes at one time using hibernate.
- Optimised for read-only performance
- Designed for big, complex queries
- Performance optimisation
  - Transparent query re-writing
- Web application - Struts/JSP/Ajax

Code Generation



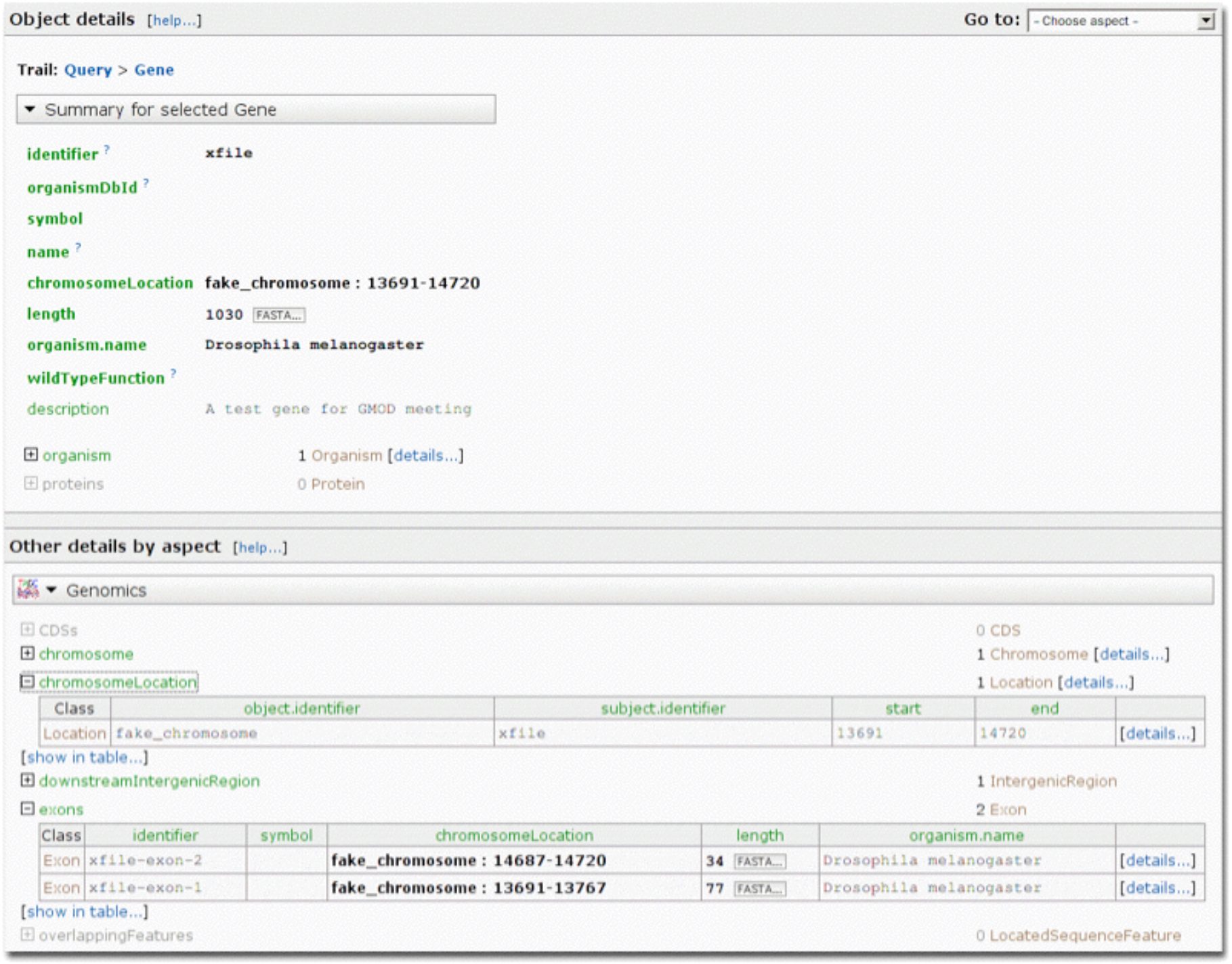
We use SOFA as core data model - similar to Chado.
Added Gene.description, compiled, loaded data (here XML + FASTA), released webapp.

Loading Data

- Read-only in production environment
- Load data from InterMine XML
- Parsers from standard formats
  - e.g. UniProt, GFF3, PSI, FASTA
- Powerful integration system

Example InterMine XML

```
<items>
   <item id="0_3" class=”” 
implements="http://www.flymine.org/model/genomic#Gene">
      <attribute name="identifier" value="xfile" />
      <attribute name="description" value="A test gene for GMOD 
meeting" />
      <reference name="organism" ref_id="0_1" />
      <collection name="transcripts">
         <reference ref_id="0_9" />
      </collection>
   </item>
   <item id="0_1" class="" 
implements="http://www.flymine.org/model/genomic#Organism">
      <attribute name="taxonId" value="7227" />
   </item>
   ...
```



Object details [help...] Go to: - Choose aspect -

Trail: Query > Gene

Summary for selected Gene

identifier ?	xfile
organismDbId ?	
symbol	
name ?	
chromosomeLocation	fake_chromosome : 13691-14720
length	1030 FASTA...
organism.name	Drosophila melanogaster
wildTypeFunction ?	
description	A test gene for GMOD meeting

organism 1 Organism [details...]

proteins 0 Protein

Other details by aspect [help...]

Genomics

CDSs 0 CDS

chromosome 1 Chromosome [details...]

chromosomeLocation 1 Location [details...]

Class	object.identifier	subject.identifier	start	end	
Location	fake_chromosome	xfile	13691	14720	[details...]

[show in table...]

downstreamIntergenicRegion 1 IntergenicRegion

exons 2 Exon

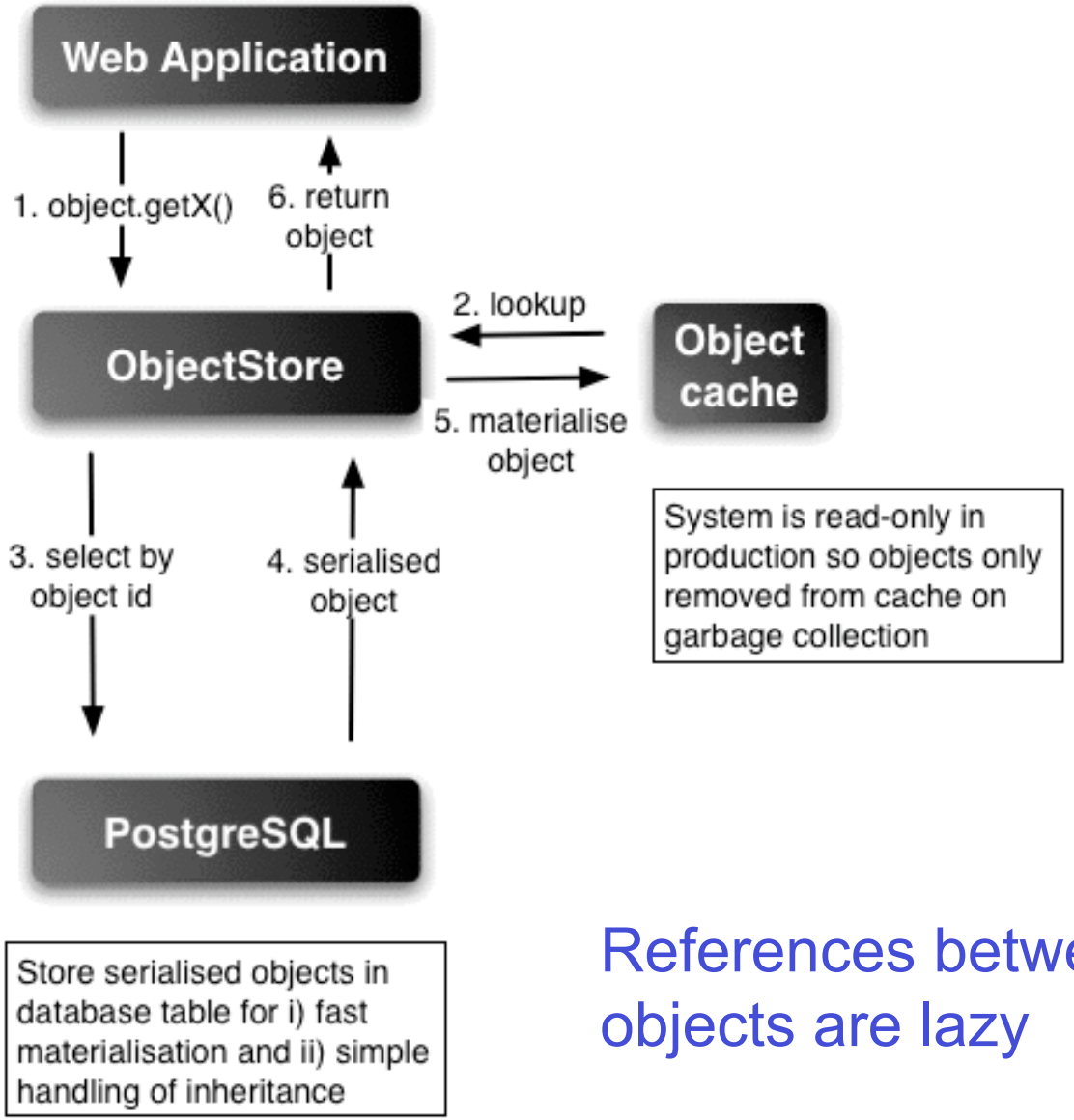
Class	identifier	symbol	chromosomeLocation	length	organism.name	
Exon	xfile-exon-2		fake_chromosome : 14687-14720	34 FASTA...	Drosophila melanogaster	[details...]
Exon	xfile-exon-1		fake_chromosome : 13691-13767	77 FASTA...	Drosophila melanogaster	[details...]

[show in table...]

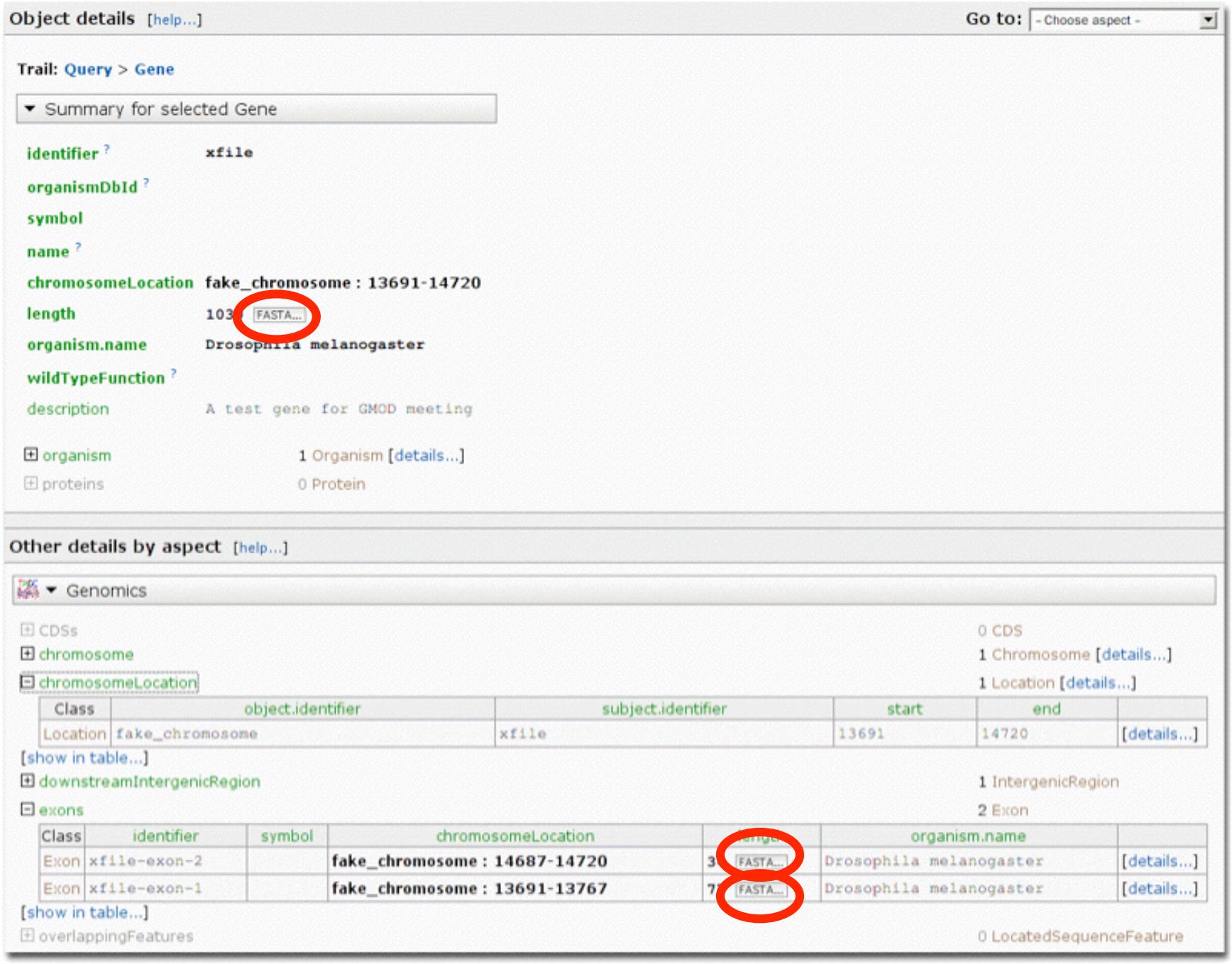
overlappingFeatures 0 LocatedSequenceFeature

Retrieving Objects

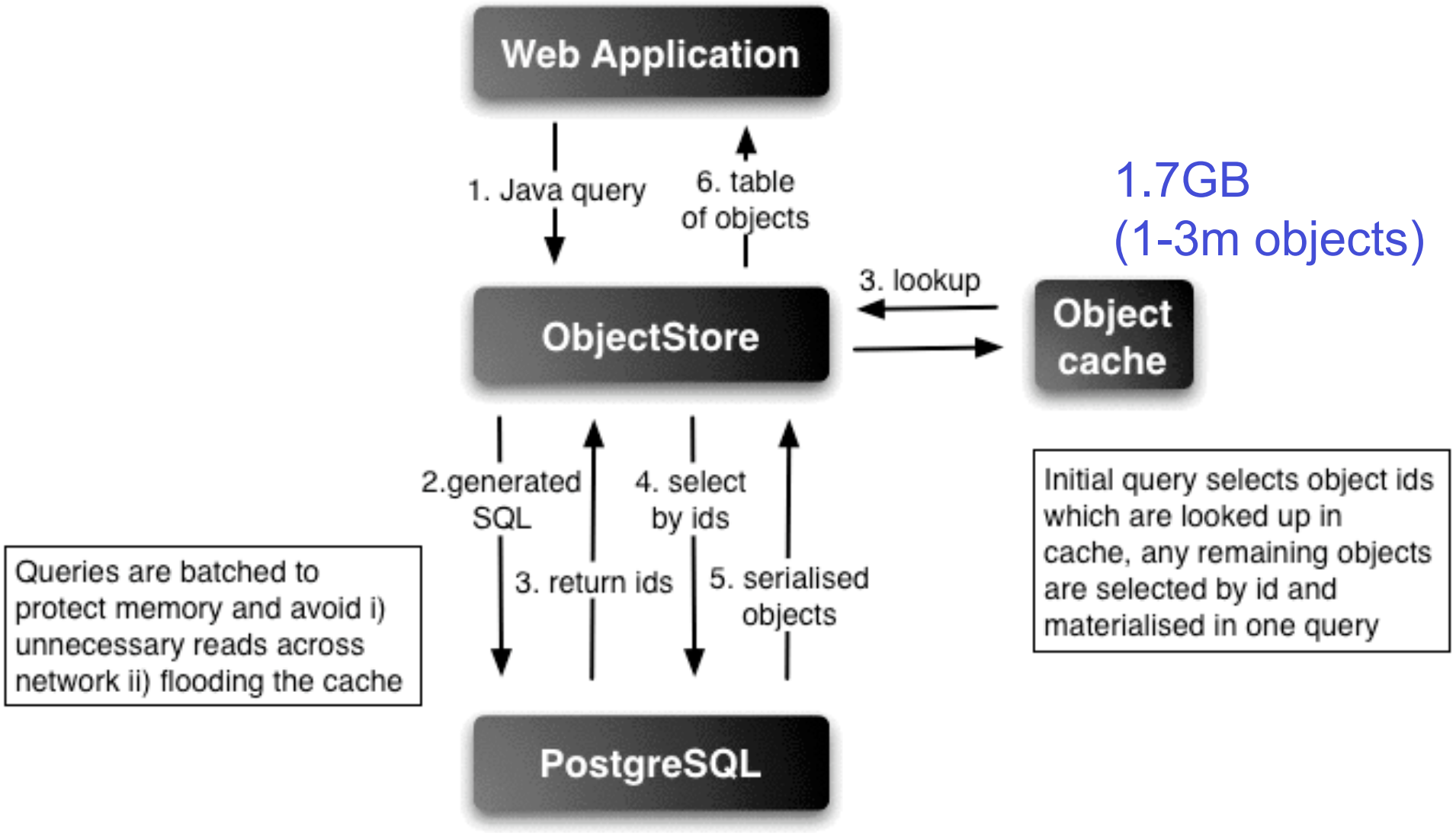
(Unrelated to commercial product)



References between objects are lazy


Object details [help...]
Go to: - Choose aspect -
Trail: Query > Gene
Summary for selected Gene
identifier ? xfile
organismDbId ?
symbol
name ?
chromosomeLocation fake_chromosome : 13691-14720
length 103 FASTA...
organism.name Drosophila melanogaster
wildTypeFunction ?
description A test gene for GMOD meeting
organism 1 Organism [details...]
proteins 0 Protein
Other details by aspect [help...]
Genomics
CDSs 0 CDS
chromosome 1 Chromosome [details...]
chromosomeLocation 1 Location [details...]
Class object.identifier subject.identifier start end
Location fake_chromosome xfile 13691 14720 [details...]
[show in table...]
downstreamIntergenicRegion 1 IntergenicRegion
exons 2 Exon
Class identifier symbol chromosomeLocation organism.name
Exon xfile-exon-2 fake_chromosome : 14687-14720 FASTA... Drosophila melanogaster [details...]
Exon xfile-exon-1 fake_chromosome : 13691-13767 FASTA... Drosophila melanogaster [details...]
[show in table...]
overlappingFeatures 0 LocatedSequenceFeature

Executing a Query



InterMine Queries

Quicksearch

```
Java API:
  Query q = new Query();
  QueryClass qcObj = new QueryClass(Gene.class);
  q.addFrom(qcObj);
  q.addToSelect(qcObj);

  QueryField qf = new QueryField(qcObj, "identifier");

  SimpleConstraint sc = new SimpleConstraint(qf, ConstraintOp.MATCHES, new
        QueryValue("x-%"));
  q.setConstraint(sc);

IQL:
  SELECT DISTINCT a1_.identifier AS a2_ FROM org.flymine.model.genomic.Gene
AS a1_ WHERE a1_.identifier LIKE 'x-%'


Perl API:
  my $genes = InterMine::Gene::Manager->get_genes(query => [
                                identifier => { like => 'x-%' },],);
```

Larger Query

Results for template: **Gene --> Transcripts and exons + chromosomal locations and lengths.**

For a particular gene, list the transcripts and exons and show the chromosomal locations and lengths of the transcripts and exons.

Page size 10

First Previous Next Last

Gene > identifier	Gene > transcripts > identifier	Gene > transcripts > chromosome > identifier	Gene > transcripts > chromosomeLocation > start	Gene > transcripts > chromosomeLocation > end	Gene > transcripts > length	Gene > transcripts > exons > identifier
CG2328	CG2328-RA [MRNA]	2R	5491054	5492592	1468	CG2328:1
CG2328	CG2328-RA [MRNA]	2R	5491054	5492592	1468	CG2328:2

Displaying all 2 rows

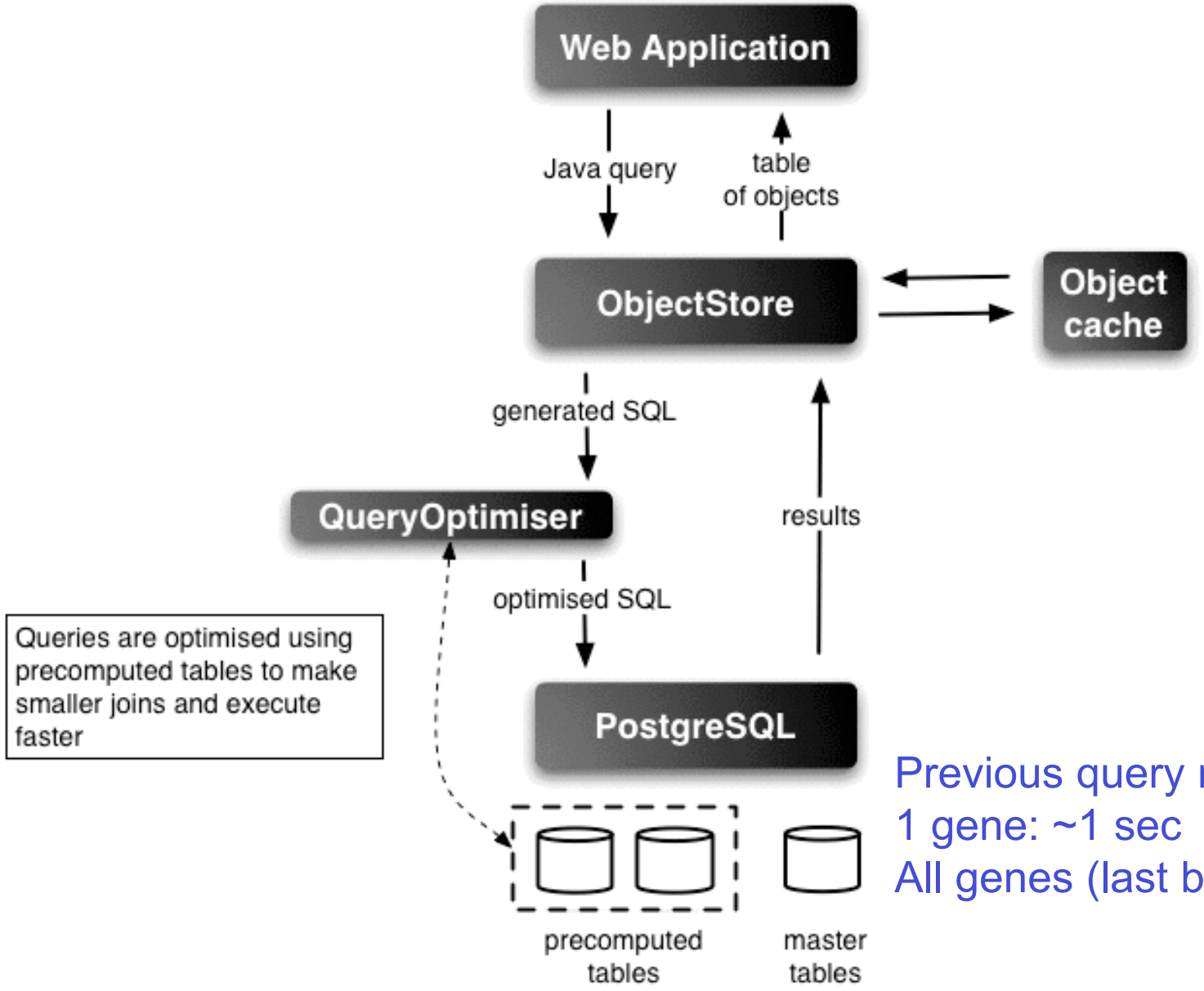
First Previous Next Last

- Joins 7 classes - all are on select list of query
- Here, one gene (two rows) ~2 seconds
- All genes, all organisms ~300,000 rows in 36 sec (not via
- pre-computation)

```
public class BakeOff {
    public static void main(String[] args) throws Exception {
        // code to get the "xfile" gene
        ObjectStore os = ObjectStoreFactory.getObjectStore("os.production");
        Query q = new Query();
        QueryClass qcObj = new QueryClass(Gene.class);
        q.addFrom(qcObj);
        QueryField qf = new QueryField(qcObj, "identifier");
        q.addToSelect(qf);
        SimpleConstraint sc = new SimpleConstraint(qf, ConstraintOp.EQUALS, new QueryValue("xfile"));
        q.setConstraint(sc);
        System.err.println("query: " + q);
        Results res = os.execute(q);

        // a Results object is a List of Lists
        List rr = (List) res.get(0);
        Gene gene = (Gene) rr.get(0);

        System.err.println ("symbol: " + gene.getIdentifier());

        // a BioEntity in FlyMine has a collection of Synonym objects -
        // we need Synonym.value for each Synonym
        System.err.print ("synonyms: ");
        Iterator synIter = gene.getSynonyms().iterator();
        while (synIter.hasNext()) {
            Synonym syn = (Synonym) synIter.next();
            System.err.print (syn.getValue() + ' ');
        }
```

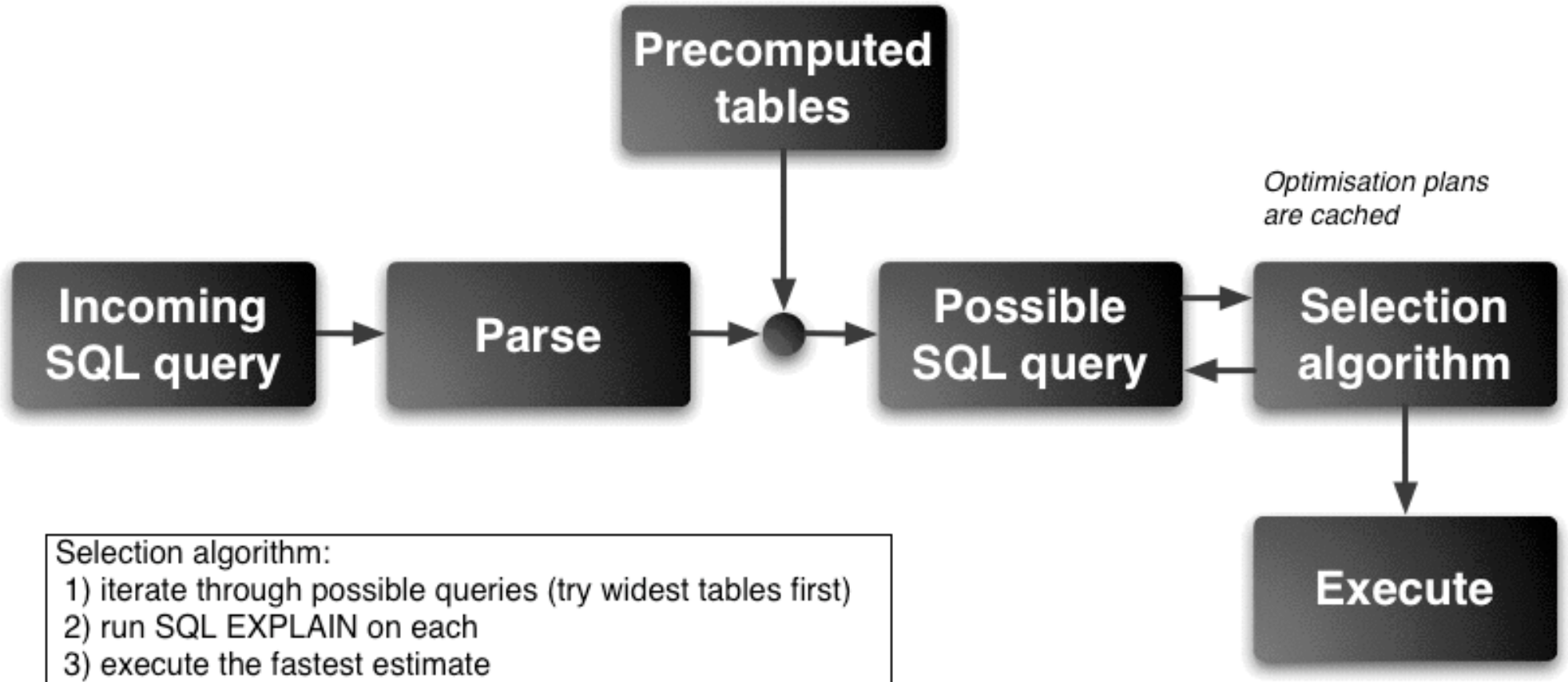
```
System.err.println ("description: " + gene.getDescription());

        // get the class name, but we already know that the gene is a Gene
        System.err.println ("type: " + gene.getClass().getName());

        // make a List from a the Set of exons for this Gene
        List exons = new ArrayList(gene.getExons());
        Exon exon1 = (Exon) exons.get(0);
        Exon exon2 = (Exon) exons.get(1);

        // get the start and end via the Location object
        System.err.println ("exon1 start: " + exon1.getChromosomeLocation().getStart());
        System.err.println ("exon1 end: " + exon1.getChromosomeLocation().getEnd());
        System.err.println ("exon2 start: " + exon2.getChromosomeLocation().getStart());
        System.err.println ("exon2 end: " + exon2.getChromosomeLocation().getEnd());

        // write out the first cds
        List cdss = new ArrayList(gene.getCDSs());
        FlyMineSequence flymineSequence = FlyMineSequenceFactory.make((CDS) cdss.get(0));

        // use BioJava to output the sequence
        Annotation annotation = flymineSequence.getAnnotation();
        annotation.setProperty(FastaFormat.PROPERTY_DESCRIPTIONLINE,
                               gene.getIdentifier() + " cds");
        SeqIOTools.writeFasta(System.err, flymineSequence);
    }
}
```

Query Optimisation



Previous query now:
1 gene: ~1 sec
All genes (last batch): ~1sec

Query Optimisation



Selection algorithm:
1) iterate through possible queries (try widest tables first)
2) run SQL EXPLAIN on each
3) execute the fastest estimate

Heuristics to improve performace:
a) avoid optimising very simple queries
b) run immediately if a one table query found
c) minimise number of EXPLAINs needed
d) avoid EXPLAINs on very large joins
e) bail out if taking too long

Query Optimisation - implications

- Performance optimisation not tied to schema design
- Can adapt performance optimisation to usage
- Template queries pre-computed (40 run per gene details page - renders in seconds)

Future Directions

- Import of any Chado database
- Improvements to build process
- User groups for sharing of templates/ bags
- Public bags/ tagging
- Relative genome coordinate based queries
- REST-style web services from templates
- Elaboration of bag details page:
  - graphical widgets, bag comparison, protein interaction graph functions, gene expression, generalised GoSTAT, chromosome distribution, enriched publications…

Cons

- slow build times (which are getting better)
- lots of disk space but it's a data warehouse and disks are cheap
- configuration is still a bit complicated, but improving fast
- we can't do much with locations yet except query starts and ends, although we do have the overlappingFeatures collections now...
- export is still limited to tab or comma delimited or FASTA
- we don't handle sequence very well:
  e.g. each chromosome sequence is stored in one big text field in PostgreSQL
- we can't do queries involving sizes of collections of things (e.g. find the genes with only 1 transcript)

Acknowledgements

Richard Smith
Kim Rutherford
Matthew Wakeling
Xavier Watkins
Julie Sullivan
Rachel Lyne
Hilde Janssens
François Guillier
Philip North
Gos Micklem

Andrew Varley, Mark Woodbridge, Tom Riley, Peter Mclaren, Debashis Rana, Wenyan Ji, Markus Brosch, Florian Reising

www.flymine.org www.intermine.org

FlyMine is funded by the Wellcome Trust (grant no. 067205), awarded to M. Ashburner, G. Micklem, S. Russell, K. Lilley and K. Mizuguchi.

Bioinformatics Position in Berkeley

www.berkeleybop.org/jobs

- Work with Chris Mungall and Gos Micklem on Chado and InterMine
- Gather requirements and design data models for novel, high-throughput, large-scale experiments
- Build a workflow system, a la UCSC, for data submission
- Integrate data from WormBase and FlyBase
- Integrate database with GBrowse, BioMart, and other software
- Applications and questions may be sent to Nicole Washington (nlwashington@lbl.gov)