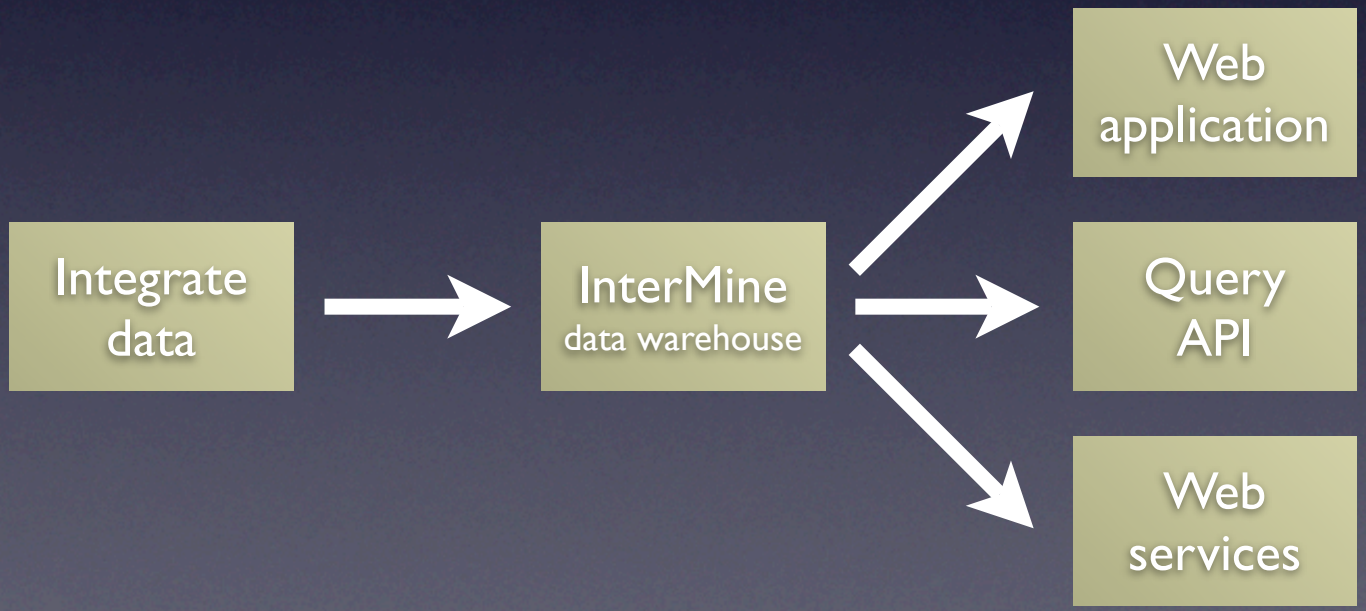
InterMine and Chado

Richard Smith
University of Cambridge

Overview

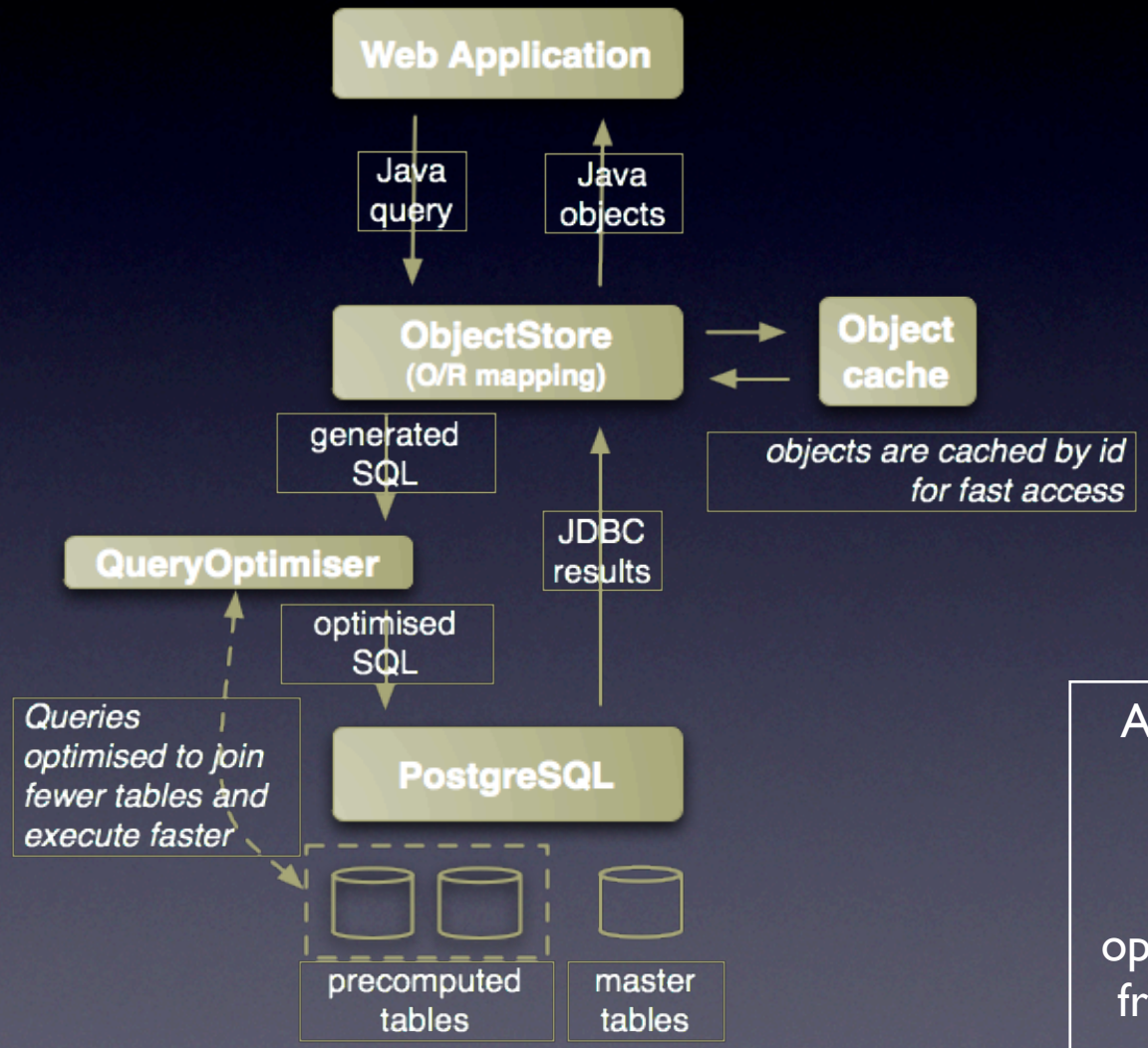
- Query-optimised data warehouse system
- Java, object-based data model
- Flexible querying



Projects

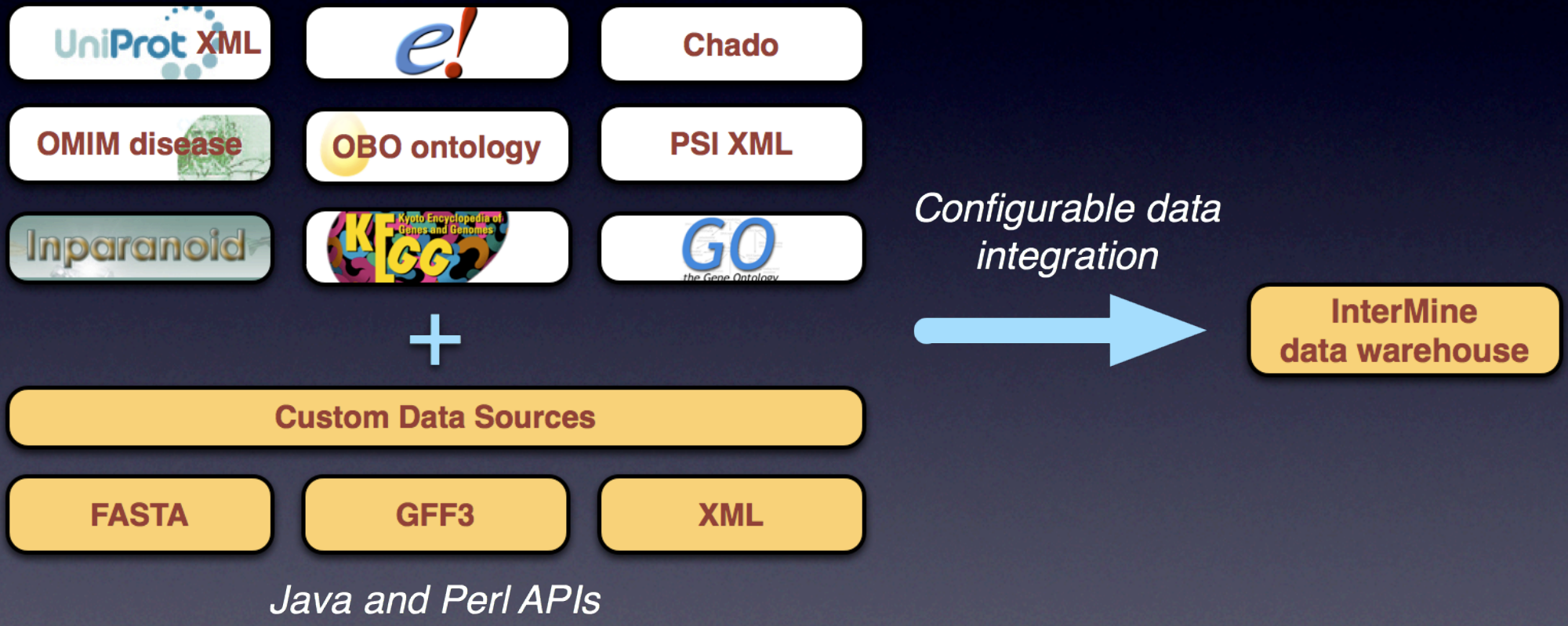
- FlyMine - www.flymine.org
  - 30+ data sources, *Drosophila & Anopheles*
- modENCODE - www.modencode.org
  - *C. elegans/D. melanogaster high throughput*
- BOKU & IMP - Vienna
- MitoMiner - mitochondria
- MilkMine - milk proteins
- Model organisms

Architecture



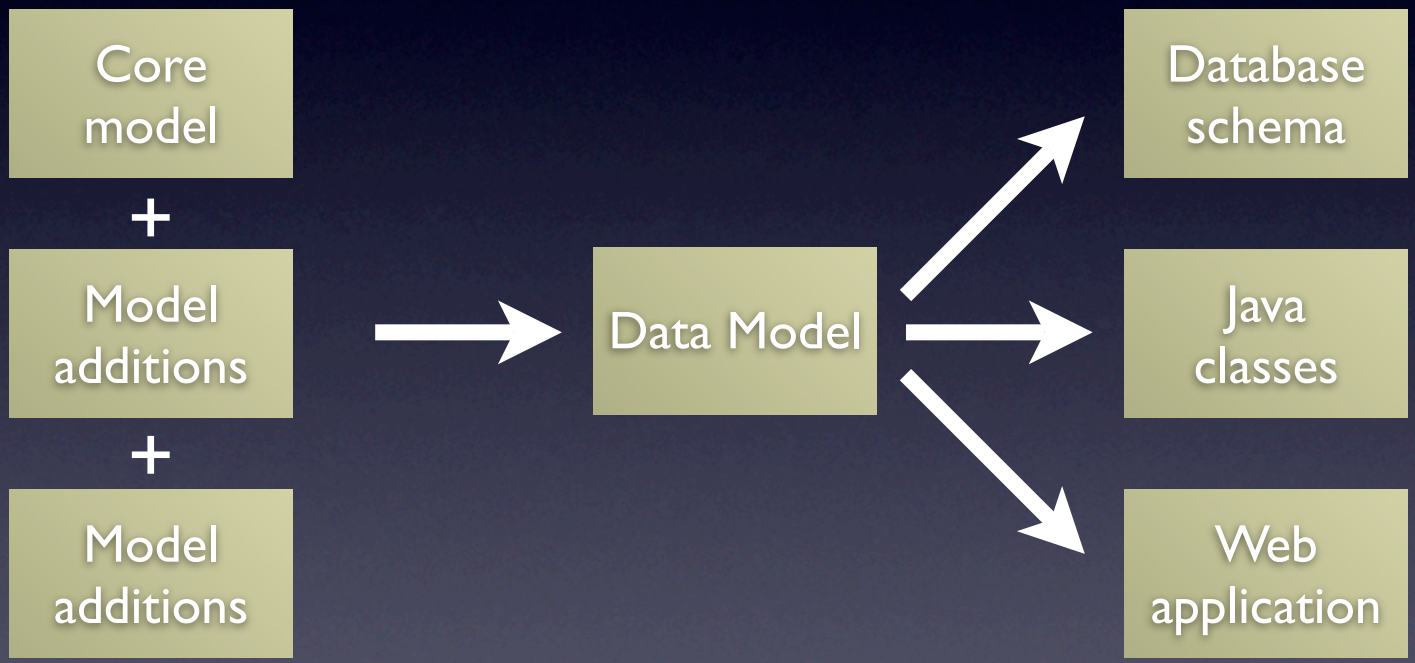
Adapt performance to actual usage
+
Performance optimisation separate from schema design

Data Integration



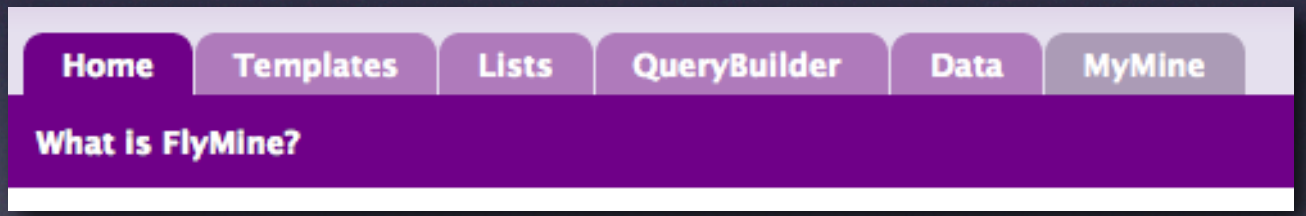
Auto-generation

- Low overhead to extending data model

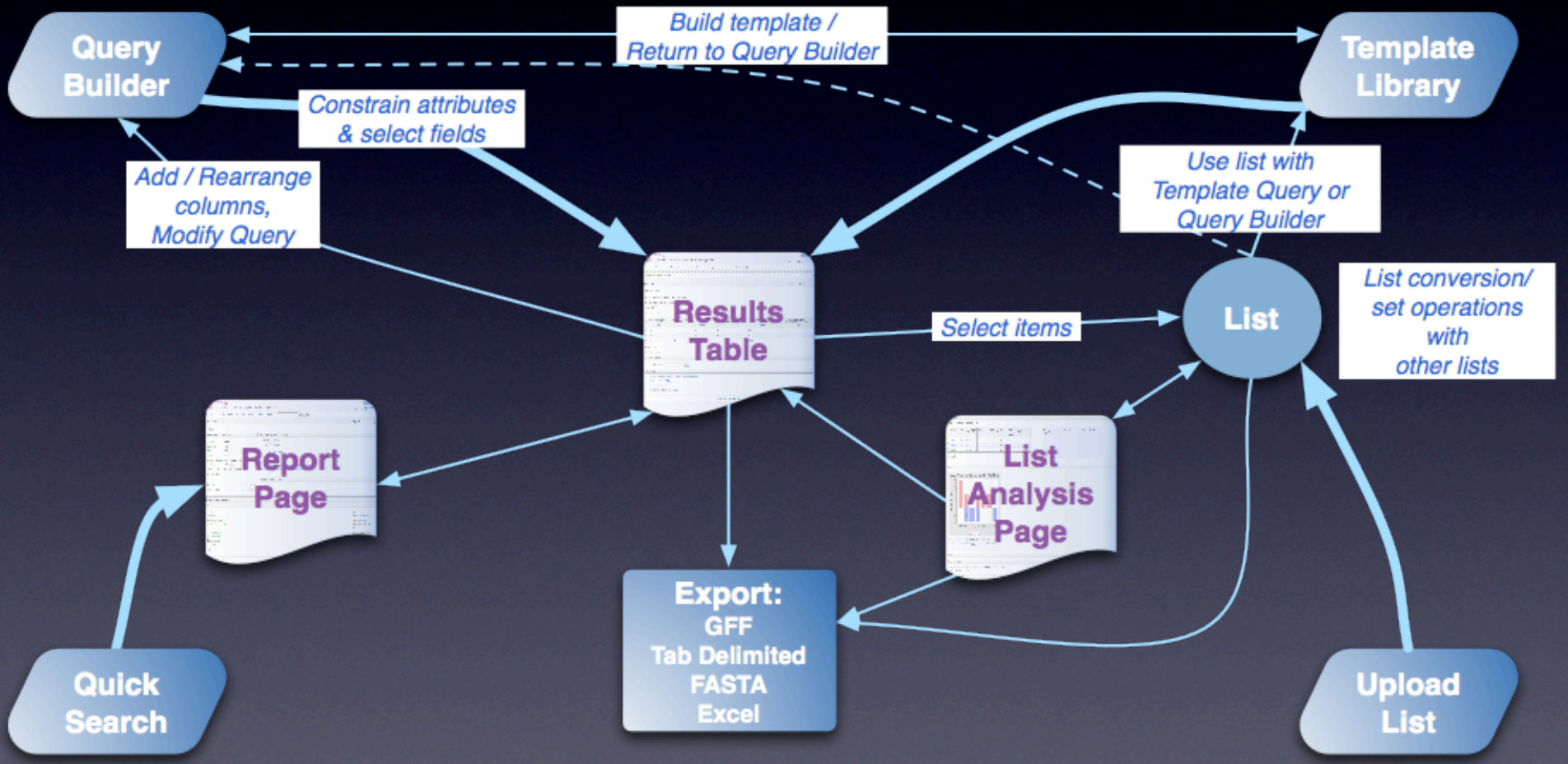


Web Application

- Works for any data model
- Highly configurable
- Configuration from within web interface



Webapp Overview

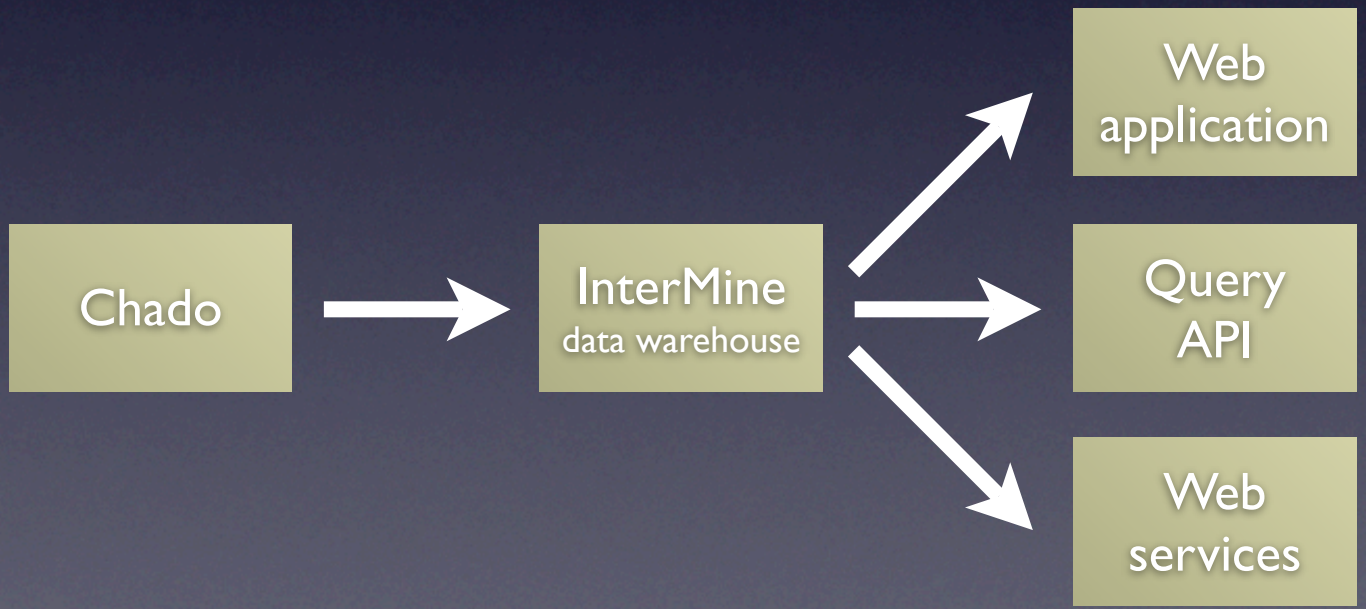


Chado sources

- FlyBase - genome, publications, alleles
- WormBase - genome only
- modENCODE DCC submissions

Import any Chado

- Periodic import of Chado data
- Work for any Chado instance
- Default templates and widgets



Data model

- Java class per SO term
- Inheritance to represent is_a relationships
- part_of -> Java references/collections
- featureprops -> attributes
- Similar approach to Chado for locations

Data Loading

ChadoSequenceProcessor
- reads: sequence module + dependencies
- creates: features, locations, synonyms, pubs
- sets: identifiers as attributes
- sets: references/collections from part_of feature_relationships

modENCODEProcessor
- DCC metadata

FlyBaseProcessor
- e.g. alleles
- e.g. cyto_range

WormBaseProcessor

Possible improvements

- Cleaner Java API
- Java code -> configuration file
- Point and click setup
- Default templates and widgets
- Derive data model automatically

Acknowledgments

Biologists Hilde Jannsens, Rachel Lyne

Developers Richard Smith, Jakub Kulaviak, Julie Sullivan, Matthew Wakeling, Xavier Watkins

Sys Admin Dan Tomlinson

modENCODE Sergio Contrino, Kim Rutherford

PI Gos Micklem