RESEARCH METHODS

The bulk of the work under this proposal falls under Specific Aims 1-3. Specific aim 4 represents a smaller effort but a considerable benefit, and Specific 5 covers the software development practices that will underpin the other specific aims.

SPECIFIC AIM 1: Apply InterMine infrastructure to generate Mines alongside each of the target Model Organism Databases (MODs)

The core aim of this proposal is to generate an InterMine data warehouse instance for each of the model organism databases: SGD (budding yeast), RGD (rat) and ZFIN (zebrafish). This means that the features described above for FlyMine will become available for the data collated by the different MODs. This plan addresses a number of requirements of the MODs and the main benefits are:

1) the entire MOD data collection will be available for query in a flexible, integrated manner

2) users will be able to perform custom bulk queries and export data in a number of formats

3) an unprecedented level of interoperability between MODs, which greatly facilitates comparative analysis, will be achieved

4) users will have access to other InterMine features including templates, list operations, widgets, web services etc (see FlyMine features section)

5) a simple method will be provided to include new types of data that are not currently in the MOD (see Specific Aim 4)

6) MOD data will be made available in a flexible manner through web-services to any other database or software package.

Each of the MODs will run their own instance of InterMine (a 'Mine') on hardware at their own site. Each Mine will be populated with data exported from the corresponding MOD and from outside data repositories, and will have a stand-alone web application, like that of FlyMine. However, we plan to incorporate elements of this web application directly into the existing MOD web interface providing seamless integration for users. Note that we do not intend to replace or provide an alternative to the existing MOD web interface, but rather to enhance it by means of the Mine.

The proposed work will be carried out by one full time developer at each MOD site (MOD developers) for one year (with the exception of ZFIN, see Budget Justification), supported by one full time developer working alongside the InterMine team in Cambridge (Cambridge developer). The MOD developers will have the day-to-day direct contact with each MOD required to understand the data they hold as well as the specific requirements of the user community. The Cambridge developer will provide close links and coordination between the MOD developers and the InterMine team (for immediate help and to raise issues or feature requests) as well as providing direct help with the development process: they will train, ensure commonality between the different Mines, act as a general co-developer for model and parser development and interoperability. The initial effort needed to establish each Mine will be carried out by the MOD and Cambridge developers together, but the longer-term aim is that the routine maintenance of each Mine will be performed by the existing MOD teams. To support this transition and any further developments, the Cambridge developer will be kept in post for an additional year.

Below, we outline the various steps that will be carried out to derive an InterMine data warehouse from each MOD:

Develop data model and write parsers: The import of data will be achieved by writing a custom parser to read data from each MOD database schema. This will be done by writing parser code that reads data directly from the source database, or by writing a parser that operates on flat files dumped from the database. This approach has already been taken within the FlyMine project to obtain bulk data from, e.g. Chado (Mungall et al, 2007) and Ensembl (Hubbard et al, 2007) databases. Most existing InterMine parsers are written in Java but other languages can be used depending on the skill-sets of the MOD personnel who will be maintaining the system in the longer term. Many of the data currently held by the MODs have already been modelled as part of the FlyMine project. We will map the MOD data to this data model, the core genome annotation components of which are based on the Sequence Ontology (Eilbeck et al., 2005): this means that terms from the Sequence Ontology and the relationships between them are used to define parts of the data model used by InterMine-based systems. By adopting this standard ontology, InterMine-based systems can ensure that all genomic feature types are described by standard, widely used, terms.

InterMine has been developed to allow straightforward extension of the data model, which will allow us to accommodate specific data from the MODs. Where multiple MODs hold the same type of data, we will coordinate the data modelling between sites. A benefit of this approach is that a common model allows re-use of components (data parsers, display tools, template queries) and eases interoperation of the resulting databases (discussed further in specific aims 2,3,4).

Database building: Having developed the data model and parsers as above, it will be possible to build the various Mines. Each Mine will be a read-only data warehouse that, once in production, will be re-built periodically according to the release cycle and rate of data accumulation in each MOD. FlyMine (27m rows) is built in under a day and because this is larger than the MODs (SGD: 5m, RGD: 18m rows, ZFIN: 9m rows) we believe it will be realistic to release the Mines on an at least fortnightly cycle. Much of the FlyMine build time is taken up with integration of diverse datasets (>30): this involves running queries during the build to ensure that equivalent entities from different data sources are merged correctly. Within this proposal the integration step will be less significant because the MODs will provide the core data from a database containing already-merged data. Such data will be loaded as a single source without the need for the above build-time queries, and so the build time will be significantly shorter. The need to make distinct builds will lead to a lag between the state of the data in each MOD and the corresponding Mines. This kind of lag is inevitable when building databases and is also true for the MODs themselves with respect to their data sources. We will work to ensure that the timing of Mine releases minimizes the impact to researchers and takes into account particularly large changes to the MODs.

Timing: We propose a three-phase approach: In the **first phase**, we will establish a prototype at each site within the first few months. This will be a 'complete' Mine with a limited scope (e.g. all genes and some related data but not all data types) and, depending on the standard development procedures of each MOD will undergo phased usability studies with local then a wider set of researchers and/or will immediately be made available to the whole user community to invite comments. In the **second phase** we will establish routine public releases of the Mine, containing a large and increasing proportion of the data held by each MOD. The **final phase** will start before the end of year 1 and will establish the maintenance of each Mine as part of the routine operations of each MOD. This will start during phase 2 with the MOD developers training other members of the MOD team, and will continue in year 2 of the project with the help of the Cambridge developer who will be available to assist and advise on all aspects of the production build and any maintenance/ development that is needed.

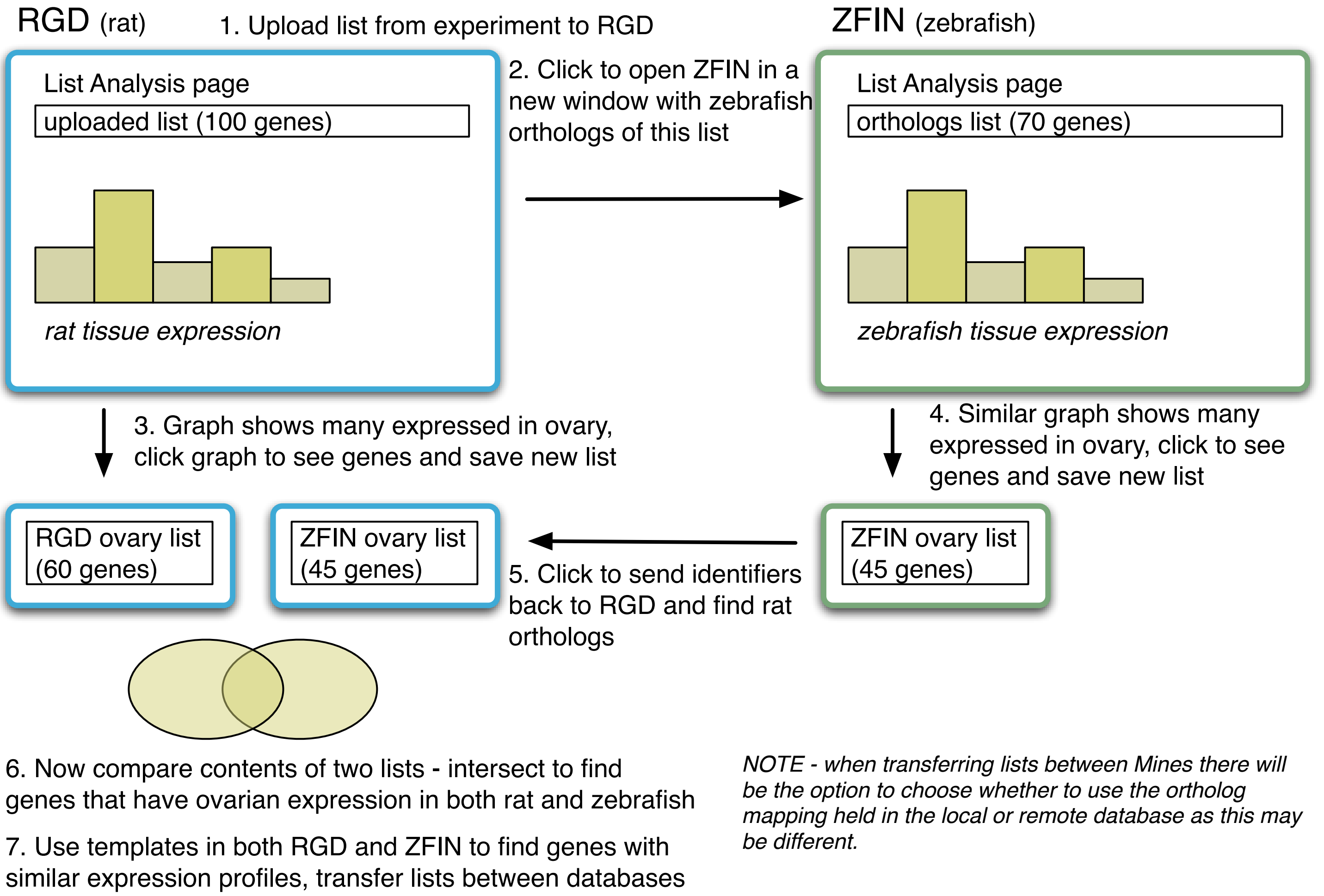
SPECIFIC AIM 2: Enable the different Mines to be interoperable and, thus, provide greater cross-organism integration

Specific Aim 1 outlined how we plan to use a common underlying platform, InterMine, to build data warehouses for the three different MODs. The fact that the MODs will be using the same platform provides a tremendous opportunity to increase the degree of interoperability between the three databases, and indeed, with any other available InterMine databases, e.g. FlyMine. Such integration will increase the ease with which data from different organisms can be compared. In addition, the proposed model for integration will make the high-quality data in the MODs available to other databases and programs by means of web services.

One way of facilitating cross-organism interoperation relies on a mapping between orthologs in the different organisms, many of which have been generated computationally (e.g. Inparanoid, Remm et al, 2001) and/or by curation (e.g. TreeFam, Li et al., 2006). The Gene Ontology Consortium and others are active in defining orthologs. As the Mines are capable of representing multiple parallel ortholog mappings, we will make use of the best mapping or mappings available at the time of the project and (see below) allow users to choose which mapping to use in queries. Given the above mappings, it is straightforward to provide simple links between the Mines. For example, when viewing a gene report page in one Mine, links will lead to analogous reports for its orthologs, if available, in other Mines. This will be extended to operate on lists: e.g. when viewing a list analysis page in one Mine, a link will be available to generate the corresponding list of orthologs in another Mine with a single click. This will give immediate access to the sequences, widgets, and results of template queries for that list of orthologs. Although it may not be meaningful to consider orthology between yeast genes and those of rat or zebrafish, it is still useful to be able to associate data across these organisms (see later). This can be based on sequence similarity or other features, for instance GO annotation or protein domain content.

List transfer: A powerful way of associating data in the different Mines is by transferring lists between them and this will be straightforward as they will be based on a common InterMine platform. Transferring lists and performing analyses in two different Mines will enable rapid cross-genome comparison to be made. For instance in SGD one may be working with a list of genes and the protein domain enrichment widget has shown that a set of protein domains is enriched within them. It would be of interest to know the GO term distribution for genes containing the same domains in RGD and ZFIN. This can be accomplished by making a list of the protein domains and we will make it a one step process to transfer the list to another Mine and view the appropriate list analysis page there. Orthology relationships can also be used: given a starting list one could easily find the subset of genes that behave the same way (e.g. expressed in the ovary) in two organisms (see Figure 3 for an example analysis workflow using orthology). In having a common InterMine platform for the different Mines, many of the data will be displayed in similar formats and this will ease manual comparison. Note that we will make it clear, when ortholog mappings are used, whether the conversion is done in the database providing the list or the one that is receiving it. This is important because, in general, one might not be able to rely on different Mines using the same mappings. Such functionality is not currently available in InterMine and will be implemented as part of this project.

Automatic ortholog mapping: The availability of templates through web service URLs (which will allow remote programs and databases to run queries on the Mines via web links) will be extended as part of this project to include automatic conversion between orthologs. This means that it will be possible to embed the results from widgets or any template query from one Mine, within report or list analysis pages of another Mine. For instance, when viewing a list of zebrafish genes in the ZFIN Mine, a widget provided by FlyMine could provide the gene expression tissue profiles or mutant phenotypes of the *Drosophila* orthologs. The super-user at the hosting Mine would select high-value templates (those most useful for comparative analysis) from other Mines and configure that they appear in report pages. Then, the InterMine extension, developed as part of this project, will be able to automatically find the appropriate ortholog and execute the remote query when requested. We will allow the user to select the method by which orthologs are determined (e.g. curated, computationally generated) and, for computationally predicted orthologs, allow them to specify a cut-off confidence score. Likewise it will be possible to search for template queries present in a remote Mine that are useful for comparative analysis, and execute them for a different organism: such templates would be clearly marked within the host Mine's interface with the Mine they originate from.


RGD (rat)
ZFIN (zebrafish)
List Analysis page
uploaded list (100 genes)
rat tissue expression
List Analysis page
orthologs list (70 genes)
zebrafish tissue expression
RGD ovary list (60 genes)
ZFIN ovary list (45 genes)
ZFIN ovary list (45 genes)

1. Upload list from experiment to RGD

2. Click to open ZFIN in a new window with zebrafish orthologs of this list

3. Graph shows many expressed in ovary, click graph to see genes and save new list

4. Similar graph shows many expressed in ovary, click to see genes and save new list

5. Click to send identifiers back to RGD and find rat orthologs

6. Now compare contents of two lists - intersect to find genes that have ovarian expression in both rat and zebrafish

7. Use templates in both RGD and ZFIN to find genes with similar expression profiles, transfer lists between databases

NOTE - when transferring lists between Mines there will be the option to choose whether to use the ortholog mapping held in the local or remote database as this may be different.

Figure 3: For a particular list of rat genes derived from some experiment, an example that transfers lists between Mines to find the subset of those genes with ovarian expression in both rat and zebrafish.

For example, a researcher has uploaded a list of yeast genes to the ZFIN Mine and wishes to refine that list according to the pathways the gene products appear in. The yeast pathway data are immediately available but in addition it will be possible for pathway templates from the RGD and FlyMine Mines to be executed within the ZFIN interface. These remote templates will return results that can be associated with the appropriate yeast genes through the orthology maps. This would allow the researcher to refine their list based on data from three organisms without leaving the ZFIN Mine interface. Although the above mechanism does not allow true federation of queries over multiple Mines, the ease with which lists can be saved, moved between Mines and combined means this is not a serious limitation.

Importantly, the above approach to embedding the widgets and templates of remote Mines in the user interface of a local Mine, does not need to depend on orthology. The wide and growing use of standard naming systems (ontologies and controlled vocabularies) means that finding equivalence between Mines will be greatly simplified: for instance given a set of GO terms in one Mine, it will be possible to both run a template query that finds genes annotated with those terms in the local Mine and displays their pathway annotation, but also to embed remote templates that do the same thing in remote Mines for other organisms and locally display their results. In this way, without any extra effort on the part of the user, it will be possible to cross-compare data from multiple Mines.

The above examples illustrate that our proposal will enable unprecedented interoperability between the different MODs. The same mechanism will allow interoperability with *Drosophila* and *C. elegans* data through the Mines generated as part of the modENCODE project, and FlyMine. In addition, there are further developments of the core InterMine/ FlyMine project underway: the InterMine and BioMart (Durinck et al 2005) teams have started collaborating on interoperation between the InterMine and BioMart systems. The intention is that by using the same cross-query mechanism as BioMart does, that Mines will be able to interoperate both with other Mines and with BioMart instances. The proposed project will directly benefit from any advances made in this area.

SPECIFIC AIM 3: Configure the Mines, implement custom widgets and integrate tools

User Interface: The InterMine web application works automatically with any data that have been loaded but many aspects are configurable. The overall appearance can be customized by means of Cascading Style-sheets (CSS), specific icons and text defined through 'properties' files. The Mines will be configured to reflect the appearance of their parent MODs. Each Mine will be able to draw on the common pool of existing template queries, but another element of configuration will be to create new template queries with default values for the data specific to each MOD. The super-user can configure where in the web interface particular templates are displayed by means of 'tagging' – i.e. annotating the template with a keyword in the live web application. For instance, any given template can be tagged so that it will be executed on a gene report page, or displayed as an example query within a particular data category's 'home' page, or so that it appears as one of the example queries on the front page of the web site.

Similarly, public lists can be created and published by a super-user at any time, and tagged to appear in different locations within the web interface. We expect curators at each MOD to find this a useful feature: for example, if they curate a paper that identifies a particular list of genes, they can immediately create that list in the Mine for convenient access and analysis by the research community. Although the default states for most other configurations can be inherited from the FlyMine project, editing XML files can also easily change them. We will aim to share configuration between the sites where possible.

Widgets: FlyMine includes a number of integrated tools and data analysis and visualization 'widgets' (For examples, see Figure 1). Widgets perform simple analyses or display of data and are most often applied to lists of e.g. genes or proteins. Rather than selecting each analysis widget in turn, the performance of InterMine-based systems is sufficiently high that when viewing the properties of a list, all appropriate widgets are automatically run. When integrating new types of data into FlyMine we consider the types of widgets that would be useful in presenting the data. A framework exists for easy incorporation of new widgets, and an important part of this proposal is to add widgets appropriate for the MOD data. Some widgets within FlyMine are generally applicable to data from other organisms while some work for specific *Drosophila* data sets. Typically they are interactive so that e.g. clicking on a bar from a bar chart will display the corresponding subset of e.g. genes, for export or creation on a new list. As part of this proposal we will implement widgets that are specific to the MOD datasets, for instance providing widgets that calculate if a set of genes is enriched for any disease or phenotype annotation.

To add a widget to calculate enrichment of a new attribute or ontology simply requires designing appropriate queries, while graphical widgets require both query design and some specific Java code. However, developing completely new types of widget, and, for instance, implementing statistical algorithms would remain within the remit of the InterMine/FlyMine projects.

Tool integration: There are two possible levels of integration of InterMine with external tools: **i)** data can be exported in a format suitable for immediate import into a separately-run tool (e.g. .sif format for CytoScape (Shannon et al, 2003); **ii)** tools can be integrated into InterMine itself so that they can be seamlessly launched as part of the user interface. Examples in this latter category include GBrowse (Stein et al, 2002) for interactive visualization of genome annotation and Jmol to allow interactive rendering of protein structures. We will integrate the Mines with tools that have already been used and/or developed by the MODs (e.g. GBrowse, RGD's GViewer (http://www.gmod.org/flashgviewer). We will also include other GMOD tools as appropriate e.g. the Ontology browser that is under development.

Collectively, the above widgets and tools will provide remarkable additional functionality for each of the MODs. For example, we will be able to satisfy one of RGD's goals of assessing assignments from one ontology with respect to another: starting with a GO term or terms of interest, a list of genes can be generated, and in the process of viewing this list, the enrichment widget would identify terms in other ontologies that are enriched. Thus one would immediately be able to find out if genes involved in a particular biological process were enriched for any molecular function. A similar approach could be taken with respect to other types of annotation not based on ontologies.

Integration with existing MOD website: It will be possible (release due early Q2 2008) to execute InterMine template queries through a RESTful web service and receive results as HTML. By this method it will be possible to embed the results of any template query (or widget) in another web page. This means, for example, that the existing gene report pages in one of the MODs could display the results of a template query from the corresponding Mine that searched for predicted transcription factor binding sites in the upstream region within a gene. Likewise, a QTL report page could search for all genes within the corresponding region of the genome. This provides a mechanism to include results of more complex data mining queries in the MOD pages, and also data that are not yet included in the MOD database itself, but which have been integrated into the corresponding Mine (see specific Aim 4). Likewise, for example, each gene report in InterMine has a section that indicates which public lists the gene is a member of; this information could also be embedded in the MOD gene report page.

SPECIFIC AIM 4: Merge other external data sources into the MOD-specific Mines

InterMine is not just capable of executing complex queries rapidly but is also a sophisticated data integration platform. The existing MODs will form the core of each Mine, but through the FlyMine project, InterMine will be able to import and integrate a wide variety of data formats. Therefore this proposal provides an opportunity, for relatively little effort, to increase the scope of data covered by each MOD through their respective Mines; each time a Mine is built, the latest versions of supplementary data sources will be included, too. Note, that because data from each Mine can be incorporated into the report pages of each MOD (Specific Aim 1), the MODs will be able to present Mine data seamlessly, even though it has not yet been incorporated in the core database of each MOD. By exploiting the data modelling and data parsers developed in the various parts of the project for all of the MODs, this project will lead to a more consistent and comprehensive set of data being represented across all the MODs.

Figure 4 summarizes the coverage of data types by the different databases. A number of observations can be made about this table: **1)** Many data types of interest have already been incorporated into FlyMine so that we already have experience modelling and querying them. **2)** Many data types are found across the three MODs, indicating that there will be data available to carry out comparative analysis. **3)** In some cases MODs provide particular data types, but these are not fully integrated with the main databases e.g. transcripts in SGD and genotypes in RGD. Adding these to the Mines will make this possible. **4)** There are cases in which the MOD does not cover the data at all, but for which InterMine already has a parser (See Table 1) available for an appropriate source, and data is available e.g. Protein/ protein interaction data from IntAct for RGD. **5)** There are cases where writing a new parser would benefit all sites and will benefit the existing InterMine user community. e.g. a parser for dbSNP data and genetic interactions from BioGrid (Breitkreutz et al 2008). We will take into account the relative importance to the different MODs, data availability and use to the community in deciding work priorities.

An example where additional data integration will be useful and which is a priority for RGD are data produced as part of the Medical College of Wisconsin PhysGen project (http://pga.mcw.edu). This has already generated 435,000 measurements. As complete integration within RGD would be challenging, the Mine would be an ideal destination for these data. This is also the case for Rat SNP data.

	FlyMine	SGD	RGD	ZFIN
Genome Annotation				
sequence	■	■	▨	▨
genes	■	■	■	■
transcripts	■	▨	■	■
exons	■	■	▨	▨
regulatory regions	■	■	□	□
microarray probes (expression/tiling)	■	■	□	■
Affymetrix probes	■	▨	□	■
markers	□	□	■	■
ESTs	■	□	■	■
insertions/deletions	■	▨	□	■
Proteins				
sequence	■	■	▨	▨
feature locations	■	■	□	□
domain content	■	■	■	▨
protein-protein interactions	■	■	□	□
3D structure	■	■	□	□
Gene Expression				
microarray experiments	■	■	□	■
in situ hybridisations	■	□	□	■
Comparative				
curated orthologs	□	□	□	■
curated paralogs	□	▨	□	■
computed orthologs	■	■	■	□
computed paralogs	■	▨	□	□
synteny	□	■	□	□
whole genome alignments	□	■	□	□
Genetics				
Alleles	□	□	■	■
SNPs	□	□	▨	■
Genotypes	□	□	▨	■
linkage	□	□	■	■
QTLs	□	□	■	□
Genetic Interactions	□	■	□	□
Stocks				
Mutant Lines	□	□	□	■
Transgenic lines	□	□	□	■
Strains	□	□	■	■
Other				
Gene Ontology	■	■	■	■
RNAi screens	■	□	□	■
disease	■	□	■	▨
phenotypes	■	■	■	■
anatomy	□	□	□	■
pathways	■	■	■	□
publications	■	■	■	■

Key:

Integrated in MOD, possible to query	■
Available from MOD but not in main database	▨
Not yet available from MOD (Note that this does not necessarily mean appropriate data is available)	□

SPECIFIC AIM 5: Develop code to a high quality, and make it freely available under an Open Source license in a well-documented form through the Generic Model Organism Database (GMOD) Consortium

Availability: InterMine and FlyMine are well-established Open Source projects. All code is available under the LGPL license (http://www.gnu.org/licenses/lgpl.html), which means that it is freely available for academic and commercial use and we actively encourage contributions in the form of developments, tests, bug fixes and documentation. Code can be obtained from an open Subversion (http://subversion.tigris.org/) repository. We will make all code generated as part of this project available through the same repository (see http://www.intermine.org/wiki/SVNCheckout). The advantage of centralizing resources in this way is that there is a single location globally at which all possible modules are available. This project will provide a number of extensions to the InterMine project, all of which will be freely available and many of which may be useful to other projects both current and future. Specific categories of developed code include:

- data parsers for additional data types
- data model enhancements to support new data types
- core enhancements to InterMine to support interoperability and automatic ortholog mapping
- implementation of particular widgets to support MOD data
- integration of MOD and other tools specific to this project

Quality: FlyMine and InterMine are robust and large-scale projects that have been engineered with solid modular design principles and with a view to long-term maintainability and development. The team responsible is composed of experienced software engineers and computer scientists. One quarter of the code-base of over 200,000 lines is unit tests that are run automatically in response to changes checked into the code repository. All code conforms to strict coding standards, which requires that all classes and methods include javadoc comments. This means that the code base is very well documented. All code generated as part of the current proposal will be developed to conform to the above standards and principles: this will be important to maximize the future utility of enhancements, to minimize maintenance efforts and ensure the overall project remains robust.

Documentation: As described above, there will be fine-grained documentation of all code through javadoc comments. In addition there is extensive documentation in the form of a wiki at http://www.intermine.org that covers the building and maintenance of new data warehouses and many other aspects of the smooth-running of InterMine based systems. This documentation is under continual refinement and members of the proposed project will be given write-access to the wiki to document their developments. In particular, it is important that new data parsers are well-documented so that they can be effectively reused by third parties.

GMOD: InterMine is a component project within the Generic Model Organisms Database (GMOD) Consortium (http://www.gmod.org). The mission of GMOD is to develop, distribute, and maintain software tools and procedures for shared use by existing and emerging model organism databases. GMOD tools are characterized by their adherence to common standards, such as the Sequence Ontology, and their distribution as open source projects. GMOD tools are widely used in the animal and plant genome research community. The InterMine project is described on the GMOD website, along with links to the code respository. In this way, InterMine is highly visible to the Model Organism Database community as will be the outputs of this project.

Milestones

Figure 5 shows the major activities and milestones for the project. Milestones are shown as “X” characters and ongoing activities as solid cells. Greater levels of effort are shown as shaded cells of increasing intensity. See the bottom of the table for the key. Note that the University of Oregon (ZFIN) will be following a 50% effort timetable so that the first year of the timetable will be stretched to cover two years.

	1q1	1q2	1q3	1q4	2q1	2q2	2q3	2q4
General Management								
Hire all personnel	X							
Establish mailing list/ call schedule	X							
InterMine bootcamp	X							
Site visits to MODS			X	X X		X	X	X
Specific Aim #1: *Develop Mines for MODs*								
Install servers & software	X							
Data modelling	🟧	🟧	🟧	🟨	🟨	🟨	🟨	🟨
Develop data loaders/ templates/ configuration	🟥	🟥	🟧	🟧	🟨	🟨	🟨	🟨
Build prototype and test databases	🟨	🟨						
Internal prototype database	X							
Test database for selected community feedback		X						
Production database build and release			🟨	🟨	🟨	🟨	🟨	🟨
Initial public release of databases			X					
InterMine warehouses part of normal release cycle				X				
Training of MOD staff for future release cycle			🟧	🟥	🟧	🟨		
Support of MODs by Cambridge	🟥	🟥	🟨	🟨	🟧	🟧	🟧	🟧
Specific Aim #2: *Develop Interoperation*								
Ortholog import		🟨						
Ortholog list interchange between databases			🟨					
Ability to run remote template queries			🟧	🟧	🟨			
Ability to run remote widgets				🟧	🟧	🟨		
Specific Aim #3: *Widgets and Tool integration*								
Develop widgets			🟨	🟨	🟨	🟨	🟨	🟨
Integrate with MOD viewing/ analysis tools		🟨	🟨	🟨	🟨	🟨		
Specific Aim #4: *Extend data coverage*								
Identify and load new data sources:								
existing InterMine data loaders			🟨	🟨				
new data loaders			🟨	🟧	🟨	🟨	🟨	🟨
Specific Aim #5: *Software management*								
Common SVN accounts established	X							
Remote login accounts established	X							
Unit testing of new software	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨
Software documentation	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨

Key:

🟨	light development
🟧	moderate development
🟥	heavy development
X	Milestone, end of first month of quarter
X	Milestone, end of second month of quarter
X	Milestone, end of third month of quarter

Figure 5: Timeline & Milestones

Note, the schedule for Year 1 will be stretched over 2 years for the ZFIN project