



# SADI for GMOD: An RDF/OWL Interface for GMOD

<http://code.google.com/p/sadi/wiki/SADIforGMOD>

## What is SADI for GMOD?

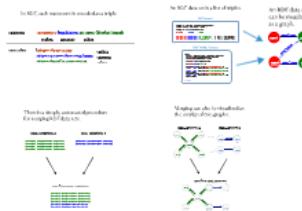
- SADI stands for Semantic Automated Discovery and Integration; it's our standard for RDF web services (more info later)
- SADI for GMOD is a set of CGI scripts for accessing sequence feature data as RDF
- DAS for RDF!

Service Name	Input	Relationship	Output
get_feature_info	feature identifier	is about	feature description
get_features_overlapping_region	generic coordinate	overlaps	collection of feature descriptions
get_feature_in_region	generic coordinate	is represented by	DNA, RNA, protein, domain sequence
get_feature_regions	feature description	has part / covers	collection of feature descriptions
get_overlapped_features	feature description	is part / overlaps	collection of feature descriptions

## What is the purpose of SADI for GMOD?

- Provide an easy way for GMODs to share their data as RDF, using a standardized protocol (SADI)
- Provide infrastructure for data integration across GMODs and other biology resources
- Support development of smarter bioinformatics software:
  - => Distributed queries
  - => Automated construction of web service workflows
  - => Assembly of datasets from multiple sources

## RDF primer



## What SADI for GMOD services are currently available?

To bootstrap participation, we are mirroring several GMODs (as of late):

- `Chado` – C. elegans
- `CGD` – C. albicans
- `CGPS` – C. persicae
- `PhageDB` – *lambda*
- `Gramene` – A. thaliana
- `DBG` – *Drosophila* genomes
- `BDG` – S. cerevisiae
- `TIGR37` – T. gracilis genome
- `GMOD` – GMOD genome

SADI for GMOD services are provided for each, e.g. [http://sadi\)framework.org/service/sadi/gmod/get\\_Status\\_info](http://sadi)framework.org/service/sadi/gmod/get_Status_info)

The SADI for GMOD services are in the public SADI registry:

[http://sadi\)framework.org/service/sadi/gmod/get\\_Status\\_info](http://sadi)framework.org/service/sadi/gmod/get_Status_info)

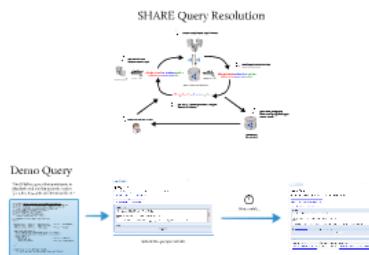
In the way, you can register your own SADI services here too:

[http://sadi\)framework.org/service/sadi/gmod/get\\_Status\\_info](http://sadi)framework.org/service/sadi/gmod/get_Status_info)

## How do I use the services? (for GMOD users)



## Demo (distributed query)



## How do I set up the services? (for GMOD providers)

- Load your GMOD files into a MySQL database (will soon support Chado, also)
- Install SADI for GMOD dependencies with CPAN
- Download the SADI for GMOD module and unpack into `perl-bin`
- Set DB connection parameters in `perl-bin/sadi_gmod/sadi_gmod.conf`

```
GMODDBI = MySQL
db_dsn = dbi:mysql:database=chado
db_user = dbuser
base_url = http://dbuser:password@localhost/gmod/
```
- Configure Chado mappings in `perl-bin/sadi_gmod/doiexec2.conf`

```
GMOD_TO_GMOD
PREFIXES = Chado
URIPREFIX = chado
URIPREFIXES = chado
URIPREFIXES = Chado
```
- Register the services in public SADI registry: [http://sadi\)framework.org/register](http://sadi)framework.org/register)

## Future plans

- Chado support (soon!)
- add BLAST services/anything else that you want!
- use cases and demos
- more GMOD mirrors
- page on GMOD wiki
- distribute with Tripal or GBrowse?

## Acknowledgements

### Team

Mark Wilkinson, Principal Investigator  
Mark Wilkinson, Lead Developer (SADI framework and data modeling)  
Luke McCarthy, Lead Programmer, SADI & GMOD  
Edward Kawas, Lead Programmer, SADI

### Funding



[http://sadi\)framework.org/](http://sadi)framework.org/)

Ben Vandervalk<sup>\*1</sup>, Luke McCarthy<sup>1</sup>, Edward Kawas, Michel Dumontier,<sup>2</sup> Mark Wilkinson<sup>1</sup>  
James Hogg Research Institute, St. Paul's Hospital, University of British Columbia<sup>1</sup>  
Carleton University<sup>2</sup>  
<sup>\*</sup>ben.vvalk@gmail.com

[http://sadi\)framework.org/](http://sadi)framework.org/)



# SADI for GMOD: An RDF/OWL Interface for GMOD

<http://code.google.com/p/sadi/wiki/SADIforGMOD>

## What is SADI for GMOD?

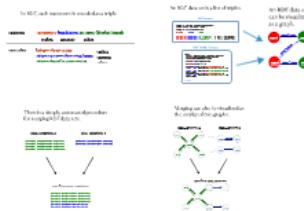
- SADI stands for Semantic Automated Discovery and Integration; it's our standard for RDF web services (more info later)
- SADI for GMOD is a set of CGI scripts for accessing sequence feature data as RDF
- DAS for RDF!

Service Name	Input	Relationship	Output
get_feature_info	feature identifier	is about	feature description
get_features_overlapping_region	generic coordinate	overlaps	collection of feature descriptions
get_feature_in_region	generic coordinate	is represented by	DNA, RNA, protein, domain sequence
get_feature_regions	feature description	has part / covers	collection of feature descriptions
get_overlapped_features	feature description	is part / overlaps	collection of feature descriptions

## What is the purpose of SADI for GMOD?

- Provide an easy way for GMODs to share their data as RDF, using a standardized protocol (SADI)
- Provide infrastructure for data integration across GMODs and other biology resources
- Support development of smarter bioinformatics software:
  - => Distributed queries
  - => Automated construction of web service workflows
  - => Assembly of datasets from multiple sources

## RDF primer



## What SADI for GMOD services are currently available?

To bootstrap participation, we are mirroring several GMODs (as of late):

- *AnnotDB* – C. elegans
- *CGD* – C. albicans
- *CGP* – C. elegans
- *Phage* – D. melanogaster
- *Gramene* – A. thaliana
- *DBG* – Arabidopsis genes
- *BIG* – S. cerevisiae
- *TIGD* – T. gracilis genome
- *Chado* – C. elegans genome

The SADI for GMOD services are in the public SADI registry:

[http://sadiframework.org/service\\_registry/service\\_registry.cgi?service\\_id=1](http://sadiframework.org/service_registry/service_registry.cgi?service_id=1)

In the way, you can register your own SADI service here too:

[http://sadiframework.org/service\\_registry/service\\_registry.cgi?service\\_id=2](http://sadiframework.org/service_registry/service_registry.cgi?service_id=2)

## How do I set up the services? (for GMOD providers)

- Load your GMOD files into a MySQL database (will soon support Chado, also)
- Install SADI for GMOD dependencies with CPAN
- Download the SADI for GMOD install and unpack into `perl-bin`
- Set DB connection parameters in `perl-bin/sadi_rmod/sadi_gmod.conf`

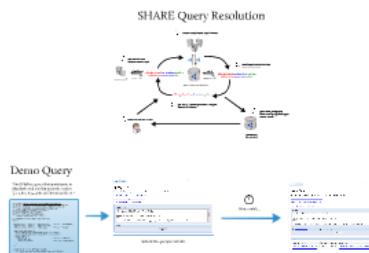
```
GMODDBI = MySQL
db_dbs = db_annotation, db_annotation_index
db_dbs = db_annotation, db_annotation_index
base_url = http://dbase.org/gmod/testdb/gmod/
```
- Configure Chado mappings in `perl-bin/sadi_gmod/diodes2.conf`

```
GMOD_TO_GMOD
perl-bin > Chado
mappings = column
relationships = column
```
- Register the services in public SADI registry: <http://sadiframework.org/register>

## How do I use the services? (for GMOD users)



## Demo (distributed query)



## Future plans

- Chado support (soon!)
- add BLAST services/anything else that you want!
- use cases and demos
- more GMOD mirrors
- page on GMOD wiki
- distribute with Tripal or GBrowse?

## Acknowledgements

### Team

Mark Wilkinson, Principal Investigator  
Mark Wilkinson, Bioinformatics Core (informatics and databases)  
Luke McCarthy, Lead Programmer, SADI & SWS  
Edward Kawas, Lead Programmer, SADI

### Funding



<http://sadiframework.org/>

Ben Vandervalk<sup>\*1</sup>, Luke McCarthy<sup>1</sup>, Edward Kawas, Michel Dumontier,<sup>2</sup> Mark Wilkinson<sup>1</sup>  
James Hogg Research Institute, St. Paul's Hospital, University of British Columbia<sup>1</sup>  
Carleton University<sup>2</sup>  
<sup>\*</sup>ben.vvalk@gmail.com

<http://sadiframework.org/>

# What is SADI for GMOD?

- SADI stands for Semantic Automated Discovery and Integration; it's our standard for RDF web services (more info later)
- SADI for GMOD is a set of CGI scripts for accessing sequence feature data as RDF
- "DAS for RDF"

Service Name	Input	Relationship	Output
get_feature_info	database identifier	is about	feature description
get_features_overlapping_region	genomic coordinates	overlaps	collection of feature descriptions
get_sequence_for_region	genomic coordinates	is represented by	DNA, RNA, or amino acid sequence
get_child_features	feature description	has part / derives into	collection of feature descriptions
get_parent_features	feature description	is part of / derives from	collection of feature descriptions

# What is the purpose of SADI for GMOD?

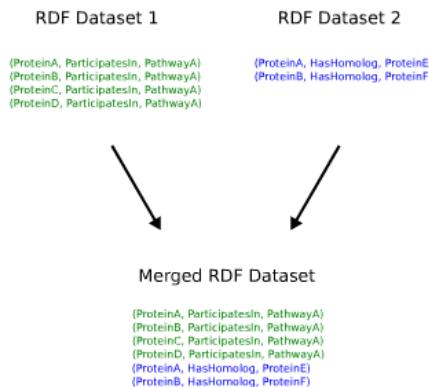
- Provide an easy way for GMODs to share their data as RDF, using a standardized protocol (SADI)
- Provide infrastructure for data integration across GMODs and other biology resources
- Support development of smarter bioinformatics software:
  - => Distributed queries
  - => Automated construction of web service workflows
  - => Assembly of datasets from multiple sources

# RDF primer

In RDF, each statement is encoded as a triple.

Statement:	<b>"Hexokinase 1 participates in the human glycolysis pathway."</b>		
	<i>Subject</i>	<i>Predicate</i>	<i>Object</i>
RDF Triple:	( <a href="http://lsrn.org/UniProt:P19367">http://lsrn.org/UniProt:P19367</a> ,	<i>Subject</i>	
	<a href="http://semanticscience.org/resource/SIO_000062">http://semanticscience.org/resource/SIO_000062</a> ,	<i>Predicate</i>	
	<a href="http://lsrn.org/KEGG_PATHWAY:hsa00010">http://lsrn.org/KEGG_PATHWAY:hsa00010</a> )	<i>Object</i>	

There is a simple, automated procedure for merging RDF data sets.



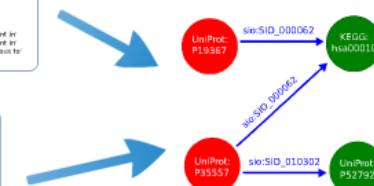
An RDF data set is a list of triples.

`<@prefix sko:> <http://purl.org/linked-data/resource#> .  
<@prefix Univat:> <http://purl.org/univat#> .  
<@prefix Klod:> <http://purl.org/klod#> .`

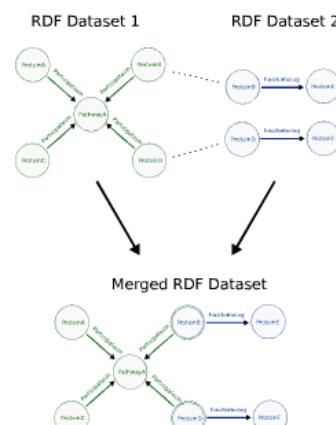
`# subject predicate object`

`UnivatP12347 sko:isSKO_0000962 Klod#0000010 . # $0..000962 = 1) participant in  
UnivatP12347 sko:isSKO_0000962 Klod#0000010 . # $0..000962 = 1) participant in  
UnivatP12347 sko:isSKO_0000962 Klod#0000010 . # $0..000962 = 1) participant in`

An RDF data set can be visualized as a graph.



Merging can also be visualized as the overlay of two graphs.



In RDF, each statement is encoded as a triple.

Statement: “Hexokinase 1 participates in the human glycolysis pathway.”

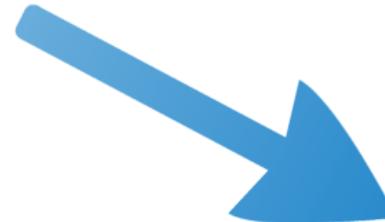
*Subject*      *Predicate*      *Object*

RDF Triple: (<http://lsrn.org/UniProt:P19367>, *Subject*  
[http://semanticscience.org/resource/SIO\\_000062](http://semanticscience.org/resource/SIO_000062), *Predicate*  
[http://lsrn.org/KEGG\\_PATHWAY:hsa00010](http://lsrn.org/KEGG_PATHWAY:hsa00010)) *Object*

# An RDF data set is a list of triples.

## N3 Format

```
@prefix sio: <http://semanticscience.org/resource/> .  
@prefix UniProt: <http://lsrn.org/UniProt:> .  
@prefix KEGG: <http://lsrn.org/KEGG:> .  
  
# subject predicate object  
  
UniProt:P19367 sio:SIO_000062 KEGG:hsa00010 . # SIO_00062 = 'is participant in'  
UniProt:P35557 sio:SIO_000062 KEGG:hsa00010 . # SIO_00062 = 'is participant in'  
UniProt:P35557 sio:SIO_010302 UniProt:P52792 . # SIO_10302 = 'is homologous to'
```



## RDF/XML Format

```
<rdf:RDF  
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"  
    xmlns:sio="http://semanticscience.org/resource/"  
    xmlns:UniProt="http://lsrn.org/UniProt:"  
    xmlns:KEGG="http://lsrn.org/KEGG:>  
  
<!-- subject predicate object -->  
  
<rdf:Description rdf:about="http://lsrn.org/UniProt:P35557">  
  <sio:SIO_000062 rdf:resource="http://lsrn.org/KEGG:hsa00010"/>  
  <sio:SIO_010302 rdf:resource="http://lsrn.org/UniProt:P52792"/>  
</rdf:Description>  
<rdf:Description rdf:about="http://lsrn.org/UniProt:P19367">  
  <sio:SIO_000062 rdf:resource="http://lsrn.org/KEGG:hsa00010"/>  
</rdf:Description>  
  
</rdf:RDF>
```



set is a list of triples.

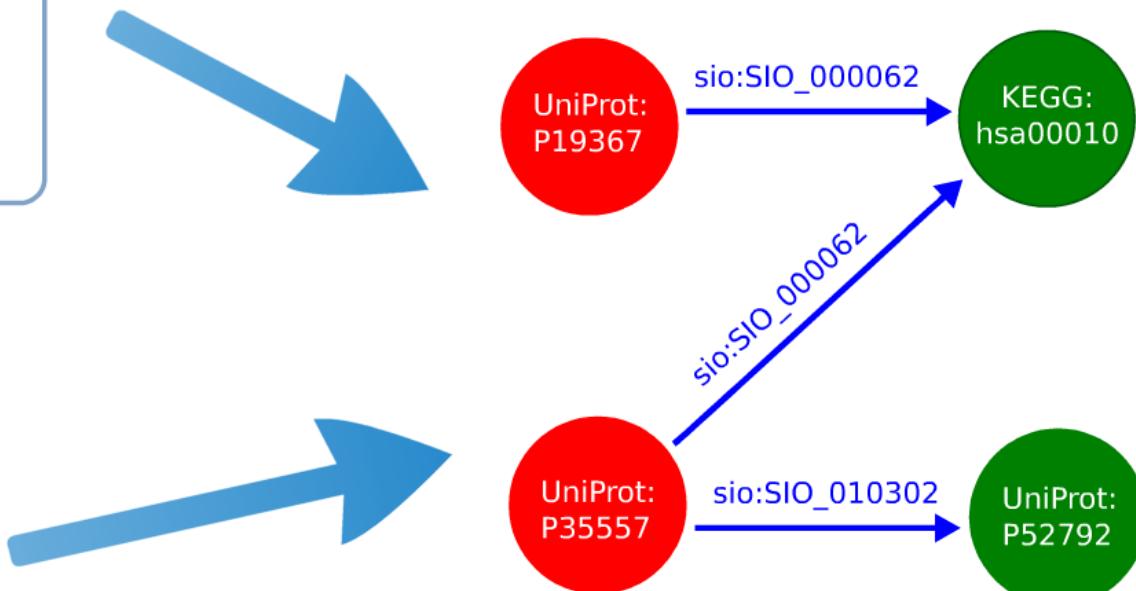
### N3 Format

```
ounce.org/resource/> .  
UniProt:> .  
KEGG:> .  
  
object  
  
EGG:hsa00010 . # SIO_00062 = 'is participant in'  
EGG:hsa00010 . # SIO_00062 = 'is participant in'  
UniProt:P52792 . # SIO_010302 = 'is homologous to'
```

### RDF/XML Format

```
www.w3.org/1999/02/22-rdf-syntax-ns#"  
semanticscience.org/resource/"  
http://lsrn.org/UniProt:"  
http://lsrn.org/KEGG:">  
  
object -->  
  
about="http://lsrn.org/UniProt:P35557">  
rdf:resource="http://lsrn.org/KEGG:hsa00010"/>  
rdf:resource="http://lsrn.org/UniProt:P52792"/>  
  
about="http://lsrn.org/UniProt:P19367">  
rdf:resource="http://lsrn.org/KEGG:hsa00010"/>
```

An RDF data set can be visualized as a graph.



There is a simple, automated procedure for merging RDF data sets.

RDF Dataset 1

(ProteinA, ParticipatesIn, PathwayA)  
(ProteinB, ParticipatesIn, PathwayA)  
(ProteinC, ParticipatesIn, PathwayA)  
(ProteinD, ParticipatesIn, PathwayA)

RDF Dataset 2

(ProteinA, HasHomolog, ProteinE)  
(ProteinB, HasHomolog, ProteinF)

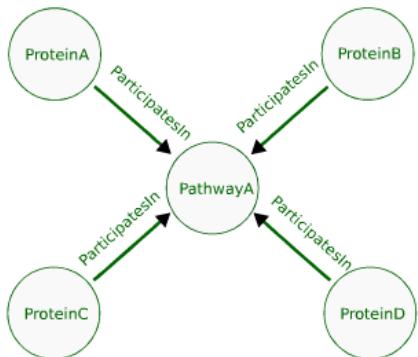


Merged RDF Dataset

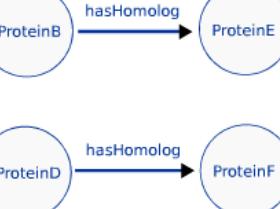
(ProteinA, ParticipatesIn, PathwayA)  
(ProteinB, ParticipatesIn, PathwayA)  
(ProteinC, ParticipatesIn, PathwayA)  
(ProteinD, ParticipatesIn, PathwayA)  
(ProteinA, HasHomolog, ProteinE)  
(ProteinB, HasHomolog, ProteinF)

Merging can also be visualized as the overlay of two graphs.

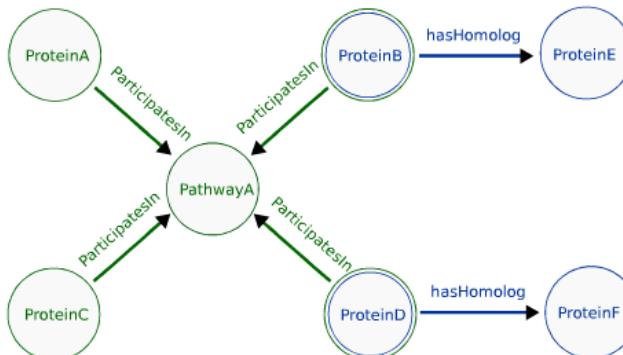
RDF Dataset 1



RDF Dataset 2



Merged RDF Dataset



# What SADI for GMOD services are currently available?

To bootstrap participation, we are mirroring several GMODs (9 so far):

- AmoebaDB -- *E. histolytica*
- CGD -- *C. albicans*
- CryptoDB -- *C. hominis*
- FlyBase -- *D. melanogaster*
- Gramene -- *A. thaliana*
- PlasmoDB -- *P. falciparum*
- SGD -- *S. cerevisiae*
- ToxoDB -- *T. gondii* (RH strain)
- TriTrypDB -- *L. major*

SADI for GMOD services are provided for each, e.g.

[http://s7.semanticscience.org/~ben/cgi-bin/FlyBase/get\\_feature\\_info](http://s7.semanticscience.org/~ben/cgi-bin/FlyBase/get_feature_info)

The SADI for GMOD services are in the public SADI registry.

The screenshot shows a web interface for the SADI registry. At the top, there are tabs for 'Validate', 'Register', 'Services' (which is selected), and 'SPARQL'. Below the tabs, a section titled 'Registered services' displays a list of 422 registered services. Each service entry includes a 'Service URL' (e.g., <http://irom3.semanticscience.org/sadi-service/bt2caso>), a brief description (e.g., 'Name getUnProtIdByProteinName'), and a 'Description' field. A note indicates that values are from [http://semanticcorpora.org/resource/SIO\\_010013](http://semanticcorpora.org/resource/SIO_010013). The list continues with other entries like 'getUnProtIdByPMCID', 'getMGIIDByPMCID', and 'getDBIDByPMCID'.

<http://sadiframework.org/registry/services/>

By the way, you can register your own SADI services here too.



<http://sadiframework.org/registry/register/>

To bootstrap participation, we are mirroring several GMODs (9 so far):

- AmoebaDB -- *E. histolytica*
- CGD -- *C. albicans*
- CryptoDB -- *C. hominis*
- FlyBase -- *D. melanogaster*
- Gramene -- *A. thaliana*
- PlasmoDB -- *P. falciparum*
- SGD -- *S. cerevisiae*
- ToxoDB -- *T. gondii* (RH strain)
- TriTrypDB -- *L. major*

SADI for GMOD services are provided for each, e.g.

[http://s7.semanticscience.org/~ben/cgi-bin/FlyBase/get\\_feature\\_info](http://s7.semanticscience.org/~ben/cgi-bin/FlyBase/get_feature_info)

# The SADI for GMOD services are in the public SADI registry.

Validate Register Services SPARQL

## Registered services

422 services are registered.

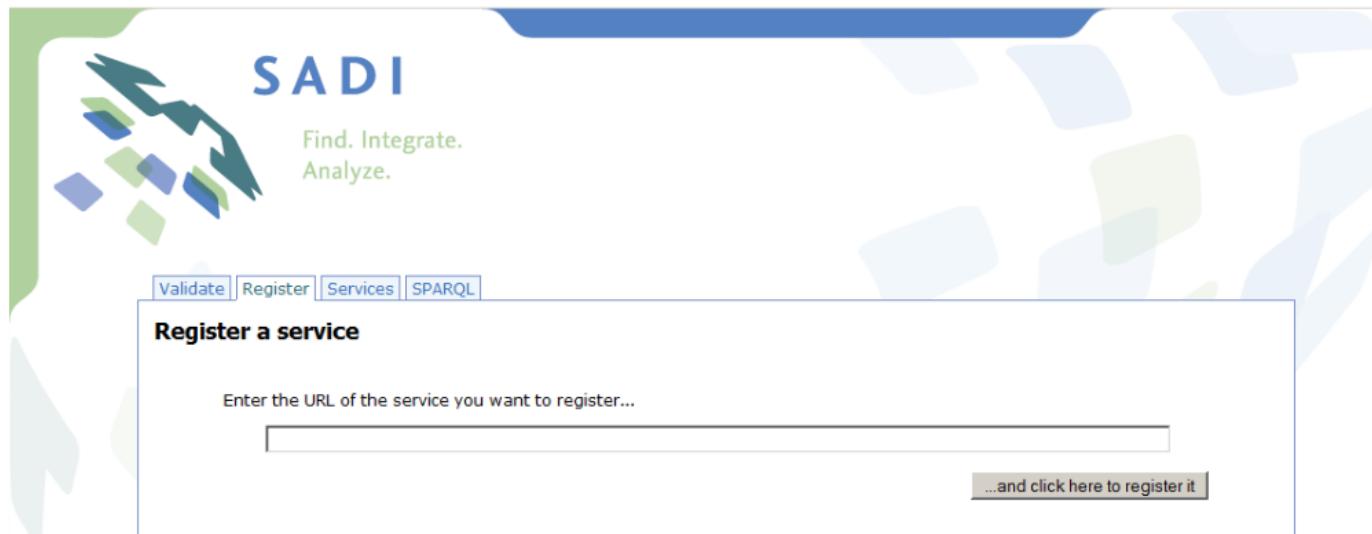
Service URL
+ <a href="http://unbsj.biordf.net/util-sadi-serv/pdf2ascii">http://unbsj.biordf.net/util-sadi-serv/pdf2ascii</a>
+ <a href="http://unbsj.biordf.net/util-sadi-serv/getWikipediaPageByTopic">http://unbsj.biordf.net/util-sadi-serv/getWikipediaPageByTopic</a>
- <a href="http://unbsj.biordf.net/util-sadi-serv/getUniProtIdByProteinName">http://unbsj.biordf.net/util-sadi-serv/getUniProtIdByProteinName</a>

**Name** getUniProtIdByProteinName  
**Description** Finds UniProtId of protein by its name  
**Properties attached** [is attribute of](#)  
(with values from [http://semanticscience.org/resource/SIO\\_010043](http://semanticscience.org/resource/SIO_010043))

+ <a href="http://unbsj.biordf.net/util-sadi-serv/getPMIDByPMCID">http://unbsj.biordf.net/util-sadi-serv/getPMIDByPMCID</a>
+ <a href="http://unbsj.biordf.net/util-sadi-serv/getPMIDByKeyPhrase">http://unbsj.biordf.net/util-sadi-serv/getPMIDByKeyPhrase</a>
+ <a href="http://unbsj.biordf.net/util-sadi-serv/getPMCIDBvPMID">http://unbsj.biordf.net/util-sadi-serv/getPMCIDBvPMID</a>
+ <a href="http://unbsj.biordf.net/util-sadi-serv/getPDFByPMID">http://unbsj.biordf.net/util-sadi-serv/getPDFByPMID</a>
+ <a href="http://unbsj.biordf.net/util-sadi-serv/getPDFByPMCID">http://unbsj.biordf.net/util-sadi-serv/getPDFByPMCID</a>

<http://sadiframework.org/registry/services/>

By the way, you can register your own SADI services here too.



HEART &  
STROKE  
FOUNDATION  
OF BC & YUKON



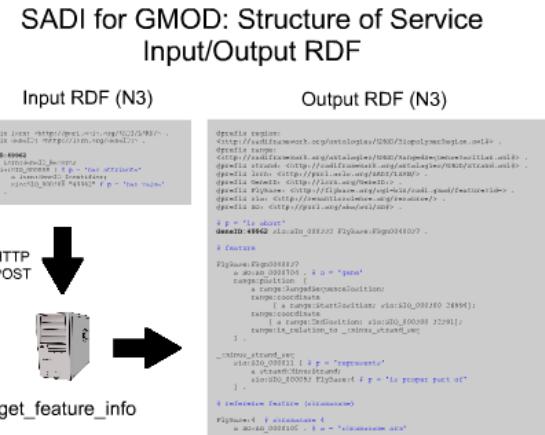
Development of SADI is generously supported by CANARIE, the Heart and Stroke Foundation of B.C. and Yukon, the Canadian Institutes of Health Research, and Microsoft Research.

<http://sadiframework.org/registry/register/>

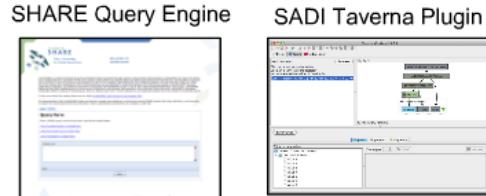
# How do I use the services? (for GMOD users)

## SADI in a Nutshell

- **to invoke a SADI service:**
    - HTTP POST an RDF document to the service URL
  - **to get service metadata:**
    - HTTP GET on service URL
    - returns an RDF document with service name, description, etc.
  - **structure of input/output data is described in OWL**
    - service provider specifies one *input* OWL class and one *output* OWL class
  - **strengths of SADI**
    - no framework-specific messaging formats or ontologies
    - supports batch processing of inputs
    - supports long-running services (asynchronous services)



SADI Client Software



*SPARQL Query => SADI Workflow*  
<http://biordf.net/cardioSHARE/query>

## Design SADI Workflows

# SADI in a Nutshell

- **to invoke a SADI service:**
  - HTTP POST an RDF document to the service URL
- **to get service metadata:**
  - HTTP GET on service URL
  - returns an RDF document with service name, description, etc.
- **structure of input/output data is described in OWL**
  - service provider specifies one *input OWL class* and one *output OWL class*
- **strengths of SADI**
  - no framework-specific messaging formats or ontologies
  - supports batch processing of inputs
  - supports long-running services (asynchronous services)

# SADI for GMOD: Structure of Service Input/Output RDF

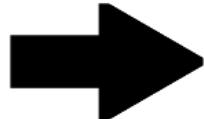
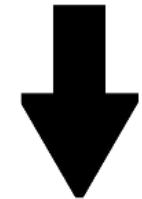
Input RDF (N3)

```
@prefix lsrn: <http://purl.oclc.org/SADI/LSRN/> .  
@prefix GeneID: <http://lsrn.org/GeneID:> .  
  
GeneID:49962  
  a lsrn:GeneID_Record;  
  sio:SIO_000008 [ # p = 'has attribute'  
    a lsrn:GeneID_Identifier;  
    sio:SIO_000300 "49962" # p = 'has value'  
  ] .
```

Output RDF (N3)

```
@prefix region: <http://sadiframework.org/ontologies/GMOD/BiopolymerRegion.owl#> .  
@prefix range: <http://sadiframework.org/ontologies/GMOD/RangedSequencePosition.owl#> .  
@prefix strand: <http://sadiframework.org/ontologies/GMOD/Strand.owl#> .  
@prefix lsrn: <http://purl.oclc.org/SADI/LSRN/> .  
@prefix GeneID: <http://lsrn.org/GeneID:> .  
@prefix FlyBase: <http://flybase.org/cgi-bin/sadi.gmod/feature?id=> .  
@prefix sio: <http://semanticscience.org/resource/> .  
@prefix SO: <http://purl.org/obo/owl/SO#> .  
  
# p = 'is about'  
GeneID:49962 sio:SIO_000332 FlyBase:FBgn0040037 .  
  
# feature  
  
FlyBase:FBgn0040037  
  a SO:SO_0000704 . # o = 'gene'  
  range:position [  
    a range:RangedSequencePosition;  
    range:coordinate [ a range:StartPosition; sio:SIO_000300 26994];  
    range:coordinate [ a range:EndPosition; sio:SIO_000300 32391];  
    range:in_relation_to _:minus_strand_seq  
  ] .  
  
_:minus_strand_seq  
  sio:SIO_000011 [ # p = 'represents'  
    a strand:MinusStrand;  
    sio:SIO_000093 FlyBase:4 # p = 'is proper part of'  
  ] .  
  
# reference feature (chromosome)  
  
FlyBase:4 # chromosome 4  
  a SO:SO_0000105 . # o = 'chromosome arm'
```

HTTP  
POST



get\_feature\_info

# Input RDF (N3)

```
@prefix lsrn: <http://purl.oclc.org/SADI/LSRN/> .  
@prefix GeneID: <http://lsrn.org/GeneID:> .
```

**GeneID:49962**

```
a lsrn:GeneID_Record;  
sio:SIO_000008 [ # p = 'has attribute'  
    a lsrn:GeneID_Identifier;  
    sio:SIO_000300 "49962" # p = 'has value'  
] .
```



```
@prefix sio: <http://semanticscience.org/resource/> .  
@prefix SO: <http://purl.org/obo/owl/SO#> .  
  
# p = 'is about'  
GeneID:49962 sio:SIO_000332 FlyBase:FBgn0040037 .  
  
# feature  
  
FlyBase:FBgn0040037  
    a SO:SO_0000704 . # o = 'gene'  
    range:position [  
        a range:RangedSequencePosition;  
        range:coordinate  
            [ a range:StartPosition; sio:SIO_000300 26994];  
        range:coordinate  
            [ a range:EndPosition; sio:SIO_000300 32391];  
        range:in_relation_to _:minus_strand_seq  
    ] .  
  
_:minus_strand_seq  
    sio:SIO_000011 [ # p = 'represents'  
        a strand:MinusStrand;  
        sio:SIO_000093 FlyBase:4 # p = 'is proper part of'  
    ] .  
  
# reference feature (chromosome)  
  
FlyBase:4 # chromosome 4  
    a SO:SO_0000105 . # o = 'chromosome arm'
```

# SADI Client Software

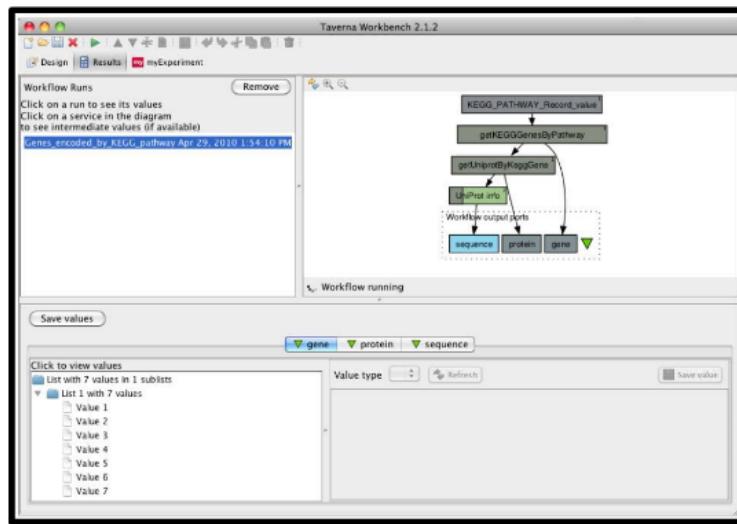
# SHARE Query Engine



## *SPARQL Query => SADI Workflow*

<http://biordf.net/cardioSHARE/query>

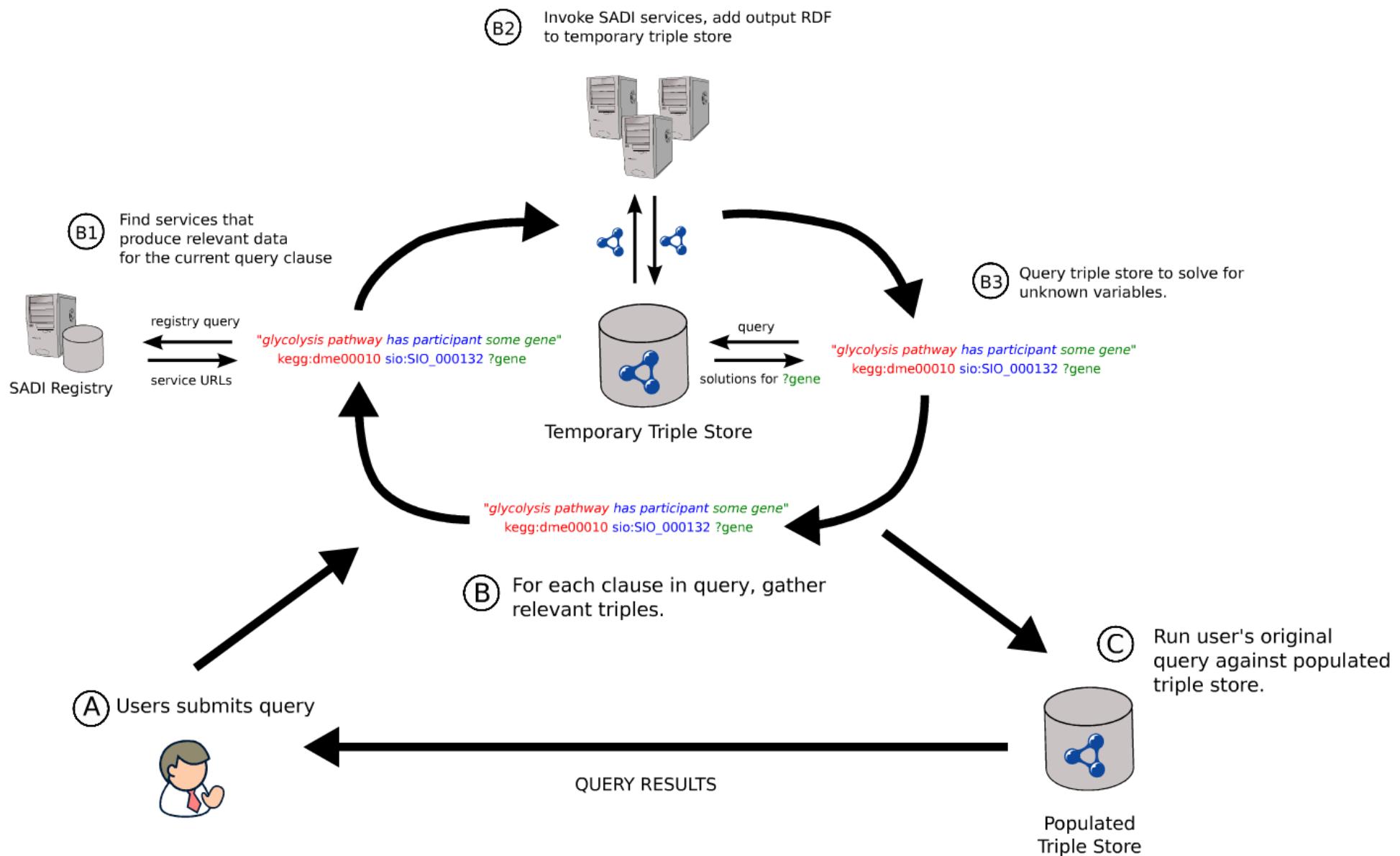
# SADI Taverna Plugin

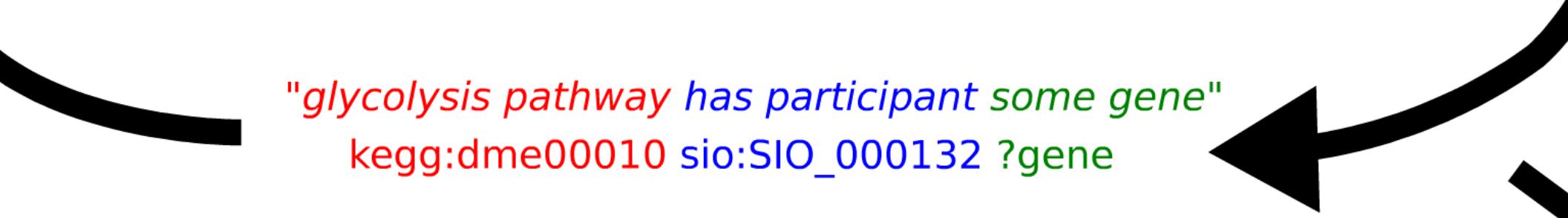


## *Design SADI Workflows*

*http://sadiframework.org/content/2010/05/03/sadi-taverna-plugin-tutorial/*

# SHARE Query Resolution





*"glycolysis pathway has participant some gene"*  
kegg:dme00010 sio:SIO\_000132 ?gene

- ④ For each clause in query, gather relevant triples.

---

## QUERY RESULTS

B1

Find services that produce relevant data for the current query clause



SADI Registry

registry query  
service URLs

"glycolysis pathway has participant some gene"  
kegg:dme00010 sio:SIO\_000132 ?gene

Tempor

"glycolysis pathway  
kegg:dme00010

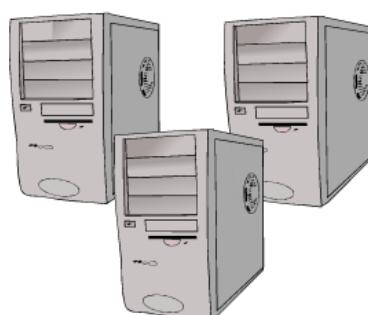
B

For each clause, find services that produce relevant data

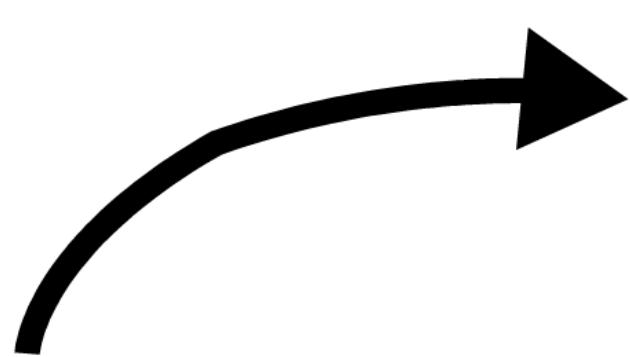
# How to query tools

B2

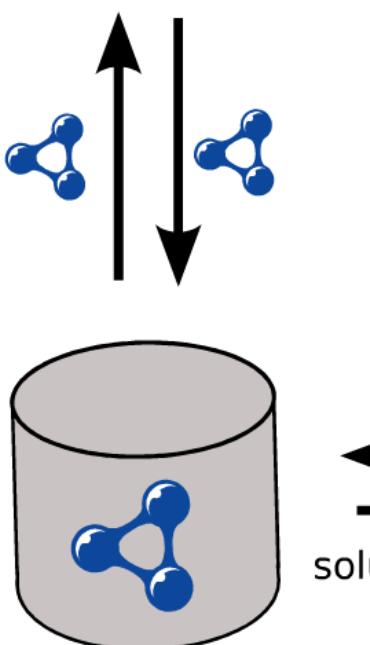
Invoke SADI services, add output RDF  
to temporary triple store



ause



*"glycolysis pathway has participant some gene"*  
kegg:dme00010 sio:SIO\_000132 ?gene

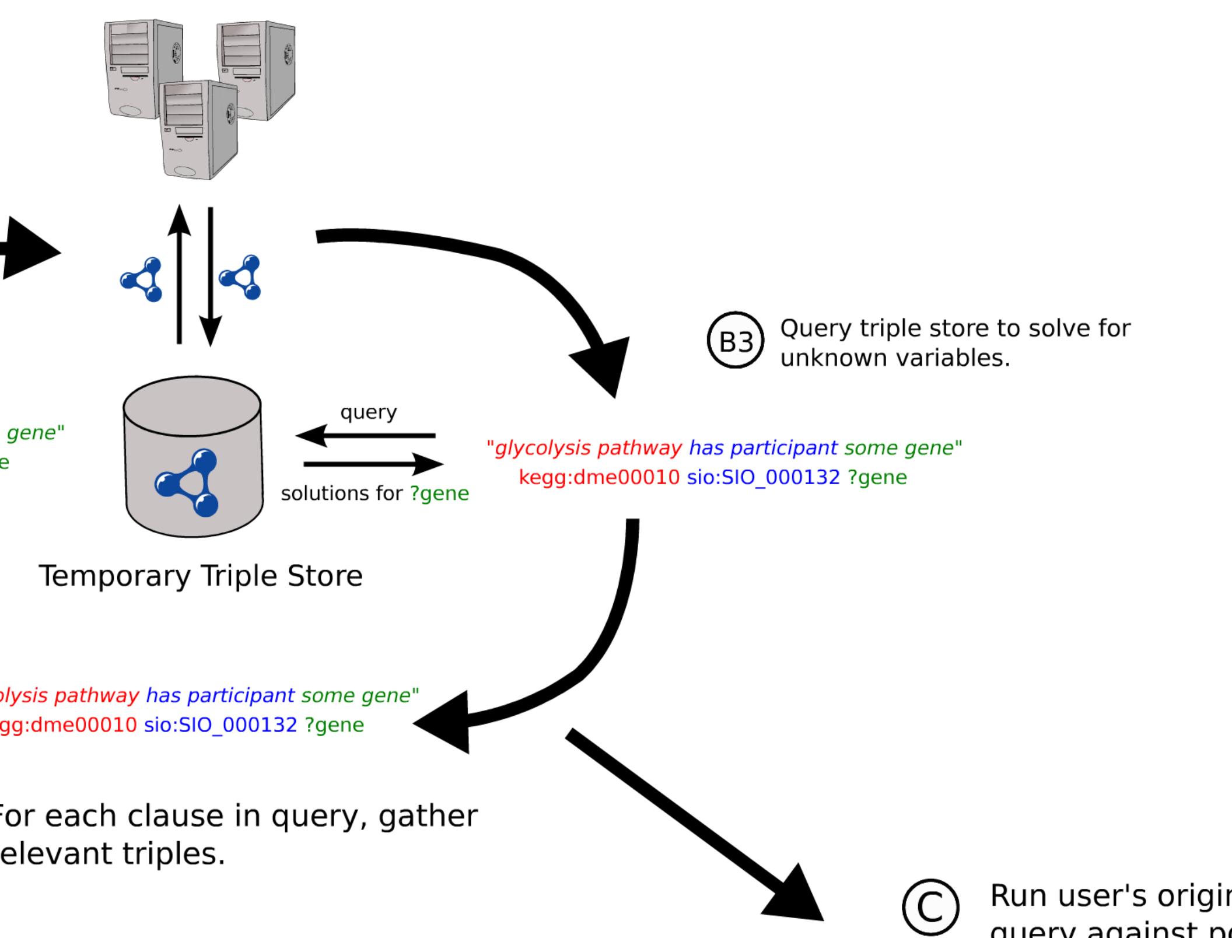


Temporary Triple Store

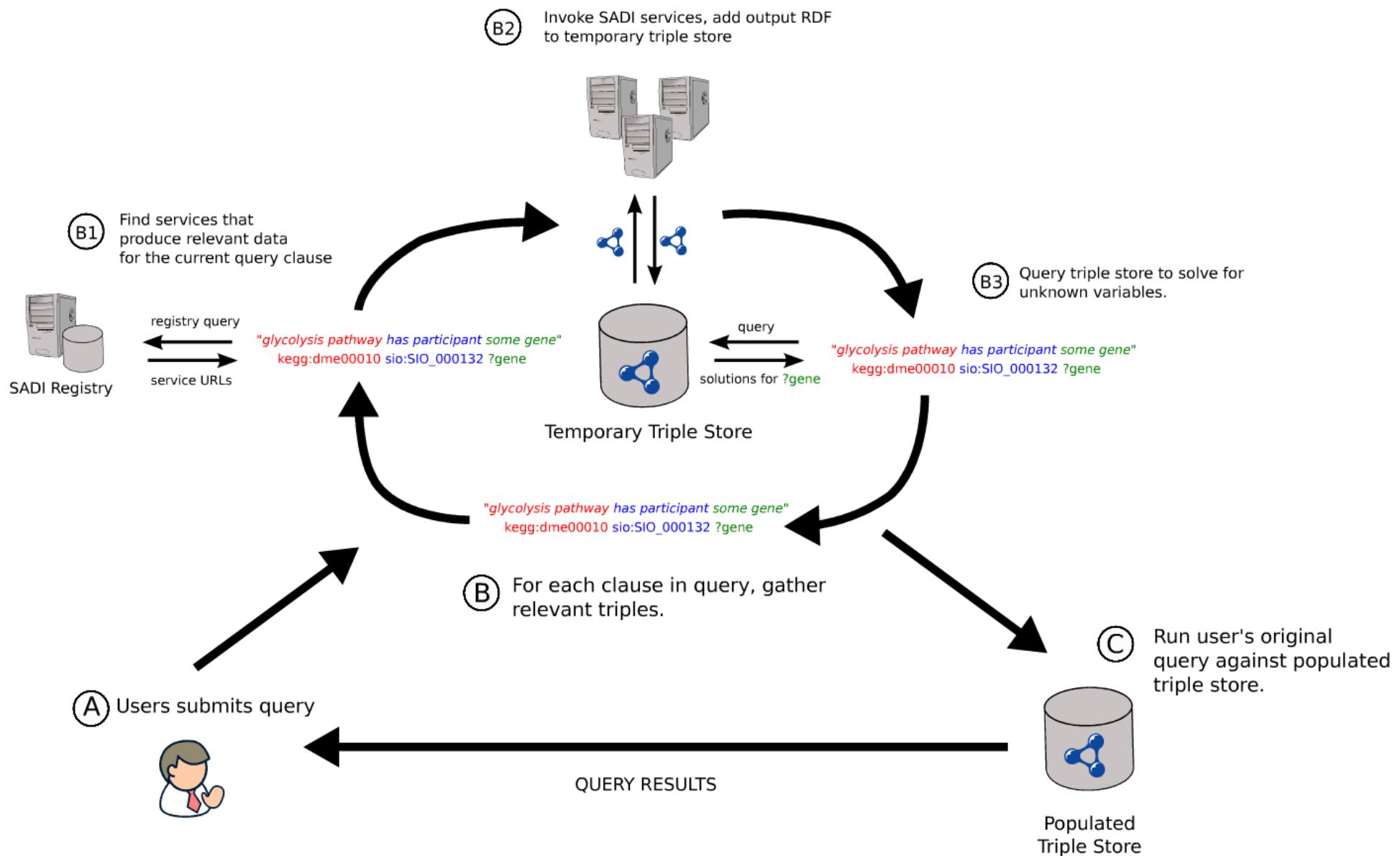
B3

*"glycolysis pathway has pa*  
kegg:dme00010 sio:SIO





# SHARE Query Resolution



# Demo Query

"Find FlyBase genes that participate in glycolysis and overlap genomic region 5,919,623..6,344,662 on chromosome 3L"

```
PREFIX feature: <http://s7.semanticscience.org/~ben/cgi-bin/FlyBase/feature?id=>
PREFIX range: <http://sadiframework.org/ontologies/GMOD/RangedSequencePosition.owl#>
PREFIX region: <http://sadiframework.org/ontologies/GMOD/BiopolymerRegion.owl#>
PREFIX strand: <http://sadiframework.org/ontologies/GMOD/Strand.owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX sio: <http://semanticscience.org/resource/>
PREFIX pathway: <http://lsrn.org/KEGG_PATHWAY:>

SELECT ?flybase_gene_record ?startpos ?endpos ?strand_type
WHERE {

    pathway:dme00010  sio:SIO_000132 ?kegg_gene_record .      # SIO_000132 = 'has participant'
    ?kegg_gene_record  owl:sameAs      ?flybase_gene_record .
    ?flybase_gene_record sio:SIO_000332 ?flybase_feature .      # SIO_000332 = 'is about'

    ?flybase_feature region:position [
        range:coordinate [ rdf:type range:StartPosition; sio:SIO_000300 ?startpos ];
        range:coordinate [ rdf:type range:EndPosition; sio:SIO_000300 ?endpos ];
        range:in_relation_to [
            sio:SIO_000210 [
                rdf:type ?strand_type;
                sio:SIO_000093 feature:3L
            ] # SIO_000210 = 'represents'
            ] # SIO_000093 = 'is proper part of'
    ];
];

FILTER ((?endpos >= 5919623) && (?startpos <= 6344662))
FILTER ((?strand_type = strand:MinusStrand) || (?strand_type = strand:PlusStrand))

}
```



PREFIX owl: <<http://www.w3.org/2002/07/owl#>>

PREFIX sio: <<http://semanticscience.org/resource/>>

PREFIX pathway: <[http://lsrn.org/KEGG\\_PATHWAY:](http://lsrn.org/KEGG_PATHWAY:)>

SELECT ?flybase\_gene\_record ?startpos ?endpos ?strand\_type

WHERE {

pathway:dme00010 sio:SIO\_000132 ?kegg\_gene\_record . # SIO\_000132 = 'has part'  
?kegg\_gene\_record owl:sameAs ?flybase\_gene\_record .  
?flybase\_gene\_record sio:SIO\_000332 ?flybase\_feature . # SIO\_000332 = 'is about'

?flybase\_feature region:position [

range:coordinate [ rdf:type range:StartPosition; sio:SIO\_000300 ?startpos ];

range:coordinate [ rdf:type range:EndPosition; sio:SIO\_000300 ?endpos ];

range:in\_relation\_to [

sio:SIO\_000210 [ # SIO\_000210 = 'represents'  
rdf:type ?strand\_type;

sio:SIO\_000093 feature:3L # SIO\_000093 = 'is proper'  
]

]

];

FILTER ((?endpos >= 5919623) && (?startpos <= 6344662))

FILTER ((?strand\_type = strand:MinusStrand) || (?strand\_type = strand:PlusStrand))

}

**Query**

Browse

## Query form

Enter a SPARQL query in the text box below and click the submit button.

[A list of example queries is available here.](#)

[Learn how to build your own query here.](#)

[A list of predicates is available here.](#)

SPARQL query:

```
PREFIX sio: <http://semanticscience.org/resource/>
PREFIX pathway: <http://lsrn.org/KEGG_PATHWAY:>

SELECT ?flybase_gene_record ?startpos ?endpos ?strand_type
WHERE {
    pathway:dme00010      sio:SIO_000132  ?kegg_gene_record .      # SIO_000132 = 'has participant'
```

Ready.

Submit the query to SHARE.

A screenshot of a web-based SPARQL query interface. On the left, there is a vertical dropdown menu with an upward arrow at the top and a downward arrow at the bottom. The text 'participant' is visible on the left side of the interface. At the bottom, there is a large blue button with a white outline.

RE.



Wait a while...



A screenshot of a web-based SPARQL query interface. At the top, there are tabs for 'Query' (which is selected) and 'Browse'. Below the tabs, the word 'Query' is displayed in bold. There is a text input field with placeholder text 'Enter a SPARQL query...'. Below the input field, there are several links: 'A list of examples', 'Learn how to use SPARQL', and 'A list of prefixes'. On the right, there is a section titled 'SPARQL code' containing a SPARQL query. At the bottom, there is a 'View results' button with a warning icon.

Query results

flybase\_gene\_id  
[http://lsrn.org/flybase\\_gene\\_id](http://lsrn.org/flybase_gene_id)  
[http://lsrn.org/flybase\\_gene\\_id](http://lsrn.org/flybase_gene_id)

## Query form

Enter a SPARQL query in the text box below and click the submit button.

[A list of example queries is available here.](#)

[Learn how to build your own query here.](#)

[A list of predicates is available here.](#)

SPARQL query:

```
PREFIX sio: <http://semanticscience.org/resource/>
PREFIX pathway: <http://lsrn.org/KEGG_PATHWAY:>

SELECT ?flybase_gene_record ?startpos ?endpos ?strand_type
WHERE {
    pathway:dme00010      sio:SIO_000132  ?kegg_gene_record .      # SIO_000132 = 'has participant'
```

 [View results as RDF](#). There were warnings executing the query. Click for details.

### Query results

flybase_gene_record	startpos	endpos	strand_type
<a href="http://lsrn.org/FLYBASE:FBgn0001258">http://lsrn.org/FLYBASE:FBgn0001258</a> 	6252592	6255793	<a href="http://sadiframework.org/ontologies/GMOD/Strand.owl#MinusStrand">http://sadiframework.org/ontologies/GMOD/Strand.owl#MinusStrand</a>
<a href="http://lsrn.org/FLYBASE:FBgn0035679">http://lsrn.org/FLYBASE:FBgn0035679</a>	6086304	6087836	<a href="http://sadiframework.org/ontologies/GMOD/Strand.owl#MinusStrand">http://sadiframework.org/ontologies/GMOD/Strand.owl#MinusStrand</a>

# How do I set up the services? (for GMOD providers)

1. Load your GFF files into a `Bio::DB::SeqFeature::Store` (will soon support Chado, also)
2. Install SADI for GMOD dependencies with CPAN
3. Download the SADI for GMOD tarball and unpack into `cgi-bin`
4. Set DB connection parameters in `cgi-bin/sadi.gmod/sadi.gmod.conf`

```
[GENERAL]
db_adaptor = Bio::DB::SeqFeature::Store
db_args     = -adaptor DBI:mysql
              -dsn      dbi:mysql:database=flybase
base_url    = http://flybase.org/cgi-bin/sadi.gmod/
```

5. Configure Dbxref mappings in `cgi-bin/sadi.gmod/dbxref.conf`

```
[DBXREF_TO_LSRN]
SwissProt = UniProt
UniProtKB = UniProt
SwissProt/TrEMBL = UniProt
...
```

6. Register the services in public SADI registry: <http://sadiframework.org/registry>

3. Download the SADI for GMOD tarball and unpack into cgi-bin

4. Set DB connection parameters in cgi-bin/sadi.gmod/sadi.gmod.conf

```
[GENERAL]
db_adaptor = Bio::DB::SeqFeature::Store
db_args     = -adaptor DBI::mysql
              -dsn      dbi:mysql:database=flybase
base_url    = http://flybase.org/cgi-bin/sadi.gmod/
```

5. Configure Dbxref mappings in cgi-bin/sadi.gmod/dbxref.conf

```
[DBXREF_TO_LSRN]
SwissProt = UniProt
UniProtKB = UniProt
SwissProt/TrEMBL = UniProt
...
...
```

6. Register the services in public SADI registry: <http://sadiframework.org/re>

```
db_args      = -adaptor DBI::mysql  
              -dsn       dbi:mysql:database=flybase  
base_url    = http://flybase.org/cgi-bin/sadi.gmod/
```

## 5. Configure Dbxref mappings in cgi-bin/sadi.gmod/dbxref.conf

```
[DBXREF_TO_LSRN]  
SwissProt = UniProt  
UniProtKB = UniProt  
SwissProt/TrEMBL = UniProt  
...  
...
```

## 6. Register the services in public SADI registry: <http://sadiframework.org>

Ben Vandervalk\*,<sup>1</sup> Lu  
James Hogg Rese

# Future plans

- Chado support (soon!)
- add BLAST service (anything else that you want?)
- use cases and demos
- more GMOD mirrors
- page on GMOD wiki
- distribute with Tripal or GBrowse?

# Acknowledgements

## Team

Mark Wilkinson: Principal Investigator

Michel Dumontier: Principal Investigator (ontologies and data modelling)

Luke McCarthy: Lead Programmer, SADI & SHARE

Edward Kawas: Perl Programmer, SADI

## Funding



Canada's Advanced Research and Innovation Network  
Le réseau évolué de recherche et d'innovation du Canada



HEART &  
STROKE  
FOUNDATION



Microsoft  
Research

*<http://sadiframework.org/>*