intriguing

arousing the curiosity or interest

but also

making secret plans to do something illicit

Biomoby

Integration and intriguing semantics

Martin Senger
martin.senger@gmail.com

First thing first...

- There are two Biomoby branches
  - this update is about “Moby-S” (Moby Services)
  - the other one is “S-Moby” (Semantic Moby)
    - http://semanticmoby.org/
- Acknowledgement
  - Mark Wilkinson, PI and creator of Biomoby
  - many groups around the world working with and for Biomoby, e.g.
    - Generation Challenge Programme of the Consultative Group for International Agricultural Research
    - The PlaNet Consortium (a network of European plant databases)
    - The Australian Centre for Plant Functional Genomics
    - The National Institute for Bioinformatics, Spain (Genome Espania)
- Where to find more
  - http://biomoby.org

Biomoby in a nutshell

...for those not yet initiated

I need data.
Why should I use Biomoby?

- Because you get data from hundreds of services
- Because these data and services can interoperate (exchange their data)
- Because you need to run programs to consume data (semi-)automatically
  - – if you can get what you need just by clicking on web pages, you do not need Biomoby

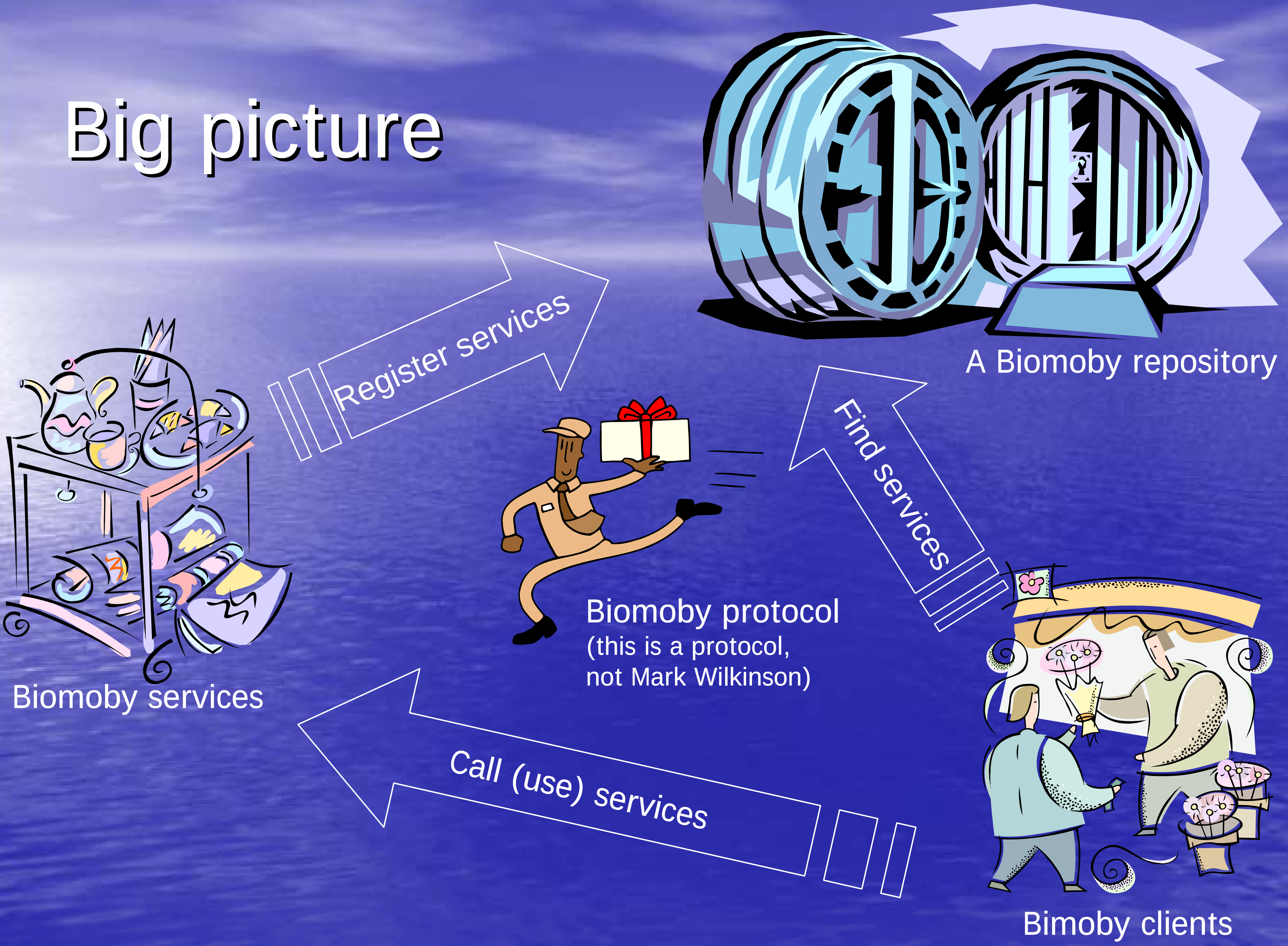
I have data.
Why should I use Biomoby?

- Because your data can be shared (accessed by others)
- Because Biomoby helps to get your data visible (almost without programming)
  - it does not help, however, to create web pages showing your data in web browsers
- Because you can add-value to your data by linking them to other Biomoby-aware data

What, actually, is Biomoby?

- A registry (a computer) that knows where to find services around the world
- A registry (a computer) that knows what data are being served by these services, and how the data are related to each other
- A standard (a specification) telling how to access such data (how to call such services)
- Growing number of software tools (programs) that allow to provide, to get, to browse and to combine such data
- A community of dedicated (and often nice) people to help, and to have a beer with you...

Big picture

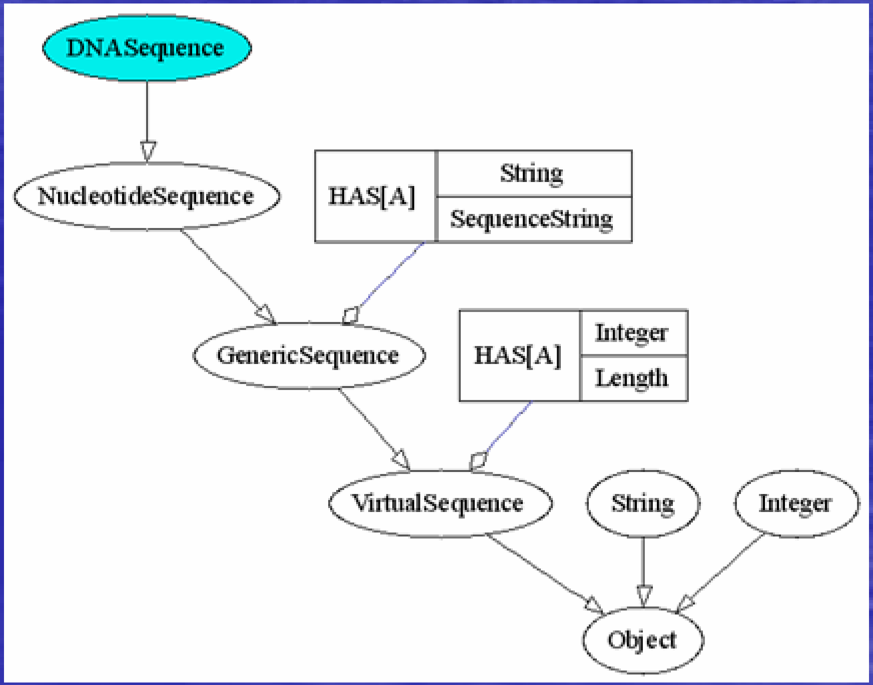

Register services
A Biomoby repository
Find services
Biomoby protocol
(this is a protocol,
not Mark Wilkinson)
Biomoby services
Call (use) services
Bimoby clients

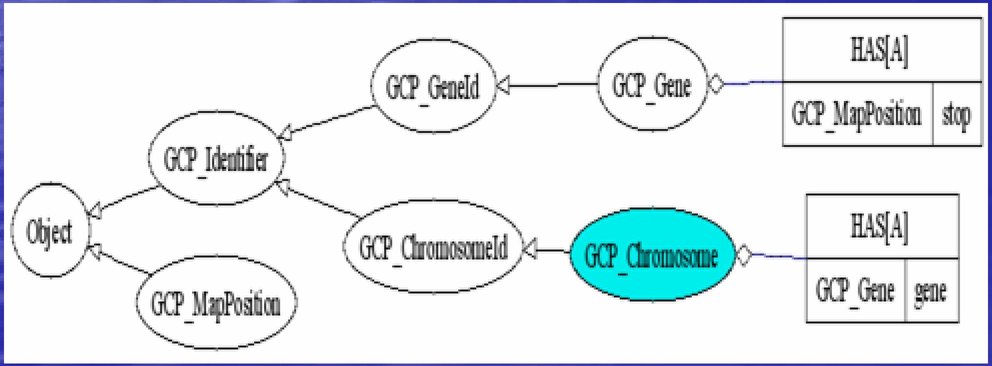
Bottom line

- *Biomoby services* are your responsibility
  - you are a service provider, you implement your service (but Biomoby project has tools to help you - Moses for Java, Perl libraries, ...)
- *Biomoby data types* are community responsibility
  - otherwise it would limit how they can be shared and re-used
  - you are part of the community: register your data types

What is registered

- Ontology 1: Data types
  - What data represent and how they are related
  - They all sit in one hierarchical tree (ISA)
  - They have children
    - HAS (more of this kind)
    - HASA (maximum one)



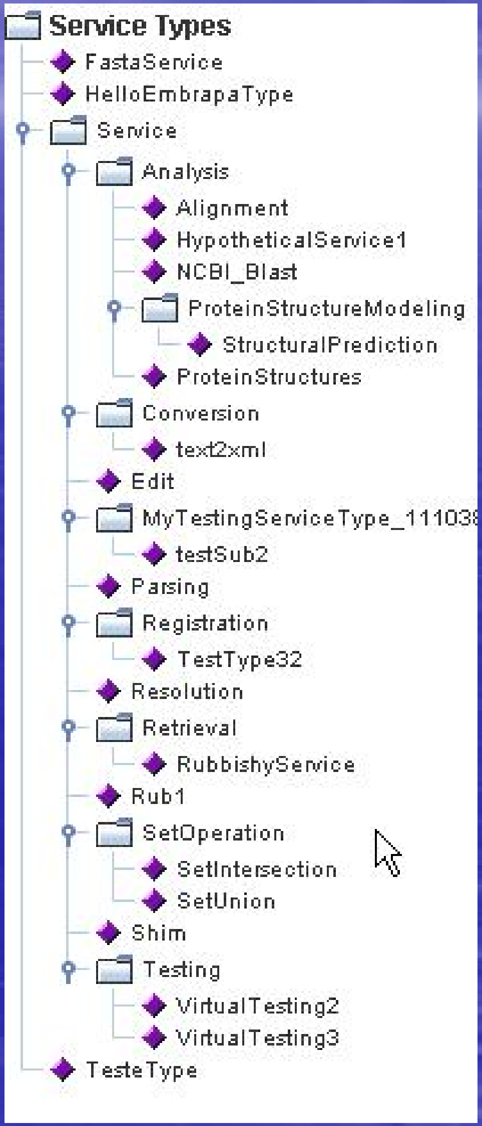


What is registered

- Ontology 2: Namespaces
  - define the scope of your data
    - geographically (where a database is located)
      - e.g. “NIAS_OryzaMutant”
    - semantically (what kind of database data are in)
      - Example: If you have a datum identified by a string “163483”, you have no clue what it is, unless you say “the namespace is “NCBI_gi”. Another example of a namespace is “ICIS_Germplasm”.
  - no hierarchy – just a plain control vocabulary

What is registered

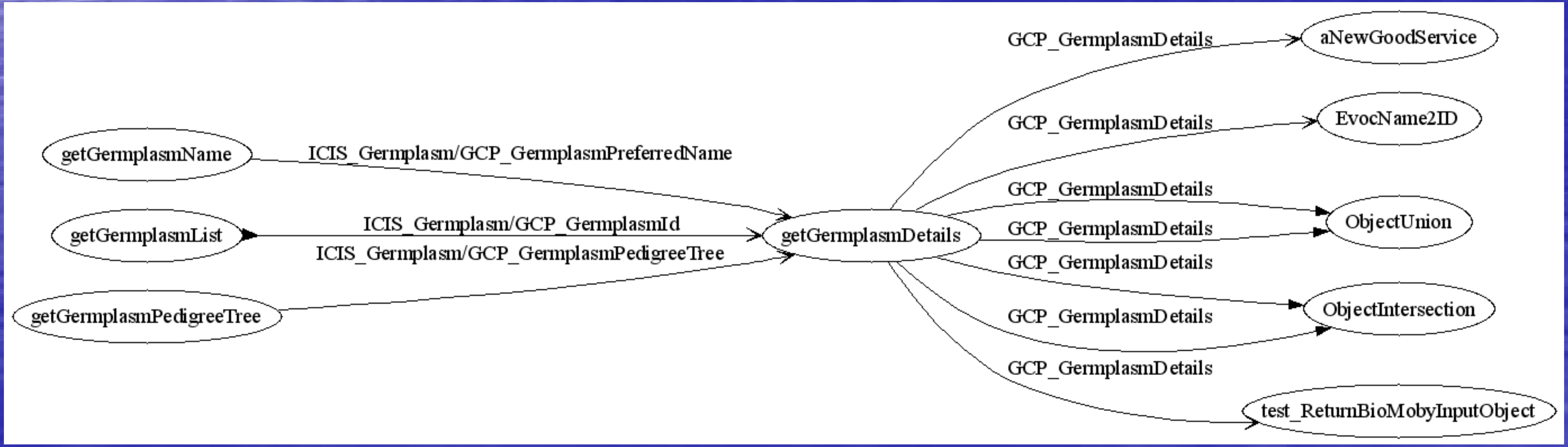
- Ontology 3: Service Types
  - a hierarchy of all kinds of services
  - it helps to discover your service
  - it is not yet mature enough
    - changes expected here
    - collaboration with myGrid,...



- Service Types
  - FastaService
  - HelloEmbrapaType
  - Service
    - Analysis
      - Alignment
      - HypotheticalService1
      - NCBI_Blast
      - ProteinStructureModeling
        - StructuralPrediction
      - ProteinStructures
    - Conversion
      - text2xml
    - Edit
    - MyTestingServiceType_11103
      - testSub2
    - Parsing
    - Registration
      - TestType32
    - Resolution
    - Retrieval
      - RubbishyService
    - Rub1
    - SetOperation
      - SetIntersection
      - SetUnion
    - Shim
    - Testing
      - VirtualTesting2
      - VirtualTesting3
  - TesteType

What is registered

- Ontology 4: Services
  - where they are (an endpoint)
  - where to find more about them (a URL with an RDF document that is partly maintained by the service provider)
  - what input and output data they can consume and provide



Biomoby major trick how to gain interoperability between services

- Each service must understand data type as declared in the registry
  - this is usual
- Each service must be able to ignore more specific data, if they come, and not to break itself on them
  - this is usual in programming languages but it is not that common in Web Services world
  - it is possible because data types are related in a hierarchy

Biomoby update

...what happened over the past year

New funding...

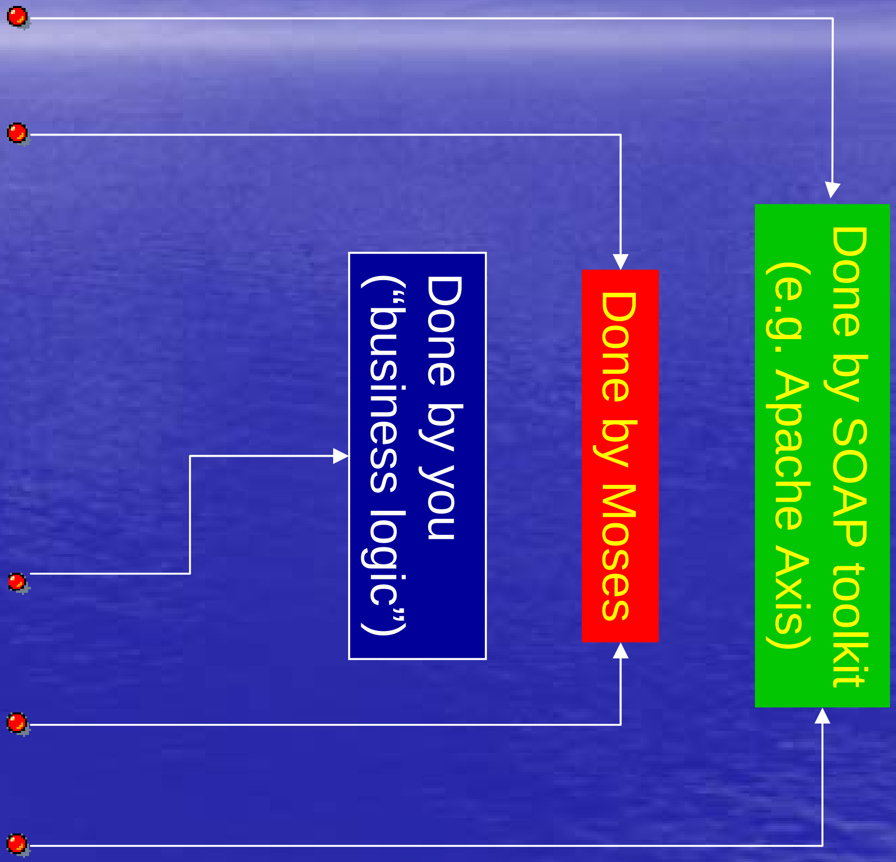
- Keep and enhance current Biomoby
- Research on Biomoby 2
  - more about service discovery
    - semantics as a hype or reality?
  - cautious approaches to S-Moby
    - could we have just one?

jMoby: Biomoby for Java

- major pieces are
  - Java libraries (API) for accessing registry
    - Central.java
  - Generators of Biomoby service skeletons (MoSeS = Moby Services Support)
    - a framework that you extend by your own implementation to create your own services
    - *coming soon:* fully generated services accessing data using BioCASE, Soaplab and Hibernate
      - no need to write any implementation code for services
  - Dashboard...

To write a Biomoby service, one needs:

- To extract data from a SOAP envelope
- To expect incoming data in different encoding (data can be a String or a byte array)
- To extract data from a Biomoby XML envelope
- To separate data into individual "jobs" (a request can consist of many of them)
- [To get installation parameters from the surrounding servlet engine]
- To do something meaningful with data (to create results)
- To convert results back into response "jobs"
- To wrap results into a Biomoby XML
- To send data back in a SOAP envelope

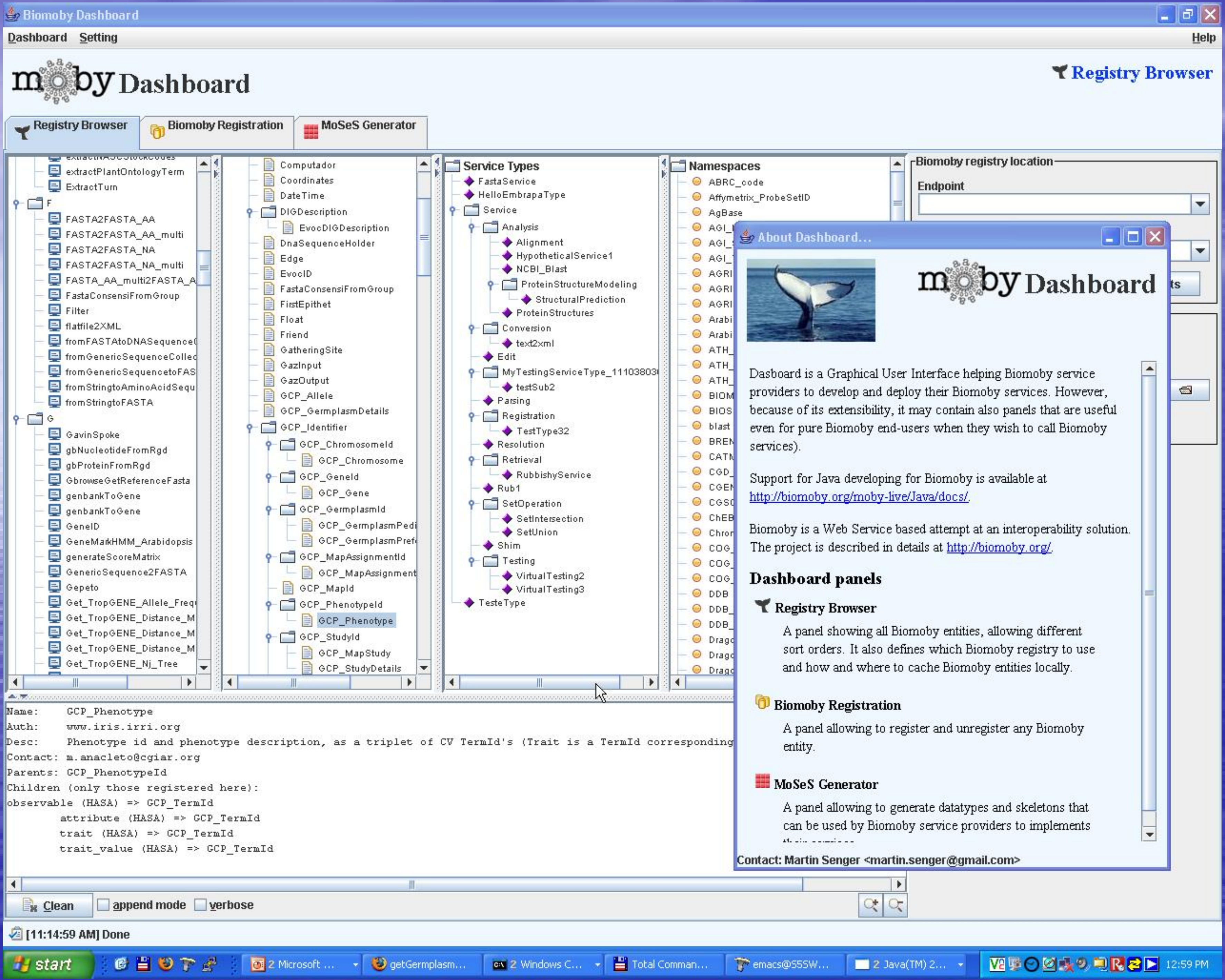

Done by you ("business logic")
Done by Moses
Done by SOAP toolkit (e.g. Apache Axis)

An example: a full Biomoby service "HelloBiomobyWorld"

```
package org.jmoby.tutorial.service;

import net.jmoby.samples.HelloBiomobyWorldSkel;
import org.biomoby.shared.MobyException;
import org.biomoby.shared.parser.MobyPackage;
import org.biomoby.shared.parser.MobyJob;
import org.biomoby.shared.datatypes.*;

public class HelloBiomobyWorldImpl
   extends HelloBiomobyWorldSkel {
   public void processIt (MobyJob request, MobyJob response,
                          MobyPackage outputContext)
      throws MobyException {
   set_greeting (response, new MobyString ("Hello, World!"));
   }
}
```



```
Name:    GCP_Phenotype
Auth:    www.iris.irri.org
Desc:    Phenotype id and phenotype description, as a triplet of CV TermId's (Trait is a TermId corresponding
Contact: m.anacleto@cgiar.org
Parents: GCP_PhenotypeId
Children (only those registered here):
observable (HASA) => GCP_TermId
       attribute (HASA) => GCP_TermId
       trait (HASA) => GCP_TermId
       trait_value (HASA) => GCP_TermId
```

About Dashboard...

moby Dashboard

Dasboard is a Graphical User Interface helping Biomoby service providers to develop and deploy their Biomoby services. However, because of its extensibility, it may contain also panels that are useful even for pure Biomoby end-users when they wish to call Biomoby services).

Support for Java developing for Biomoby is available at http://biomoby.org/moby-live/Java/docs/.

Biomoby is a Web Service based attempt at an interoperability solution. The project is described in details at http://biomoby.org/.

Dashboard panels

Registry Browser

A panel showing all Biomoby entities, allowing different sort orders. It also defines which Biomoby registry to use and how and where to cache Biomoby entities locally.

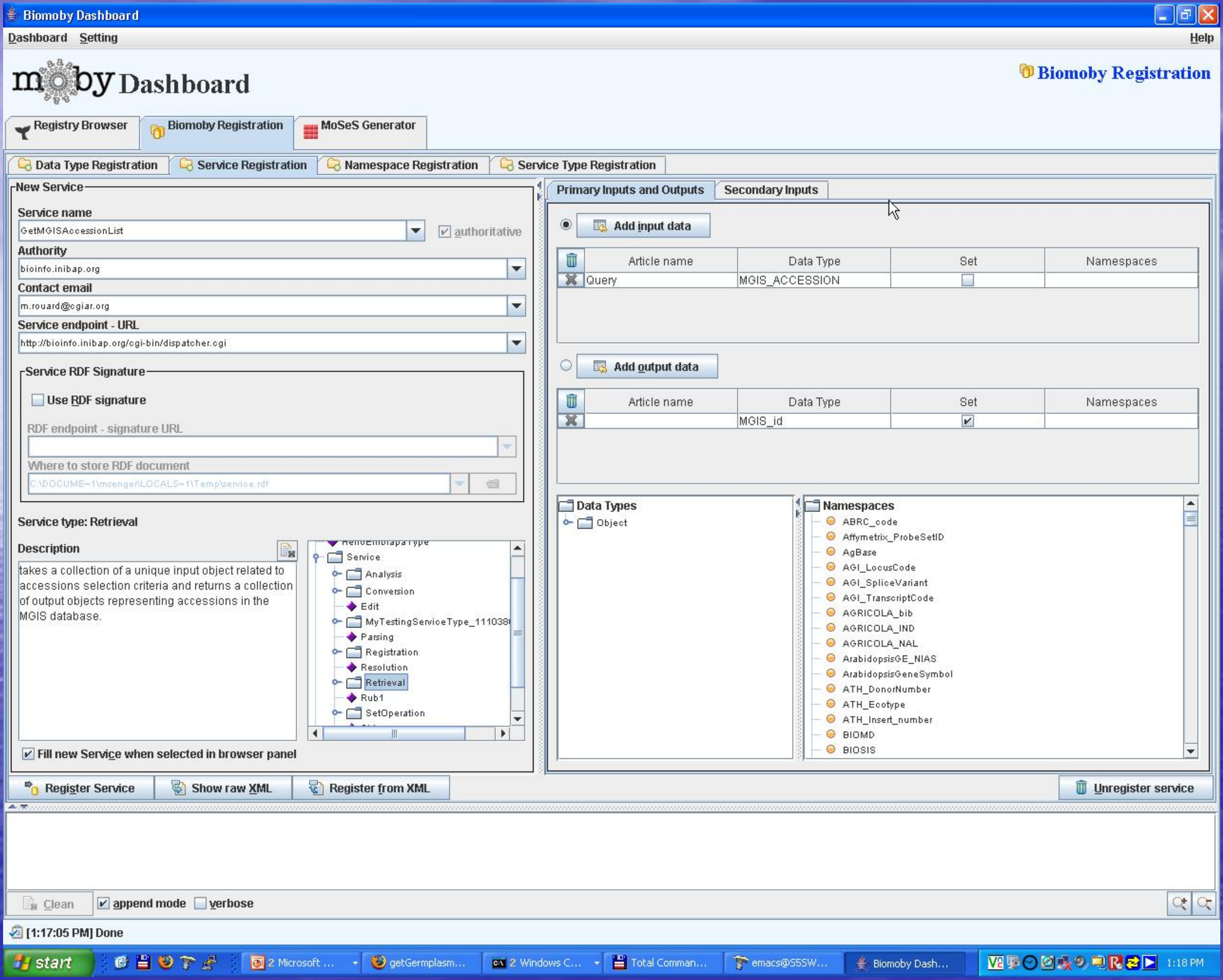
Biomoby Registration

A panel allowing to register and unregister any Biomoby entity.

MoSeS Generator

A panel allowing to generate datatypes and skeletons that can be used by Biomoby service providers to implements

Contact: Martin Senger <martin.senger@gmail.com>



New Service

Service name: GetMGISAccessionList [✓ authoritative]

Authority: bioinfo.inibap.org

Contact email: m.rouard@cgiar.org

Service endpoint - URL: http://bioinfo.inibap.org/cgi-bin/dispatcher.cgi

Service RDF Signature

☐ Use RDF signature

RDF endpoint - signature URL

Where to store RDF document: C:\DOCUME~1\msenger\LOCALS~1\Temp\service.rdf

Service type: Retrieval

Description: takes a collection of a unique input object related to accessions selection criteria and returns a collection of output objects representing accessions in the MGIS database.

Service: Analysis, Conversion, Edit, MyTestingServiceType_111038, Parsing, Registration, Resolution, Retrieval, Rub1, SetOperation

[✓] Fill new Service when selected in browser panel

Input data:

Article name	Data Type	Set	Namespaces
Query	MGIS_ACCESSION	☐	

Output data:

Article name	Data Type	Set	Namespaces
	MGIS_id	✓	

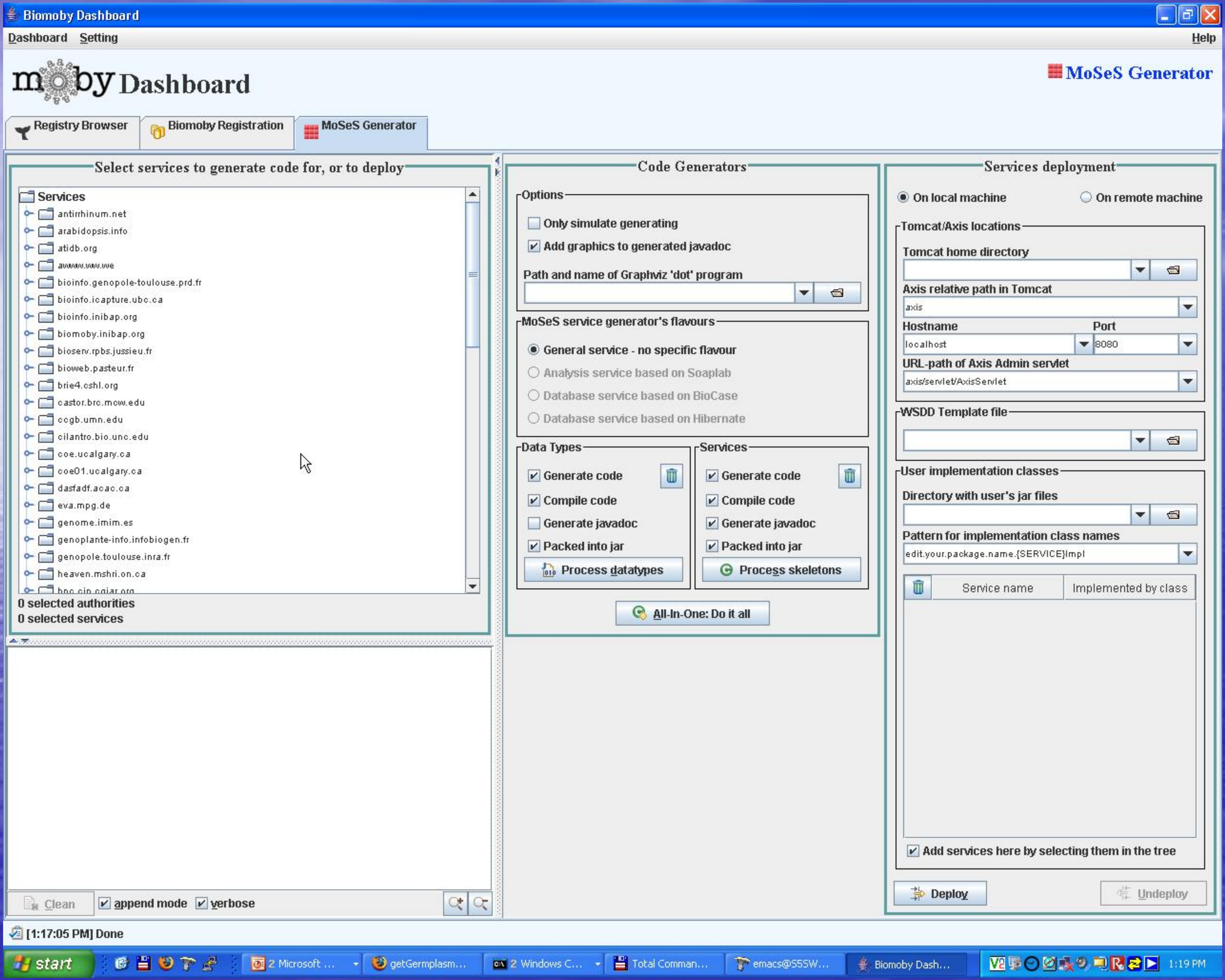
Data Types: Object

Namespaces: ABRC_code, Affymetrix_ProbeSetID, AgBase, AGI_LocusCode, AGI_SpliceVariant, AGI_TranscriptCode, AGRICOLA_bib, AGRICOLA_IND, AGRICOLA_NAL, ArabidopsisGE_NIAS, ArabidopsisGeneSymbol, ATH_DonorNumber, ATH_Ecotype, ATH_Insert_number, BIOMD, BIOSIS

[✓] append mode ☐ verbose

[1:17:05 PM] Done

2 Microsoft ... getGermplasm... 2 Windows C... Total Comman... emacs@SSSW... Biomoby Dash... 1:18 PM



Select services to generate code for, or to deploy

Services
- antirrhinum.net
- arabidopsis.info
- atidb.org
- awww.ww.we
- bioinfo.genopole-toulouse.prd.fr
- bioinfo.icapture.ubc.ca
- bioinfo.inibap.org
- biomoby.inibap.org
- bioserv.rpbs.jussieu.fr
- bioweb.pasteur.fr
- brie4.cshl.org
- castor.brc.mcw.edu
- ccgb.umn.edu
- cilantro.bio.unc.edu
- coe.ucalgary.ca
- coe01.ucalgary.ca
- dasfadf.acac.ca
- eva.mpg.de
- genome.imim.es
- genoplante-info.infobiogen.fr
- genopole.toulouse.inra.fr
- heaven.mshri.on.ca
- hpc.cip.cgiar.org

0 selected authorities
0 selected services

Clean ☑ append mode ☑ verbose

Code Generators

Options
- ☐ Only simulate generating
- ☑ Add graphics to generated javadoc

Path and name of Graphviz 'dot' program

MoSeS service generator's flavours
- ● General service - no specific flavour
- ○ Analysis service based on Soaplab
- ○ Database service based on BioCase
- ○ Database service based on Hibernate

Data Types
- ☑ Generate code
- ☑ Compile code
- ☐ Generate javadoc
- ☑ Packed into jar

Process datatypes

Services
- ☑ Generate code
- ☑ Compile code
- ☑ Generate javadoc
- ☑ Packed into jar

Process skeletons

All-In-One: Do it all

Services deployment

● On local machine ○ On remote machine

Tomcat/Axis locations

Tomcat home directory:

Axis relative path in Tomcat: axis

Hostname: localhost Port: 8080

URL-path of Axis Admin servlet: axis/servlet/AxisServlet

WSDD Template file:

User implementation classes

Directory with user's jar files:

Pattern for implementation class names: edit.your.package.name.{SERVICE}Impl

Service name	Implemented by class

☑ Add services here by selecting them in the tree

Deploy Undeploy

[1:17:05 PM] Done

There is definitely more...

- Biomoby plug-in to Taverna
- Asynchronous service invocation
- Perl-Moses
- ...

Summary

What is Biomoby good at...

- It has many running services
- It provides data models in a reasonably flexible way
- It has a potential to discover services in a modern way !
  - see also “MOBY 2” and Semantic Moby
- It has a potential to annotate services in a non-centralised way

What is Biomoby less good at...

- It has many crapped services
- It does not use fully potential of Web Services (WSDL etc.)
  - – perhaps it does not need to be SOAP-based at all (the pure HTTP can do the same here)
- The potential for service discovery by reasoning yet to be proved

Thank you...