ADMIRE Registry

15th April, 2011, Informatics Forum, Edinburgh
Carlos Buil Aranda, cbuil@fi.upm.es
Facultad de Informática, Universidad Politécnica de Madrid
Campus de Montegancedo sn, 28660 Boadilla del Monte, Madrid
http://www.oeg-upm.net
Phone: +34.91.3366605, Fax: +34.91.3524819

Outline

- Introduction to ADMIRE
- The ADMIRE Registry – What is it?
- The ADMIRE Registry – How it works?
- The ADMIRE Registry – Who uses it?
- Future work

Outline

- **Introduction to ADMIRE**
- The ADMIRE Registry – What is it?
- The ADMIRE Registry – How it works?
- The ADMIRE Registry – Who uses it?
- Future work

Introduction to ADMIRE - Basics

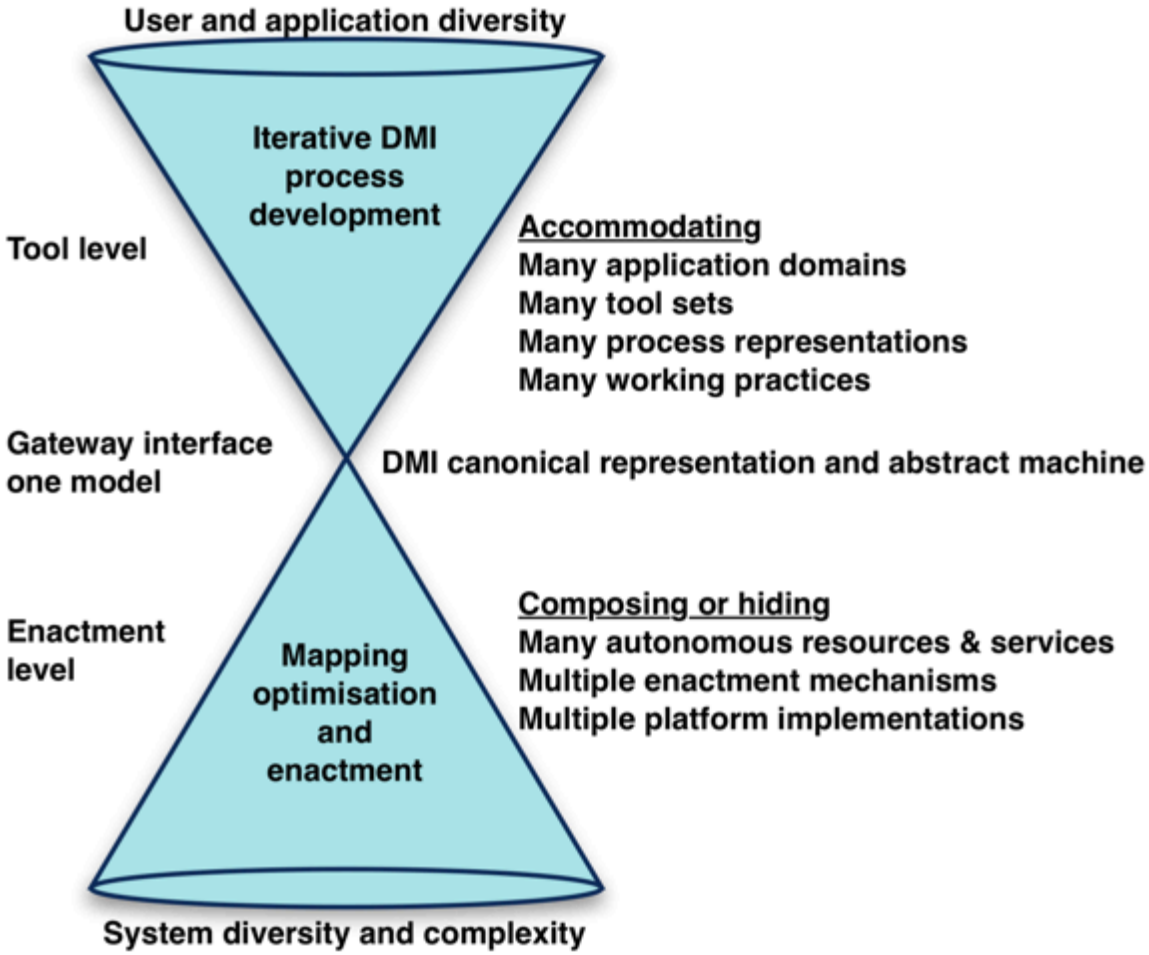
- ADMIRE – Advanced Data Mining and Integration Research for Europe
- 7th Framework Program
- Commenced in February 2008 over 36 months.
- €4.3 million in costs, and €3 million in EC funding

- University of Edinburgh, UK (Coordinator)
  - NeSC - National e-Science Centre
  - EPCC - Edinburgh Parallel Computing Centre
- Fujitsu Labs of Europe, UK
- University of Vienna, Austria
  - Institute of Scientific Computing
- Universidad Politécnica de Madrid, Spain
  - Ontology Engineering Group
- Slovak Academy of Sciences, Slovakia
  - Institute of Informatics
- ComArch S.A., Poland

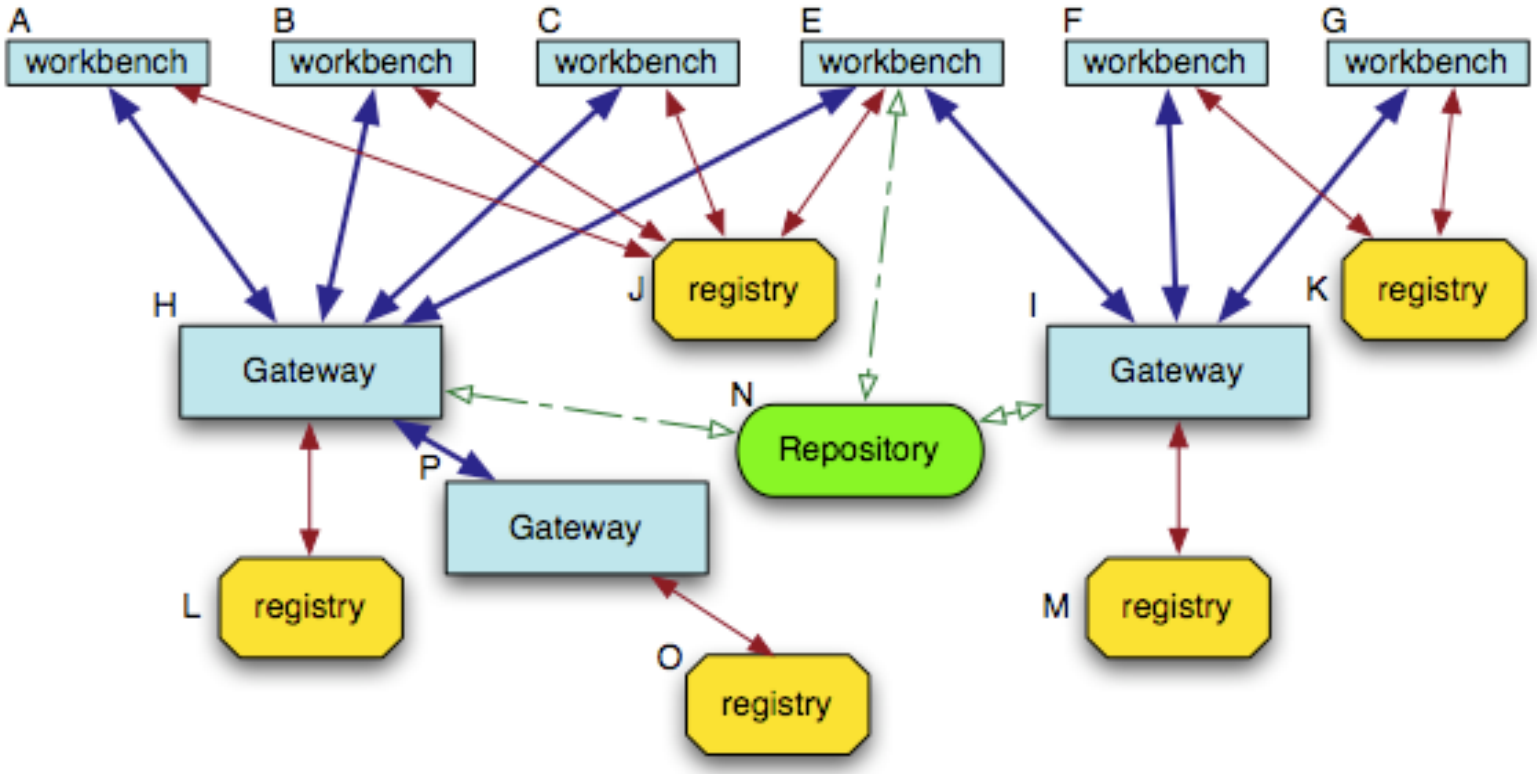
Introduction to ADMIRE - Vision

- Benefits
  - Provide power to users and developers of data mining and integration processes over large scale-out of heterogeneous, distributed data
  - Accelerate access to and increase the benefits from data exploitation
- Strategy
  - recast a selection of existing analysis and mining algorithms to enable them to take best advantage of its distributed-computing framework
  - employ a parallel processing pipeline approach for data-streaming style process to speed up data process enormously

Introduction to ADMIRE - Architecture



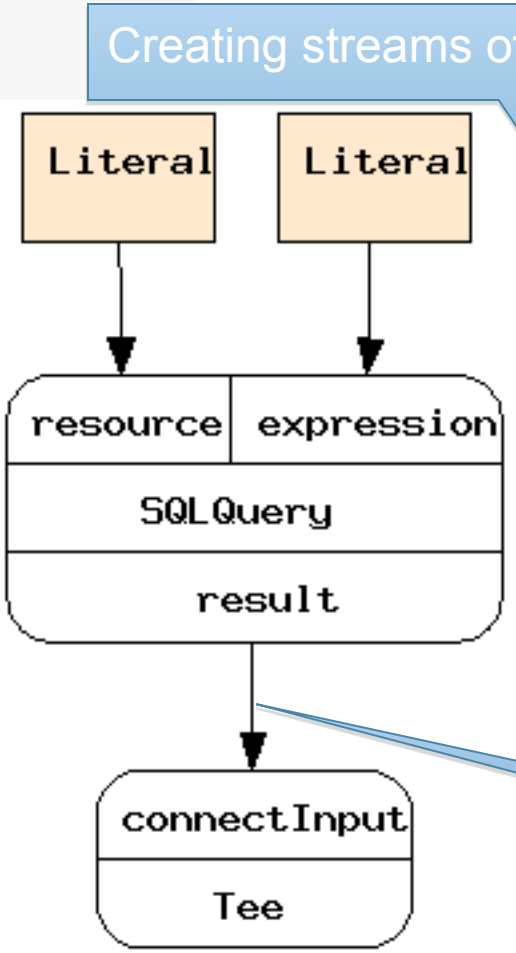
Introduction to ADMIRE - ADMIRE Framework



DISPEL – Data Intensive Systems Process-Engineering Language

- Data-intensive distributed systems
- Connection point of complex application requests and complex enactment systems
  - Benefit: method development, engineering and evolution of supported practices can take place independently in each world
- Describes enactment requests for streaming-data workflows processes
- “Process-engineering time” – transform and optimize process in preparation for enactment period

DISPEL: Simple Example

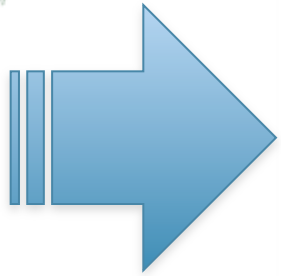


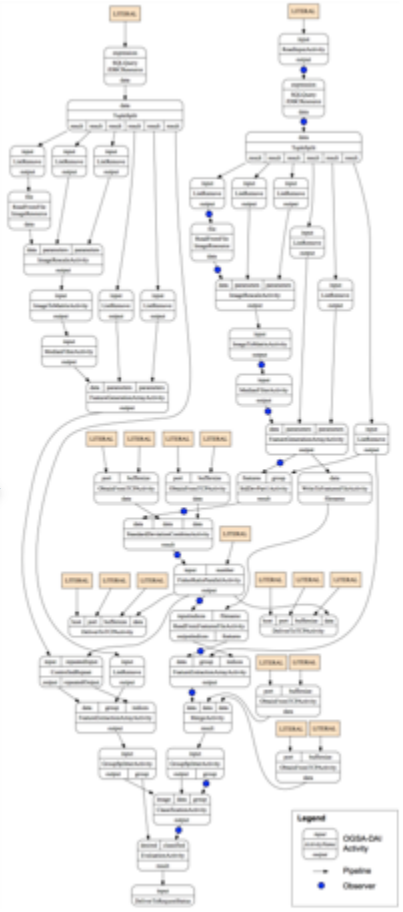

```
String sql1 = "SELECT * FROM some_table";
String sql2 = “SELECT * FROM table2”;
String resource = "128.18.128.255";

SQLQuery query = new SQLQuery;
 |- sql1, sql2 -| => query.expression;
 |- resource -| => query.resource;

Tee tee = new Tee;
query.result => tee.connectInput;
```

Creating connections

DISPEL – real use

```
use uk.org.ogsadai.SQLQuery;
use uk.org.ogsadai.TupleSplit;
use uk.org.ogsadai.ListRemove;
use uk.org.ogsadai.ReadFromFile;
use uk.org.ogsadai.ControlledRepeat;
use uk.org.ogsadai.Tee;
use uk.org.ogsadai.Split;
use uk.org.ogsadai.Echo;
use eu.admire.Results;
use eu.admire.ogsadai.activity.imageprocessing.ImageRescaleActivity;
use eu.admire.ogsadai.activity.imageprocessing.ImageToMatrixActivity;
use eu.admire.ogsadai.activity.imageprocessing.ShowMatrixContentActivity;
use eu.admire.ogsadai.activity.imageprocessing.MedianFilterActivity;
use eu.admire.ogsadai.activity.imageprocessing.FeatureGenerationArrayActivity;
use eu.admire.ogsadai.activity.imageprocessing.StdDevPart1Activity;
use eu.admire.ogsadai.activity.imageprocessing.StandardDeviationCombineActivity;
use eu.admire.ogsadai.activity.imageprocessing.FisherRatioParallelActivity;
use eu.admire.ogsadai.activity.imageprocessing.FeatureExtractionArrayActivity;
use eu.admire.ogsadai.activity.imageprocessing.GroupSplitterActivity;
use eu.admire.ogsadai.activity.imageprocessing.ClassificationActivity;
use eu.admire.ogsadai.activity.imageprocessing.EvaluationActivity;

// GenFeatures takes input data (tuple from SQL query),
// and outputs generated features and group of the images
function GenFeatures(String imageResource)
PE(<Connection data> => <Connection features , group>)
{
    // Split the tuple (filename, width, height, level, dbname, group)
    TupleSplit tsplit = new TupleSplit;

    // Remove list : from [filename] to filename)
    ListRemove nameList = new ListRemove;
    tsplit.result[0] => nameList.input;

    ListRemove widthList = new ListRemove;
    tsplit.result[1] => widthList.input;

    ListRemove heightList = new ListRemove;
    tsplit.result[2] => heightList.input;

    ListRemove levelList = new ListRemove;
    tsplit.result[3] => levelList.input;

    ListRemove dbnameList = new ListRemove;
    tsplit.result[4] => dbnameList.input;

    ListRemove groupList = new ListRemove;
    tsplit.result[5] => groupList.input;

    // Read image from file resource = PE3
    ReadFromFile reader = new ReadFromFile;
    nameList.output => reader.file;
    !- imageResource -! => reader.resource;

    // Rescale the image according to pre-assigned width and height = PE4
    ImageRescaleActivity scale = new ImageRescaleActivity;
    reader.data => scale.data;
    widthList.output => scale.parameters[0]; //width
    heightList.output => scale.parameters[1]; //height

    // Convert scaled imaged from BufferredImage object to matrix (double[])
    ImageToMatrixActivity imgToMatrix = new ImageToMatrixActivity;
    scale.output => imgToMatrix.input;

    // Filter images noise = PE5
    MedianFilterActivity filter = new MedianFilterActivity;
    imgToMatrix.output => filter.input;

    // Generate features according to pre-defined level and dbname = PE6
    FeatureGenerationArrayActivity fg = new FeatureGenerationArrayActivity;
    filter.output => fg.data;
    levelList.output => fg.parameters[0];
    dbnameList.output => fg.parameters[1];

    return PE{<Connection data = tsplit.data> => <Connection features=fg.output ; Connection
group=groupList.output>};
}

GroupSplitterActivity gsplitC = new GroupSplitterActivity;
feC.output => gsplitC.input;
```





Outline

- Introduction to ADMIRE
- **The ADMIRE Registry – What is it?**
- The ADMIRE Registry – How it works?
- The ADMIRE Registry – Who uses it?
- Future work

ADMIRE registry – What is it?

- The ADMIRE registry is in charge of registering Processing Element descriptions
- In the registry the description of a PE contains:
  - A unique name
  - A short definition of what they do
  - A precise description of their input and output streams
    - a structure of Connections
  - The (S&D)type propagation rules from inputs to outputs
  - A precise description of their properties that may permit or limit optimisation
  - Their known subtype hierarchy
  - Information about the PEs (author, organisa[illegible])

Work in progress

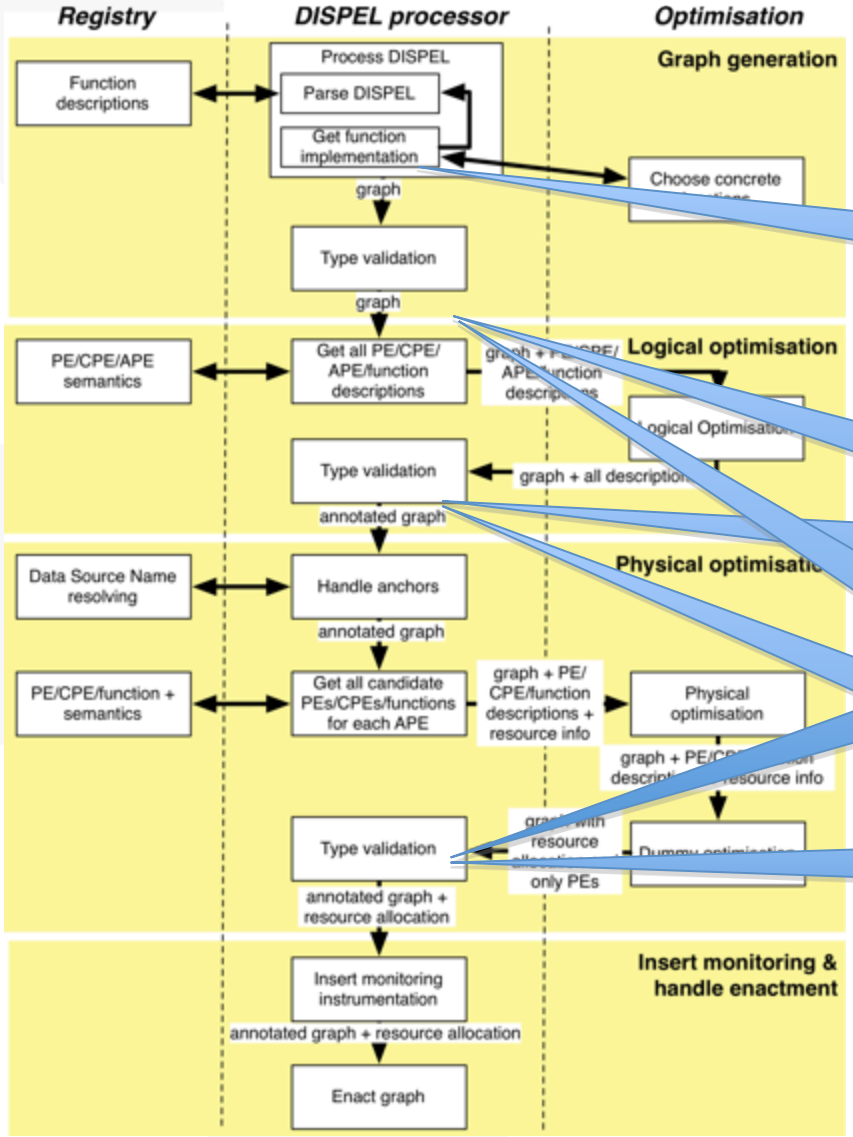
The ADMIRE Registry

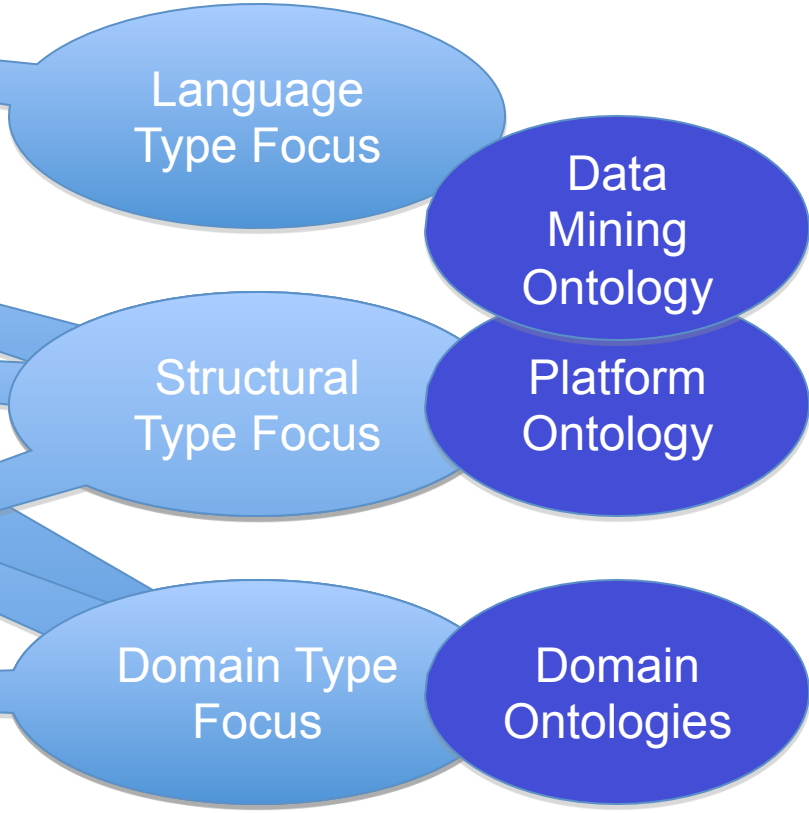
- Descriptions of PEs are stored in RDF (we are talking about data)
- The RDF graph reflects all the PE structure (in DISPEL)
  - Connections
  - STypes
  - DTypes
  - Annotation properties
    - Author
    - Description
    - Organisation
    - Version
  - Some modifiers
  - Propagation Rules are stored implicitly in the data model (see next slide)

The ADMIRE registry - A network of Ontologies

- For describing the data we use a network of ontologies (we are talking about the model)
- Data Mining Ontology (CRISP-DMI and DataMining)
  - It defines the vocabulary related to DMI, allowing end-users to understand the semantics of DMI process components.
- Platform and Operational Ontology (the conceptual level and the instances)
  - They define the terminology needed for describing processing elements of the platform (technology-independent, generic services, datasets, etc.).
- Domain Ontologies
  - They define the vocabulary and relations in the domain use cases

The ADMIRE registry - Language Processing





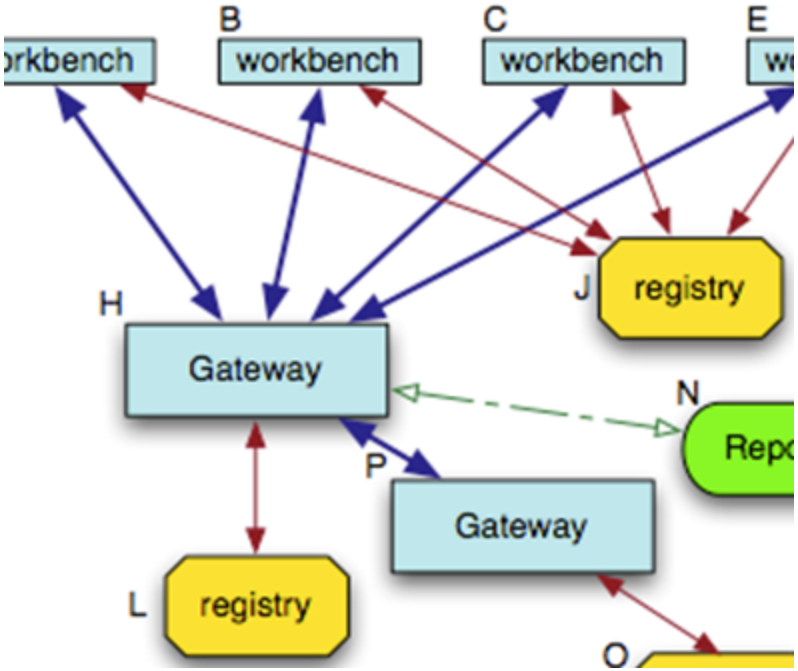
The ADMIRE Registry – An example

- Data
- Ontology

Outline

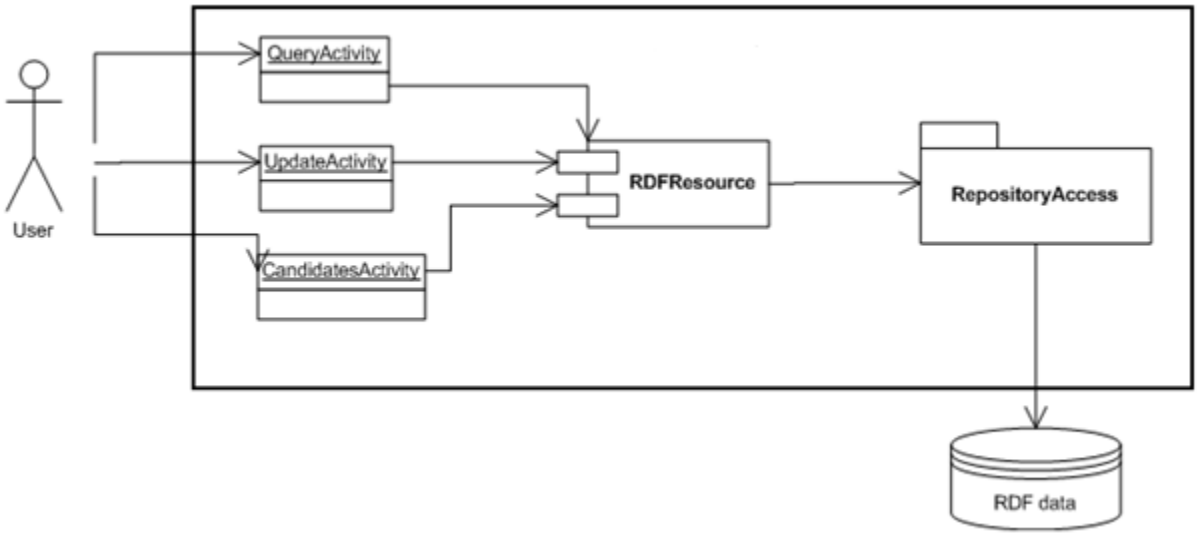
- Introduction to ADMIRE
- The ADMIRE Registry – What is it?
- **The ADMIRE Registry – How it works?**
- The ADMIRE Registry – Who uses it?
- Future work

The ADMIRE Registry



ADMIRE Registry – How it works

- RDFResource: OGSA-DAI data resource with activities for PE query and update




```
ProcessingElementDescriptor lookupProcessingElement(String name);
List<ProcessingElementDescriptor> lookupProcessingElementList();
List<String> lookupProcessingElementsName();
void registerProcessingElement(
        ProcessingElementDescriptor descriptor,
        String implementation)
    throws ProcessingElementAlreadyExistsException,
           RegistrationFailedException;
```

ADMIRE Registry – How it works

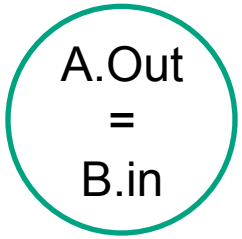
- Interfaces for
  - Registering and querying PEs:
  - Registering and querying STypes and DTypes
  - Registering and querying functions
  - Registering and querying annotations (author, etc.)
  - Registering and querying generic elements
  - Inference over the domain ontologies (Type checking)

ADMIRE Registry – How it works

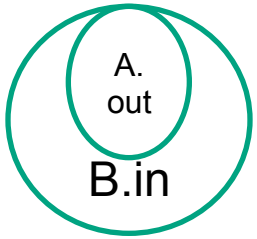
- Basically an OGSA-DAI activity for updating the RDF database
- And a set of SPARQL queries to query the data
- Plus a connection to a Description Logic reasoner for type checking

Type checking cases

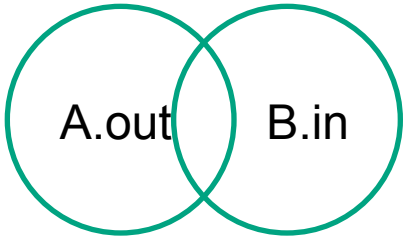
- a) A.out = B.in



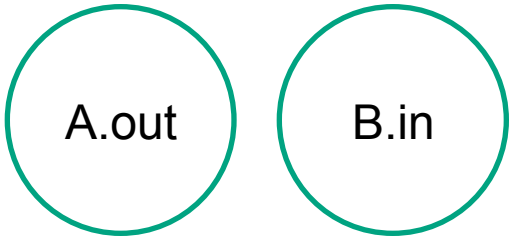
- b) A.out in B.in

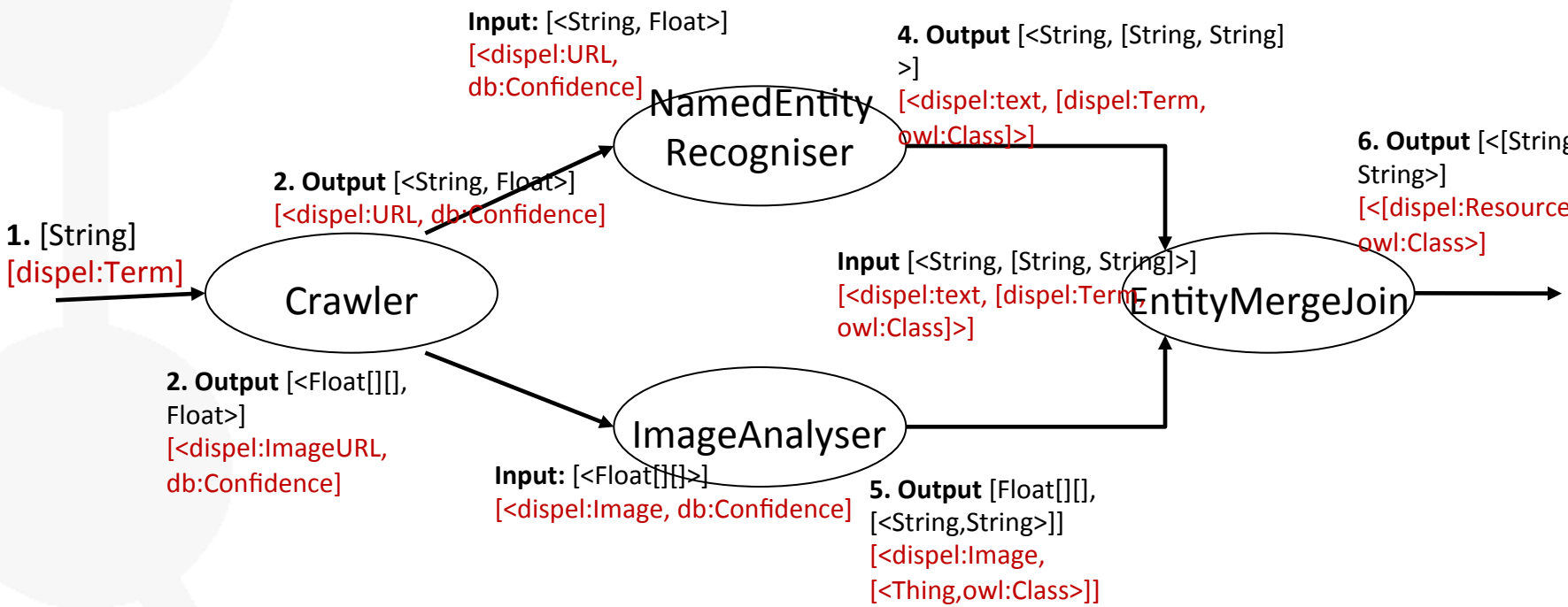


- c) A.out not disjoint with B.in



- d) A.out disjoint with B.in





1. Input: [US, politics, Obama]
2. Output: [<http://data.gob/term1.html, 0,96>, ...]
3. Output: [<http://images.google.com/img1.jpg, 0,97>, ...]
4. Output: [<http://data.gov/obama.html, [<Obama, person>, <white house, Organisation>, ...]>]
5. Output: [<http://images.google.com/obama1.jpg, [<obama, Person>, <WhiteHouse, Organisation>]>]
6. Output: [<http://data.gov/obama, http://images.google.com/obama1.jpg>],[<dbpedia:obama, dbpedia:Person>, <dbpedia:WhiteHouse, dbpedia:Organisation>]

Outline

- Introduction to ADMIRE
- The ADMIRE Registry – What is it?
- The ADMIRE Registry – How it works?
- **The ADMIRE Registry – Who uses it?**
- Future work

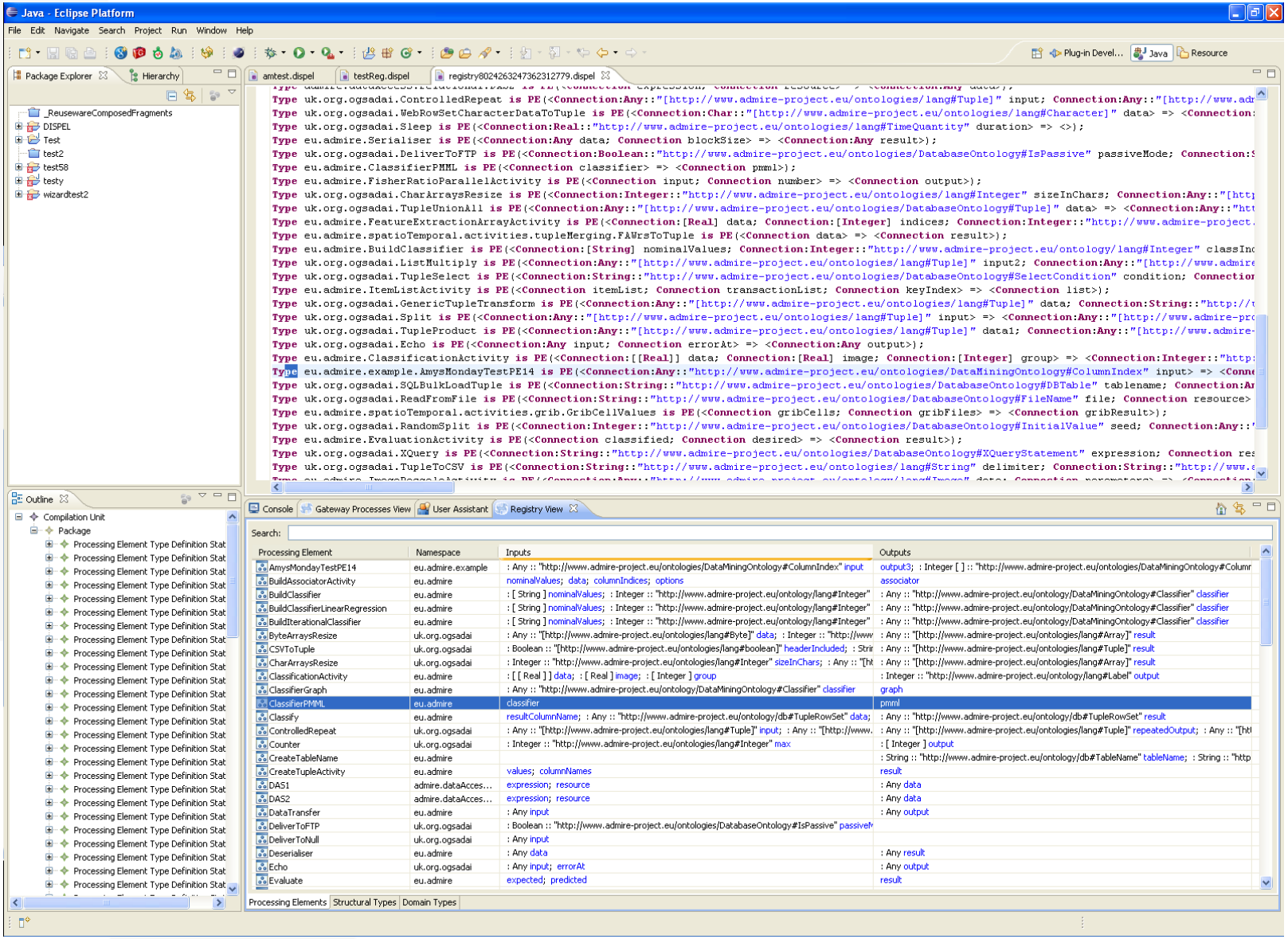
The ADMIRE Registry – Who uses it?

- The ADMIRE gateway
  - The ADMIRE gateway receives the DISPEL code and is in charge of compiling and executing it
  - In compile time the ADMIRE gateway contacts the registry importing all PEs used in that DISPEL code
  - The gateway checks
    - SType compatibility
    - DType compatibility checking and propagation
  - The Gateway submits the DISPEL code

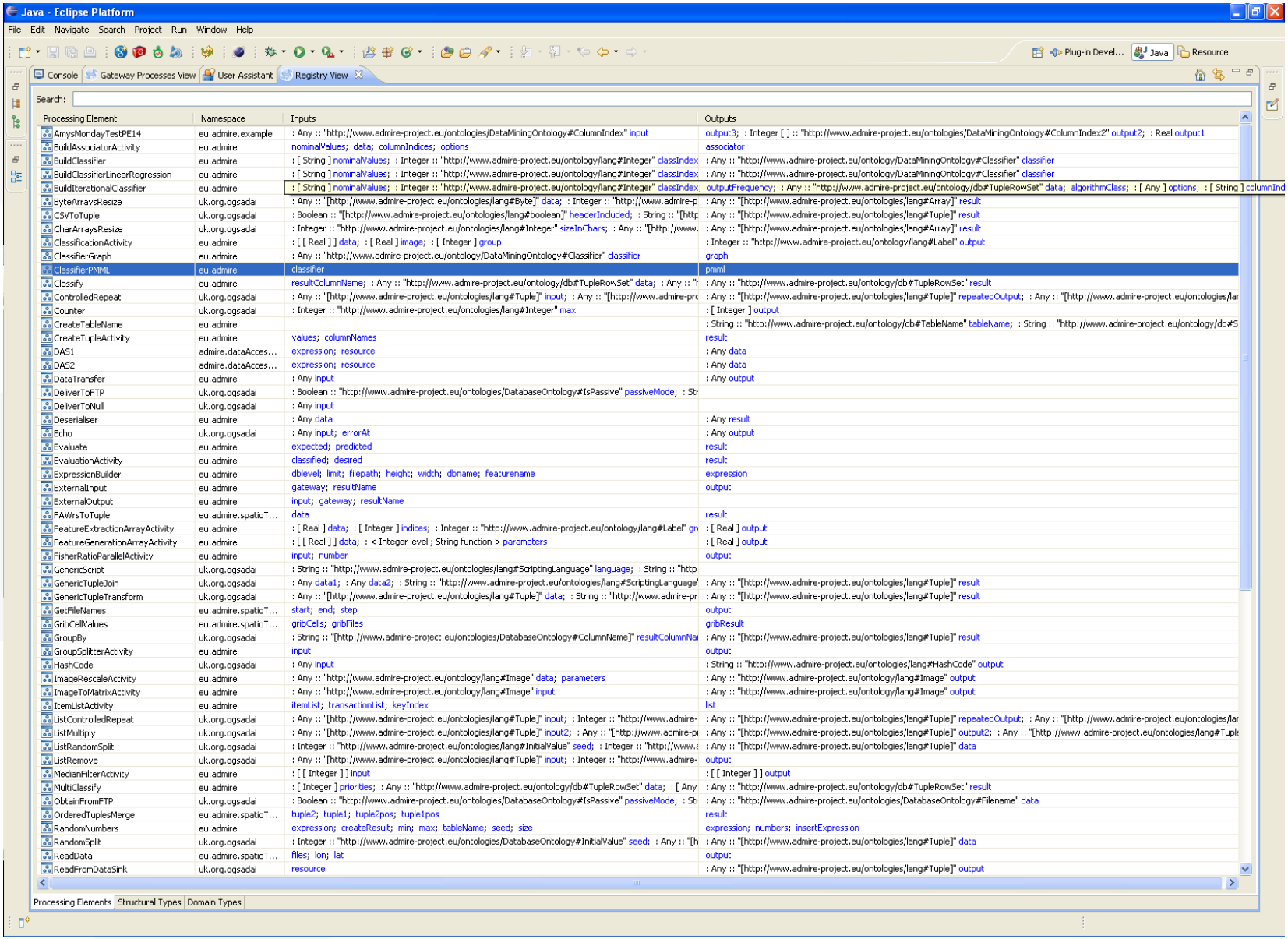
The ADMIRE Registry – Who uses it?

- The process designer
  - It is the user interface of ADMIRE
  - Graphical interface for designing DMI processes
  - Registry view:
    - Lists the PEs available with all their information

The ADMIRE Registry – Who uses it?



The ADMIRE Registry – Who uses it?

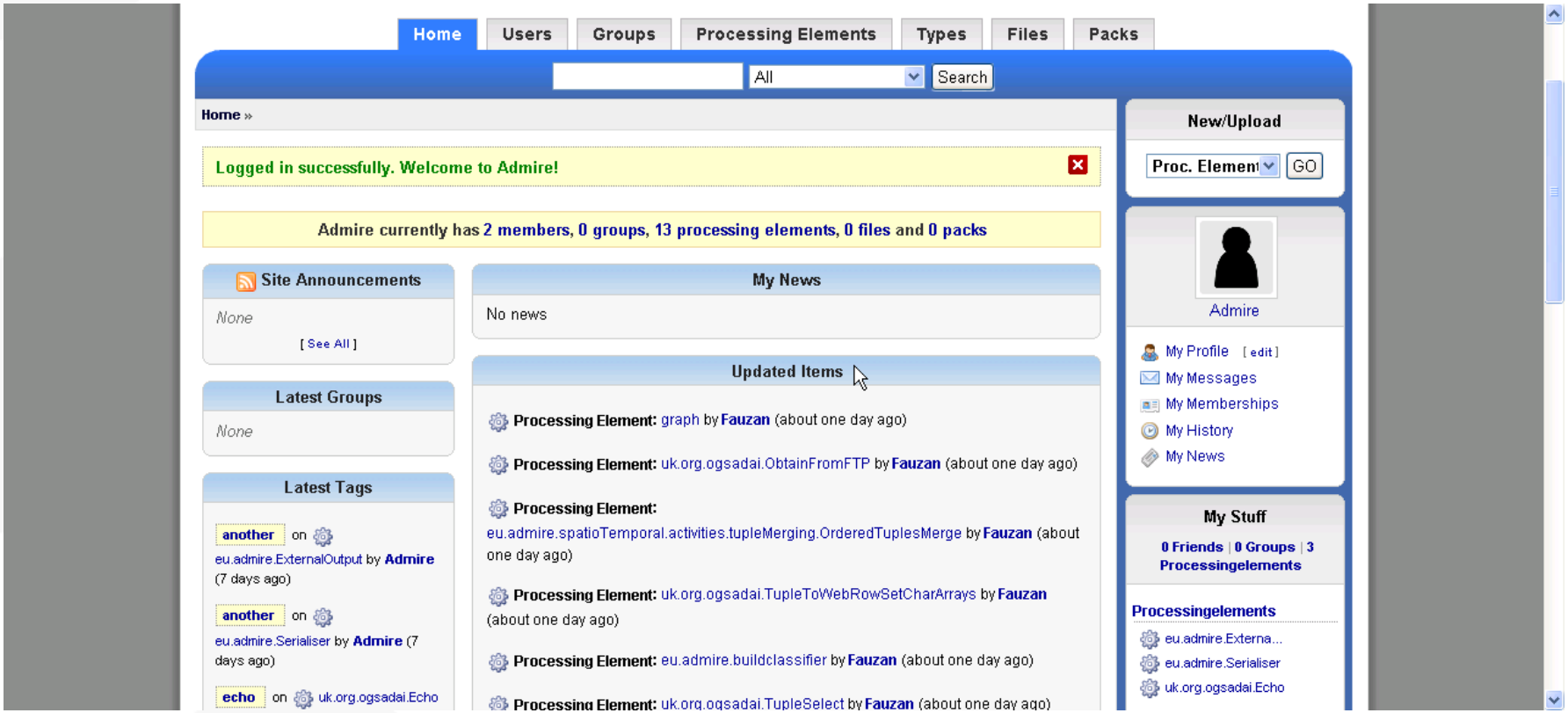


Who uses the registry? - ADMIRE myExperiment

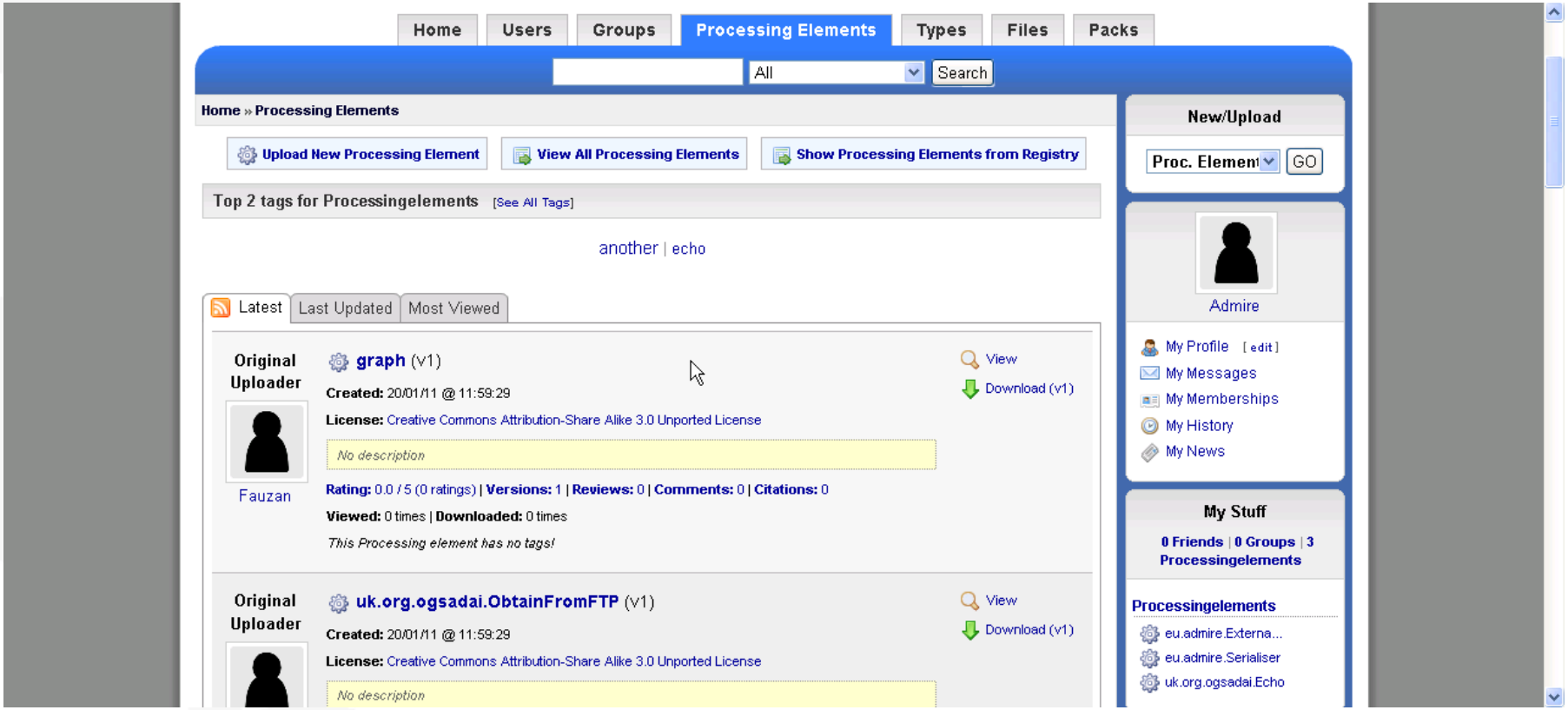
- Collaborative portal for ADMIRE
- Users can create their PEs, store them in the registry and share them within the community
- myExperiment based
- Basically it show all the information in the registry in a web-based portal

Who uses the registry? - ADMIRE myExperiment

Who uses the registry? - ADMIRE myExperiment



Who uses the registry? - ADMIRE myExperiment



Outline

- Introduction to ADMIRE
- The ADMIRE Registry – What is it?
- The ADMIRE Registry – How it works?
- The ADMIRE Registry – Who uses it?
- **Future work**

Future work

- Type checking
  - Currently basic type checking
  - In the life time of ADMIRE basic type checking and propagation
  - Reasoning needs large amounts of computational resources
  - Large amount of domain ontologies to reason with them
  - Compatibilities issues between ontologies, modules, imports of ontologies, etc.
  - All in a distributed fashion
  - Extend the current

Future work

- Distributed registry
  - All sources are distributed
  - Many registries may exist in time and work already done
  - Provide a single point of accessing all the possible registries
- Standards
  - SPARQL 1.1
    - Extend the registry for using SPARQL 1.1 Update and Fed extension
  - WS-DAI-RDF
    - Provide common methods to access the RDF data in the registry

ADMIRE Registry

15th April, 2011, Informatics Forum, Edinburgh

Carlos Buil Aranda, cbuil@fi.upm.es
Facultad de Informática, Universidad Politécnica de Madrid
Campus de Montegancedo sn, 28660 Boadilla del Monte, Madrid
http://www.oeg-upm.net
Phone: +34.91.3366605, Fax: +34.91.3524819