Data-Intensive Research with DISPEL

Oscar Corcho
Ontology Engineering Group
Universidad Politécnica de Madrid

(in collaboration with all the ADMIRE project consortium)
Special thanks to Malcolm Atkinson, from whom most of the slides have been reused
Recognition slide...

• There are many names of many people who have contributed to these slides
  – I am almost just a simple story-teller or work done by others...

• Difficult to provide all names
  – Especially when you finish compiling slides the day before.
  – This slide will be completed for the online version with all names

• For simplicity, thanks to the ADMIRE consortium members
Overview

• Motivational examples
• DISPEL: a language for data-driven research
  – Architecture
  – DISPEL components
    • Processing Elements
    • Types
    • Functions
• DISPEL processing/evaluation
  – The role of the DISPEL gateway
  – The role of the DISPEL registry
• DISPEL resources
Overview

• Motivational examples

• DISPEL: a language for data-driven research
  – Architecture
  – DISPEL components
    • Processing Elements
    • Types
    • Functions

• DISPEL processing/evaluation
  – The role of the DISPEL gateway
  – The role of the DISPEL registry

• DISPEL resources
Motivational Examples

• Astronomy: detection of quasars
• Seismology: ambient noise data processing
• CRM: customer churn and cross-selling
• Genetics: understanding mouse embryos
Motivational Examples

- Astronomy: detection of quasars
- Seismology: ambient noise data processing
- CRM: customer churn and cross-selling
- Genetics: understanding mouse embryos
Quasars

- Quasars are highly energetic cores of galaxies, where matter is falling into black holes, releasing prodigious quantities of energy in the process.
- Star-like in appearance (quasi-stellar radio sources)
- Distinguishing quasars from stars requires information from the distribution of their light across the electromagnetic spectrum.
- Most star-like objects are stars not quasars.
Detection of quasars

Traditional method:
- Spectroscopic
- Expensive
- Slow
- Single object

Alternative method:
- Photometric
- Cheaper
- Quicker
- All objects in area
- Combine multiple bands to approximate spectroscopic study

- Classification using 5 photometric bands has been shown to be good at classifying quasars.
- Research question: Does using 9 photometric bands improve the classification?
The data

- **Sloan Digital Sky Survey (SDSS)**
  - 450m astronomical features
  - 5 optical wavelength bands (u, g, r, i, and z)
  - 120,000 spectroscopically confirmed quasars
- **UKIRT Infrared Deep Sky Survey (UKIDSS)**
  - 60m astronomical features
  - 4 infrared wavelengths (Y, J, H, and K)
  - Link table with distances between objects in SDSS and UKIDSS

![Data table]

<table>
<thead>
<tr>
<th>SDSS id</th>
<th>Is Quasar?</th>
<th>u</th>
<th>g</th>
<th>r</th>
<th>i</th>
<th>z</th>
<th>UKIDSS id</th>
<th>Y</th>
<th>J</th>
<th>H</th>
<th>K</th>
</tr>
</thead>
</table>

29/05/11
Data Integration Workflow

```
SELECT s.id, q.isQuasar, u, g, r, i, z
FROM sdss s LEFT JOIN quasars q ON s.id = q.id
ORDER BY s.id
```

```
SELECT l.sdssId, u.id, Y, J, H, K
FROM ukidss u JOIN linkTable l ON u.id = l.id
ORDER BY l.sdssId
```
Data Integration Workflow

SELECT s.id, q.isQuasar, u, g, r, i, z
FROM sdss s LEFT JOIN quasars q ON s.id = q.id
ORDER BY s.id

SELECT l.sdssId, u.id, Y, J, H, K
FROM ukidss u JOIN linkTable l ON u.id = l.id
ORDER BY l.sdssId
Data Integration Workflow

SELECT s.id, q.isQuasar, u, g, r, i, z
FROM sdss s
LEFT JOIN quasars q ON s.id = q.id
ORDER BY s.id

SELECT lsdssId, u.id, Y, J, H, K
FROM ukidss u
JOIN linkTable l ON u.id = l.id
ORDER BY l.sdssId

SELECT

Ordered Merge Join

Build Classifier

ADQL Query

29/05/11
Data-Intensive Research with DISPEL
Data Integration Workflow

SELECT
  s.id, q.isQuasar, u, g, r, i, z
FROM
  sdss s LEFT JOIN quasars q ON s.id = q.id
ORDER BY s.id

SELECT
  l.sdssId, u.id, Y, J, H, K
FROM
  ukidss u JOIN linkTable l ON u.id = l.id
ORDER BY l.sdssId
SELECT s.id, q.isQuasar, u, g, r, i, z
FROM sdss s LEFT JOIN quasars q ON s.id = q.id
ORDER BY s.id

SELECT l.sdssId, u.id, Y, J, H, K
FROM ukidss u JOIN linkTable l ON u.id = l.id
ORDER BY l.sdssId
Data Integration Workflow

SELECT s.id, q.isQuasar, u, g, r, i, z
FROM sdss s LEFT JOIN quasars q ON s.id = q.id
ORDER BY s.id

SELECT l.sdssId, u.id, Y, J, H, K
FROM ukidss u JOIN linkTable l ON u.id = l.id
ORDER BY l.sdssId

SDSS

ADQL Query

Ordered Merge Join

UKIDSS

Build Classifier

SDSS id Is Quasar? u g r i z UKIDSS id Y J H K
Motivational Examples

- Astronomy: detection of quasars
- Seismology: ambient noise data processing
- CRM: customer churn and cross-selling
- Genetics: understanding mouse embryos
Ambient Noise Data Processing

- Time segmentation
- Filtering & normalization
- Cross-correlation & stacking

Data-Intensive Research with DISPEL
High level workflow
Cross-correlations
Motivational Examples

- Astronomy: detection of quasars
- Seismology: ambient noise data processing
- CRM: customer churn and cross-selling
- Genetics: understanding mouse embryos
• **Business goal**
  – Recognize customers that are probable to quit company services
  – Find out which conditions influence churning

• **Knowledge discovery phases**
  – Model training
    • Designed and executed by data analyst
    • Long-lasting and complicated
  – Model exploitation
    • Executed by domain experts (calling agents)
    • Quick and simple
Architecture

**Model training (Workbench)**

1. Training data extraction
2. Data transformation
3. Model training
4. Model evaluation
5. DeliveryToRepository

**Model exploitation (Portal)**

1. ResultsDeliveryToPortal
2. Classification
3. Test dataset extraction
4. ObtainingFromRepository
CRM Cross-Selling

• **Business goal**
  – To find out hints about additional products or services to be provided to potential customers
  – Market analysis

• **Knowledge discovery**
  – Get frequent itemsets/association/sequential rules from historical data set

\[
\text{Roaming} = \text{TRUE} \land \text{GSM\_Prepaid} = \text{TRUE} \Rightarrow \text{Voice\_mail} = \text{TRUE}
\]
Architecture

Data access → Data Preparation → Association Rule Mining → Rules delivery

<table>
<thead>
<tr>
<th>AGE</th>
<th>LONGEVITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>35</td>
<td>3</td>
</tr>
<tr>
<td>21</td>
<td>38</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AGE</th>
<th>LONGEVITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEDIUM</td>
<td>SHORT</td>
</tr>
<tr>
<td>YOUNG</td>
<td>LONG</td>
</tr>
</tbody>
</table>
Architecture

CRM Database

Data access

Data Preparation

Association Rule Mining

Rules delivery

<table>
<thead>
<tr>
<th>CID</th>
<th>PRODUCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>GSM_Prepaid</td>
</tr>
<tr>
<td>1</td>
<td>Roaming</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CID</th>
<th>GSM_PREPAID</th>
<th>Voice_mail</th>
<th>Roaming</th>
<th>Internet access</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
</tr>
</tbody>
</table>
**Architecture**

<table>
<thead>
<tr>
<th>GSM_Prepaid</th>
<th>Voice_mail</th>
<th>Roaming</th>
<th>Internet access</th>
<th>Age</th>
<th>Longevity</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
<td>YOUNG</td>
<td>LONG</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

GSM_Prepaid = TRUE & Age = YOUNG => Roaming = TRUE
Architecture

CRM Database → Data access → Data Preparation → Association Rule Mining → Rules delivery

GSM_Prepaid = TRUE & Age = YOUNG => Roaming = TRUE

...<_40:AssociationModel minimumConfidence="0.3" minimumSupport="0.01" numberOfItems="6" numberOfItemsets="12" numberOfRules="25" numberOfTransactions="4525">
  <_40:Item id="0" value="GSM_Prepaid=TRUE"/>
  <_40:Item id="1" value="Age=YOUNG"/>
  <_40:Item id="2" value="Roaming=TRUE"/>
...
Motivational Examples

• Astronomy: detection of quasars
• Seismology: ambient noise data processing
• CRM: customer churn and cross-selling
• Genetics: understanding mouse embryos
• Understand the gene function and interactions of genes in a mouse embryo
• Generate a collection of images by employing RNA in-situ hybridisation process
• Identify anatomical components expressing as a gene by annotating the images
The Numbers

- 18,000 genes’ collection for mouse embryo established by RNA \textit{in-situ} hybridisation
- 1,500 anatomical terms ontology used for annotations
- 4 Terabytes of images
  - 80\% manually annotated
  - 20\% remaining (over 85,000 images)
EURExpress-II Workflow
Motivational Examples

What do they all have in common?
The Knowledge Discovery Process

- Business Understanding
- Data Understanding
- Knowledge Delivery
- Data Preparation
- Data Integration
- Modelling
- Evaluation
- Deployment
- Knowledge Delivery

29/05/11
Common characteristics

- Need for a range of data mining and integration functionalities
- Large-scale data
  - Most of traditional/widely-available tools are not enough
  - Need to manage streaming-based models
  - Sometimes high computational demand
- Domain experts become data mining and integration experts, and even distributed computing experts
  - Such specialised human resources are difficult to find
Motivational examples

• What do we need then?
  – An all-in-one framework that combines...
    • Data integration, processing and mining
  – Extensible with domain specific requirements
    • e.g., ADQL queries in Astronomy
  – Support for reusable building blocks
    • e.g., n-fold validation
  – Support for distributed and parallel execution of workflows
  – Native support for a streaming data model
    • e.g., ordered merge joins
  – Automated optimisation
Motivational examples

• What do we need then (cont.)?
  – A framework that separates concerns of
    • Domain Experts
      – They understand the problems of their domain, and the datasets to be used
    • Data-intensive Analysts
      – They understand the knowledge discovery process and know the algorithms and techniques to be used (e.g., association rules, clustering, etc.)
    • Data-intensive Engineers
      – They understand the foundations of distributed computing, and their platforms and technologies (e.g., Grid Computing, Clouds, etc.)
Overview

- Motivational examples
- **DISPEL: a language for data-driven research**
  - Architecture
  - DISPEL components
    - Processing Elements
    - Types
    - Functions
- DISPEL processing/evaluation
  - The role of the DISPEL gateway
  - The role of the DISPEL registry
- DISPEL resources
The DISPEL Hourglass

User and application diversity

Iterative data-intensive process development

Accommodating and facilitating
Many application domains
Many tool sets
Many process representations
Many working practices

Gateway interface
one model

controlled canonical representation

System diversity and complexity

Mapping optimisation, deployment and execution

Composing and providing
Many autonomous resources
Multiple enactment mechanisms
Multiple platform implementations

Enactment level

Tool level

DE1
DE2
DE3
DAE1
DAE2
DIE1
DIE2

GrayWulf

Data-Intensive Research with DISPEL
The DISPEL Hourglass

Data-exploitation develops diversity and supports a wide range of user behaviours, tools and services.
Data-service providers compete for a consolidated load offering generic or specialised services and business models.
The DISPEL Hourglass

The main focus of this talk
DISPEL enables loosely coupling

• Domain Experts

Domain experts do not read and write DISPEL. They discuss parameters and graphs with Data-Intensive Analysts. They work by controlling enactments via their familiar tools: portals, spreadsheets, R, Matlab, ... But there are Domain experts who are also Data-Intensive Analysts! Particularly in research and academic contexts.

• Data-Intensive Analysts

Data-Intensive Analysts read and write DISPEL. They are experts in data mining, text mining, image processing, time series analysis, statistics, etc. They may discuss parameters and graphs with Domain experts. They work in a familiar development environment such as Eclipse. They discuss DISPEL patterns, sentences and performance with Data-Intensive Engineers. They expect robust enactment and effective optimisation.

• Data-Intensive Engineers

Data-Intensive Engineers read and write DISPEL, and build data-intensive platforms. They rarely meet Domain Experts. They are committed to improving all stages of DISPEL processing. They talk with Data-Intensive Analysts to help them do their work and to better understand requirements and workloads.
## Separation of concerns

<table>
<thead>
<tr>
<th></th>
<th>Architectural Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tool</td>
</tr>
<tr>
<td>Domain Experts</td>
<td>Blue</td>
</tr>
<tr>
<td>Data-Analysis Experts</td>
<td>Red</td>
</tr>
<tr>
<td>Data-Intensive Engineers</td>
<td>Yellow</td>
</tr>
</tbody>
</table>

29/05/11 Data-Intensive Research with DISPEL 46
use admire.dataAccess.relational.DAS1; // Get definition of DAS1
use admire.transforms.statistical.Stats; // Get definition of Stats
use admire.dataAccess.relational.DAS2; // Get definition of DAS2

String q1 = “SELECT * FROM db.table1”; // Define literals
String q2 = “SELECT * FROM db.table2”;
String update = “INSERT ? INTO db2.columnStatistics”; 

DAS1 das1 = new DAS1; // Create PEs
Stats stats = new Stats;
DAS2 das2 = new DAS2;

| q1; q2 | => das1.query; // Create graph
das1.result => stats.in;
stats.out => das2.data;
| update; update | => das2.SQL;
Data-Intensive Analysts. Model Deployment

Tariffs → Access → Transform

Communication → Access → Transform → Combine

Customers → Access → Transform

Contacted Customers → Access → Transform → Filter

Filter → Build predictor

Apply predictor → Filter

customers likely to leave

customers not contacted in the last 2 weeks, who have not specified that they do not want to be contacted

Set Difference → Rank → Visualise (customer care)

Visualise (managers)

29/05/11

Data-Intensive Research with DISPEL
Overview

- Motivational examples
- DISPEL: a language for data-driven research
  - Architecture
  - DISPEL components
    - Processing Elements
    - Types
    - Functions
- DISPEL processing/evaluation
  - The role of the DISPEL gateway
  - The role of the DISPEL registry
- DISPEL resources
• User-defined functions
  – encapsulating a data transforming algorithm
• PE descriptions
  – A unique name
  – A short definition of what they do
  – A precise description of their input and output streams
    • a structure of Connections
  – A precise description of their iterative behaviour
  – A precise description of their termination and error reporting
  – 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
• PEs are instantiated before they are used in an enactment
  – new PE_expression
• There may be many instances of a given PE
  – Think PE is a class
  – PEI is an instance of that class
• Assertions may refine the properties of a PE instance
  – new SQLquery with data as :[<Integer i, j; Real r; String s>]

Stating the structural type of this particular instance’s result; the programmer knows the query and schema it will be used with.
Connections

- Connections carry a stream of data
  - from a PE instance to a PE instance
  - 1 source => multiple receivers

- Typically a PE processes one element of the stream at a time

- These elements are as small as makes computational sense
  - a tuple
  - a row of a matrix

- The system is responsible for buffering and optimising their flow
  - pass by reference when possible
  - serialised and compressed for long-haul data movement
  - only buffer to disk when requested or buffer spill unavoidable
Connections

• Two types describe the values passed
  – structural type (Stype)
    • the format / representation of the elemental value
  – domain type (Dtype)
    • the `meaning’ of the elemental value

• Connections may have finite or continuous streams
  – Stream end, EoS, indicates no more data available
    • A PE transmits EoS when it has no more data to send
  – A connection may transmit a “no more” message from receiver to source

• Receiver discard throws away data
  – it sends a “no more” message immediately

• Stream literals have the form
  – | - expression - |
The default termination behaviour occurs when either all the inputs are exhausted or all the receivers of outputs have indicated they do not want more data.

- When all of a PE’s inputs have received EoS:
  - a PE completes the use of its current data
  - then sends an EoS on all of its outputs
  - then stops
- When all of a PE’s outputs have received a “no more”:
  - a PE sends a “no more” on all of its inputs
  - then stops

Termination should propagate across a distributed system:
- there may be a stop operation & external event as well.

This is the default, a PE may stop when a particular stream delivers EoS.

This is the default, a PE may stop when a particular stream receives “no more”.
package eu.admire{

Type ConverterPE is PE(
  <Connection:Any::Thing input> =>
  <Connection:Any::Thing output> );

Type Combiner is PE(
  <Connection[ ] inputs> =>
  <Connection output> );

Type ErrorStream is Connection:
  < error: String::"lang:ErrorMessage"; culprit >;

Type ProgrammableCombiner is PE(
  <Connection[] inputs;
   Connection:String::"lang:JavascriptCode” controlExpression> =>
  <Connection output;
   ErrorStream errors > );

register ConverterPE, Combiner, ErrorStream, ProgrammableCombiner;}

Language types
PEs as subtypes of PEs

```java
package eu.admire{
  use eu.admire.Combiner;
  use eu.admire.ProgrammableCombiner;

  Type SymmetricCombiner is Combiner with inputs permutable;
  Type SymmetricProgrammableCombiner is ProgrammableCombiner with inputs permutable;

  register SymmetricCombiner, SymmetricGenericCombiner;}
```

Indicate order of inputs is not significant

Make these new PE types subtypes of the previously declared types.
Overview

• Motivational examples

• DISPEL: a language for data-driven research
  – Architecture
  – DISPEL components
    • Processing Elements
    • Types
    • Functions

• DISPEL processing/evaluation
  – The role of the DISPEL gateway
  – The role of the DISPEL registry

• DISPEL resources
import coral.reef.ecology

Stype Image is Pixel[ ][ ];
Stype Camera is <Integer cNumber; String cType; Real x, y, z, theta, phi, alpha, aperture, magnification, frequency; Real[ ] settings; rest >;
Stype Illuminator is <Integer iNumber; String iType; Real x, y, z, theta, phi, beamWidth, intensity, iFrequency, duration; Real[ ] iSettings; rest >;
Stype Frame is <Time t; Image[ ] images; Camera[ ] cameras;
          Illuminator[ ] lighting>;
register Image, Camera, Illuminator, Frame;
Stypes and Dtype

```plaintext
package coral.reef.ecology{
    namespace cre "Coral_Reef_Ecology.observngStation.terms:"
    use coral.reef.ecology.Frame;
    Stype Object is <Real x, y, z, radius>;
    Stype ObjectMap is <Frame:: cre:Primary_PIV_data frame;
                           Object[ ]:: cre:Putative_Individuals objects>;

    Type ObjectRecogniser is PE (  
        <Connection: Frame:: cre:PrimaryPIVdata frames> =>  
        <Connection: ObjectMap:: cre:First_Reconstruction putativeIndividuals>  
    );
    register Object, ObjectMap, ObjectRecogniser;}
```
package coral.reef.ecology{
  namespace cre "Coral_Reef_Ecology.observingStation.terms:"
  use coral.reef.ecology.ObjectMap;

  Stype Individual is <Object:: cre:Individual_Subject confirmIndividual;
  Integer:: cre:Unique_Arbitrary_Tag idNumber;
  String:: cre:Species_Or_Inert taxa; Boolean swimmer;
  Real:: cre:Mass_Estimate1_Kilograms mass>;

  Stype IndividualMap is <Frame:: cre:Primary_PIV_data frame;
  Individual[ ]:: cre:Confirmed_Individuals individuals>;

  Type IndividualRecogniser is PE ( 
  <Connection: ObjectMap:: cre: First_Reconstruction putativeIndividuals
  > => 
  <Connection: IndividualMap:: cre:Second_Reconstruction taggedIndividuals
  > 
  );

  register Individual, IndividualMap, IndividualRecogniser;}

package coral.reef.ecology{
    namespace cre “Coral_Reef_Ecology.observingStation.terms:”;
    use coral.reef.ecology.IndividualMap;
    Stype MovingIndividual is <Individual:: cre:Tagged_Subject confirmIndividual;
        Real[]:: cre:Meters_Per_Second[] velocity;
        Real:: cre:Joules kineticEnergy>;
    Stype MovementMap is <Frame:: cre:Primary_PIV_data frame;
        MovingIndividual[]:: cre:Moving_Individuals individuals>;

    Type MovementRecogniser is PE (  
        <Connection: IndividualMap:: cre:Second_Reconstruction taggedIndividuals  
            > =>
        <Connection: MovementMap:: cre:Third_Reconstruction movingIndividuals  
            >  
    );
    register MovingIndividual, MovementMap, MovementRecogniser;}

29/05/11 Data-Intensive Research with DISPEL 64
Overview

• Motivational examples
• DISPEL: a language for data-driven research
  – Architecture
  – DISPEL components
    • Processing Elements
    • Types
    • Functions
• DISPEL processing/evaluation
  – The role of the DISPEL gateway
  – The role of the DISPEL registry
• DISPEL resources
Coupling portlet to DISPEL function

Step 1: user inputs solicited values
Step 2: portal system validates values
Step 3: portal system constructs DISPEL sentence with these values as parameters of a provided function
Step 4: portal system sends DISPEL sentence to gateway
Steps 5 to n: gateway and systems behind it validate and enact the sentence
Step n+1: gateway sends summary/partial results to progress viewer
Step n+m: gateway sends final (summary) results to result viewer
package eu.admire.seismology{
  use eu.admire.seismology.proj1portlet4invokeCorrelations;
  use eu.admire.Time;
  Time startTime = <year=1996, day = 53, seconds = 0>;
  Time endTime = <year=1996, day = 54, seconds = 0>;
  Real minFreq = 0.01; //frequencies considered in Hertz
  Real maxFreq = 1.0;
  Real maxOffset = 10*60*60;

  proj1portlet4invokeCorrelations(
    startTime, endTime, minFreq, maxFreq, maxOffset );
}

Values solicited from user and inserted into a minimal template; some of these could equally well be `constants’ embedded as hidden defaults in the template
Overview

• Motivational examples

• DISPEL: a language for data-driven research
  – Architecture
  – DISPEL components
    • Processing Elements
    • Types
    • Functions

• DISPEL processing/evaluation
  – The role of the DISPEL gateway
  – The role of the DISPEL registry

• DISPEL resources
• A DISPEL sentence is prepared...
  • Sent to a gateway...
    – which may inspect it and the sender’s credentials
    – and then accept it and initiate enactment
• Enacted in four phases
  – DISPEL language processing
    • to produce a graph and/or register definitions
  – Optimisation
  – Deployment
    • across hosting platforms
  – Execution and control
    • including termination and tidying
Architecture (e.g., ACRM)

**Workbench**

**Gateways**
- GW 1
- GW 2
- GW 3

**Repository**

**Submission of the DISPEL document (model training process)**

**Delivery of training results**

**Obtaining training results**

**Submission of the DISPEL document (model exploitation)**

**Telco domain expert**

**ACRM Portal**

**Data mining expert**

**Submission of the DISPEL document**

**Delivery of training results/Obtaining training results**

**Obtaining training results by domain expert**

29/05/11
Overview

- Motivational examples
- DISPEL: a language for data-driven research
  - Architecture
  - DISPEL components
    - Processing Elements
    - Types
    - Functions
- DISPEL processing/evaluation
  - The role of the DISPEL gateway
  - The role of the DISPEL registry
- DISPEL resources
• “Making the hourglass bottleneck narrower”
• Allowing DISPEL code to be “smaller”, while still generating large graphs
  – By means of describing patterns
    • Functions and composite PEs
  – ...with rich semantics
    • Core ontology for these descriptions
    • Domain-specific ontologies can be incorporated
  – ...plus human-focused descriptions (since domain experts must understand them)
    • Dublin core properties, social discussion, etc.
Semantics in DISPEL processing
Registry Contents

• Initial library of 81 domain-dependent and independent Processing Elements
  – DISPEL PE library initialisation file
    • Many PEs inherited from OGSA-DAI activities
    • Incorporating PEs from ADMIRE use cases

• Open distributed registration of new domain-specific (or generic) PEs to start soon
  – Sorting out the social networking part
Advanced Functionalities

- Find candidate PEs and functions
  - Find “similar” PEs
    - Queries for sibling PEs in the ontology hierarchy
  - Find “compatible” and “functionally-equivalent” PEs and functions if explicitly-defined in the ontology
  - Infer “compatible” PEs and functions
Reusing the myExperiment frontend

Show Processing Element from Registry

Enter URI to retrieve from Registry
http://138.100.11.152:8081/dai/services/ Retrieve and Show

This button will perform an massive upload
Upload All Processing Elements
### Example of a PE list

<table>
<thead>
<tr>
<th>Original Uploader</th>
<th>PE Name</th>
<th>Created</th>
<th>Credits</th>
<th>License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admire</td>
<td>classifier (v1)</td>
<td>24/01/11 @ 15:40:50</td>
<td>Admire</td>
<td>Creative Commons Attribution-Share Alike 3.0 Unported License</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>No description</td>
</tr>
<tr>
<td></td>
<td>Rating: 0.0 / 5 (0 ratings)</td>
<td></td>
<td></td>
<td>Versions: 1</td>
</tr>
<tr>
<td></td>
<td>Viewed: 0 times</td>
<td></td>
<td></td>
<td>Downloaded: 0 times</td>
</tr>
<tr>
<td></td>
<td>This Processing element has no tags!</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>eu.admire.StoreAndRegister (v1)</td>
<td></td>
<td></td>
<td>Creative Commons Attribution-Share Alike 3.0 Unported License</td>
</tr>
<tr>
<td></td>
<td></td>
<td>24/01/11 @ 15:40:49</td>
<td>Admire</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>No description</td>
</tr>
<tr>
<td></td>
<td>Rating: 0.0 / 5 (0 ratings)</td>
<td></td>
<td></td>
<td>Versions: 1</td>
</tr>
<tr>
<td></td>
<td>Viewed: 0 times</td>
<td></td>
<td></td>
<td>Downloaded: 0 times</td>
</tr>
<tr>
<td></td>
<td>This Processing element has no tags!</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>uk.org.ogsadai.ListRandomSplit (v1)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example of a structural type

Title: uk.org.ogsadai.SQLQuery
Type: Registry
Registry URI: http://138.100.11.152:8081/dai/service
Overview

• Motivational examples
• DISPEL: a language for data-driven research
  – Architecture
  – DISPEL components
    • Processing Elements
    • Types
    • Functions
• DISPEL processing/evaluation
  – The role of the DISPEL gateway
  – The role of the DISPEL registry
• DISPEL resources
Download instructions

AdmireVM can be downloaded from here (when you unzip it it will take around 6 GB):

admire3.epcc.ed.ac.uk/AdmireVM.vmwarevm.zip

It can be run using VMware Workstation or free VMware Player which can be downloaded from here:

🌐 http://downloads.vmware.com/d/info/desktop_downloads/vmware_player/3_0

Content

The image is based on Ubuntu 11.4. It contains the following Admire components:

- Admire Gateway
- Admire Execution Engine (OGSA-DAI)
- Admire Registry
- Admire Repository
- Admire Workbench
- MySQL with sample data
- Some DISPEL documents

Instructions

Username: admire; Password: admire
DISPEL resources

Hello! Welcome to ADMIRE Gateway.

What do you want to do today?
- **List DISPEL processes** - List all current Gateway processes.
- **Submit DISPEL** - Submit your DISPEL request to the Gateway.
- **Validate DISPEL** - Validate your DISPEL request and see the graphical representation.
- **View version** - View the version of the Gateway.
- **List local resources** - List local resources of the Gateway.
- **List all resources** - List all resources known to the Gateway. Some might be on remote gateways.
- **Visit** - Visit the ADMIRE Home Page.
Data-Intensive Research with DISPEL

Oscar Corcho
Ontology Engineering Group
Universidad Politécnica de Madrid

(in collaboration with all the ADMIRE project consortium)
Special thanks to Malcolm Atkinson, from whom most of the slides have been reused