



# Enabling the execution of various workflows (Kepler, Taverna, Triana, P-GRADE) on EGEE

Tamas Kukla, Tamas Kiss, Gabor Terstyanszky

Centre for Parallel computing

University of Westminster

London

Peter Kacsuk, Gergely Sipos

Computer and Automation Research Institute

Hungarian Academy of Sciences

Budapest





# Contents

- Introduction
- Workflow interoperability
- Requirements of workflow engine integration
- Realising workflow integration
- Conclusions





# Introduction

- Several widely utilised, Grid workflow management systems, such as **Triana**, **P-GRADE**, **Taverna**, **Kepler**, **CppWfMS**, **YAWL**, or the **K-Wf Grid** emerged in the last decade.
- These systems were developed by **different** scientific **communities** for various **purposes**.
- Therefore, they differ in several aspects. They use
  - different workflow **engines**
  - different workflow description **languages**
  - different workflow **formalisms**
  - different Grid **middleware**





# Different workflow engines

- Most systems are coupled with one engine:
  - Taverna uses **Freeflu**
  - Triana uses **Triana engine**
  - K-WfGrid uses **GWES** (Grid workflow execution service)
  - Older versions of P-GRADE used **Condor DAGMan**, while its recent version uses its own engine **Xen**.





# Different workflow description languages

- Most workflow systems use different workflow description languages:
  - Triana interprets **BPEL** (Business Process Execution Language) and **its own language format**.
  - Taverna workflows are represented in **SCUFL**.
  - Older versions of P-GRADE used **Condor DAG**, now it uses **its own defined language**.
  - Kepler uses **MOML**.
  - YAWL system uses **YAWL language**.
  - K-WfGrid uses **GWorkflowDL**.
- Because of this diversity, **workflows** of a system **cannot be reused** in another system.





# Different workflow formalisms

- Workflow description languages are based on various workflow formalisms.
  - Condor DAG uses directed acyclic graphs (**DAG**).
  - SCUFL is also **DAG based**, but it is extended **with control constraints**.
  - The new workflow language of P-GRADE is also **DAG based**, but it is extended **with recursion** and **nesting**.
  - YAWL and GWorkflowDL are based on **Petri Nets**
  - BPEL is **Pi-Calculus** based
- Different formalisms have **different expression capabilities**.
- Therefore, in many cases it is **not possible to express** a workflow of **one type** in the description language of **another**.





# Workflow interoperability

- In order to achieve cross-organisational **collaboration** between the different scientific communities, workflows should be able to **interoperate**, **communicate** with and/or **invoke** each other during execution.
- The WfMC defines workflow interoperability in general as:
  - "The ability of two or more Workflow Engines to communicate and work together to coordinate work."

In this definition the workflow engine is a piece of software that provides the workflow run-time environment.





# Approaches to workflow interoperability

- Various solutions can bring workflow interoperability into effect:
  - Workflow description **standardisation**
    - Would enable the **exchange of workflows** of different systems
    - **XPDL** was defined by the WfMC and **BPEL** was defined by Microsoft and IBM for this purpose, but they did not gain universal acceptance so far.
    - It is **unlikely** in the near future
  - Workflow **translation**
    - Would enable the translation from one language to another
    - Can be realised by **translating via an intermediate** workflow **language**.
      - **YAWL** and **GWorkflowDL** could also be used for this purpose. See BPEL to YAWL translator or SCUFL to GWorkflowDL converter.
    - **Cannot be applied in any case**





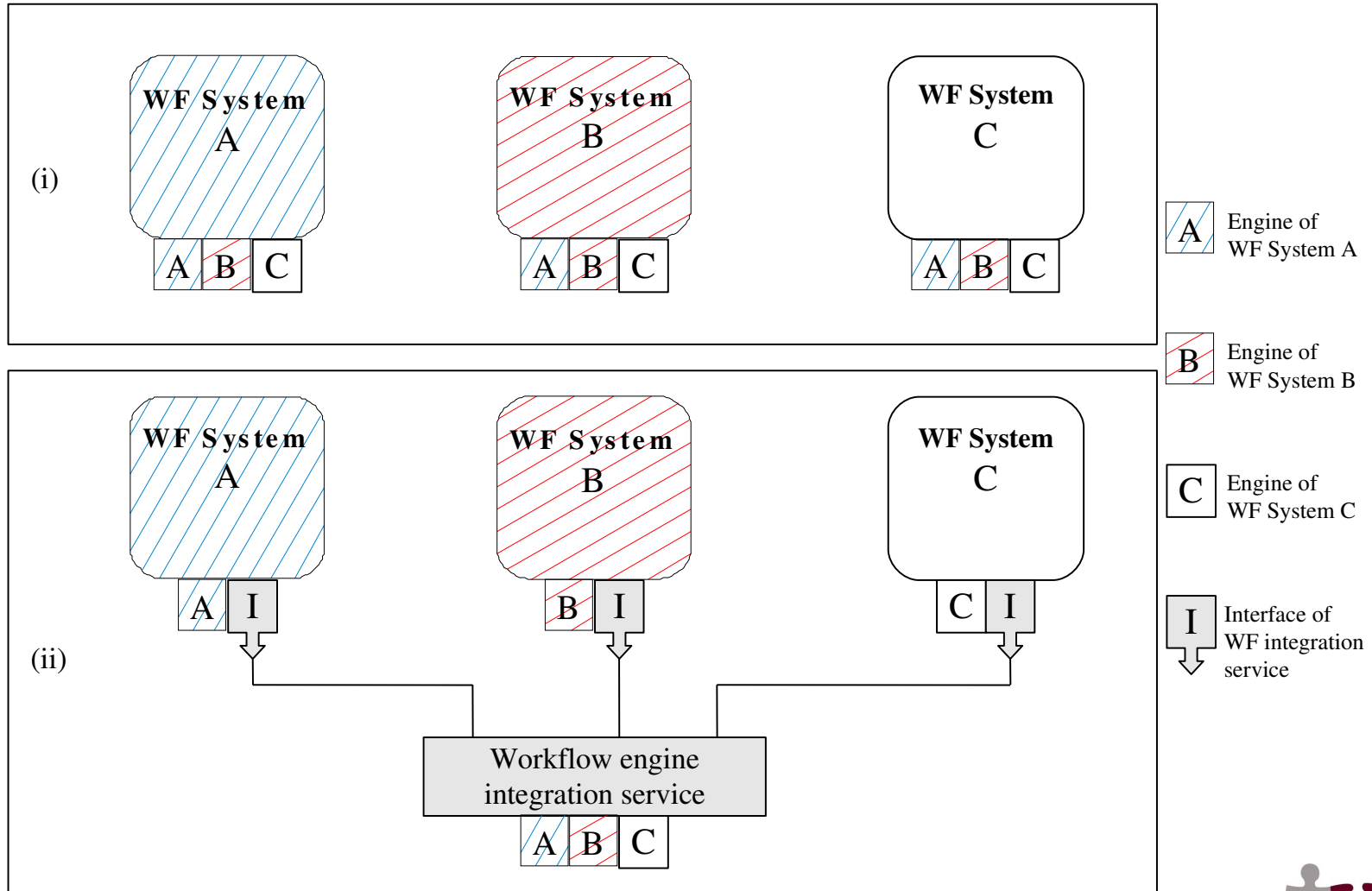


# Workflow engine integration

- An alternative approach to attain workflow interoperability could be realised by workflow engine integration.
- Executes the workflow in its **native environment** in **by its own** workflow **engine**.
- Makes workflow management **systems** to be able to **execute non-native workflows**.
- Can be realised by **loosely** or **tightly coupled integration**.



# Tightly(i) and loosely(ii) coupled engine integration





# Requirements of workflow engine integration

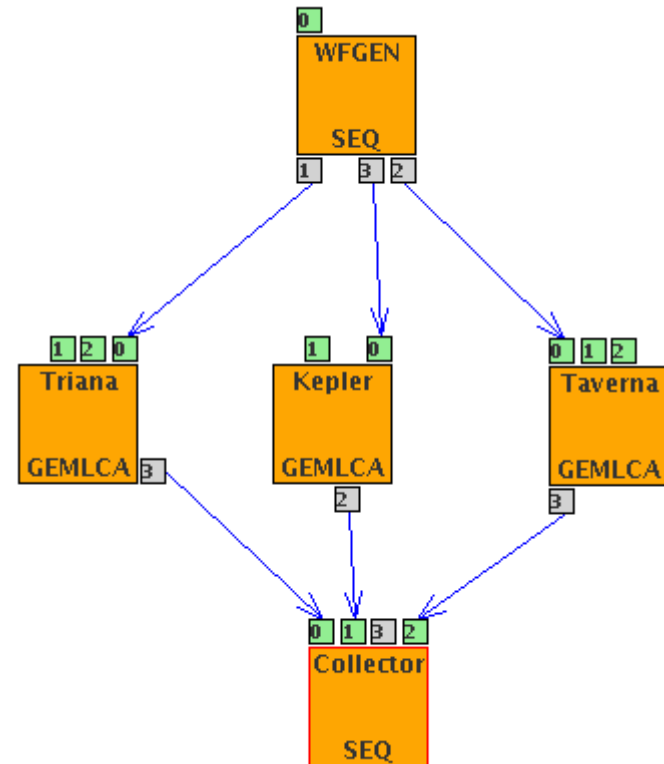
- Our aim is to **provide a solution** for workflow **sharing and interoperability** by integrating different workflow systems in the following fashion:
  - providing a **generic solution**, which can be adopted to any workflow system
  - providing a **scalable solution** in terms of both number of workflows and amount of data
  - **integration of a new workflow engine** to the system should not require code re-engineering, **only user level understanding** of the engine in question





# The concept of the target heterogeneous WF system

- In a certain type of workflow (meta workflow) other types of workflows can be executed as nodes
- The goal is that the meta workflow could be any type of WF (Taverna, Triana, Kepler, etc.)
- The embedded workflows can also be any kind of WFs





# Realising workflow integration

- To provide a generic solution:
  - It is recommended to realise **loosely coupled integration**
- To provide a scalable solution:
  - It is recommended to **utilize Grid resources** for workflow engine execution
- To make the workflow engine deployment straightforward:
  - It is recommended to **handle workflow engines as legacy applications**





# GEMMLCA

- GEMMLCA, that is unique in a sense that it is an **application repository extended with a job submitter**, allows the deployment of legacy code applications on the Grid.
- An **application can be exposed via a GEMMLCA** service and can be executed using a GEMMLCA client.
- The legacy application is stored either in the **repository of a GEMMLCA service** or on a **third party computational node** where GEMMLCA can access it.
- To **publish a legacy application** via GEMMLCA, only a **basic user-level understanding** of the legacy application **is needed**, code re-engineering is not required.
- As soon as the application is deployed, GEMMLCA is able to **submit** it using either **GT2, GT4** or **gLite** Grid middleware.
- If the workflow engine requires credentials to utilise further Grid resources for workflow execution, these are **automatically** provided by GEMMLCA through **proxy delegation**.





# Exposing workflow engines via GEMMLCA

- **Command-line workflow engines**, just like other legacy applications, **can be exposed via a GEMMLCA service**, without code re-engineering and can be automatically submitted by GEMMLCA to the Grid to a computational node.
- **Three engines** (engine of Taverna, Triana, and Kepler) have been en-wrapped by scripts so as to provide a **general command line interface** for them. This interface is the following:

```
wfsubmit.sh -w wf_descriptor
             [-p wf_input_params]
             [-i wf_input_files]
             [-o wf_output_files]
```

- Wrapper scripts are responsible for **installing** the workflow engine, **decompressing** the workflow **input files**, execute the workflow by **parametrizing and invoking** the workflow **engine** and finally **compress** the workflow **outputs** into one archive file.
- The engines were exposed using the JSR-168 based **GEMMLCA administrator portlet**.





# Exposing Taverna workflow engine on EGEE using GEMMLCA Administration Portlet

<b>DESCRIPTION</b>		Engine			
<b>PARAMETER</b>	<input type="button" value="Hide parameter"/> regExp <input type="text"/> <b>COMMANDLINE</b> <input checked="" type="checkbox"/> <b>NAME</b> <input type="text" value="-w"/> <b>VALUE</b> <input type="text" value="test.xml"/> <b>FRIENDLYNAME</b> <input type="text" value="Workflow"/> <b>MANDATORY</b> <input type="checkbox"/> <b>FIXED</b> <input type="checkbox"/> <b>FILE</b> <input type="checkbox"/> Switch <input type="checkbox"/> Non-File <b>INPUT</b> <input type="checkbox"/> Switch <input type="checkbox"/> Input <input type="button" value="Show parameter"/> <input type="button" value="Show parameter"/> <input type="button" value="Show parameter"/> <b>ORDER=</b> <input type="text" value="0"/> <input type="button" value="Add"/>	order=0 <input type="button" value="Remove"/>			
		order=1 <input type="button" value="Remove"/>			
		order=2 <input type="button" value="Remove"/>			
		order=3 <input type="button" value="Remove"/>			
		ORDER=0 <input type="text"/>	<input type="button" value="Add"/>		
		<b>AUTHORIZATION INFO</b>		<input type="button" value="Show authorizationInfo"/>	
		<b>ID</b>	Taverna-1.7-WF		
		<b>status</b>	<input type="button" value="Publish"/>		
		<b>BACKEND SPECIFIC DATA</b>		<input type="button" value="Hide backendSpecificData"/> <b>COUNT</b> <input type="text" value="1"/> <b>OUTPUT</b> <input type="text" value="STDOUT"/> <b>ERROR</b> <input type="text" value="STDERR"/> <b>JOBTYPE</b> <input type="text" value="single"/> <b>maxWallTime</b> <input type="text" value="10"/> backendId=GildaBroker <input type="button" value="Remove"/> <b>BACKENDID=</b> <input type="text" value=""/> <input type="button" value="Add"/> <b>MAXPARALLELISM</b> <input type="text" value="10"/>	
		<b>SITEINFO</b>		<input type="button" value="Hide siteInfo"/> <b>JOBMANAGER</b> <input type="text" value="FORK"/> site <input type="text" value="SITE="/> <input type="button" value="Add"/> <b>EXECUTABLE</b> Stage: <input type="text" value="/home/kukla/taverna-1.7.0/v"/> <b>PARAMPREFIX</b> <input type="text" value=""/> <b>ID=</b> <input type="text" value="0"/> <input type="button" value="Add"/>	







# Exposing Taverna workflow engine on EGEE using GEMMLCA Administration Portlet

<b>DESCRIPTION</b>		Engine	
<b>PARAMETER</b>	Hide parameter		
	regExp	<input type="text"/>	
	<b>COMMANDLINE</b>	<input checked="" type="checkbox"/>	
	<b>NAME</b>	-w	
	<b>VALUE</b>	test.xml	
	<b>FRIENDLYNAME</b>	Workflow	
	<b>MANDATORY</b>	<input type="checkbox"/>	
	<b>FIXED</b>	<input type="checkbox"/>	
	<b>FILE</b>	Switch Non-File	
	<b>INPUT</b>	Switch Input	
	order=0	Remove	
	order=1	Remove	Show parameter
	order=2	Remove	Show parameter
	order=3	Remove	Show parameter
	ORDER=0	Add	
<b>AUTHORIZATION INFO</b>	Show authorizationInfo		
<b>ID</b>	Taverna-1.7-WF		
<b>status</b>	Publish		
<b>BACKEND SPECIFIC DATA</b>		Hide backendSpecificData	
		<b>COUNT</b>	1
		<b>OUTPUT</b>	STDOUT
		<b>ERROR</b>	STDERR
		<b>JOBTYPE</b>	single
		<b>maxWallTime</b>	10
		<b>backendId=GildaBroker</b>	Remove
<b>SITEINFO</b>		Hide siteInfo	
		<b>JOBMANAGER</b>	FORK
		<b>site</b>	SITE= <input type="text"/> Add
		<b>EXECUTABLE</b>	Stage: <input type="text"/> /home/kukla/taverna-1.7.0/v
		<b>PARAMPREFIX</b>	<input type="text"/>
		<b>ID=</b>	0 Add
<b>BACKENDID=</b>		<input type="text"/>	Add
<b>MAXPARALLELISM</b>		10	

Parameters





# Exposing Taverna workflow engine using GEMMLCA Administration Portlet

<b>DESCRIPTION</b>		Engine	
<b>PARAMETER</b>	<input type="button" value="Hide parameter"/>		
	regExp	<input type="text"/>	
	<b>COMMANDLINE</b>	<input checked="" type="checkbox"/>	
	<b>NAME</b>	<input type="text" value="-w"/>	
	<b>VALUE</b>	<input type="text" value="test.xml"/>	
	<b>FRIENDLYNAME</b>	<input type="text" value="Workflow"/>	
	<b>MANDATORY</b>	<input type="checkbox"/>	
	<b>FIXED</b>	<input type="checkbox"/>	
	<b>FILE</b>	<input type="checkbox"/> Switch <input type="checkbox"/> Non-File	
	<b>INPUT</b>	<input type="checkbox"/> Switch <input type="checkbox"/> Input	
	order=0 <input type="button" value="Remove"/>		
	order=1 <input type="button" value="Remove"/>	<input type="button" value="Show parameter"/>	
	order=2 <input type="button" value="Remove"/>	<input type="button" value="Show parameter"/>	
	order=3 <input type="button" value="Remove"/>	<input type="button" value="Show parameter"/>	
	ORDER=0 <input type="button" value="Add"/>		
<b>AUTHORIZATION INFO</b>	<input type="button" value="Show authorizationInfo"/>		
<b>ID</b>	Taverna-1.7-WF		
<b>status</b>	<input type="button" value="Publish"/>		
		<b>BACKEND SPECIFIC DATA</b>	
		backendId=GildaBroker	<input type="button" value="Remove"/>
		<b>BACKENDID=</b>	<input type="text" value=""/> <input type="button" value="Add"/>
		<b>MAXPARALLELISM</b>	<input type="text" value="10"/>
		<input type="button" value="Hide backendSpecificData"/>	
<b>COUNT</b>	<input type="text" value="1"/>		
<b>OUTPUT</b>	<input type="text" value="STDOUT"/>		
<b>ERROR</b>	<input type="text" value="STDERR"/>		
<b>JOBTYPE</b>	single <input type="button" value="v"/>		
<b>maxWallTime</b>	<input type="text" value="10"/>		
<b>SITEINFO</b>	<input type="button" value="Hide siteInfo"/>		
	id=0 <input type="button" value="Remove"/>	<b>JOBMANAGER</b>	FORK <input type="button" value="v"/>
	site	<b>SITE=</b>	<input type="text"/> <input type="button" value="Add"/>
	<b>EXECUTABLE</b>	Stage:	<input type="text" value="/home/kukla/taverna-1.7.0/v"/>
	<b>PARAMPREFIX</b>	<input type="text" value="."/>	
	ID=	<input type="text" value="0"/>	<input type="button" value="Add"/>

Submission settings





# Legacy Code interface Description of the exposed Taverna engine

```
<LCEnvironment maximumParallelism="10" id="Taverna-1.7-WF" status="private"
xmlns="http://uk.ac.wmin.cpc.gemlca/schema/legacyCodeConfig">
  <description>Engine</description>
  <parameter order="0" cmdline="true" mandatory="false" fixed="false" file="true" input="true" name="-w">
    <value>test.xml</value>
    <friendlyName>Workflow</friendlyName>
  </parameter>
  <parameter order="1" cmdline="true" mandatory="false" fixed="false" file="true" input="true" name="-p">
    <value>mystring.map</value>
    <friendlyName>parameter mapping</friendlyName>
  </parameter>
  <parameter order="2" cmdline="true" mandatory="false" fixed="false" file="true" input="true" name="-i">
    <value>taverna_input.zip</value>
    <friendlyName>WF input</friendlyName>
  </parameter>
  <parameter order="3" cmdline="true" mandatory="false" fixed="false" file="true" input="false" name="-o">
    <value>result.zip</value>
    <friendlyName>WF output</friendlyName>
  </parameter>
  <backendSpecificData backendId="GildaBroker" error="STDERR" output="STDOUT" maxWallTime="10" jobType="single" count="1">
    <siteInfo id="0" jobManager="FORK">
      <site>ce.hpc.iit.bme.hu</site>
      <site>dgt01.ui.savba.sk</site>
      <site>gilda-01.pd.infn.it</site>
      <site>gilda-ce.rediris.es</site>
      <site>grid010.ct.infn.it</site>.....
      <site>iceage-ce-01.ct.infn.it</site>
      <site>sirius-ce.ct.infn.it</site>
      <site>vega-ce.ct.infn.it</site>
      <executable stage="true">auto-deploy-taverna.sh</executable>
      <paramPrefix>.</paramPrefix>
    </siteInfo>
  </backendSpecificData>
  <authorizationInfo>
    <owner>/C=UK/O=eScience/OU=Westminster/L=ComputerScience/CN=tamas kukla</owner>
    <email>kukla.tamas@gmail.com</email>
  </authorizationInfo>
</LCEnvironment>
```





# Legacy Code interface Description of the exposed Taverna engine

```

<LCEnvironment maximumParallelism="10" id="Taverna-1.7-WF" status="private"
xmlns="http://uk.ac.wmin.cpc.gemlca/schema/legacyCodeConfig">
  <description>Engine</description>
  <parameter order="0" cmdline="true" mandatory="false" fixed="false" file="true" input="true" name="-w">
    <value>test.xml</value>
    <friendlyName>Workflow</friendlyName>
  </parameter>
  <parameter order="1" cmdline="true" mandatory="false" fixed="false" file="true" input="true" name="-p">
    <value>mystring.map</value>
    <friendlyName>parameter mapping</friendlyName>
  </parameter>
  <parameter order="2" cmdline="true" mandatory="false" fixed="false" file="true" input="true" name="-i">
    <value>taverna_input.zip</value>
    <friendlyName>WF input</friendlyName>
  </parameter>
  <parameter order="3" cmdline="true" mandatory="false" fixed="false" file="true" input="false" name="-o">
    <value>result.zip</value>
    <friendlyName>WF output</friendlyName>
  </parameter>
  <backendSpecificData backendId="GildaBroker" error="STDERR" output="STDOUT" maxWallTime="10" jobType="S" ac="count="1">
    <siteInfo id="0" jobManager="FORK">
      <site>ce.hpc.iit.bme.hu</site>
      <site>dgt01.ui.savba.sk</site>
      <site>gilda-01.pd.infn.it</site>
      <site>gilda-ce.rediris.es</site>
      <site>grid010.ct.infn.it</site>.....
      <site>iceage-ce-01.ct.infn.it</site>
      <site>sirius-ce.ct.infn.it</site>
      <site>vega-ce.ct.infn.it</site>
      <executable stage="true">auto-deploy-taverna.sh</executable>
      <paramPrefix>.</paramPrefix>
    </siteInfo>
  </backendSpecificData>
  <authorizationInfo>
    <owner>/C=UK/O=eScience/OU=Westminster/L=ComputerScience/CN=tamas kukla</owner>
    <email>kukla.tamas@gmail.com</email>
  </authorizationInfo>
</LCEnvironment>

```

Parameters



# Legacy Code interface Description of the exposed Taverna engine

```

<LCEnvironment maximumParallelism="10" id="Taverna-1.7-WF" status="private"
xmlns="http://uk.ac.wmin.cpc.gemlca/schema/legacyCodeConfig">
  <description>Engine</description>
  <parameter order="0" cmdline="true" mandatory="false" fixed="false" file="true" input="true" name="-w">
    <value>test.xml</value>
    <friendlyName>Workflow</friendlyName>
  </parameter>
  <parameter order="1" cmdline="true" mandatory="false" fixed="false" file="true" input="true" name="-p">
    <value>mystring.map</value>
    <friendlyName>parameter mapping</friendlyName>
  </parameter>
  <parameter order="2" cmdline="true" mandatory="false" fixed="false" file="true" input="true" name="-i">
    <value>taverna_input.zip</value>
    <friendlyName>WF input</friendlyName>
  </parameter>
  <parameter order="3" cmdline="true" mandatory="false" fixed="false" file="true" input="false" name="-o">
    <value>result.zip</value>
    <friendlyName>WF output</friendlyName>
  </parameter>
  <backendSpecificData backendId="GildaBroker" error="STDERR" output="STDOUT" maxWallTime="10" jobType="single" count="1">
    <siteInfo id="0" jobManager="FORK">
      <site>ce.hpc.iit.bme.hu</site>
      <site>dgt01.ui.savba.sk</site>
      <site>gilda-01.pd.infn.it</site>
      <site>gilda-ce.rediris.es</site>
      <site>grid010.ct.infn.it</site>.....
      <site>iceage-ce-01.ct.infn.it</site>
      <site>sirius-ce.ct.infn.it</site>
      <site>vega-ce.ct.infn.it</site>
      <executable stage="true">auto-deploy-taverna.sh</executable>
      <paramPrefix>.</paramPrefix>
    </siteInfo>
  </backendSpecificData>
  <authorizationInfo>
    <owner>/C=UK/O=eScience/OU=Westminster/L=ComputerScience/CN=tamas kukla</owner>
    <email>kukla.tamas@gmail.com</email>
  </authorizationInfo>
</LCEnvironment>

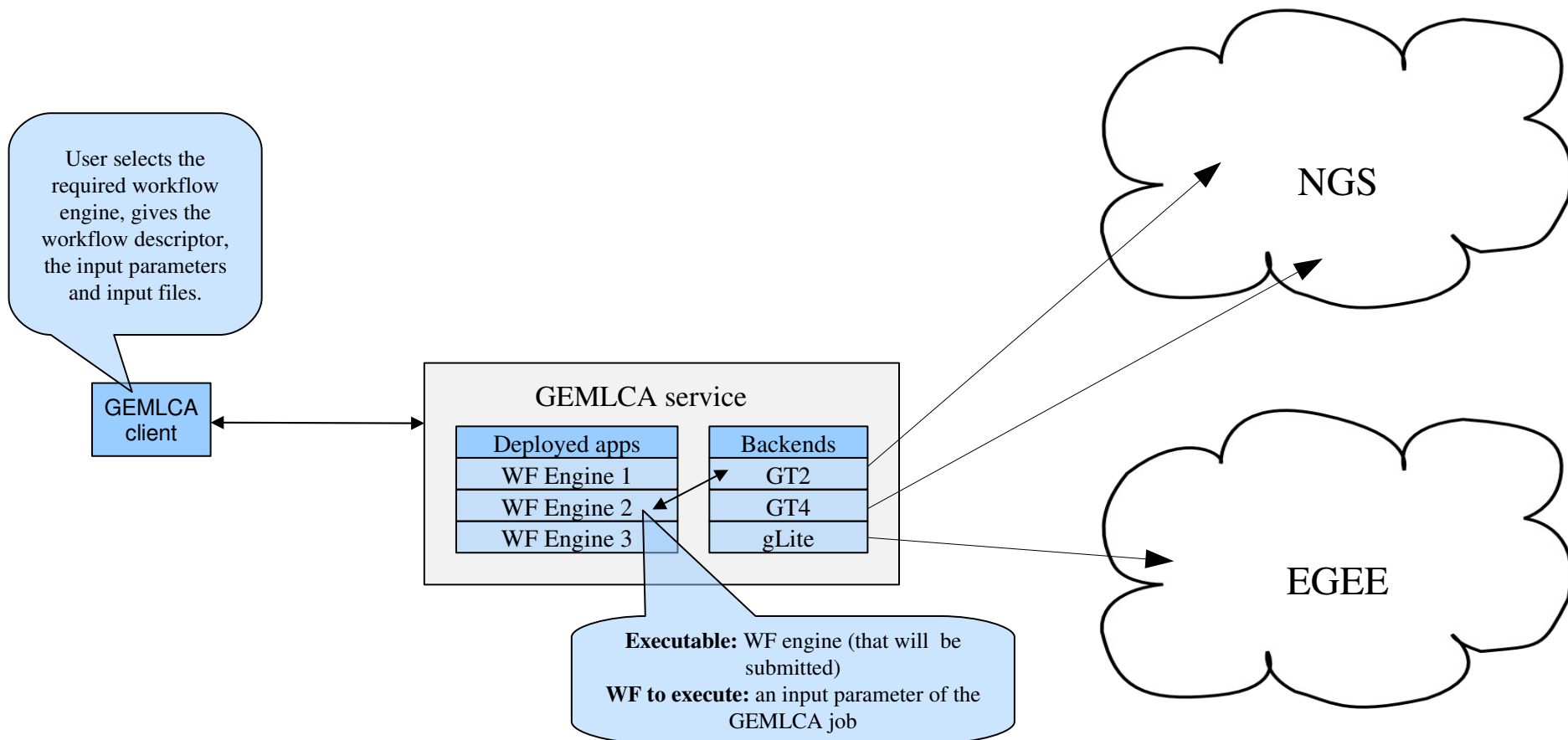
```

Submission  
settings



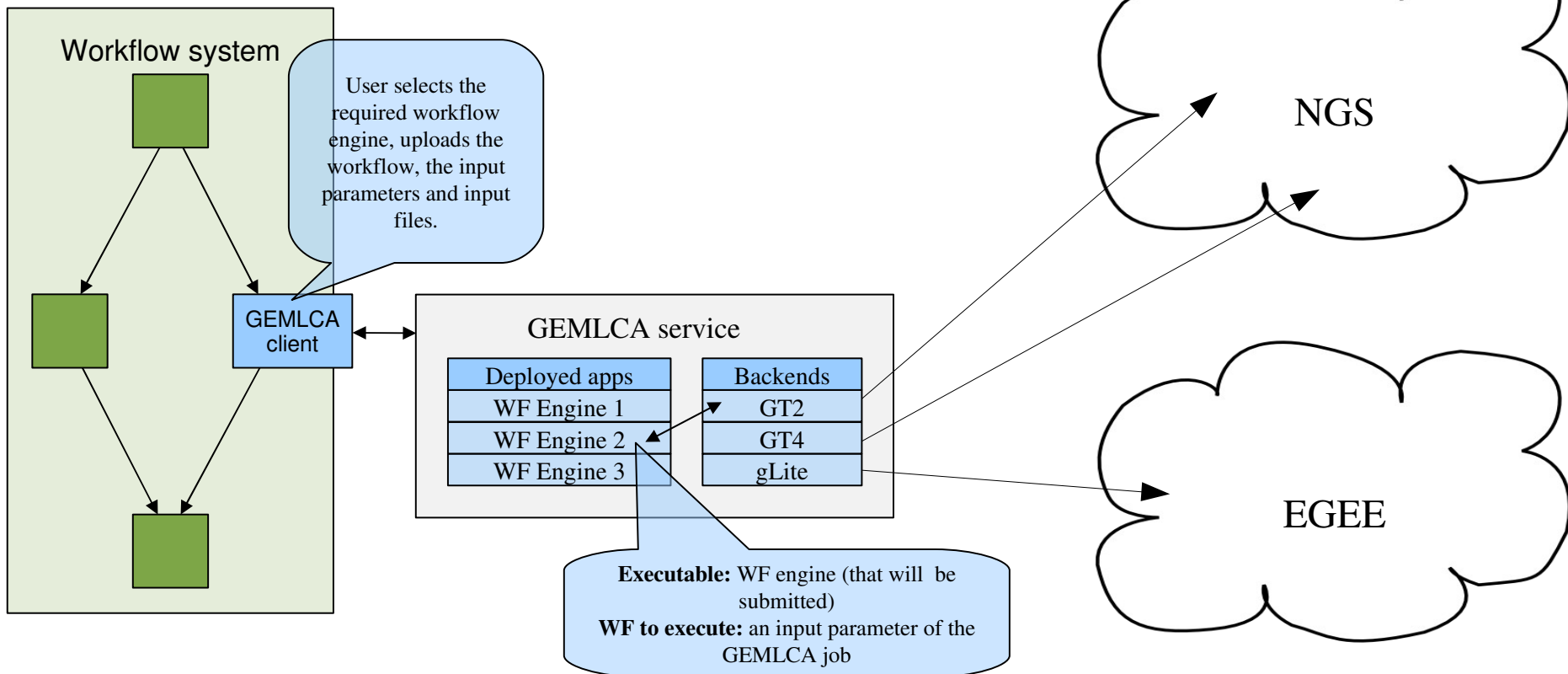


# Realisation of a Workflow engine repository and submitter via GEMMLCA





# Heterogeneous workflow nesting via GEMMLCA





# Parametrization of non-native workflow execution within the P-GRADE portal

- GEMMLCA was **integrated to the P-GRADE portal**.
- GEMMLCA jobs can be **parametrized** using a JAVA based GUI **within the P-GRADE workflow editor**.
- **Any** other workflow **system** can **adopt** this solution and integrate a GEMMLCA client.

The screenshot shows the 'Taverna properties' dialog box. The fields are as follows:

- Name: Taverna
- Job Type: GEMMLCA
- Grid: gilda\_LCG\_2\_BROKER
- Service: https://gilda-gemlca.cpc.wmin.ac.uk:8443/wsrf/services/uk/ac/w...
- Legacy Code: Taverna-1.7 - Autodeploy
- Sites: BROKER
- Parameters table:

Parameter name	Mandatory	Type	Mode	Value
wf	No	File	Input	FetchDragonImagesFromB...
map	No	File	Input	param.map
inp	No	File	Input	taverna_input.zip
out	No	File	Output	res.zip

Callouts on the left side of the dialog box:

- Selecting Grid
- Selecting GEMMLCA service
- Selecting workflow engine
- Selecting computational site

Callouts on the right side of the dialog box:

- Setting workflow descriptor
- Setting input parameters
- Setting workflow input files
- Setting workflow output file







# Case Study

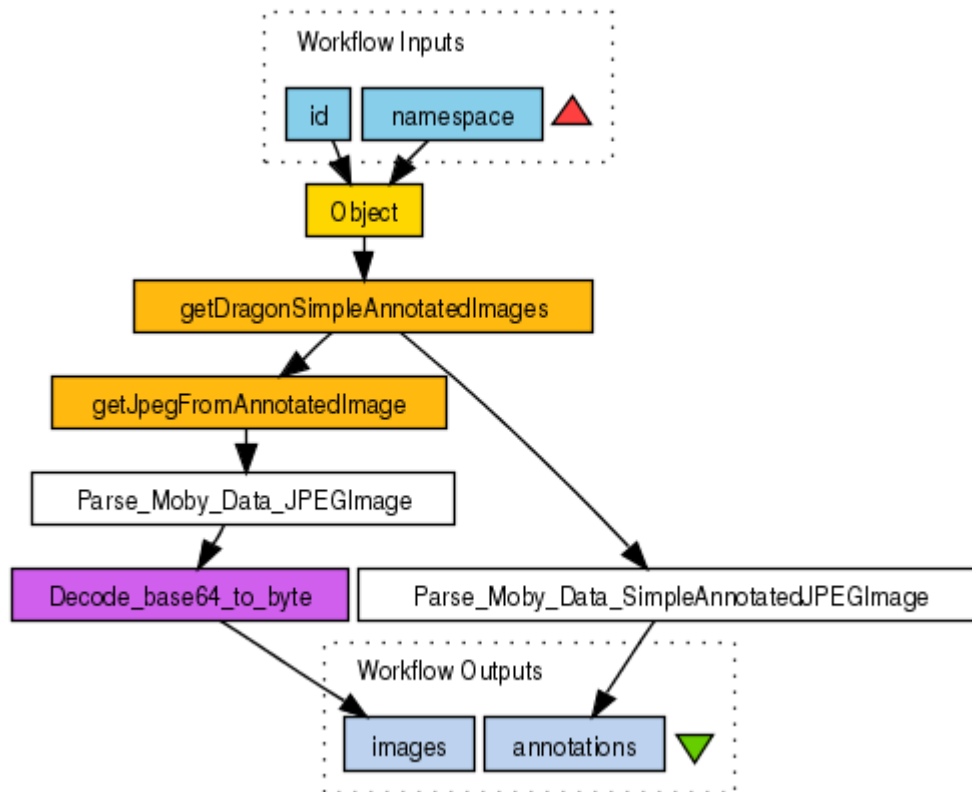
- A case study workflow, that presents how workflows of different systems interoperate, will be presented.
- It serves only demonstration purposes, it is **not a real life example**.
- A **high level heterogeneous P-GRADE workflow**, nesting a Taverna, Kepler and Triana workflows.
- The **data** that are **transferred** between the workflows are **stored files**, there is **no data transformation**.
- **If** data transformation is **needed**, user has to create a **data transformer job**.



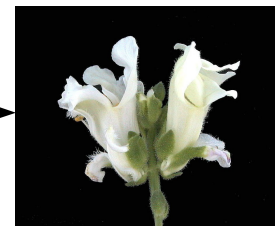


# Taverna workflow

- This workflow fetches several **images from a database**, creates a few directories and places the images into those directories as image files.

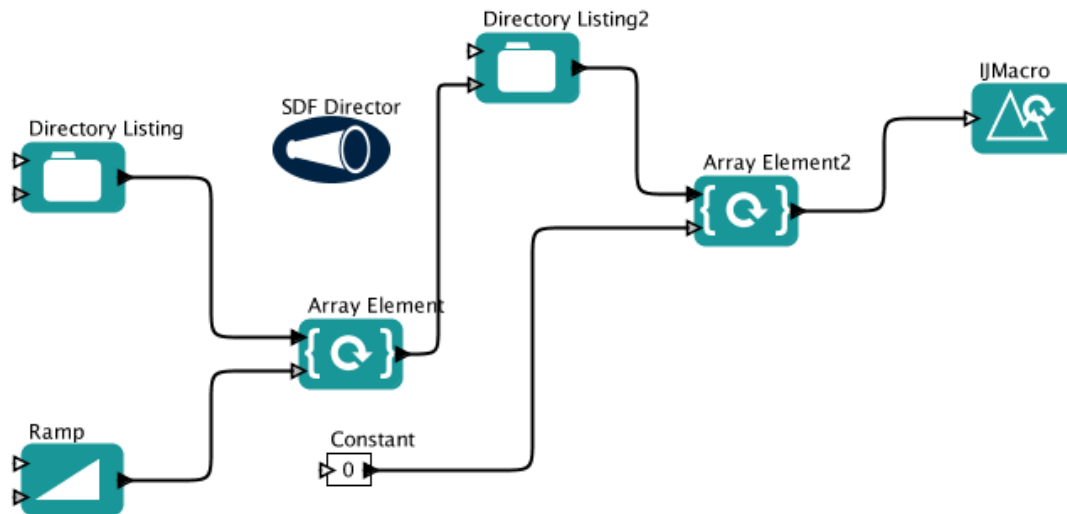


Database





# Kepler workflow

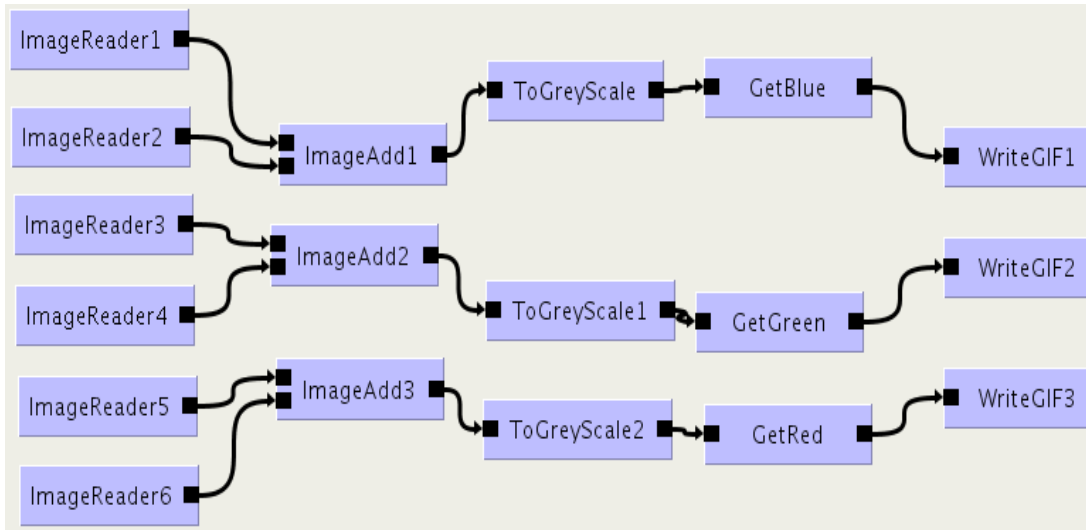


- This workflow goes through the directory structure of the archive input file and **manipulates each image** that it finds.
- The manipulation includes **edge highlighting, picture resizing and image type conversion.**

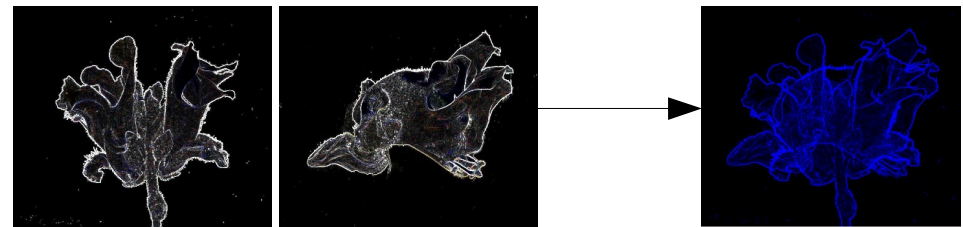




# Triana workflow

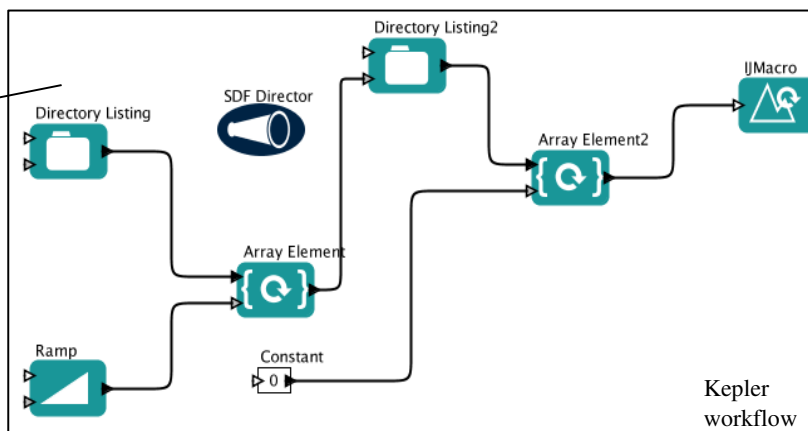
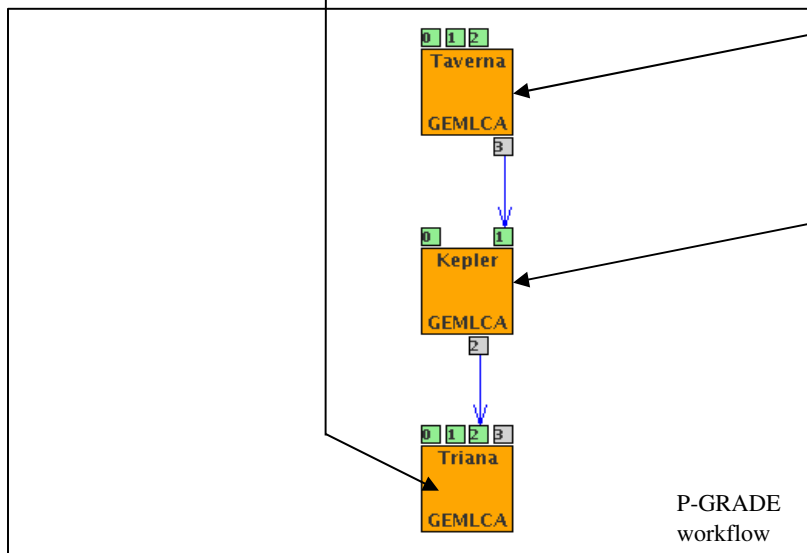
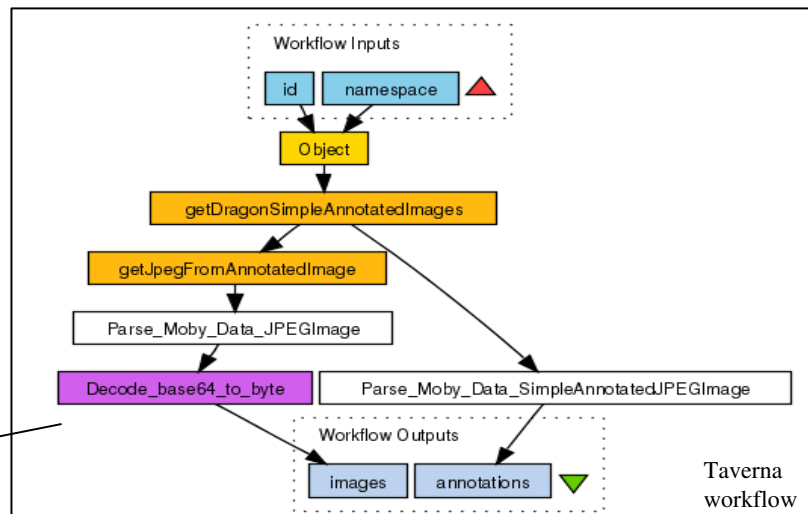
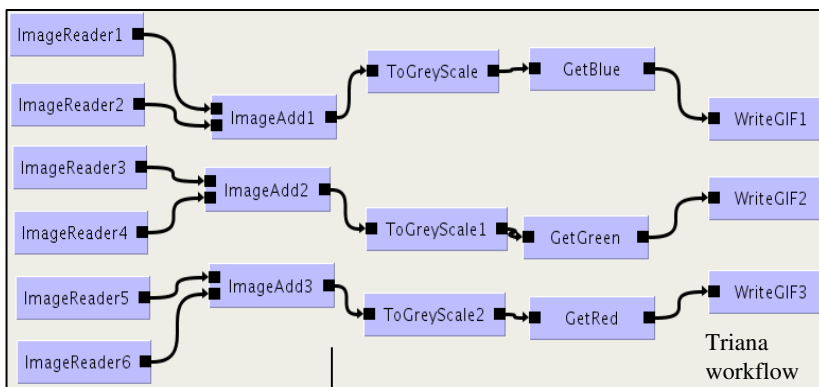


- This workflow **ouples the pictures, merges each couple** and converts the merged pictures to **greyscale** images.
- Then, **one colour component**, that can be either the blue, green or red, **is taken** of the greyscale pictures and **saved as new image file**.

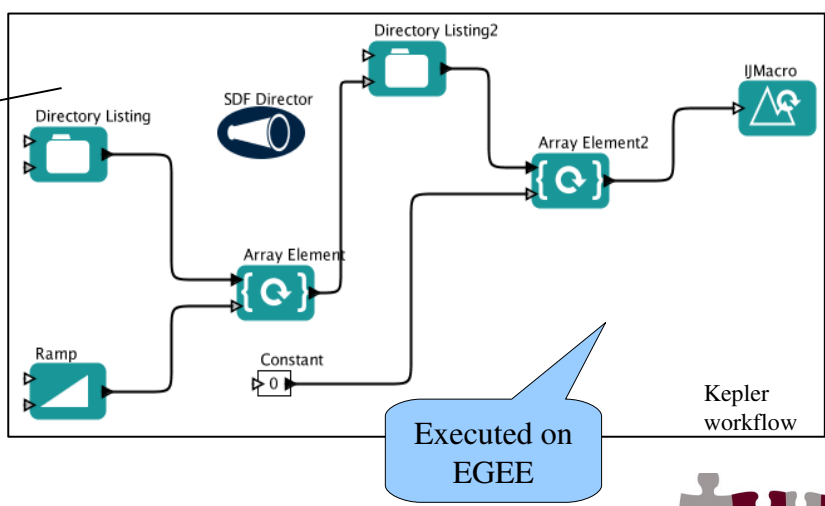
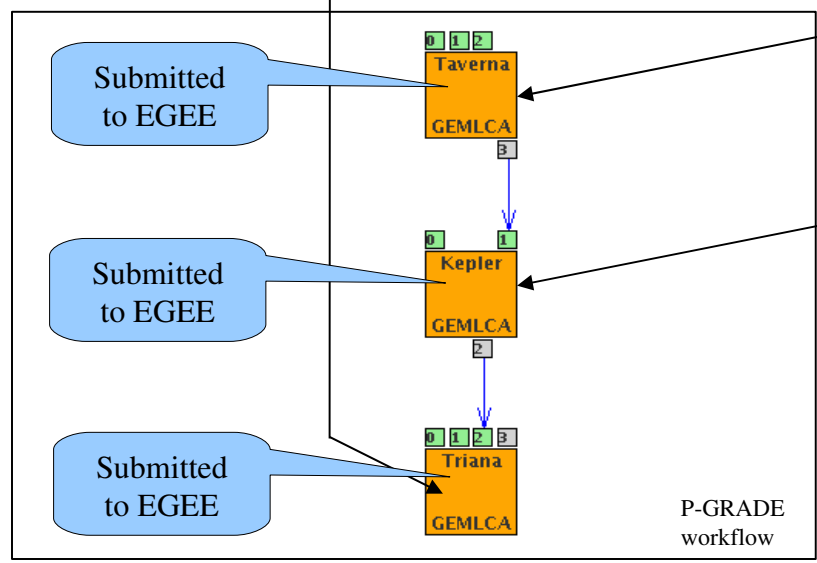
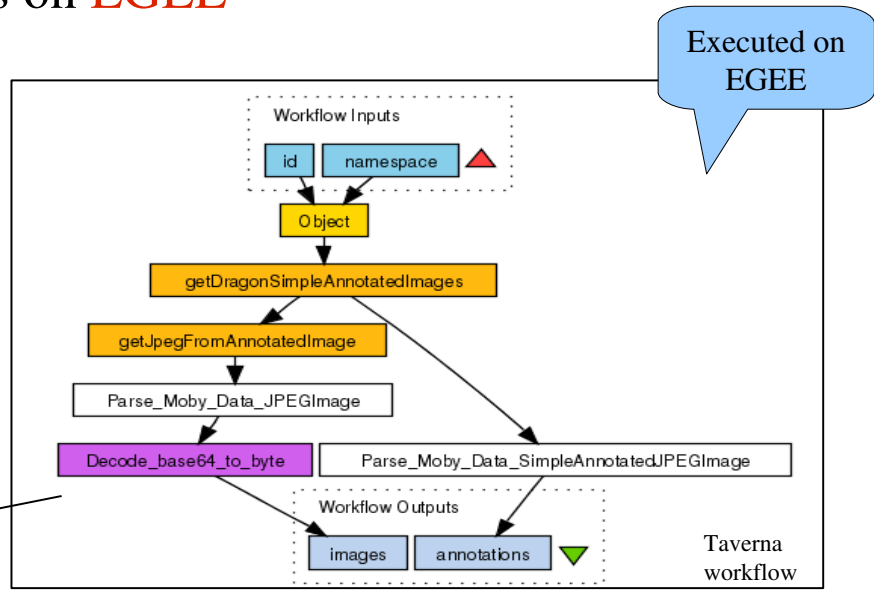
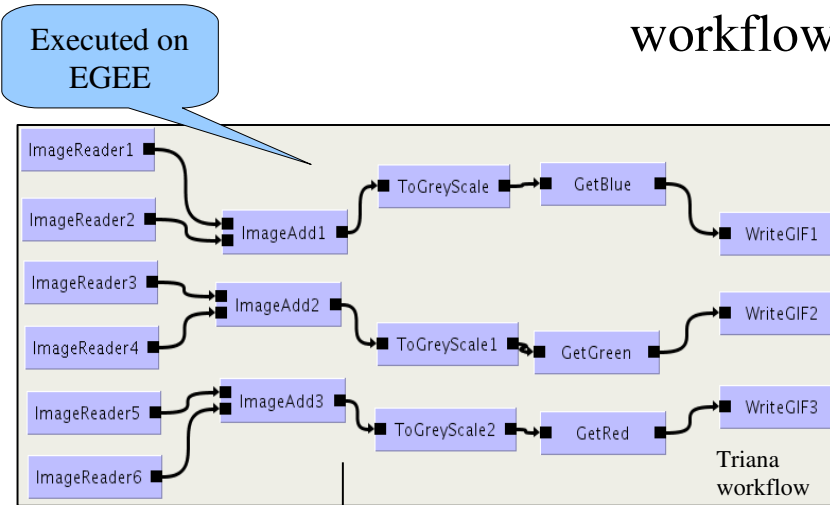




# Heterogeneous P-GRADE workflow embedding Triana, Taverna, and Kepler workflows

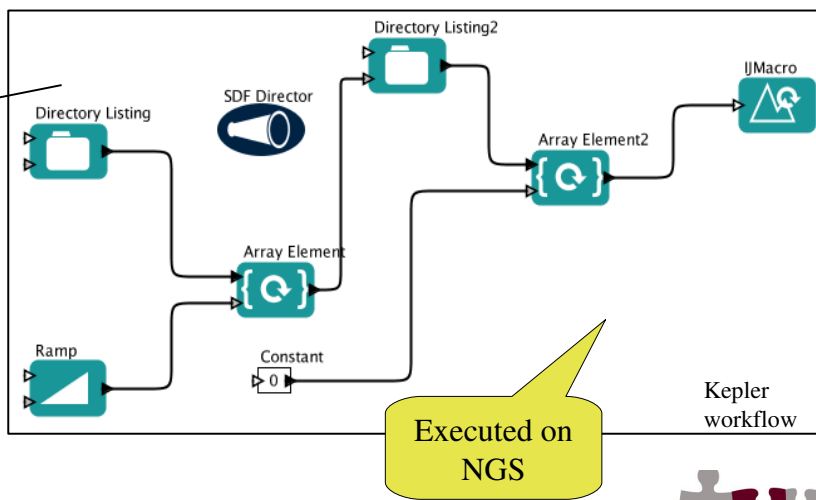
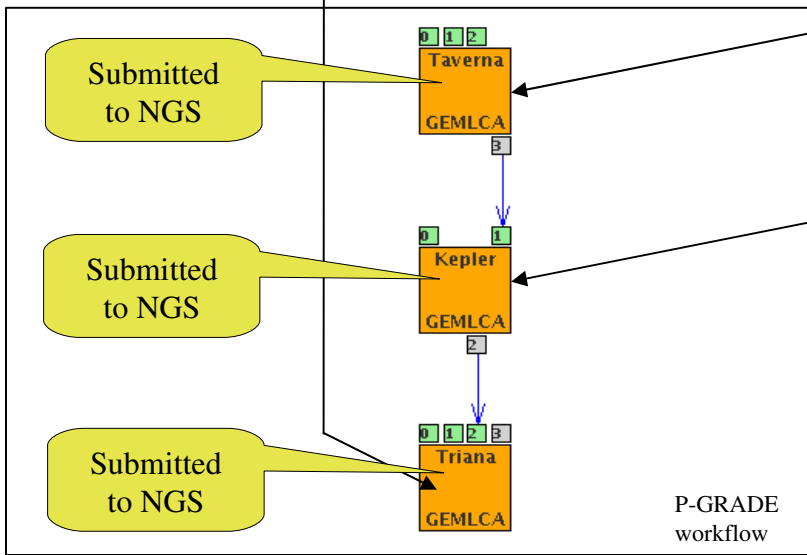
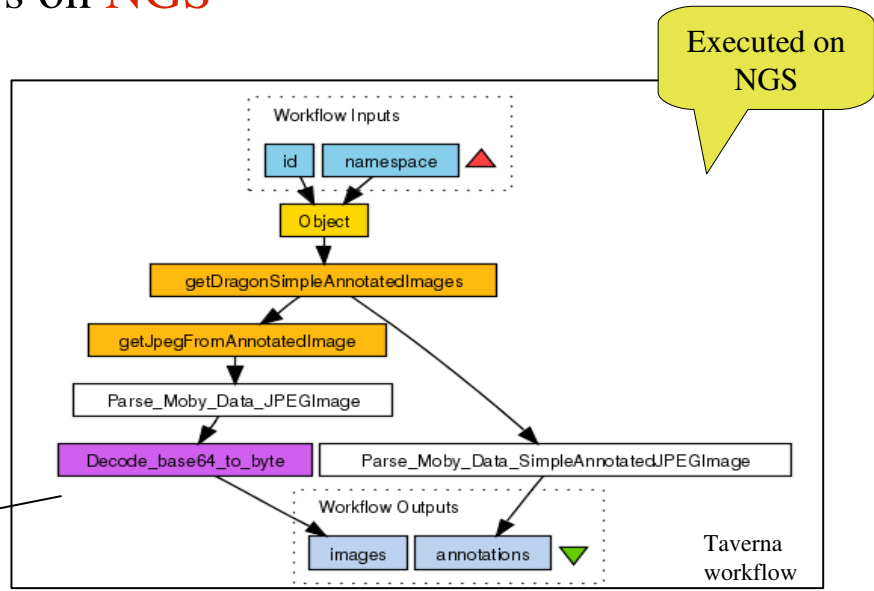
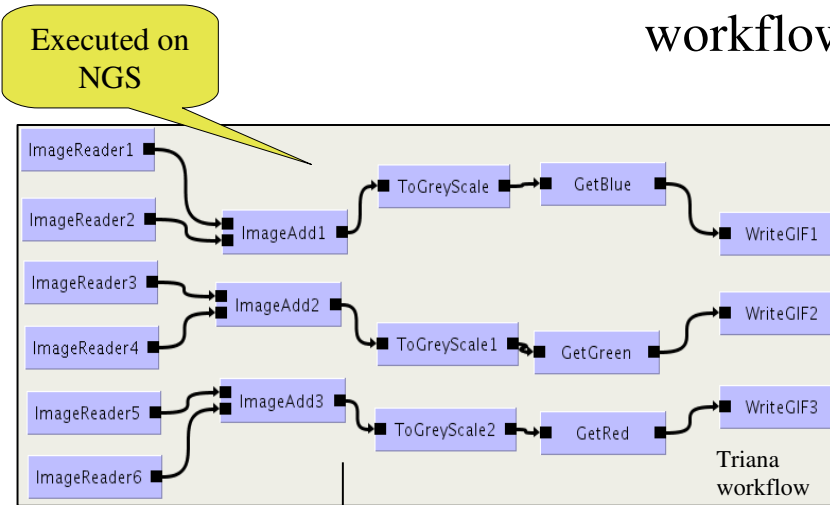


# Heterogeneous P-GRADE workflow embedding Triana, Taverna, and Kepler workflows on EGEE

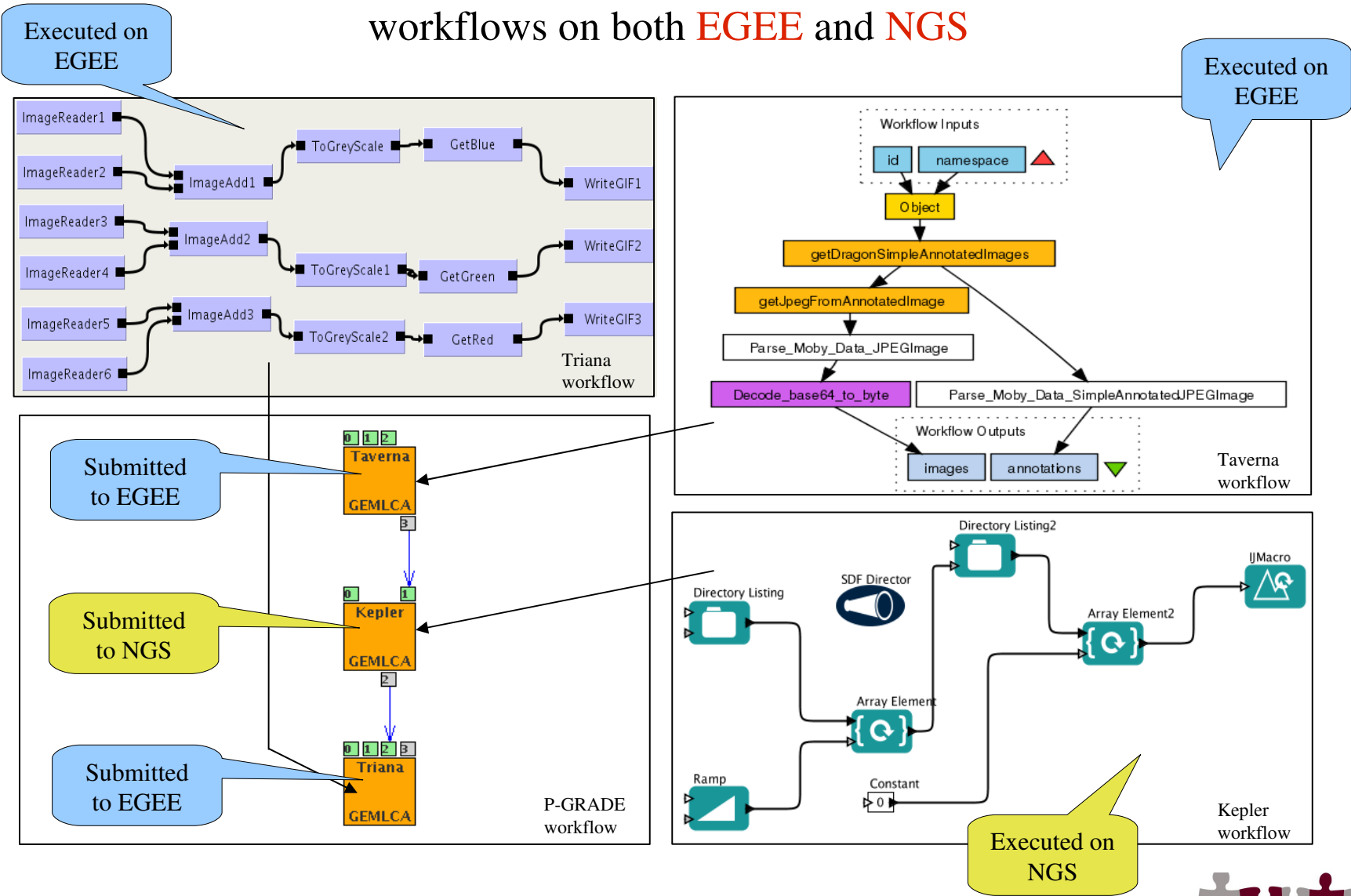




# Heterogeneous P-GRADE workflow embedding Triana, Taverna, and Kepler workflows on NGS



# Heterogeneous P-GRADE workflow embedding Triana, Taverna, and Kepler workflows on both EGEE and NGS







# Executing heterogeneous meta-workflows in the P-GRADE portal using EGEE and NGS resources

Workflow Manager

Back Refresh

**Job list**

Workflow	Job	Gridname	Hostname	Status	[ Logs ]	[ Output ]	[ Visualization ]	[ Action ]
1-Gilda-WFI-3				finished	-	<input checked="" type="checkbox"/>	Visualize All	Submit Attach Delete
	Kepler	gilda_LCG_2_BROKER	iceage- ce-01.ct.infn.it:2119/jobmanager- lcpbbs-long	finished	--		-	
	Taverna	gilda_LCG_2_BROKER	iceage- ce-01.ct.infn.it:2119/jobmanager- lcpbbs-short	finished	--		-	
	Triana	gilda_LCG_2_BROKER	iceage- ce-01.ct.infn.it:2119/jobmanager- lcpbbs-infinite	finished	--		-	

**Message:** Workflow details successfully displayed.

---

Workflow Manager

Back Refresh

**Job list**

Workflow	Job	Gridname	Hostname	Status	[ Logs ]	[ Output ]	[ Visualization ]	[ Action ]
1-NGS-Gilda-WFI-3				finished	-	<input checked="" type="checkbox"/>	Visualize All	Submit Attach Delete
	Kepler	NGS	ngs.wmin.ac.uk	finished	Out Err		-	
	Taverna	gilda_LCG_2_BROKER	ce.hpc.iit.bme.hu:2119/jobmanager- lcpbbs-gilda	finished	--		-	
	Triana	gilda_LCG_2_BROKER	vega-ce.ct.infn.it:2119/jobmanager- lcsge-gilda	finished	--		-	

**Message:** Workflow details successfully displayed.

Using only  
**EGEE**  
resources

Using both  
**NGS and**  
**EGEE**  
resources





# Executing heterogeneous meta-workflows in the P-GRADE portal using EGEE and NGS resources

Workflow Manager

Back Refresh

Workflow	Job	Gridname	Hostname	Status	[ Logs ]	[ Output ]	[ Visualization ]	[ Action ]
1-Gilda-WFI-3				finished	-	<input checked="" type="checkbox"/>	Visualize All	Submit Attach Delete
	Kepler	gilda_LCG_2_BROKER	iceage- ce-01.ct.infn.it:2119/jobmanager- lcpbbs-long	finished	--			
	Taverna	gilda_LCG_2_BROKER	iceage- ce-01.ct.infn.it:2119/jobmanager- lcpbbs-short	finished	--			
	Triana	gilda_LCG_2_BROKER	iceage- ce-01.ct.infn.it:2119/jobmanager- lcpbbs-infinite	finished	--			

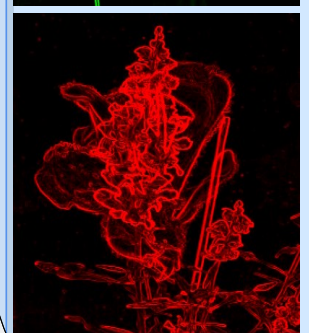
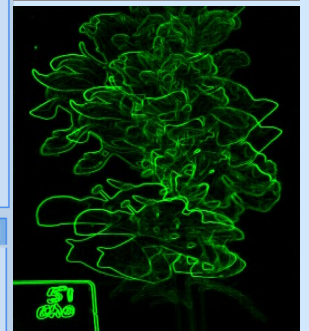
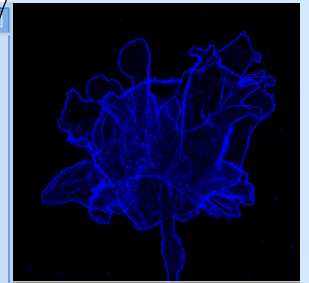
Message: Workflow details successfully displayed.

Workflow Manager

Back Refresh

Workflow	Job	Gridname	Hostname	Status	[ Logs ]	[ Output ]	[ Visualization ]	[ Action ]
1-NGS-Gilda-WFI-3				finished	-	<input checked="" type="checkbox"/>	Visualize All	Submit Attach Delete
	Kepler	NGS	ngs.wmin.ac.uk	finished	Out	Err		
	Taverna	gilda_LCG_2_BROKER	ce.hpc.iit.bme.hu:2119/jobmanager- lcpbbs-gilda	finished	--			
	Triana	gilda_LCG_2_BROKER	vega-ce.ct.infn.it:2119/jobmanager- lcpbbs-gilda	finished	--			

Message: Workflow details successfully displayed.





# Conclusions

- This presentation introduced a **general solution** for integrating and accessing heterogeneous workflow engines.
- The solution can be used to achieve workflow interoperability and sharing **at the level of workflow engine integration**.
- The solution **exposes** various workflow **engines via a GEMMLCA** service, that is capable of **submitting the engines to the Grid** using GT2, GT4 or gLite based Grid infrastructures, such as **NGS** or **EGEE**.
- It can be extended to support any further grid middleware, by implementing an additional GEMMLCA submitter backend.
- The solution **keeps the data at computational** sites, it is **scalable** in terms of **number of workflows** and **amount of data**.





# Conclusions

- Workflow engine **deployment** to this system does not require any code re-engineering, **user level understanding** is sufficient.
- The solution **can be adopted by any** workflow management **system** by integrating GEMMLCA with the selected WF system
- Our main concern is to **investigate and realise** further workflow interoperability **models** and to **extend** the solution with a **workflow repository**.
- The integration technique **can be applied** not only for workflow engine integration, but for **data middle-ware integration** as well.





# Questions?





# Thank You!

