Project Number: **RI-312579**

Project Acronym: **ER-flow**

Project Full Title:

Building an European Research Community through Interoperable Workflows and Data

Theme: **Research Infrastructures**
Call Identifier: **FP7-Infrastructures-2012-1**
Funding Scheme: **Coordination and Support Action**

Deliverable D3.3
Extended Simulation Platform

Due date of deliverable: 30/04/2014
Actual submission date: 30/04/2014
Start date of project: 01/09/2012
Duration: 24 months

Lead Contractor: University of Westminster
Dissemination Level: PU
Version: Final

ER-flow is supported by the FP7 Capacities Programme under contract no RI-312579

1 Table of Contents

2 List of Figures and Tables

2.1 Figures

2.2 Tables

3 Status and Change History

Status:	**Name:**	**Date:**	**Signature:**
Draft:	Gabor Terstyanszky	03/04/14	n.n. electronically
Reviewed:	Peter Kacsuk	28/04/14	n.n. electronically
Approved:	Steve Winter	30/04/14	n.n. electronically

Table 3.3. Deliverable Status

Version	**Date**	**Pages**	**Author**	**Modification**
1.0	03/04		G Terstyanszky	Report template Section 5 Introduction
1.1	08/04		G Terstyanszky	Section 6: enhancement requests
	10/04		B. Meilhac	Section 7: SHIWA Submission Service
	10/04		E. Michniak	Section 7: SHIWA Repository
	10/04		J. Arshad	Section 7: access to gLite and UNICORE resources
	13/04		B. Meilhac	Section 8: running workflows on the cloud
1.2	13/04		G Terstyanszky	Section 8: ER-flow usage scenarios
				Section 8: ER-flow development and execution environment
2.0	19/04		G Terstyanszky	revisioning Section 6, 7 and 8
3.0	24/04		G Terstyanszky	first complete version
3.1	28/04		P. Kacsuk	Internal review of the report
final	29/04		G Terstyanszky	Revising and compiling the final version

Table 3.4. Deliverable Change History

4 Glossary

Axx	Major Enhancement Request
ASM	Application Specific Module
CGI	Coarse-Grained Interoperability
DCI	Distributed Computing Infrastructure
EGI	European Grid Infrastructure
FGI	Fine-Grained Interoperability
Ixx	Minor Enhancement Request
MTA-SZTAKI	Magyar Tudomanyos Akademia Szamitastechnikai Kutato Intezete
NGI	National Grid Infrastructure
SSP	SHIWA Simulation Platform
UoW	University of Westminster
VO	Virtual Organisation
WF	Workflow
WE	workflow engine
WP	Work package

Table 4.1. Glossary

5 Introduction

Workflows have become essential to integrate the user and the infrastructure domain to support research communities. They help to formalize and structure scientific experiments. Workflows automate the analytical and computational steps that researchers need to go through from data selection and integration to computation and from final data presentation to visualization. Research communities have developed different workflow systems and created large numbers of workflows to run experiments. These workflow systems have different workflow description languages, enactment strategies and use different middleware to access infrastructures. It takes a significant effort and time to learn how to use workflow systems, and it requires specific expertise and skills to develop and maintain workflows. Researchers are not keen to learn new workflow systems to run their experiments as this is a time-consuming process. They would prefer workflows sharing, i.e. automatic porting of workflows across computing infrastructures and workflow systems to optimise their efforts. Currently, the major obstacle of workflow sharing is that workflow systems are not interoperable. To address workflow interoperability the “Sharing Interoperable Workflows for Large-Scale Scientific Simulations on Available DCIs” (SHIWA) project [1] developed the Coarse-Grained Interoperability (CGI) concept. SHIWA created and deployed a production-level CGI service, the SHIWA Simulation Platform (SSP).

The ER-flow research communities have a different expertise in the workflow technology. The Computational Chemistry and Life Sciences community has created MOTEUR, Taverna and WS-PGRADE workflows. The Heliophysics community developed Taverna workflows. The Astrophysics community was relatively new to the workflow technology. The first two communities used both command-line and graphical user interfaces while the third community only command-line interfaces to manage workflows. They execute workflows on different infrastructures such as clusters, services grids, web services, etc. To support them the SHIWA Simulation Platform must support different workflow systems and provide access to different computing infrastructures.

WP3 and WP5 analysed the community requirements and developed the ER-flow strategy to address computing infrastructure and workflow challenges. This strategy defined two environments: a development and an execution environment. The development environment is the SHIWA Simulation Platform. The execution environment contains community gateways deployed and managed by the ER-flow communities within the framework of the SCI-BUS project [2]. These environments are connected to two SHIWA services (SHIWA Repository and SHIWA Submission Service). WP3 focused on supporting communities developing WS-PGRADE workflows on the ER-flow development environment and running them on both environments in Year 1. WP3 completed two major development tasks: creating a web service based submission service and integrating the GEMLCA and SHIWA Repository to enable remote execution of non-native workflows through community gateways in Year 2.

WP3 and WP5 collected community requirements and got community feedback on the simulation platform. They are obtained through the ER-flow Bugzilla [3], SHIWA User Forum [4] and WP5 feedback report [5]. ER-flow defined three phases to collect, analyse and address these requests. WP3 categorised these requests into major and minor ones. It created a schedule to implement these requests classifying them as short-, mid- and long-term requests. WP3 upgraded the SHIWA Simulation Platform deploying several enhancement requests to improve its usability and user experience in Year 1 and Year 2. (See D3.2 report [6] and Section 7 of this report).

In Section 6 the report outlines the enhancement and feature requests that the research communities and technology providers identified and highlights those requests WP3

addressed. Section 7 describes how the selected major and minor enhancement requests have been implemented in Year 2. In Section 8 we give a short summary of the current version of the ER-flow Development and Execution Environment. We also present the ER-flow usage scenarios which support local and remote execution of non-native workflows using the CGI concept and the new SHIWA Submission Service. Finally, Section 8 contains conclusions presenting a short evaluation of the new services and listing some of the further developments.

6 Community Requirements for Managing Workflows

ER-flow collected research communities' enhancement requests and requirements towards the SHIWA Simulation Platform in two phases in Year 1 and in one more phase in Year 2. WP3 defined two types of enhancement requests: major and minor and introduced three categories: short-, medium- and long-term requests.

6.1.1 Phase 1: October – November 2012

In **Phase 1** WP5 compiled a list of workflows of the Astrophysics, Computational Chemistry, Heliophysics and Life Science community which they wanted to port to the SHIWA Simulation Platform. WP3 and WP5 analysed these workflows and defined enhancement requests and platform requirements needed to run these workflows. Table 6.1 and 6.2 presents the identified major and minor enhancement requests.

No.	major enhancement requests	partners	Schedule
A01	To upgrade the **SHIWA Portal** to offer better performance and services needed by the research communities.	UoW	ST
A02	To create an **export/import service** to support automatic upload/download operation between the SHIWA Portal and SHIWA Repository	SZTAKI + UoW	ST
A03	To implement **single-sign-on** for the SHIWA Portal and the SHIWA Repository	SZTAKI + Uow	ST
A04	To support **robot certificate** to offer simple access for both workflow developers and e-scientists	SZTAKI	ST
A05	To provide either the **end-user interface** or **ASM based interface** for e-scientist considering their requirements	UoW	MT
A06	To re-engineer the GT4 based **GEMLCA Service** replacing with a web services based submission service	UoW	MT
A07	To **monitor the components of the simulation platform** (portal + repository + submission service + workflow systems)	SZTAKI + UoW	MT
A08	To create **personalized access to the SHIWA Repository**, based on the user profile	UoW	MT
A09	To collect **information about workflow execution** (domain, user, etc.) including both meta and sub-workflows	SZTAKI	MT
A10	To create a **personalised view** of the SHIWA Repository (my own workflows, my favourite workflows, etc.)	UoW	MT
A11	To support **monitoring of non-native sub-workflow execution** and retrieve proper fault information	SZTAKI	MT
A12	To define and publish the **SHIWA Repository API** to support upload, listing, searching and downloading workflows	UoW	LT
A13	To define the **SHIWA API** to expose the simulation platform services which manage workflow execution	SZTAKI + UoW	LT
A14	To **integrate** login with **EGI SSO, facebook, gmail**, etc.	SZTAKI + UoW	LT
A15	To enable **automatic workflow execution environment deployment** on the cloud and investigating workflow execution on the cloud	SZTAKI + UoW	LT
A16	To support **workflow downloading** in CGI bundle format without the SHIWA desktop concept	SZTAKI + UoW	LT
A17	To extend **ASM with SHIWA API** to allow users to manage workflow data and execution	SZTAKI	LT

Table 6.1: Major enhancement requests in phase 1

Legends: ST – short-term MT – medium-term LT – long-term
green colour - completed in phase 1 and phase 2
blue colour - completed in phase 3

The ER-flow research communities submitted 17 major enhancement requests in this phase. They classified four as short-term, seven as mid-term and six as long-term requests. Since ER-flow is support action project WP3 decided to target only A01-A02 and A04-A06 requests in Year 1 and A15 in Year 2 since they can significantly improve the useability and user experience of the simulation platform. The work package implemented A01, A02 and A04 requests in the period of November 2012 - January 2013 and upgraded the simulation platform to SSP 4.1. (See details in D3.2.) WP3 started working on A06 to replace the GEMLCA Service with a new submission service.

WP3 and WP5 identified further 14 minor enhancement requests in phase 1. They specified five as short-term and nine as mid-term requests. WP3 implemented all short-term enhancement requests and three of the mid-term requests. (See details in D3.2.)

No.	minor enhancement requests	partners	schedule
I01	To introduce **pre-defined domains** and **sub-domains** and re-allocate workflows to relevant domains	UoW	ST
I02	To support proper **workflow versioning** in the SHIWA Repository	UoW	ST
I03	To modify the **welcome page of the SHIWA Repository** and provide direct access to the browse page of abstract workflows	UoW	ST
I04	**Controlling characters** entered into the SHIWA Repository (lowercase, uppercase, white spaces should not matter in the repository)	UoW	ST
I05	To enable **automatic re-submission of failed non-native sub-workflows**	SZTAKI	ST
I06	To assign **URL to each workflow implementation** uploaded to the repository (the URL should guide to workflow implementation stored in the repository)	UoW	MT
I07	To create and **display lists of** the latest 5-10 workflows and the most frequently used 5-10 workflows	UoW	MT
I08	To enable **access to the simulation platform via Facebook**	SZTAKI	MT
I09	To enable **downloading Taverna 2 workflows** from the myExperiment repository	UoW	MT
I10	To enable specifying input file type (for example GIF file for ImageMerger wf)	UoW	MT
I11	To replace the existing validation strategy with **community based validation** and implement it	UoW	MT
I12	To **transfer workflows** uploaded to the SHIWA Repository to the EGI App database	UoW	MT
I13	To allow users to move from the **end user view** to the **power user view** by a single click assuming that user will access the end user view after login	SZTAKI	MT
I14	To **test workflows** uploaded to the repository on cloud or on local resources by a single click	UoW	MT

Table 6.2: Minor enhancement requests in phase 1

6.1.2 Phase 2: December 2012 - April 2013

In this phase the key event was the ER-flow Application Porting Workshop held in London in 19 – 22 March 2013. WP3 elaborated new manual and tutorials. The work package ran a portal and repository tutorial at the workshop for researchers of the four ER-flow

communities plus for researchers of the hydrometeorology community (DRIHM project) and seismology community (VERCE project). The workshop participants created the second set on enhancement requests (See Table 6.3 and Table 6.4). They defined two more mid-term major and four short-term and two mid-term enhancement requests.

No.	major enhancement requests	partners	schedule
A18	To enable collection and processing workflow execution statistics	UoW	MT
A19	To support **interactive workflow nodes**	UoW	MT

Table 6.3: Major enhancement requests in phase 2

WP3 implemented A18 and I15-I16 enhancement requests in the period April – June 2013. The work package added these enhancement requests to the simulation platform and upgraded it to SSP v4.2 in July 2013

No.	minor enhancement requests	partners	schedule
I15	To provide access to **vlemed gLite resources**	UoW	ST
I16	To upgrade **export-import service** for MOTEUR workflow system	CNRS UoW	ST
I17	To offer **remote access** to the SHIWA Repository and the SHIWA Submission Service	SZTAKI UOW	ST
I18	To provide access to **UNICORE resources**	UoW	ST
I19	To implement **CGI support** for the **UNICORE workflow system**	UoW	ST
I20	To restrict **number of users** who can user the portal at the same time	UoW	MT
I21	To extend **CGI support** for the **DISPEL workflow system**	UoW	MT

Table 6.4: Minor enhancement requests in phase 2

6.1.3 Phase 3: May – July 2013

The key event of this phase was the second ER-flow project meeting. WP3 demonstrated the SSP v4.2 presenting the new and upgraded features and services. The four research communities evaluated this version and compiled the third set of enhancement requests presented in in D5.1 “User evaluation of the simulation platform” report

No.	major enhancement requests	partners	schedule
A20	To support white-box based meta-workflow creation and execution	UoW	MT

Table 6.5: Major enhancement requests in phase 3

The third set of the enhancement requests contains one mid-term major and nine short-term minor requests. See Table 6.5 and 6.6.

No.	minor enhancement requests	partners	schedule
I22	To provide access to **astro gLite resources**	UoW	ST
I23	To reduce or **eliminate**, if possible, need of **entering redundant information** of the workflows in the repository and simulation platform	SZTAKI UOW	ST
I24	To moderate and **clean up the repository contents** on a regular basis	UoW	ST
I25	To use, if possible, all the **information available in the Taverna workflow bundle** to minimize the process of entering parameter information	UoW	ST
I26	To create a **"richer" user profile** which must hold information about allocation and usage of resources	UoW	ST
I27	To **display** on request **who exported and when workflows** from the repository and/or who executed it from the SSP	SZTAKI UoW	ST
I28	To enable **deletion and transfer ownership of workflows**	UoW	ST
I29	To improve workflow management when a group is deleted	UoW	ST

Table 6.6: Minor enhancement requests in phase 3

WP3 completed A06 and implemented A15, A20 and I22-I24 enhancement requests in the period of September 2013 – February 2014. The work package added these enhancement requests to the simulation platform and upgraded it to SSP v5.0 in March 2014

WP3 was working on several major and minor enhancement requests in the period September 2013 – February 2014:

phase 1:
major requests: A06 & A15 minor requests:
phase 2
major requests: minor requests: I17 -I18 & I21
phase 3
major requests: A20 minor requests: I22 - I24

See details of the implemented enhancement requests in Section 7 and Section 8.

7 Upgrading the ER-flow Development and Execution Environment

7.1 Major Upgrades

7.1.1 SHIWA Submission Service enhancement requests: A06, I17

Problem: The SHIWA Simulation Platform uses the GEMLCA Service to manage non-native workflows. The GEMLCA Service has been integrated to the SHIWA Portal to submit non-native workflows to GT2/GT4 resources. It allows users to create, configure and execute non-native workflows on a specific non-native workflow engine to run (e.g. Taverna, Kepler, Moteur) using the GEMLCA Repository and the SHIWA Repository.

The ER-flow research communities identified four major limitations of the GEMLCA Service which prevent efficient management of non-native workflows on their community gateways. First, it is implemented as a GT4 service and it requires GT4 deployment. It is a major limitation for the science gateways of research communities which are considered as execution environments to run workflows. Second, the GEMLCA Service supports submission only to GT2 and GT4 based computing resources, while communities also want to submit workflows to gLite and UNICORE resources. Third, currently there are two repositories in the simulation platform: GEMLCA Repository and SHIWA Repository. The first one stores execution-enabled workflows plus workflow engine data and its executable(s) whilst the second stores only workflow data. As a result, there are two data formats to describe workflow and workflow engine data, plus two GUIs to manage this data. Fourth, the GEMLCA Service has some performance issues during workflow execution because of the implemented caching solution.

Design: WP3 designed a new service, called the new SHIWA Submission Service, to replace the current GEMLCA Service in order to address the limitations listed above. The basic design aim was to minimise modifications of services which use the GEMLCA Service. The solution was to create service which acts as an intermediate layer between the SHIWA Portal and the SHIWA Repository. The **Error! Reference source not found.** shows how the new submission service is integrated to the SHIWA Simulation Platform.

There were two major development tasks. First, WP3 has to extend the SHIWA Repository to allow management of workflow engine data and enable workflow engine execution. This development incorporates the integration of workflow engines management and extending the communication between the repository and the submission service. Second, the existing submission service has to be re-implemented to enable communication among the DCI Bridge, the SHIWA Portal and the SHIWA Repository. WP3 implemented the Search and Process API. The Search API provides search operations for the SHIWA Portal and the SHIWA Repository to find workflows and workflow engines. The Process API allows workflow submission through the DCI Bridge.

Implementation: The SHIWA Submission Service has two components: a front-end that allows applications to communicate with the submission service and a back-end that requests, manages and processes workflow information. This back-end contains two main components. The *Repository Communication* that requests data from the SHIWA Repository through a web service deployed in the repository. The *JSDL Modificator* changes the configuration file used by the DCI Bridge to allow workflow submission.

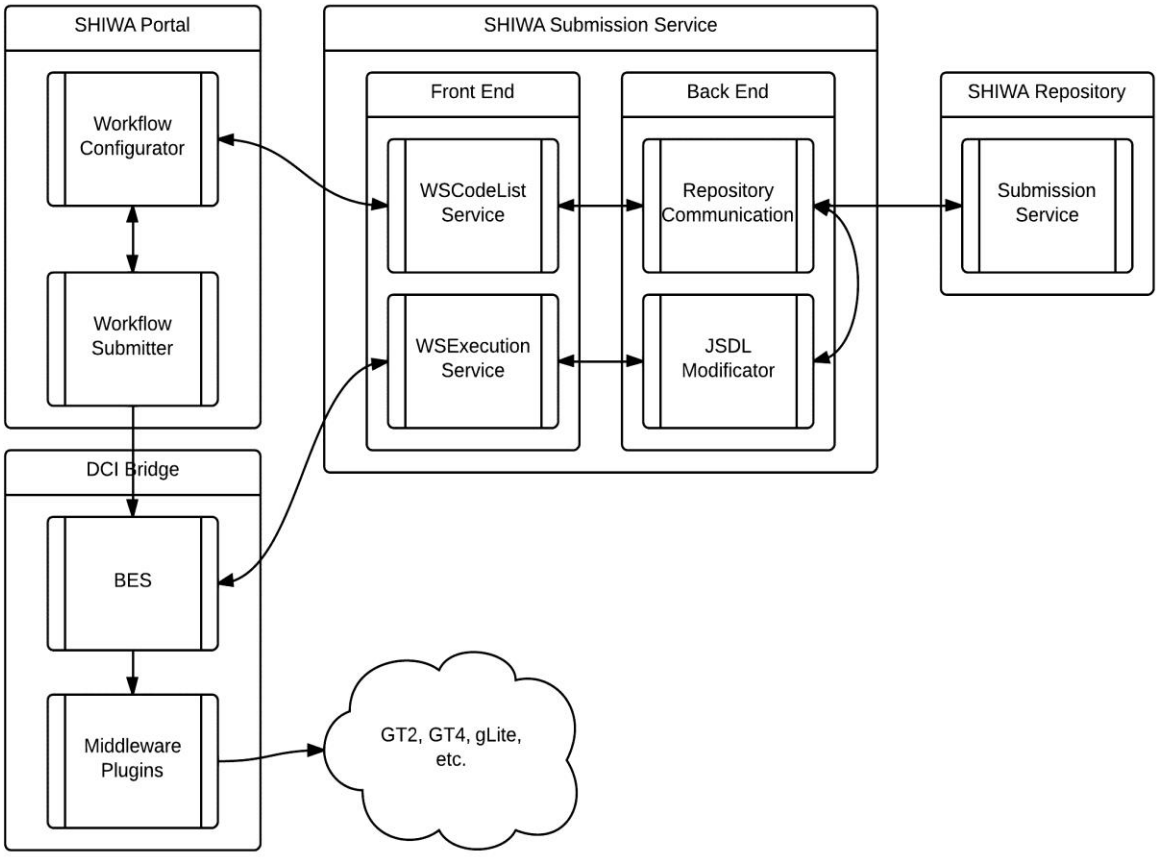

Figure 7.1: SHIWA Submission Service in the SHIWA Simulation Platform

The SHIWA Submission service communicates with the DCI Bridge, the SHIWA Portal and the SHIWA Repository (See Fig. 7.1):

- **Repository – submission service communication**. The *Repository Communication* component calls a web service, called the *Submission Service*, deployed in the SHIWA Repository, to get workflow or workflow engine data or both. This web service retrieves, formats and returns data from the repository to the SHIWA Submission Service.

- **Portal – submission service communication:** The *Workflow Configurator* in the portal retrieves data from the SHIWA Repository through the submission service. The configurator gets information about the workflow from the submission service through the *WSCodeListService* web service. This web service gets:

  - The list of *Submittable Execution Nodes* (SENs) that has been marked as *submittable*, i.e. they are configured to be submitted through a portal.
    Remark: The term SEN is explained in the repository part of this report.
  - The parameters of SENs included on the list, and
  - The resource where the SENs can be submitted.

  DCI Bridge – submission service communication: The Figure 7.2 presents how the submission service has been integrated to the DCI Bridge. Having the workflow data the SHIWA Portal configures the workflow, creates its JSDL file and submits it to the DCI Bridge. The JSDL file contains the description of the configured workflow with all

data and parameters required to execute the workflows such as files location, workflow executables, the middleware used for the submission, etc.

If the workflow is a non-native one, the DCI Bridge forwards the workflow ID and the JSDL file to the *WSExecutionService* of the submission service. The submission service should modify this JSDL file before forwarding it to the DCI Bridge. The *WSExecutionService* modifies the JSDL file according to:

- The SEN configuration within the SHIWA Repository.
- The computing resource chosen where the SENs can be submitted.
- Workflow engine configuration (if the workflow engine has to be deployed or exists on the chosen resource and if extra information are required).

Finally, the modified JSDL is sent back to the DCI Bridge which submits the workflow to the computing infrastructure specified in the JSDL file.

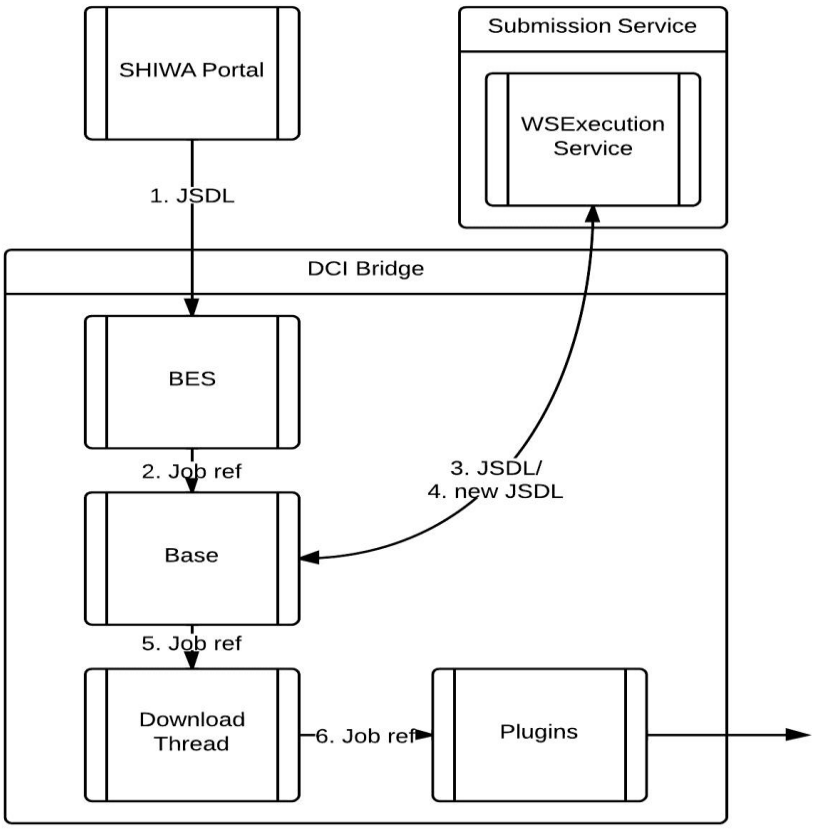

Figure 7.2: DCI Bridge and submission service integration

Usage: MTA-SZTAKI has modified the configuration page of non-native workflows in the gUSE framework. The new configuration page (See Fig. 7.3) has the same structure as the previous configuration page used with the GEMLCA Service. Workflow developers have to specify the following parameters:

- Service type SHIWA service which identifies the SHIWA Submission Service required to execute non-native workflows
- SHIWA Submission Service URL of the submission service which manages execution of non-native workflows
  There could be central (SHIWA Submission Service) and local (community gateway's submission service) submission services.

- SHIWA Repository URL of the workflow repository which stores the non-native workflow
- Submittable Enable Node (SEN) name of the non-native workflow
- Resource VO where the workflow is executed

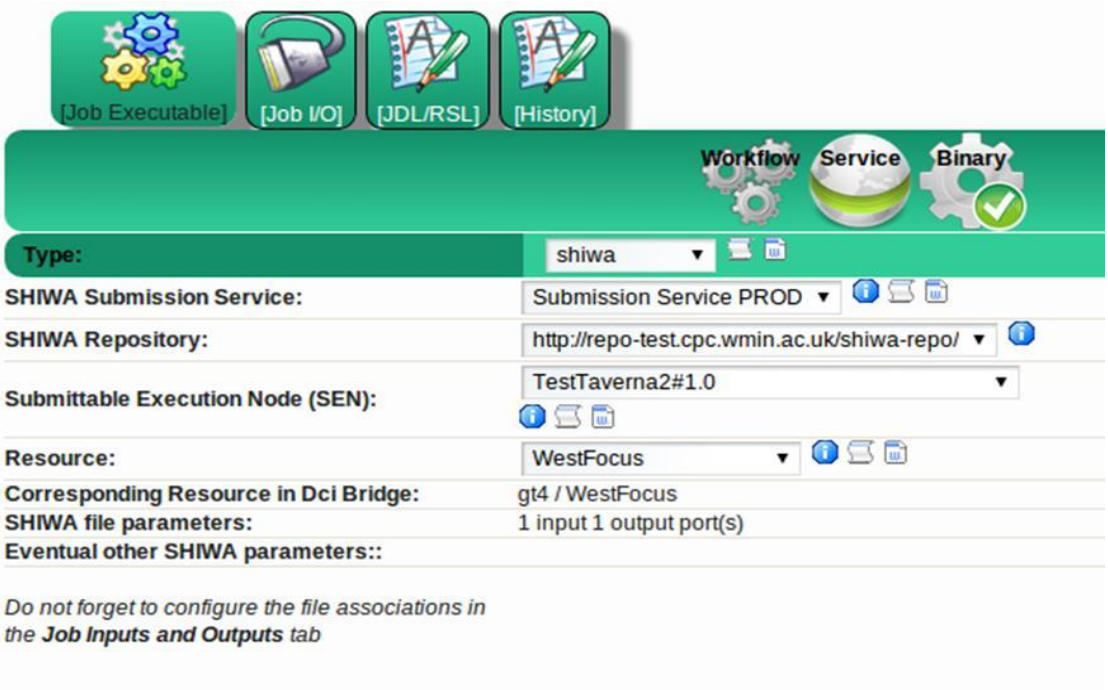

Figure 7.3: Configuration page for non-native workflows

7.1.2 Workflow engine data management enhancement requests: A06, I17

Problem: In the CGI concept each workflow is assigned to a workflow engine in the SHIWA Repository. Workflow developers have to specify the workflow engine and its version as a parameter of the concrete workflow. At the beginning of the ER-flow project the GEMLCA Service managed non-native workflows as legacy applications. This service combines non-native workflows with non-native workflow engines to execute them. The GEMLCA Service has the built-in GEMLCA Repository which handles workflow execution data and non-native workflow engines data (data, metadata, binaries, configuration, dependencies, etc.). Workflow engine developers have to use the GEMLCA Repository Portlet (or GEMLCA Admin Portlet) to specify, upload, edit and delete workflow engine data. As a result, they have to use two GUIs: GEMLCA Repository to manage workflow engine data and SHIWA Repository to handle workflow data.

Design: The aim was integrate the workflow engine management to the SHIWA Repository and to make the GEMLCA Repository obsolete, i.e. to have a single repository in the SHIWA Simulation Platform that is able to manage both the workflows and the workflow engines. The key design requirement was not to have any change in the previous workflow management or at least to minimise these changes. As a result, the extended SHIWA Repository must have the ability to store information regarding the different configurations and settings of workflow engines and their execution environments. Major front-end and back-end changes addressed in the design process include, but are not limited to:

- The ability to create, duplicate or delete a **Concrete Workflow Engine**.
  - This is associated to a **Workflow Engine**. One workflow engine can have many concrete workflow engines (or implementations).
  - A concrete workflow engine has an **owner** and a **visibility**.

- A workflow engine is flagged as **Submission Enabled** if it is supported by the SHIWA Submission Service.

- The ability to **upload files** and associate them to a workflow engine
  - These files can be classified as either **data** or **executable**. This allows the user to specify, in a generic manner, a new workflow engine for On-The-Fly Deployment.
- The ability to define submission and execution middleware known as Back-end **Configurations.**
  - These settings allow the user to also specify, with absolute flexibility, a type of middleware or execution site.
- The ability to **toggle** a concrete workflow (or workflow implementation) as **Submittable** for use in the SSP.
  - This has replaced the **Deploy to GEMLCA** functionality. The operation may have changed but the connotations are the same; Toggling a concrete workflow as Submittable allows the user to access it and use it from the SSP.
  - Before a concrete workflow can be toggled as submittable it must:
    - Be publicly accessibly.
    - Have a properly defined execution node.
    - Have a workflow definition file.
    - Have a domain

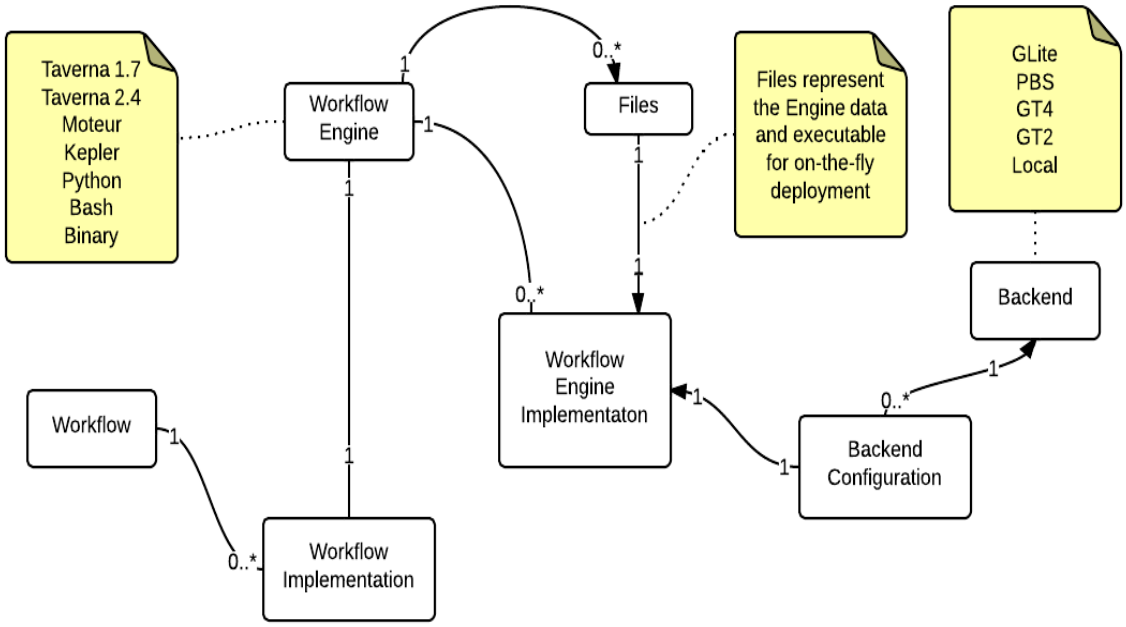

Figure 7.4: Workflow and workflow engine management in the extended SHIWA Repository

Implementation: WP3 extended the SHIWA Repository adding the workflow engine data management as a new functionality. All previous relationships between concrete workflows and workflow engines have been maintained. As a result, if a concrete workflow was execution enabled in the GEMLCA Repository it remained execution enabled in the extended SHIWA Repository. If a concrete workflow was deployed to the GEMLCA Service it is likely (if not certain) that it has a configured execution node, in this case the same concrete workflow can be **Toggled Submittable** by the owner in the repository - the same way one might have deployed it to the GEMLCA Service.

To execute non-native workflows the new SHIWA Submission Service (see Section 7.1.1) retrieves data of the workflow and the workflow engine from the SHIWA Repository and

combines the workflow and the workflow engine to enable the execution of the non-native workflow. The Entity Relation diagram in Fig. 7.5 shows the new additions to the data structure and their relationships. The new tables are: ***we_implementation, workflow_engine_files, be_instance, be_attr, uploaded_file***.

Usage: Workflow engine developers have to define workflow engine data using the SHIWA Repository GUI. They should also upload binaries, configuration and dependency data of workflow engines into the repository through the same GUI. Workflow developers have to select first the workflow engine, and then its version from a drop-down list of workflow engines and to associate a workflow engine with a concrete workflow. They identify the resource where the workflow can be executed. The concrete workflow stores this information in the SHIWA Repository. The SHIWA Portal retrieves this information from the repository. Next, either the SHIWA Submission Service (if the workflow is non-native one) or the WS-PGRADE workflow engine (if the workflow is native one) forwards this information towards the DCI Bridge which submits the workflow to the specified resource.

Two tutorials have been produced and are available through the simulation platform that explain how users can use these new features on the SHIWA Portal and on the SHIWA Repository.

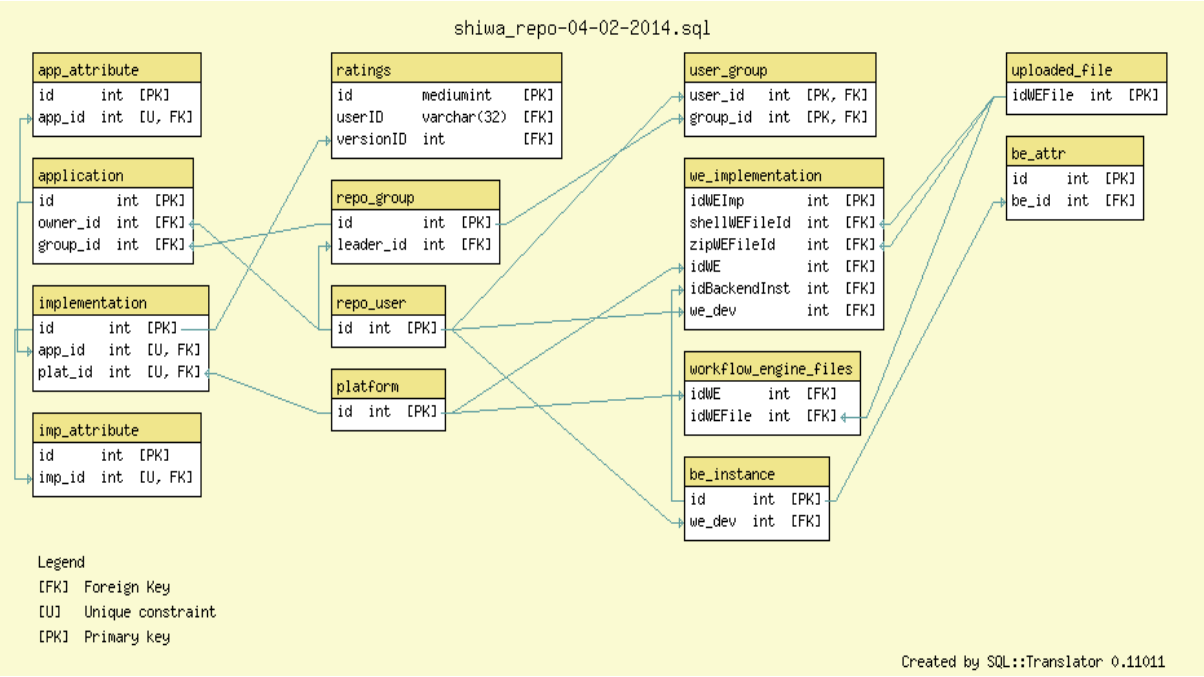

Figure 7.5: Data structure of extended SHIWA Repository

7.2 Minor Upgrades

Access to gLite and UNICORE resources enhancement requests: I18 – I22

Problem: In the SHIWA project the SHIWA Portal provided access to the shiwa-vo. The technology providers offered access to gLite, GT2 and GT4 based Virtual Organisations. The ER-flow research communities want to access further gLite and UNICORE resources. The Astrophysics community wants to use the gLite based astro, inaf and planck Virtual Organisations, while the Life Sciences uses the gLite based vlemed Virtual Organisation to run workflows. The Computational Chemistry community needs access to the UNICORE based resources.

Implementation: WP3 provided access to three gLite based VOs i.e. astro, inaf and planck and the UNICORE based MoSGrid VO. The process to accomplish this task is described below:

Access to Astrophysics VOs:

The Astrophysics community uses three gLite based VOs i.e. astro.vo.eu-egee.org, planck and inaf VO. The process of integrating these three VOs with the SHIWA Simulation Platform involves configuration at two levels i.e. middleware and gateway levels.

- **Middleware Configuration**: The middleware configuration for the three astro VOs included installing and setting up the gLite middleware to enable job submission using the interface provided by the middleware. The gLite middleware was installed and specific configurations for the three VOs were performed. These configurations involved setting up VO specific parameters such as VOMS Server, Distinguished Name etc. A sample middleware configuration for the inaf VO is presented in Figure 7.6.

```
VO_INAF_SW_DIR=$VO_SW_DIR/inaf
VO_INAF_DEFAULT_SE=$SE_HOST
VO_INAF_STORAGE_DIR=$CLASSIC_STORAGE_DIR/inaf
VO_INAF_VOMS_SERVERS="'vomss://voms.cnaf.infn.it:8
443/voms/inaf?/inaf' 'vomss://voms-
01.pd.infn.it:8443/voms/inaf?/inaf'"
VO_INAF_VOMSES="'inaf voms.cnaf.infn.it 15010
/C=IT/O=INFN/OU=Host/L=CNAF/CN=voms.cnaf.infn.it
inaf' 'inaf voms-01.pd.infn.it 15010
/C=IT/O=INFN/OU=Host/L=Padova/CN=voms-
01.pd.infn.it inaf'"
VO_INAF_VOMS_CA_DN="'/C=IT/O=INFN/CN=INFN CA'
'/C=IT/O=INFN/CN=INFN CA'"
```

Figure 7.6: Sample middleware configuration for inaf VO

- **Gateway Configuration**: The gateway configuration involves configuring the DCI Bridge for the gUSE gateway used by the SHIWA Simulation Platform. In order to achieve this task, VO specific configurations were performed with the DCI Bridge which include parameters such as BDII, WMS etc. A sample DCI Bridge configuration for the inaf VO is presented in Figure 7. 7.

The configurations for both the middleware and the gateway for the astro VOs were obtained via direct communication with the administrators for these VOs.

Testing: As with the configuration, the testing for the gLite based astro VOs was performed at two different levels i.e. command-line interface and via the SHIWA Simulation Platform, particularly via the SHIWA Portal as described below.

- **Command Line Interface Tests**: The testing performed using the gLite command-line interface is achieved using the interface provided by the gLite middleware and facilitates direct interaction with the functions provided by the middleware. As compared with gateway based testing, the command-line interface provides more control and flexibility when using the functionality provided by the middleware.
- **SHIWA Simulation Platform Test**: This form of testing involved using SHIWA Portal to configure and run workflows with test jobs.

The overall testing involved submitting hundreds of jobs via both the command-line interface and the SHIWA Portal synchronously. The test jobs included both simple “hello world” jobs and also workflows requiring input/output functionalities. Furthermore, the tests were performed by the Westminster team as well as users from the Astrophysics community.

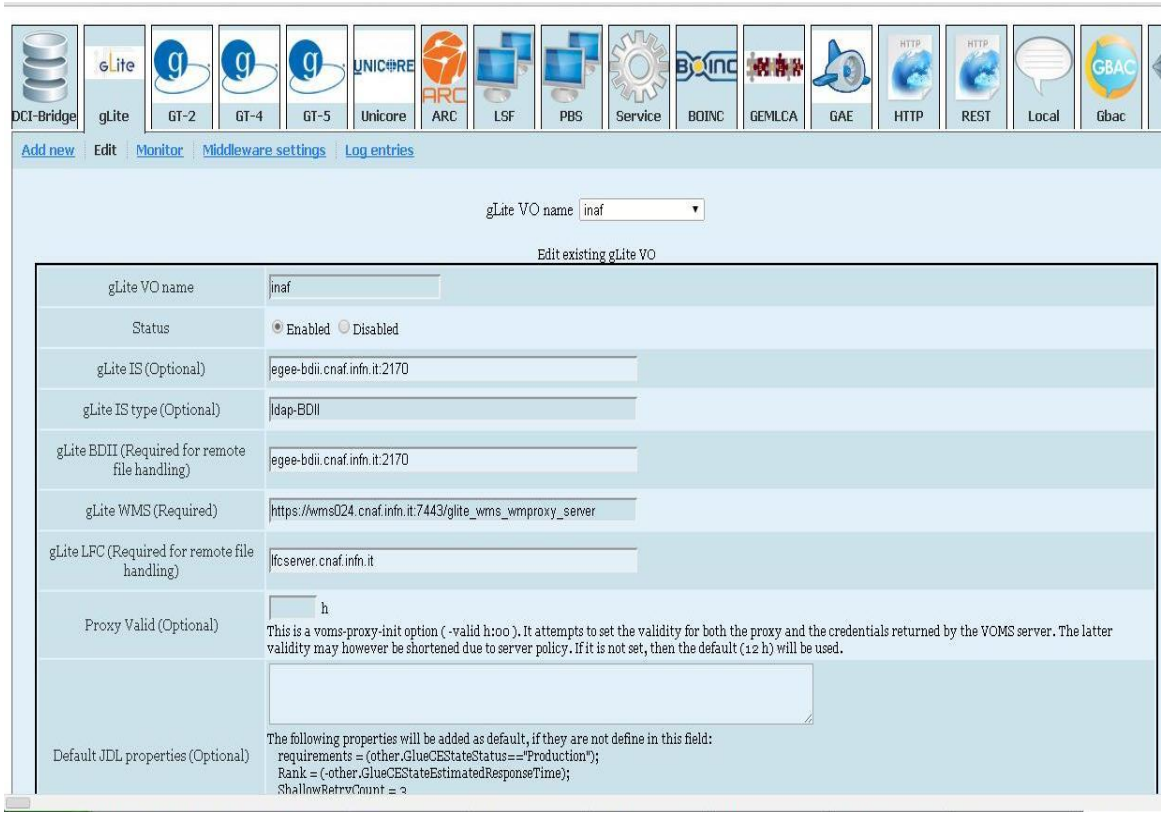

Figure 7.7: Sample Gateway configuration for inaf VO

Test Results: With respect to the results of the above mentioned evaluation, a significant difference was observed in the performance between the command-line interface and the SHIWA Portal. For the initial experiments, the tests for command-line interface reported a success rate of 90% for all three VOs whereas the test results for the SHIWA Portal presented a completely different picture. The success rates for initial tests from SHIWA portal were 50% for astro VO, and less than 20% for both the inaf and the planck VO.

An investigation was conducted in coordination with the technical team at the astro VO. As part of this process the configuration for the three VOs was verified along with rigorous debugging of the resources provided by the three VOs. As a result the second phase of

experiments yielded better performance via the SHIWA Portal with success rate of 80% for the astro VO and 50% for both the inaf and the planck VO. These are encouraging developments but further investigation is still ongoing to achieve a 100% success rate for these VOs.

Problems and issues: During the installation and configuration of the gLite based astro VOs, a number of problems were encountered as explained below:

- **WMS related problems**: As described earlier an important part of the configuration of gLite based resources is the WMS which is then responsible to manage job execution within the gLite environment. However, during the configuration of astro VOs, it was discovered that sometimes one of the WMSs for a specific VO might not respond as expected. Therefore, having the correct configuration for the most reliable WMS is essential. This problem is easily addressed via CLI as users have the options to select a specific WMS and/or CE. Another issue identified is that one VO may have many WMSs however, the gUSE gateway currently only allows to have one WMS for a VO. The ability to configure a primary and multiple secondary WMSs for a VO was reported as an enhancement requests.
- **Proxy Certificate Issues**: The astro VOs use X509 proxy certificates for authentication and authorization purposes. Furthermore, the gUSE gateway used by SHIWA Simulation Platform provides the ability to associate VO specific attributes to a proxy certificate using VOMS extensions. However, our investigation revealed that when a multi-WMS is used as an endpoint to configure the DCI Bridge, the VO specific attributes are not translated successfully resulting in authorization error for the specific user. The problem was addressed using a concrete WMS as an endpoint to configure the DCI Bridge however the feature of "multi-wms" is handy for load balancing and sustainable service.
- **Open Issues**: As explained in the previous section, the current performance of the inaf and the planck VO is not satisfactory with success rates of around 50%. This is primarily due to a very complex erroneous situation currently under investigation. Due to the reasons currently being investigated, the proxy of a user is corrupted in a way that if a user submits a workflow as a user of the inaf VO, the workflow and its associated files land on the VO resource as a planck user thereby denying the execution of the workflow. This problem is currently being investigated in cooperation with the technical team at the VO as well as the technical team at MTA-SZTAKI.

Access to Computational Chemistry VO

The compchem VO provides access to UNICORE based German NGI resources and is primarily used by the scientists from the MoSGrid community. The integration of UNICORE with the SHIWA Portal is performed by configuring the DCI Bridge. These configurations involve VO specific parameters such as the security credentials of the source and destination resource. Figure 7.8 presents a sample DCI Bridge configuration for the UNICORE resource.

Testing: The evaluation of the UNICORE access was conducted via the SHIWA Simulation Platform by the Westminster team as well as a member of the Computational Chemistry community. The testing involved submitting simple jobs for example “hello world” jobs and complex workflows using I/O facilities. The results for these experiments produced a 100% success rate for the UNICORE job submission facility.

Problems and issues: The SHIWA Simulation Platform currently supports both gLite and UNICORE resources. However, in order to provide access to these different types of grid middleware from the same platform has not been trivial. In particular, these grids use different authentication and authorization methods i.e. gLite based resources use X509 certificates whereas the UNICORE resources use SAML assertions. Although the gUSE

gateway distribution claims that it can host both these grid resources from the same portal, our investigation has proved it otherwise. Therefore, a workaround was developed and deployed by using two different Apache Tomcat instances to host two separate DCI Bridges. In this setting, one of the DCI Bridges acts as a slave bridge to the master node. The current deployment of the SHIWA Simulation Platform makes use of this setting to provide access to both the UNCIORE and gLite VOs.

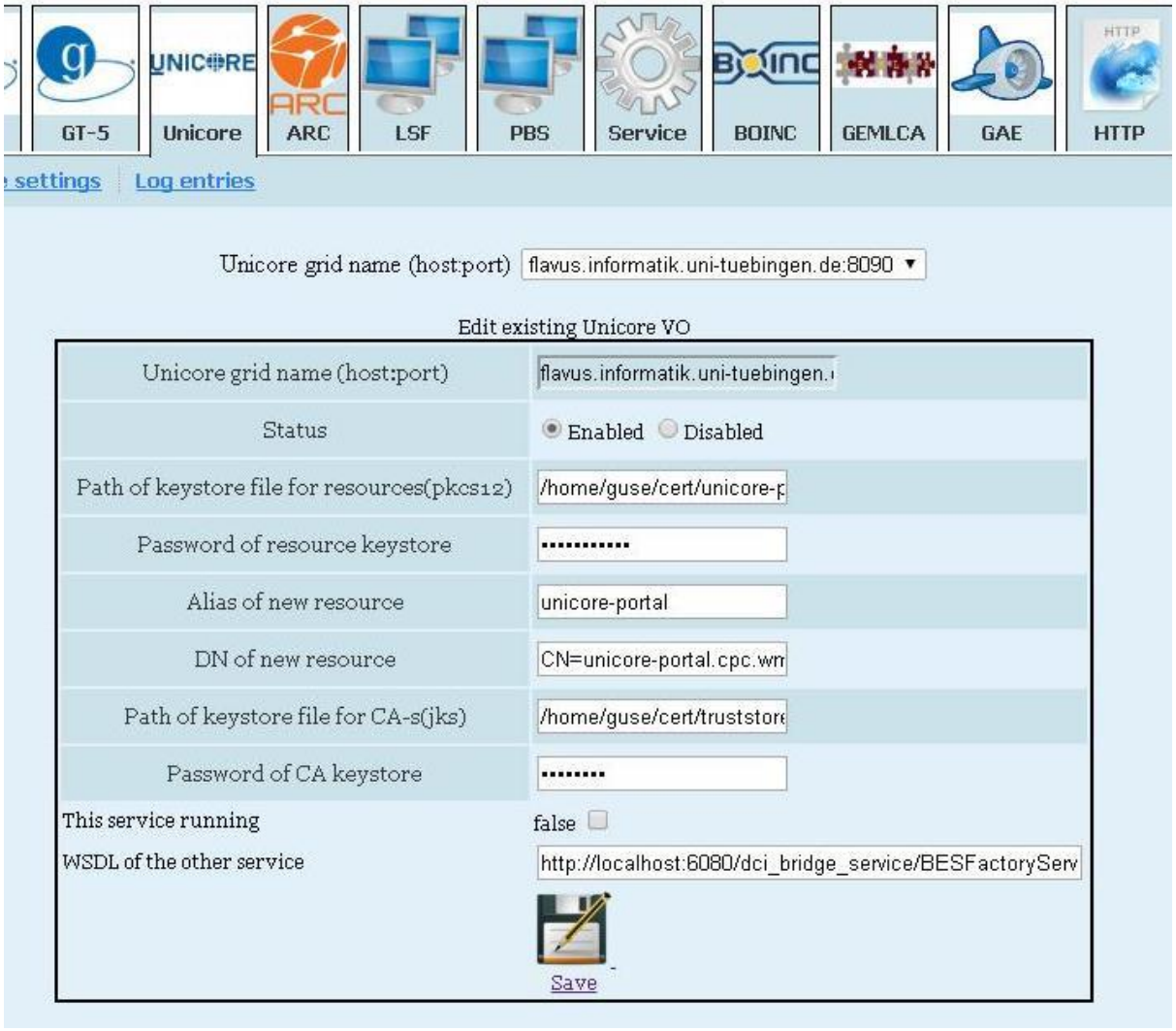

Figure 7.8: Sample DCI Bridge Configuration for the UNCIORE

7.2.1 Minor enhancements of the SHIWA Repository

7.2.1.1 Enhancement requests submitted by WP5

WP5 defined several enhancement requests in D5.1 report and submitted further requests through the SHIWA User Forum. There have been several minor upgrades in the repository, mainly focusing on improving the user-interface:

- upgrade from Primefaces 2.2 to **Primefaces 3.0.1**: source: **SHIWA User Forum**
  - the aim was remove a dependency on Flash based file uploads.

- access to the **SHIWA User Forum** through the repository homepage using an iFrame (Fig 7.9) source: **SHIWA User Forum**

.

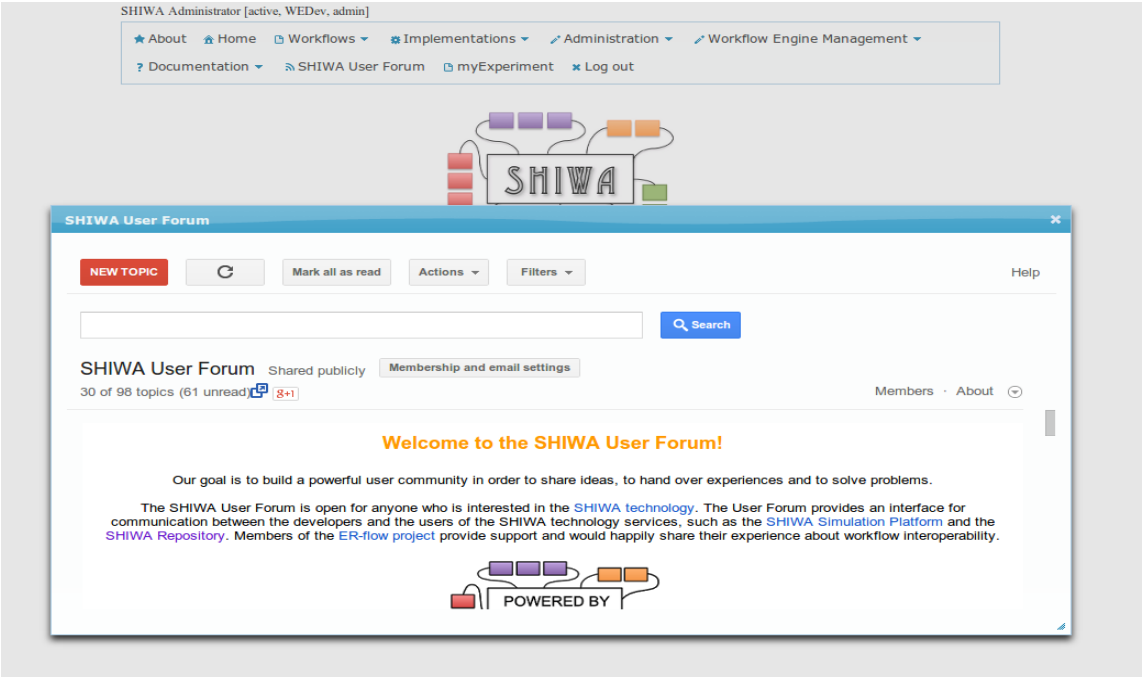

Figure 7.9: Access to the SHIWA User Forum

- adding **help prompts** to the repository GUI to explain content on the repository's pages: source: **SHIWA User Forum**
  - there are a few new attributes, concepts, features and symbols that are used, these help prompts help explain these and give valuable context to the page

.

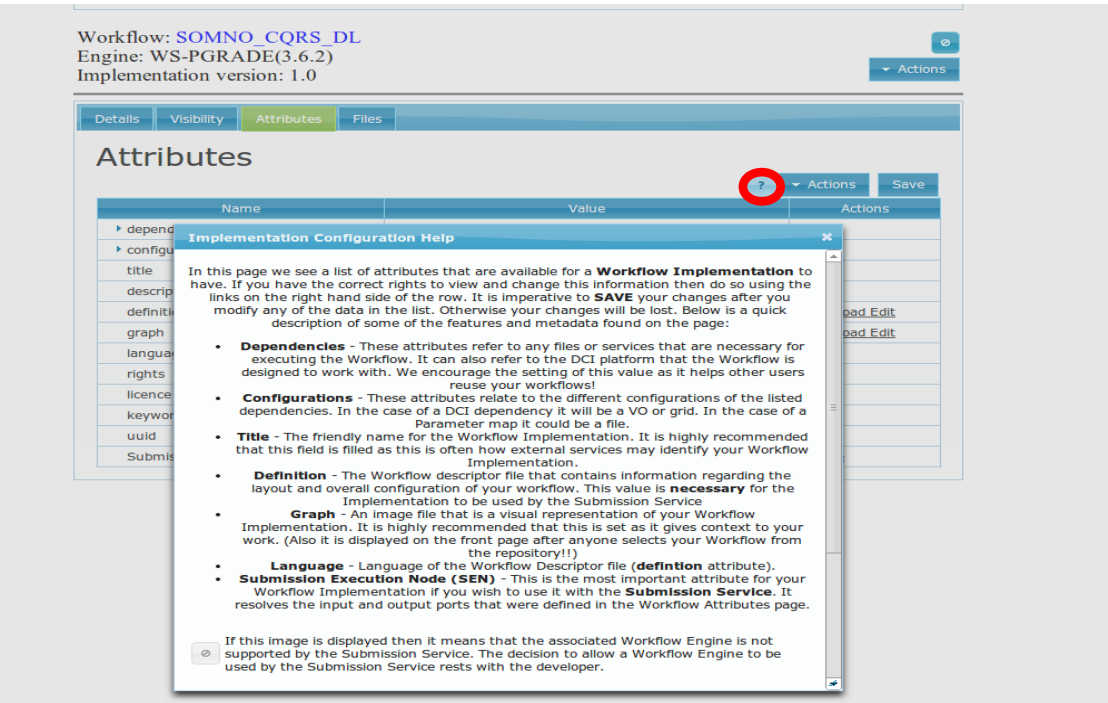

Figure 7.10: Help prompts to explain attributes

- tracking and adding new attributes and data, such as**, tracking user access**, **logging workflow** and **workflow implementation views**: source: **I07**

- this also enables the user to **search** or **sort workflows by popularity** calculating by a workflow having a certain percentage of the total views.

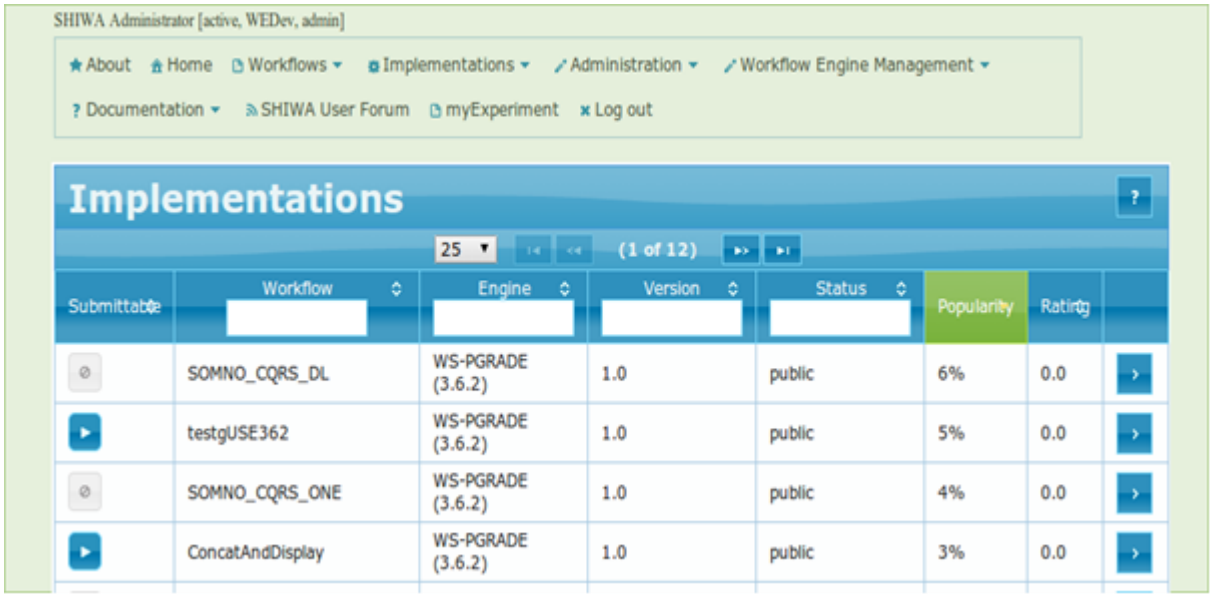

Figure 7.11: Searching and sorting workflows by popularity

- creating a **list of workflows by group** through an adding an extra tab:
  source: **SHIWA User Forum**
  - this allows the user to get an idea of the volume of work produced by a certain group and how it is represented on the repository.

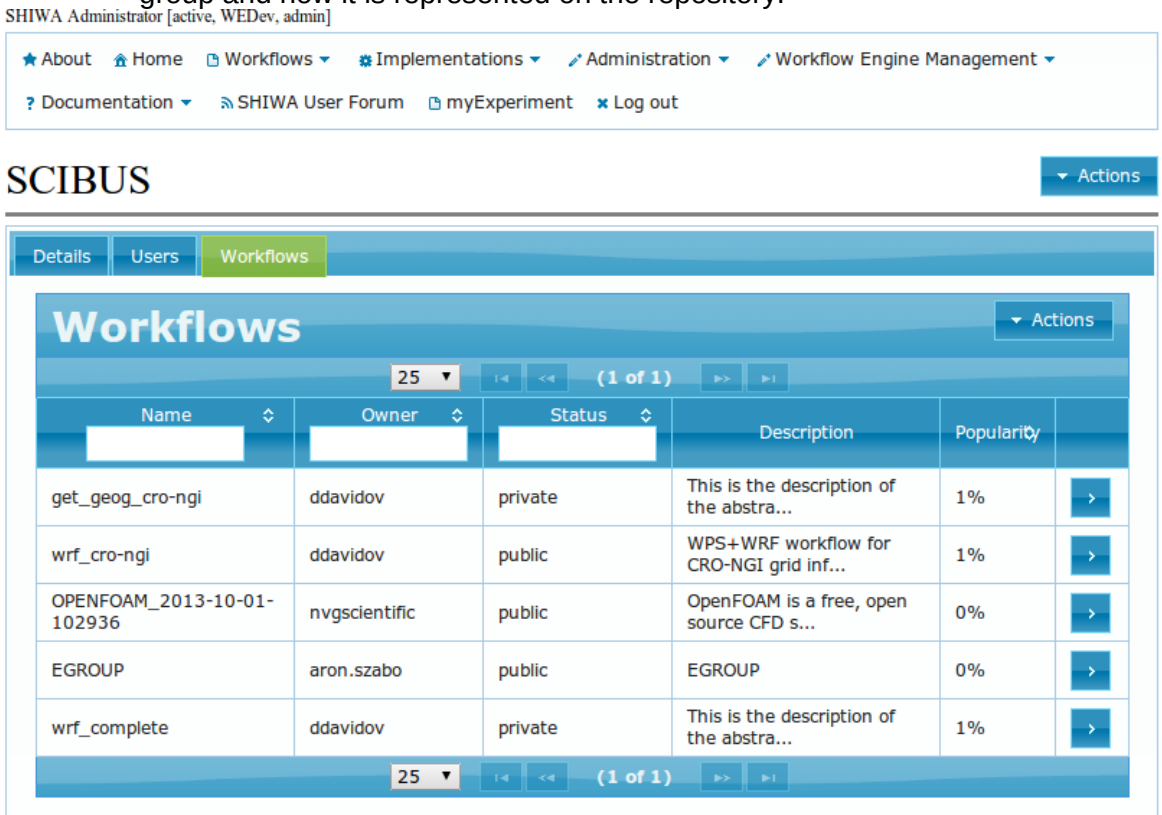

Figure 7.12: Creating a list of workflows belonging to a group

7.2.1.2 ER-FLOW Bugzilla bug reports and enhancement requests on the SHIWA Repository

WP3 consolidated the SHIWA Repository implementing several bugs and enhancements requests reported in the ER-FLOW Bugzilla.. The work package created a new product inside the ER-flow Bugzilla (SHIWA-Repo 3.1 product) and now it is used as the standard product. We present an exhaustive list of the closed/resolved bugs that have been implemented and incorporated to the SHIWA Repository along with the closing statement.

resolved bugs and enhancement requests:

- **bug 05** – in the workflow export operation the workflow name is not being timestamped
  - fixed in the SHIWA Repository 3.1 - workflow name now has timestamp,.
- **bug 06** – in the workflow export operation the image of graph is not being transferred to the SHIWA Repository
  - fixed in the SHIWA Repository 3.1. - workflow graph is uploaded to the repository
- **bug 07 -** error message is thrown when attempting to delete a workflow implementation
  - fixed by raising an exception if the workflow implementation id exists in the implementation table
- **bug 09** - changing displayed names of workflows is not allowed
  - fixed in the SHIWA Repository 3.1 - truncated workflow names are visible in all data tables.
- **bug 12 -** changing workflow name is not allowed
  - fixed in the SHIWA Repository 3.1.
- **bug 15** - changing column name of "status" column from workflow and workflow implementation table and browse view is not allowed
  - fixed in the SHIWA Repository 3.1.
- **bug 16 -** browsing workflow view as public user and searching private workflows
  - fixed in the SHIWA Repository 3.1.
- **bug 17** – there is a workflow name length limit (too short)
  - bug fix deployed as version shiwa-repo 3.0.2 on production server. (Now
- **bug 18** - upgrade PrimeFace version to version 3+
  - fixed in the SHIWA Repository 3.1.
- **bug 19 -** execution parameter default value
  - fixed in the SHIWA Repository 3.1.
- **bug 20** – selecting list of domains from browse workflow & workflow implementation view
  - fixed in the SHIWA Repository 3.1.
- **bug 21** – while importing a workflow from the SHIWA Repository filter avaliable "public bundles"
  - The bug is partially invalid as much of the filtering takes place on the SHIWA Portal. However, due to the nature of the recent upgrade this can be considered resolved. A review should take place to confirm that the current behaviour maps to policy.
  - Note: since the closing of this bug more work has taken place in line with the SHIWA Submission Service and versioning policy and we are confident it is resolved in the SHIWA Repository 3.1.
- **bug 22** - poor visual tabulation in table views
  - fixed in the SHIWA Repository 3.1.
- **bug 23** – workfloow implementation rating - database connection broken
  - fixed by removing the Hibernate API and replacing with eclipseLink in the SHIWA Repository 3.1
- **bug 24** - remove the "Validation" tab from repo tool bar
  - fixed in the SHIWA Repository 3.1.
- **bug 25** - access rights confusion: private implementations of public workflows are visible
  - fixed in the SHIWA Repository 3.1.
- **bug 27 -** introduce logging system

- logging is still problematic, however since new components have been added logging has been implemented. Also tracking user access and views of wf/imps has been added.

closed bugs and enhancement requests:

bug 10 - duplicating workflow implementation and associating the new workflow implementation with another workflow
- due to the constraints of the Submittable Execution Node this is not possible.

bug 26 - exporting workflows from the SHIWA Portal to the SHIWA Repository including empty files
- constraint of the SHIWA Desktop API

All other bugs and enhancement requsts not reported here were either reported invalidly or closed by the reporter before any other action was taken.

7.2.1.3 Enhancement requests to be addressed

These requests are planned developments and they are in the **ER-FLOW Bugzilla** for the **SHIWA-Repo 3.1** product It can be found at: http://bit.ly/L9DWiD or from navigating through the Bugzilla browse lists at: https://bugzilla.cpc.wmin.ac.uk/

short-term requests

bug 31 - update workflow implementation attribute tab

bug 39 - flag a workflow if it has been exported from the SHIWA Portal. It will aid the filtering of the SSP import list and searching through repository contents.

bug 47 - improve workflow and workflow implementation browse view search policy with successive filters

bug 49 - make all public implementations of a workflow private if a public workflow made private

bug 53 - normalise workflow domains and sub-domains
- to enforce and expand on the list of domains currently available through the repository.

long-term requests

bug 32 - when a new workflow implementation is created, its execution node should be pre-filled with default value extracted from the workflow it is created from

bug 51 - enforce the setting of certain fields when creating a workflow or exporting from the SHIWA Portal.
- The aim is to minimise the amount of unsortable and unsearchable data in the SHIWA Repository. If users were to provide a short description or a domain for their workflow before they export it would aid reusability.

bug 52 - improve the DCI dependency management by creating an up-to-date list of available DCIs and install a drop down box for the users to select from.
- the current field is not used by workflow developers -> the plan is to normalise the values that can be entered here.
- it includes **bug 41 -** DCIs listing page at the repository

7.2.2 CGI enabling the DISPEL workflow engine enhancement request: I21

Problem: The Seizmology community in the VERCE project deployed a gUSE based community gateway. The community runs simulations that produce synthetic seismograms for Earth models that are compared to seismograms of previous earthquakes The simulation combines a pre- and post-processing phase. The community implemented the pre-processing phase as WS-PGRADE workflows those executed through the VERCE community gateway. The post-processing of the simulation results is executed by DISPEL

workflows. The VERCE community gateway uses the WS-PGRADE workflow system as the host native workflow system. It is not able to run DISPEL workflows. As a result, the outputs of the pre-processing phase must be transferred to the DISPEL workflow system to complete the simulation.

Implementation: To improve the simulation first, the CGI support should be extended for the DISPEL workflow system. Secondly, the pre-processing WS-PGRADE workflows should be combined with the post-processing CGI enabled DISPEL workflows in meta-workflows. WP3 elaborated the CGI support for the DISPEL workflow system. The work package tested this CGI support using two sets of test workflows. WP3 has deployed a prototype CGI support for the DISPEL workflow system inside the Westminster domain.

Usage: Currently, WP3 is testing the DISPEL CGI support with seizmology workflows.

7.2.3 SHIWA Repository clean-up enhancement request: I24

Problem: The SHIWA Repository contains five types of workflows:

- community production workflows: These workflows are the final outputs of the workflow development. They can be considered as production workflows and researchers can use them. Workflow developers can declare these workflows either private or public.
- community test workflows: Workflow developers create a few workflow implementations in the development phase. These workflows are private workflows.
- platform test workflows: WP3 developed workflows to test the SHIWA Portal, the SHIWA Repository, the SHIWA Submission Service and the whole SHIWA Simulation Platform itself.
- trainee's workflows: Participants of the community and project training events create native, non-native and meta workflow.
- demonstration workflows: WP3 elaborated five demonstration workflows: native workflow, non-native workflow, meta workflow, black-box based meta workflow and white box based meta workflow. These workflows are also used at the training events to explain participants how to create their own workflows.

The SHIWA Repository v3.0 stored 217 public and 119 private workflows. WP3 analysed the usage of these workflows and made three conclusions. First, the trainees' workflows are native, non-native and meta workflows created during Tutorial 1, Tutorial 2 and Tutorial 3, respectively. Second, the trainees do not use these workflows after the training events. Thirdly, platform test workflows can be categorised into SHIWA Portal, SHIWA Repository, SHIWA Submission Service and SHIWA Simulation Platform tests. There are many similar test workflows in each category.

Implementation: WP3 compiled a list of workflows available in the SHIWA Repository and assigned them to the five types listed above. WP3 and WP5 checked this list and identified those workflows they need. They decided to remove all training workflows and large number of platform testing workflows. WP3 archived all the workflows stored in v3.0. The work package transferred 136 public and 99 private workflows to the upgraded SHIWA Repository of version 3.1.

8 Coarse-Grained Workflow Interoperability in ER-flow

WP3 created the SHIWA Submission Service and extended the SHIWA Repository to improve support for the CGI concept. These two services enabled the ER-flow Execution Environment to run non-native workflows (See Section 8.1). Previously there was only one usage scenario to execute non-native workflows. This scenario was based on the SHIWA Simulation Platform. The extended SHIWA Repository and the new SHIWA Submission Service enables running of non-native workflows through community gateways (See Section 8.2.). To further extend workflow execution options WP3 investigated how to run workflows on the cloud. (See details in Section 8.3).

8.1 ER-flow Development and Execution Environment

ER-flow defined two environments in Year 1:

- ER-flow development environment with power user view and
- ER-flow execution environment with end user view and/or ASM portlets.

8.1.1 ER-flow development environment (Fig. 8.1)

The development environment is the SHIWA Simulation Platform (SSP). It contains a portal (SHIWA Portal), a submission service (SHIWA Submission Service), a workflow repository (SHIWA Repository), and a proxy server (SHIWA Proxy Server) to support the Coarse-Grained Interoperability (CGI) concept.

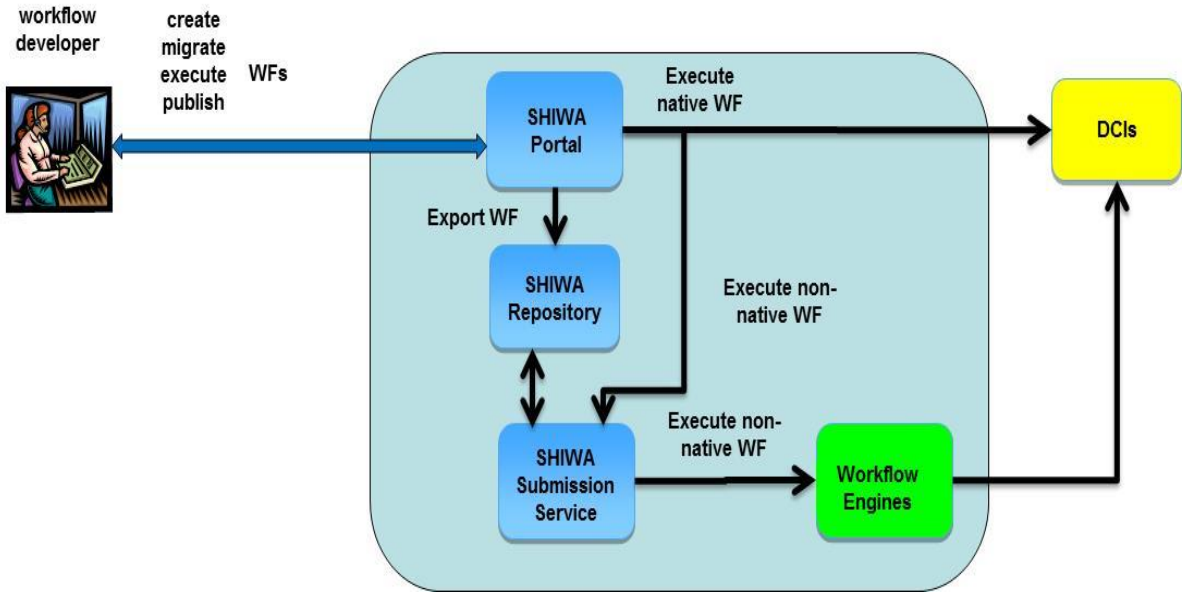

Figure 8.1: ER-flow Development Environment

Workflow developers can create and execute native (WS-PGRADE), non-native (Kepler, MOTEUR, Taverna, etc.) and meta-workflows in this environment. The WS-PGRADE workflow engine of the SHIWA Portal submits directly native workflows while non-native workflows are executed through a submission service (GEMLCA Service in Year 1 and SHIWA Submission Service in Year 2) and a non-native workflow engine.

8.1.2 ER-flow Execution Environment

The Astrophysics, Computational Chemistry, Heliophysics and Life Science community deployed gUSE based community gateways within the framework of the SCI-BUS project in Year 1 to develop and run WS-PGRADE workflows (See Fig. 8.2). The gateways were remotely connected to the SHIWA Repository to enable exporting and importing workflows. However the community gateways are able to manage only native WS-PGRADE workflows because they were not connected to the GEMLCA Service. The ER-flow communities

expressed their interest in running non-native workflows through the community gateways. To address this interest WP3 extended the SHIWA Repository and developed the SHIWA Submission Service.

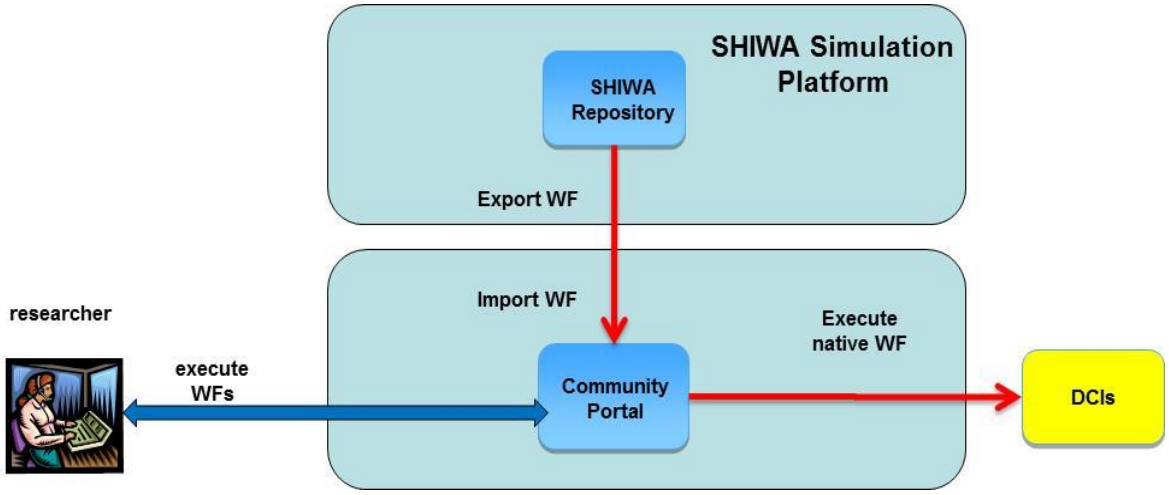

Figure 8.2: ER-flow Execution Environment in Year 1

Having the SHIWA Submission Service researchers and workflow developers can submit non-native workflows through community gateways. (See Fig 8.3).

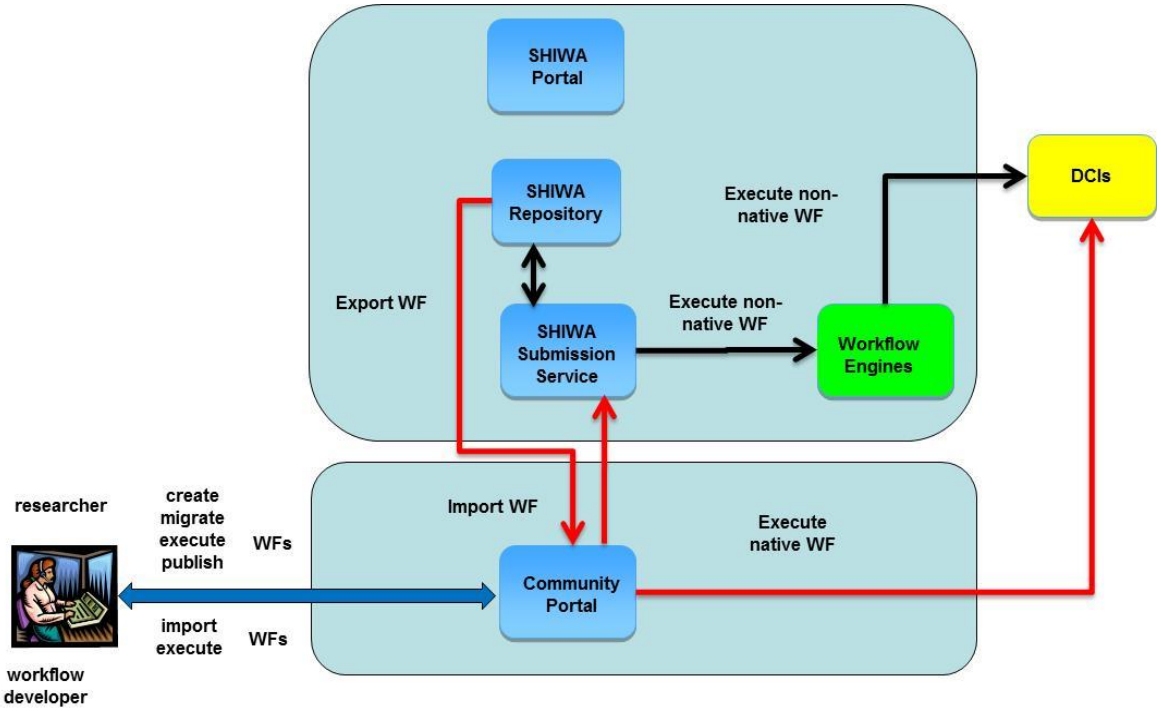

Figure 8.3: ER-flow Execution Environment in Year 2

8.2 ER-flow Usage Scenarios

The GEMLCA Service supports execution of non-native workflows through the SHIWA Simulation Platform (or the ER-flow Development Environment). ER-flow communities deployed gUSE based community gateways (or portals). They wanted to run non-native workflows through these gateways. WP3 considered providing remote access from these gateways to the GEMLCA Service but its GT4 dependency was a major obstacle. The work package designed and implemented the new SHIWA Submission Service to enable execution of non-native workflows through the community gateways. Having the new

submission service WP3 defined three usage scenarios to enable execution of non-native workflows through the community gateways and the simulation platform:

- scenario 1 (Fig. 8.4): Researchers develop and run non-native workflows on the SHIWA Simulation Platform using its services as local services.
- scenario 2: (Fig 8.5) Research communities deploy and manage their own community gateway which is remotely connected to the SHIWA Repository and the SHIWA Submission Service. They use the SHIWA Submission Service of the simulation platform to run non-native workflows.
- scenario 3: (Fig. 8.6) Research communities deploy and run their own community gateway and a submission service which are both connected to the SHIWA Repository. The SHIWA Repository is used as a remote service and the submission service as a local one.

WP3 deployed a prototype SHIWA Simulation Platform consisting of the extended SHIWA Repository (See details in 7.1.2) and the new SHIWA Submission Service (See details in 7.1.1) in March 2014. The work package ran the prototype platform in parallel with the production platform for one month. WP3 switched off the previous production platform and replaced it with the prototype platform in April 2014.

WP3 created the virtual images of the SHIWA services, i.e. of the SHIWA Repository, of the SHIWA Portal and of the SHIWA Submission Service using Juju and deployed all these services on the Westminster cloud to further improve the availability of the simulation platform. As the next step the work package will create the virtual image of the whole simulation platform to support its one-click deployment.

8.2.1 ER-flow usage scenario 1: running non-native workflows from the SHIWA Portal through the centrally deployed SHIWA Submission Service (Fig. 8.4)

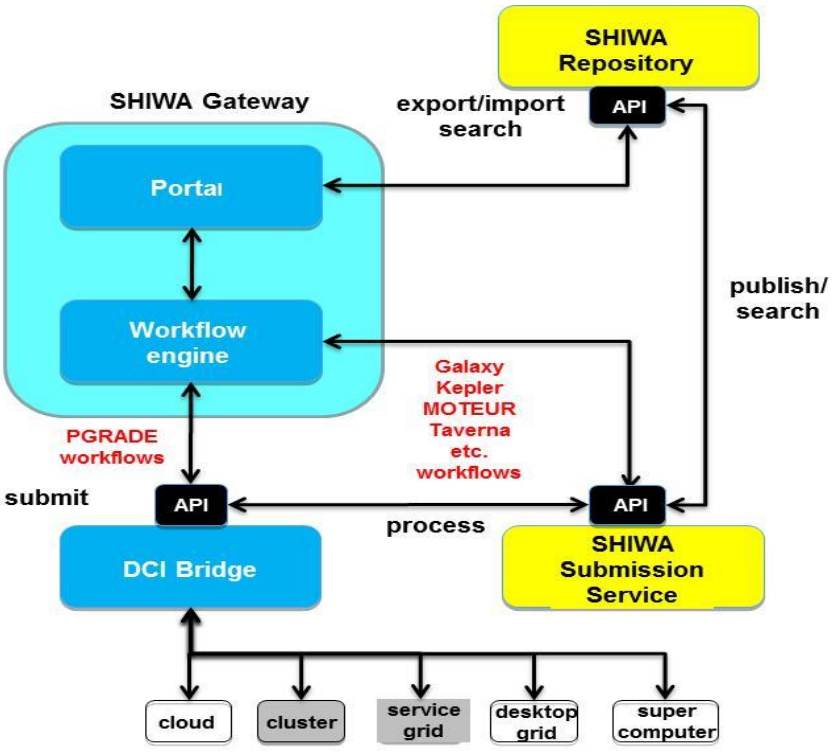

Figure 8.4: ER-flow usage scenario 1

This usage scenario uses only the SHIWA Simulation Platform. Researchers and workflow developers run non-native workflows accessing the SHIWA Repository and the SHIWA Submission Service of the simulation platform through the SHIWA Portal. Research communities without community gateways can use the simulation platform as both development and execution environment. Research communities with community gateways can use the simulation platform for developing and testing new features and services for example connecting a community gateway to new computing resources, creating the CGI support for a workflow system, etc.

WP3 recommends scenario 1 communities that have just started using the workflow technology and do not have either expertise or staff to deploy and manage community gateways.

8.2.2 ER-flow usage scenario 2: running non-native workflows from the community gateway through the centrally deployed SHIWA Submission Service (Fig. 8.5)

In this usage scenario researchers and workflow developers use the SHIWA Repository and the SHIWA Submission Service of the SHIWA Simulation Platform as remote services and their community gateway as a local service. They have to configure their gateway to enable remote access to the SHIWA Repository to perform workflows export and import operations, and access to the SHIWA Submission Service to submit non-native workflows. They can use the SHIWA Simulation Platform as the ER-flow Development Environment to create and test workflows and their community gateway as the ER-flow Execution Environment to run the workflows accessing the repository and submission service remotely.

The advantage of this scenario is that the communities should not deploy and manage the SHIWA Submission Service. Its disadvantage is that the DCI Bridge of the SHIWA Portal should provide access to all computing infrastructures used by the research communities.

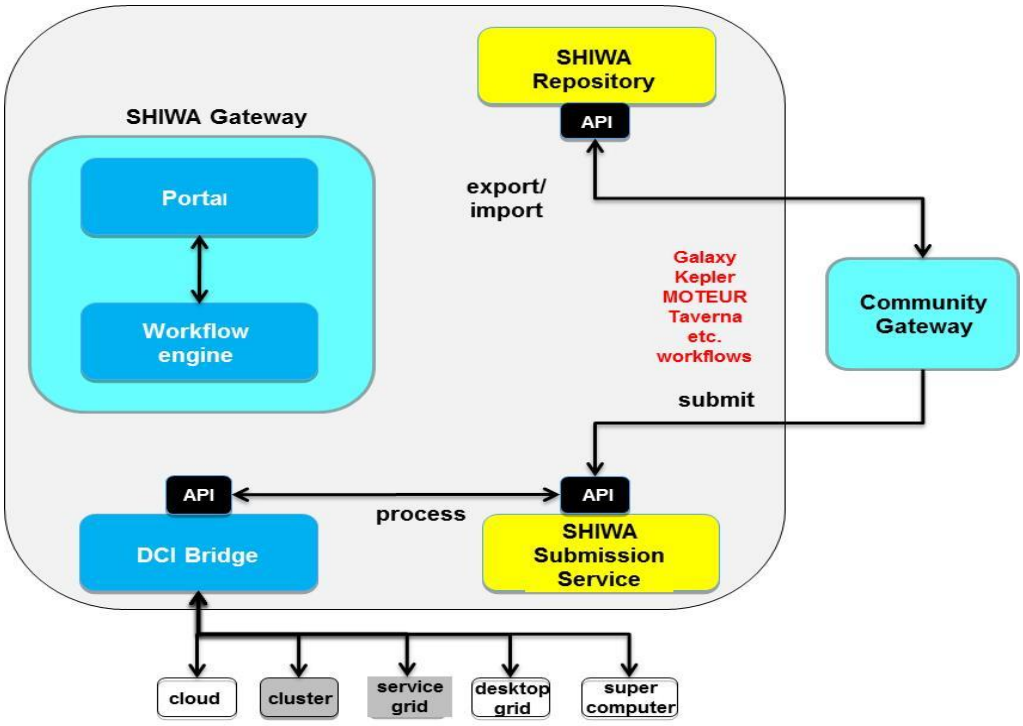

Figure 8.2: ER-flow usage scenario 2

8.2.3 ER-flow usage scenario 3: running non-native workflows from the community gateway through the locally deployed submission service (Fig. 8.6)

In this scenario the communities must deploy and connect the SHIWA Submission Service to their community gateway. Researchers and workflow developers use the community gateway as both development and execution environment. They need access remotely only to the SHIWA Repository to download and upload workflows.

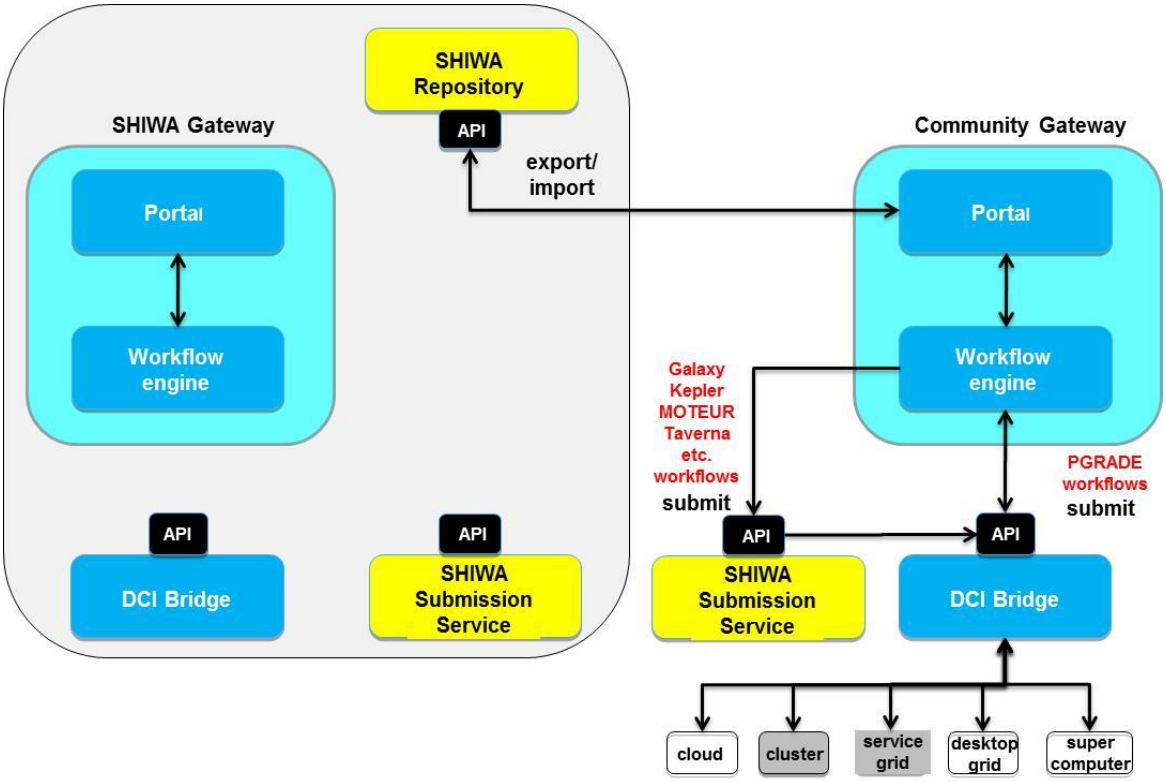

Figure 8.6: ER-flow usage scenario 3

The advantage of this scenario is that the community has a full control over the development and execution environment excluding the SHIWA Repository.

8.3 Running Workflows on the Cloud

The availability and reliability of service grid resources has its own limitations. If the resources on which the workflow is executed is out of service the workflow fails. To address this challenge WP3 investigated how to execute workflows on the cloud. The coming sub-sections present the output of this investigation.

8.3.1 Direct cloud submission (Fig. 8.7)

The SHIWA Portal can use the DCI Bridge to provide a direct access to an Amazon EC2-based cloud. This solution introduces a master-slave DCI Bridge concept to access the cloud. An image containing a slave DCI Bridge is created and uploaded in the cloud images repository. When a submission is requested, the master DCI Bridge contacts the Amazon EC2 frontend service and a virtual machine is initialized using the stored image. Once launched, the master DCI Bridge can contact the slave bridge and the job is submitted locally on the slave virtual machine.

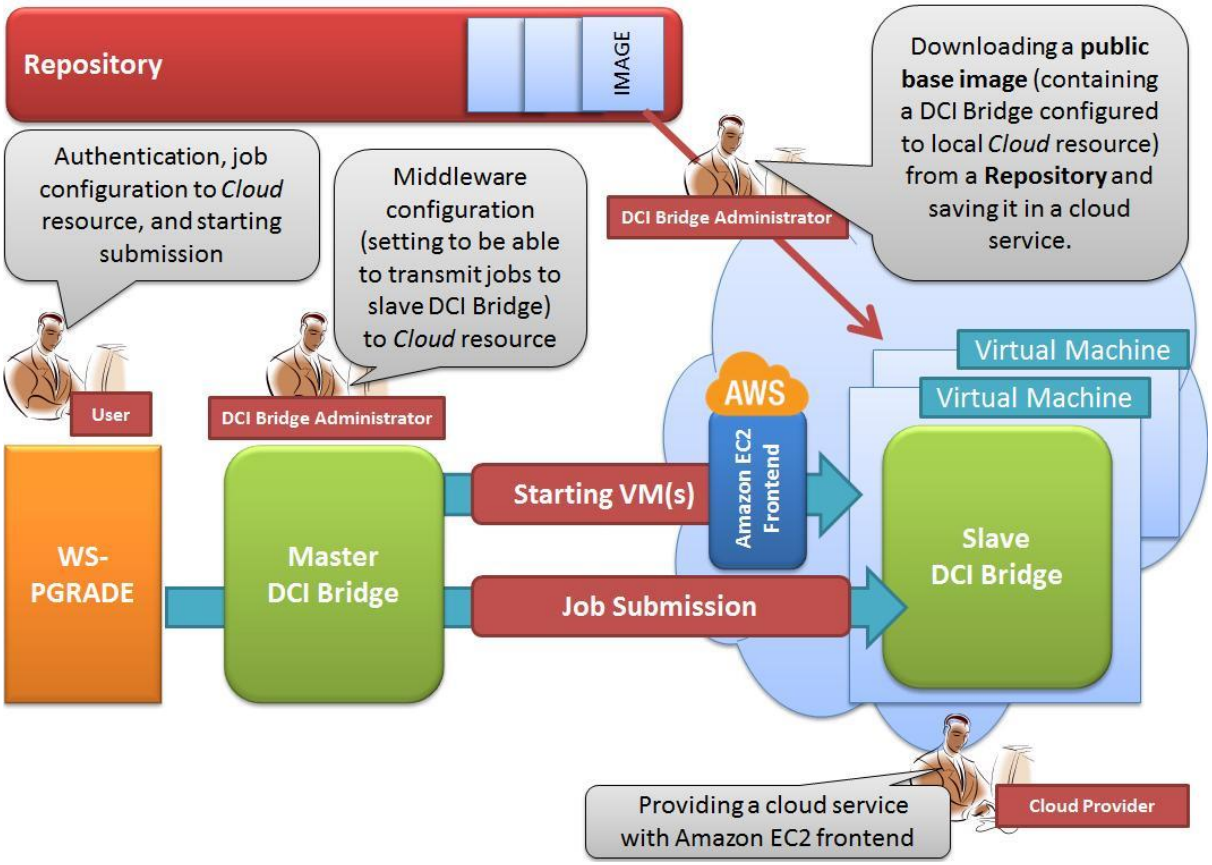

Figure 8.7: Overview of the direct cloud access process

8.3.2 Indirect cloud submission (Fig. 8.8)

The idea behind the indirect cloud submission is to use a brokering platform to submit a workflow to a cloud. This platform is the CloudBroker. It is an application store for high performance computing (HPC) applications in the cloud. Through its API the portal is able to submit workflows to the cloud using the DCI Bridge as submitter.

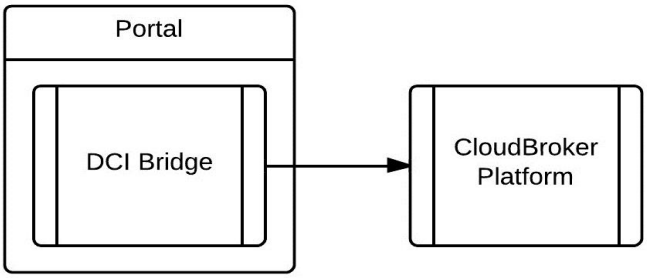

Figure 8.8: Overview of the indirect cloud submission

The Westminster Cloud has been added as a cloud resource to the CloudBroker. WP3 configured a gUSE gateway to handle the indirect cloud submission with this technology. Currently, there is a bug on the portal side when a submission is done on the Westminster Cloud. The DCI Bridge cannot run in normal mode, each workflow submitted returns an error at the end due to some files that could not be deleted. A workaround is to run the DCI Bridge in debug mode preventing files to be deleted but generating a huge amount of data at every submission.

8.3.3 Juju-based submission (Fig. 8.9)

Juju is a project frontend by Ubuntu and Canonical. Its main focus is to provide a generic solution to cloud orchestration and a platform for deploying and developing services or software for example Hadoop, Liferay, MySQL, Wordpress, etc. on to cloud resources in a dynamic and scalable fashion. Such services or software are known as “charms” and are referenced in an official catalogue to be reused by the community. The Westminster team have configured and deployed a Juju environment on the Westminster cloud infrastructure. This consists of a Juju controller machine and number of virtual instances that represent deployed components. Juju offers a frontend (simply known as the Juju-GUI) that provides a drag-and-drop interface to further simplify the service deployment and configuration. Juju also has a well-defined API which has prompted investigation into a DCI Bridge plugin for Juju, allowing more flexibility in services or infrastructures deployment aiming single-click workflow submission to the cloud. The DCI Bridge plugin would contact a python web service that would manage virtual machines deployment, workflow execution, monitor functions, results gathering and virtual machines lifetime.

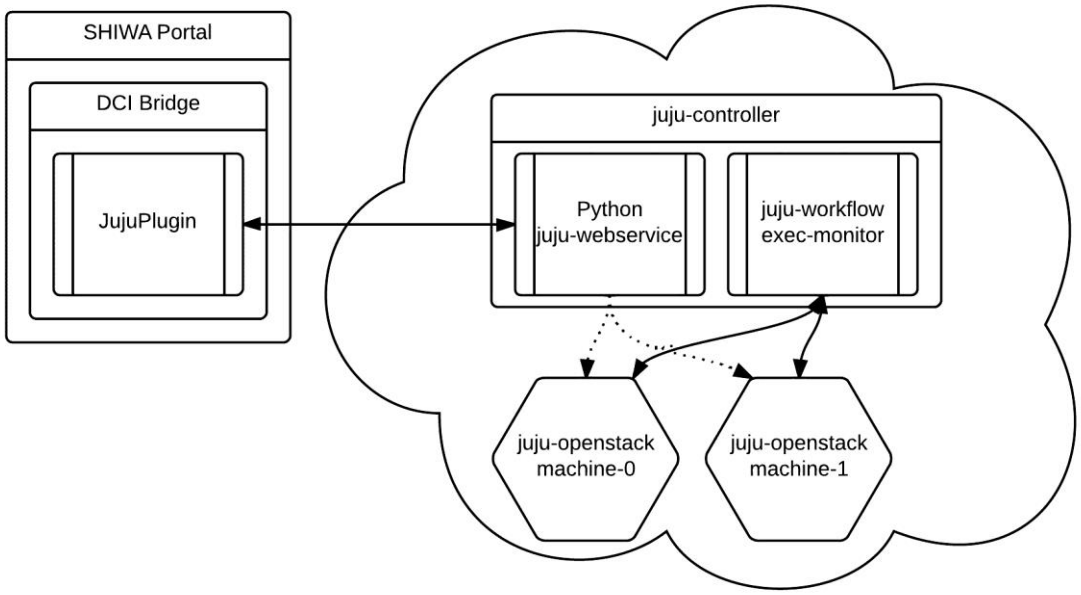

Figure 8.9: Juju-based workflow submission

9 Conclusion

WP3 implemented three major and six minor enhancement requests in the period of September 2013 and March 2014. The work package also fixed seventeen bugs reported in the ER-flow Bugzilla. We want to emphasize that the ER-flow Bugzilla and the SHIWA User Forum established and provides efficient communication among the ER-flow technology providers and the ER-flow research communities. Both technology providers and workflow developers used to these communication channels and they raise and discuss issues through these channels on a regular basis.

As a result of the bug fixes and enhancements both the ER-flow Development and the ER-flow Execution environment provides more advanced and better features and services than previously. The major achievement was the extension of the SHIWA Repository to enable management of both workflows and workflow engines and the implementation of the SHIWA Submission Service to improve execution of non-native workflows. Having these services WP3 created a prototype simulation platform in March 2014. The work package replaced the previous production platform in April 2014 with the prototype platform and created v5.0 of the SHIWA Simulation Platform after one month-long test period. To improve availability of the simulation platform WP3 created virtual images of all major SHIWA services, i.e. of the SHIWA Portal, of the SHIWA Repository and of the SHIWA Submission Service, and runs these services on the Westminster cloud. As the next step the work package will create the virtual image of the whole simulation platform to enable its one-click deployment and launching.

WP3 defined three usage scenarios to support execution of non-native workflows: one for the SHIWA Simulation Platform and two for the ER-flow – SCI-BUS community gateways. The last two scenarios allow execution non-native workflows through an environment which is under community control.

ER-flow communities prefer running their workflows on gLite and UNICORE based resources. The communities had several problems running workflows on these resources. For example several virtual organisations were switched off during the ER-flow project, several virtual organisation had availability and reliability issues, etc. WP3 also started an investigation how to execute workflows on the cloud. The work package investigated direct and indirect workflow execution on the cloud and experimented with the Juju based workflow deployment on the cloud.

References

[1] SHIWA project: www.shiwa-workflow.eu

[2] SCI-BUS project https://www.**sci-bus**.eu

[3] ER-flow Bugzilla https://bugzilla.cpc.wmin.ac.uk/

[4] SHIWA User Forum https://www.erflow.eu/shiwa-user-forum

[5] ER-flow D5.1 report https://documents.egi.eu/public/ShowDocument?docid=1856

[6] ER-flow D3.2 report https://documents.egi.eu/public/ShowDocument?docid=1859