

7 WS-PGRADE/gUSE and Clouds

Zoltán Farkas, Ákos Hajnal, and Péter Kacsuk

Abstract. Cloud computing is becoming more and more popular compared to grid computing due to its virtualization capabilities, flexibility and elasticity. In order to keep up with this trend, WS-PGRADE/gUSE has been extended with the possibility to use cloud infrastructures in workflows, either directly as described in Chap. 4 or via the CloudBroker Platform service. In this chapter we present the integration of WS-PGRADE/gUSE and the CloudBroker Platform, which is particularly useful in the case of accessing commercial clouds and/or using preregistered applications in commercial and academic clouds.

7.1 Introduction

In this chapter we present the integration of WS-PGRADE/gUSE with the CloudBroker Platform (CBP) [CBP] to enable usage of cloud computing resources in WS-PGRADE/gUSE-based scientific workflows. The advantage of using the CloudBroker Platform is that it offers a service capable of hiding the differences between using different types of cloud infrastructures, and thus provides a simplified method to connect different computing infrastructures to WS-PGRADE/gUSE. Another benefit of the integration is that it enables Software-as-a-Service-like (SaaS) and Infrastructure-as-a-Service-like (IaaS) cloud usage from the users' point of view, and fits completely into the philosophy how WS-PGRADE/gUSE uses the different computing infrastructures. Additionally, the integration can transparently be used by the different customization methodologies offered by WS-PGRADE/gUSE, like the end user view, the Remote API, or the ASM API. A further advantage of using CBP appears in the case of commercial clouds, where the cost of running a workflow in the cloud can be estimated and the actual final cost can be provided via CBP. This feature is not available in the direct cloud access solution described in Chap. 4.

Cloud infrastructures can be accessed not only through the CloudBroker Platform, but directly as well, using their EC2 interfaces. In this chapter we briefly introduce the features of the CloudBroker Platform, so readers get a basis for the following part, in which we present the integration of WS-PGRADE/gUSE and its features.

7.2 The CloudBroker Platform

The CloudBroker Platform offers a generic interface to run a predefined set of software on resources offered by different cloud providers. That is, CBP offers the SaaS service model of cloud services. It currently supports Eucalyptus [EUCA], Amazon EC2 [Amazon], IBM SmartCloud [IBMCloud], OpenStack [OpenStack] (using EC2 and Nova interfaces), and OpenNebula [OpenNebula] computing resources.

The main entities in the CBP are users, storages, resources, software, accesses, and pricing. Every action performed in the CBP happens with an authenticated user entity. The users may have different roles: standard (only allowed to run jobs), advanced (standard roles extended with access management), and admin (advanced roles extended with new resource definition and user management). Each user belongs to an organization that is managed by the organization administrator.

Various storages are used to store input and output files of jobs executed through the CBP. Currently Amazon S3 [S3], Nirvanix, Walrus, and Rados S3 storages are supported. Resources reflect accesses to different types of computing resources, like an Amazon EC2 account or an OpenNebula-based cloud provider access. Each resource makes use of Storage access to store data of jobs run in the given resource.

In order to provide the SaaS model, the set of usable software and executables have to be defined in the CBP. The given software may have a number of versions with a set of executables offered. For example, the Gromacs software package consists of at least 50 different executables in the CloudBroker Platform. In the following, we will refer to a CBP software and CBP executable pair as a CBP application. In order to have some CBP application available on a given resource, a deployment belonging to the selected Resource must be defined. A deployment can be produced by following two methods: either a deployment script with a package should be provided that will perform the installation using a base image, or it is possible to define an existing virtual machine image that already contains the software package.

Each user may or may not have access to CBP resources and applications. Only users with access to a given application and resource may run the selected Application on the selected resource. Beside accesses, visibility of resources, applications, and storages can be multiple: private (only the user who defined the entity may use it), protected (users belonging to the organization of the user who defined the entity may use it), and public (every user of the CBP may use it).

In the CBP it is possible to define prices for the usage of the different entities, for example, the execution cost of one Gromacs job may include a \$1 one-time fee and \$0.25 for each CPU hour consumed by the job. All these costs appear at the organization level, meaning every user's spending will be charged against the balance of the organization to which he or she belongs. The operators of the CBP ask for a fixed percentage of these prices. Of course, in case of academic or local

cloud infrastructures, defining such prices is not necessary at all; thus using such resources can be completely free for the users.

Access to the CBP is possible at different levels: the web interface, a Java API, and a REST API. The web interface (a sample job submission shown in Fig. 7.1) offers the most convenient way for users to run jobs, and exposes all the functionalities of the CBP, like resource editing, invoicing, and user management (given that sufficient roles are available). The Java API offers a convenient way to access most of the CBP functionalities from software written in Java, like WSPGRADE/gUSE. Finally, the REST API can be used to access all CBP functionalities through a RESTful [REST] interface, basically providing a programming language-independent way to access the CBP services.

The screenshot shows the CloudBroker Platform web interface. At the top, there is a blue header with the text "CloudBroker Platform" and a search bar. Below the header, there is a navigation menu with buttons for Home, Users, Software, Resources, Jobs, Invoices, and Information. The "Jobs" button is highlighted. Below the navigation menu, there is a "Jobs" section with a sub-menu for "Entry Options" and "Expert Options". The "Entry Options" sub-menu is selected. The form contains several fields: "Name" (text input), "Software" (dropdown menu with "AutoDock Suite 4.2.5" selected), "Executable" (dropdown menu with "AutoDock Suite 4.2.5 autodock4" selected), "Resource" (dropdown menu with "OpenStack EC2 MTA SZTAKI BIFI" selected), "Region" (dropdown menu with "OpenStack EC2 MTA SZTAKI BIFI BIFI" selected), and "Instance type" (dropdown menu with "OpenStack EC2 MTA SZTAKI BIFI m1.tiny (1 core, 2.0 GB memory)" selected). To the right of the form, there is a table titled "Estimated Costs" with the following data:

Estimated Costs	
Per submission [USD]:	0.00
Per GB input data [USD]:	0.16
Per GB output data [USD]:	0.01
Per hour [USD]:	0.00
VAT [USD]:	0.00
Total for first hour [USD]:	0.00

Fig. 7.1 CloudBroker Platform web interface

As already mentioned, CloudBroker offers the SaaS service model: clients invoking the CBP can select any of the preregistered applications to run, and provide input files. However, it is possible to achieve an IaaS-like operation with the help of wrapper software: the task of the wrapper software is simply to run one of its input files with a fixed name. If a client invoking the CBP wants to use the IaaS-like execution to run its own executable, then the client simply has to select the wrapper application, and provide the desired executable as an additional input file called as "execute.bin".

7.3 Integrated WS-PGRADE/CloudBroker Platform to Access Multiclouds

The integration of WS-PGRADE/gUSE and the CloudBroker Platform is based on the idea of using the CBP as the entry point to cloud infrastructures. Thus, CBP is used as a hub to access the different cloud resources and the applications deployed on them. It follows from this that the integrated solution is capable of handling every cloud infrastructure accessible by the CBP. Such infrastructures include OpenNebula-based, OpenStack-based, Eucalyptus, Amazon, and IBM clouds. Notice that CloudBroker Platform plays the same role for clouds as DCI Bridge for all the DCIs.

Table 7.1 Comparison of IaaS and SaaS models

	IaaS model	SaaS model
Enables users to run their own executable	Yes	No
What has to be pre-deployed in the virtual machine image	A single wrapper application	Every application that users would like to use
What has to be configured in the CloudBroker Platform	Only the wrapper application	Every application that users would like to use
Level of security	Low: user can run anything	High: users can run only pre-registered, tested apps
Ease of use (user's perspective)	Easy: very similar to existing WS-PGRADE/gUSE interface	Easy: user simply selects from the predefined Software
Ease of use (portal administrator's perspective)	Easy: only two additional properties have to be set	Very easy: no need to set additional properties
Ease of use (CloudBroker administrator's perspective)	Easy: only one application has to be configured for each cloud resource	Hard: every application should be configured
Easy of use (cloud administrator's perspective)	Easy: only one VM image has to be deployed	Hard: either a number of VM images have to be deployed or one VM image must be updated a number of times

Additionally, the integrated solution offers two modes of operation: SaaS-like and IaaS-like. The SaaS-like operation mode enables the gateway users to run applications deployed on the CBP, whereas in case of the IaaS-like operation the users are able to run their own applications, with the help of a hidden wrapper software in the background. Table 7.1 summarizes the properties and requirements of the IaaS and SaaS operation models in the WS-PGRADE/gUSE and CBP scenarios.

As can be seen in Table 7.1, IaaS is a better choice than SaaS, given that we trust that the users will not submit malicious executables to the provided cloud resources. In turn, the SaaS model provides better security, but the overhead on configuration is notable. From the users' point of view, IaaS is only required if they would like to run their own executables; otherwise they will be fine with the SaaS model.

In the case of existing WS-PGRADE/gUSE workflows configured to run on non-cloud DCIs with executables provided by the users, the IaaS model offers a quick way to migrate the existing workflows to cloud resources: users simply have to reconfigure the workflow nodes to run on the CBP with the wrapper application.

7.4 Architecture

The high-level architecture of the WS-PGRADE/gUSE and CBP integration is shown in Fig. 7.2. WS-PGRADE with the different portlets and application-specific views is located at the top of the architecture. The user can connect to WS-PGRADE/gUSE to run workflows, or to the CBP directly to run single jobs. Users can use the authentication portlet to set their CBP credentials (username and password), and the workflow portlet can be used to configure workflows based on available resource and application information fetched from the CBP. Once a workflow is properly configured, it can be submitted. From this point on, the workflow interpreter (WFI) component of gUSE is responsible for creating job instances to be managed by the DCI Bridge. If a node/job of the submitted workflow is configured for execution in a cloud, the DCI Bridge's CBP plugin is responsible for managing the execution of the job with the help of the CBP service (creates corresponding CloudBroker jobs and uploads input files from the portal server to the cloud storage attached to the selected cloud resource). From this point on, it is the CBP service's task to create the requested virtual machine based on the selected application's deployment for running the job, and to perform any necessary data transfer between the virtual machine and the cloud storage used.

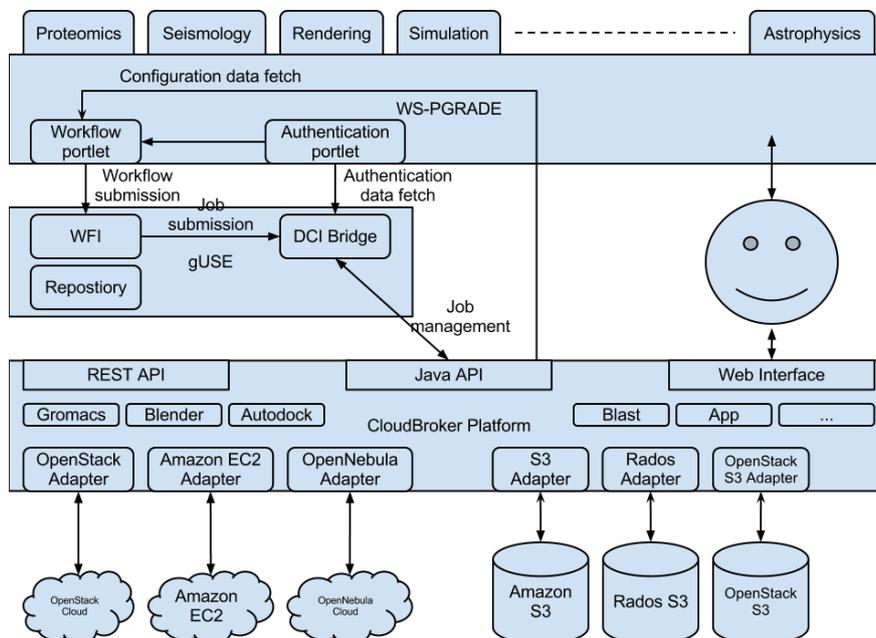


Fig. 7.2 WS-PGRADE/gUSE and CloudBroker integration architecture

All communication between WS-PGRADE/gUSE and the CBP service requires authentication, which is based on usernames and passwords. Thus, in order to use the CBP services, one should possess such credentials. In the case of the WS-PGRADE's workflow developer mode (where all the functionalities like workflow creation and configuration are exposed for the user), these credentials have to be entered, meaning that the portal user must have registered in the targeted CBP service, and during this registration step has received the necessary username and password. However, the need to enter these credentials can be eliminated by the "robot certificate" feature of WS-PGRADE/gUSE, where the workflow developer can attach different credentials to the nodes of the workflow he/she has developed, and users of this workflow will make use of the attached credentials in a transparent way (see Chap. 6 for details of robot certificates).

7.4.1 User Interface for Configuring Nodes for Clouds

The web interface WS-PGRADE/gUSE has been extended at different points: a new authentication portlet has been created to offer a way to enter credentials for each CBP service configured in the DCI Bridge, the concrete workflow configuration has been extended with possibilities to configure properties of CBP jobs with cost estimate display, and cost is displayed for executed workflows as well.

CloudBroker Authentication Portlet

One gateway can be connected to several CBP services. For example, in the SCI-BUS project CloudBroker provides two platforms for project members. The first one is the development platform where the research experiments of the gateway and platform developers can be carried out. The second one is the production platform that is a stable cloud access service provided for the gateway end-users and application developers.

For every CBP service connected to the gateway there is a separate plug-in instance in the DCI Bridge. Portal users have to provide their credentials for every CBP service in order to be able to configure and run jobs on the given service: the username and the password. Fig. 7.3 shows a view of the CloudBroker authentication portlet. As it can be seen in Fig. 7.3, three CBP services are configured: “platform”, “scibus” and “Dumm”. Once the user has entered or modified the credentials to be used for a given CBP service, the “Save authentication data” button can be used to save the new credentials. The portlet also checks if the provided credentials are valid (performs authentication with the selected CBP service), and notifies the user if any problems were detected.

The screenshot shows a web form titled "Authentication for CloudBroker". Below the title, it says "Authentication data for middlewares requiring HTTP(s) Basic Authentication:". There are three sections, each for a different service:

- platform (cloudbroker)**: Username: zfarikas@szaki.hu, Password: *****
- scibus (cloudbroker)**: Username: zfarikas@szaki.hu, Password: *****
- Dumm (cloudbroker)**: Username: [empty], Password: [empty]

At the bottom of the form is a button labeled "Save authentication data".

Fig. 7.3 CloudBroker authentication portlet

Concrete Workflow Configuration

Once the user has set proper CBP credentials, workflow nodes can be configured to run on an available CBP service. Users have two possibilities: either they can select from the predefined set of applications provided by the CBP service, or they can upload their own application given that the wrapper-enabled CBP application has been configured in the DCI Bridge’s admin interface (details can be found in Sect. 2.15 of the DCI Bridge Admin manual).

Type:	cloudbroker
Name:	scibus
Software:	Use Own Executable or
Executable code of binary:	Recently stored: Cpy.sh Browse...
Resource:	OpenNebula MTA SZTAKI SZTAKI
Region:	OpenNebula MTA SZTAKI SZTAKI SZTAKI OpenNebula
Instance type:	OpenNebula MTA SZTAKI SZTAKI SZTAKI OpenNebula x86_64
Argument string:	A

shows the configuration of a workflow node where a predefined executable (“Autodock 4.2.3” in the figure) is selected from the list of applications registered on the corresponding CBP service. As shown in the figure, the portlet can display an estimated cost for the execution of the job given that using the selected application and resource is not free of charge. The estimated cost is presented for the job as if it would execute only once, and would run for one hour.

Type:	cloudbroker	Estimated Cost Per submission [USD]: 0 Per GB input data [USD]: 0,289 Per GB output data [USD]: 0,289 Per hour [USD]: 0,17 VAT [USD]: 0 Total for first hour [USD]: 0,17
Name:	platform	
Software:	AutoDock 4.2.3	
Executable:	AutoDock 4.2.3 autodock4	
Resource:	Amazon EC2 CloudBroker GmbH	
Region:	Amazon EC2 CloudBroker GmbH EU (Ireland)	
Instance type:	Amazon EC2 CloudBroker GmbH Micro - 64 bit	
Argument string:	M	

Type:	cloudbroker
Name:	scibus
Software:	Use Own Executable or
Executable code of binary:	Recently stored: Cpy.sh Browse...
Resource:	OpenNebula MTA SZTAKI SZTAKI
Region:	OpenNebula MTA SZTAKI SZTAKI SZTAKI OpenNebula
Instance type:	OpenNebula MTA SZTAKI SZTAKI SZTAKI OpenNebula x86_64
Argument string:	A

Fig. 7.4 Predefined software and executable configuration compared to wrapping method usage

Type:	cloudbroker
Name:	scibus
Software:	Use Own Executable or
Executable code of binary:	Recently stored: Cpy.sh Browse...
Resource:	OpenNebula MTA SZTAKI SZTAKI
Region:	OpenNebula MTA SZTAKI SZTAKI SZTAKI OpenNebula
Instance type:	OpenNebula MTA SZTAKI SZTAKI SZTAKI OpenNebula x86_64
Argument string:	A

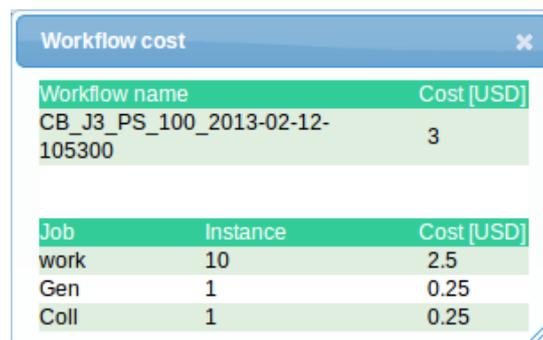
also shows a node configuration where the user is making use of the wrapping method. As it can be seen, the property “Software” indicates that the user is about

to use her own executable, the previously uploaded executable is highlighted (if applicable), and the user has the possibility to upload the executable she wants to run with the help of the Browse button.

Once the workflow has been configured and saved, WS-PGRADE stores the workflow's description in an XML file with all the relevant properties set. This XML file is used by the workflow interpreter to control the workflow execution.

Cost Display for Executed Jobs

As soon as at least one CloudBroker job has finished in a workflow, WS-PGRADE is capable of displaying the costs that occurred during the execution of the workflow. The DCI Bridge stores the costs after the jobs' termination as described in Sect. 7.4.2. In case of the overall workflow cost, the costs of the individual jobs are summarized and presented for the user as shown in Fig. 7.5 upon request. As one can see, the total workflow cost, and the cost of individual jobs (with the total number of job instances that resulted in the job summary cost) are presented for the user. Thus, the user has a clear view on how much the workflow did cost, and which jobs were dominant in the workflow cost.



Workflow name		Cost [USD]
CB_J3_PS_100_2013-02-12-105300		3

Job	Instance	Cost [USD]
work	10	2.5
Gen	1	0.25
Coll	1	0.25

Fig. 7.5 Extensions of DCI Bridge

7.4.2 Execution of Cloud-Directed Jobs

After the workflow configuration is saved, the execution of the workflow is managed by the workflow interpreter (WFI) and the DCI Bridge. These two components also had to be modified to enable cloud-based job execution:

- WFI: the workflow interpreter parses the workflow's configuration and prepares the JSDL [OGSA-BES] that is submitted to the DCI Bridge. This part of the WFI had to be extended by adding the CloudBroker-specific parts to the JSDL document.
- DCI Bridge: this component had to be extended with a CBP plugin that is capable of performing job submission and management with the CBP by

using the Java API of CBP. The DCI Bridge fetches authentication data from WS-PGRADE as set in the authentication portlet.

Modifications of the Workflow Interpreter (WFI) to Support CBP Integration

The WFI is responsible for scheduling submission of jobs to different DCIs after evaluating job status changes received from the DCI Bridge. For this, the WFI uses the XML description of the workflow and available job instance information. Once the WFI realizes that a job can be executed, it takes the job's description from the workflow XML, collects the access information of input files belonging to the job, creates the JSDL document describing the job, and finally submits the job to the DCI Bridge. In the case of jobs set to run on a CBP service, the WFI simply maps the XML attributes specifying the application, resource, region and instance to the ones used in the JSDL.

Extensions of DCI Bridge

The DCI Bridge has been extended with a CBP plugin that is responsible for communicating with CBP services. Not only one, but also a number of CBP services can be configured in the DCI Bridge. These services are identified by a name and each has its own configuration, but all of them rely on the CBP plugin to communicate with the target service.

DCI Bridge has two aspects: a configuration aspect and a job execution aspect. In the configuration aspect administrators may enable the different plugins and set their configuration, whereas in the job execution aspect the DCI Bridge accepts job submission requests in the form of OGSA-BES standard JSDL documents and ensures their execution in the selected DCI. These two aspects can be accessed concurrently; that is, it is possible to configure the DCI Bridge while it accepts and processes jobs. The configuration aspect is reachable through a simple web interface, whereas the job execution aspect can be used through a web service interface.

Configuration Aspect

One gateway may be connected to a number of CBP services. In the configuration view the DCI Bridge administrators may enable or disable the CBP plugin (resulting in disabling access to all the CBP services configured), and may add or modify existing CBP services. Figure 7.6 shows the configuration of a CBP service.

The screenshot shows a web form titled 'Modify CloudBroker data' for a service named 'scibus'. The form contains the following fields and options:

- Name:** scibus
- Status:** Enabled Disabled
- URL:** https://scibus.cloudbroker.com
- User management:** Individual
- Software for running own executable:** gUSE Wrapper 1.0
- Executable for running own executable:** guse_wrapper.sh
- Running own executable function is enabled only when the 'Software' and 'Executable' fields are filled.**
- This service running:** true
- WSDL of the other service:** (empty text box)

At the bottom of the form, there is a 'Save' button with a pencil icon.

Fig. 7.6 Settings of a CloudBroker Platform service in the DCI Bridge

Different properties can be set for a given CBP service, like the name of the service, the API URL of the targeted CBP service, the user management type (the only available option is “Individual”, meaning every WS-PGRADE user has to provide its own credentials to use the specified CBP service, or the robot certificate extension should be used), and CBP application for running own executable (enabling the IaaS working model). Description of other properties is available in the DCI Bridge admin manual [DCIBRIDGE]. A properly configured DCI Bridge CBP service can be used to run jobs.

Job Management Aspect

DCI Bridge accepts job submission requests through its OGSA-BES interface in the form of JSDL documents. Notice that the WFI, when using WS-PGRADE, automatically generates this JSDL document; thus this is completely hidden from the users. In the JSDL, gUSE makes use of a few extensions to the original JSDL standard to have a way to specify CBP-specific settings, for example the name of the selected CBP application or the resources to be used for the job’s execution.

The DCI Bridge performs submission of jobs according to the following steps:

1. DCI Bridge collects all the input files either from the file or workflow storage (WFS) services of gUSE based on the information stored in the incoming JSDL.
2. A new CBP job is created using the Java API of the CBP.
3. Different properties of the CBP job are set based on the incoming JSDL, like the job’s name, the application to be run, or the resource to be used for execution. DCI Bridge asks the CBP to collect the output files of the job after its termination as an archive (resulting in much faster job finishing times compared to the case when output files are transferred back one-by-one).

4. The DCI Bridge packs the input files collected in step 1 into an archive. Once the archive is ready, the DCI Bridge invokes the CBP API to add the archive to the CBP job (created in step 2) as an input file. This also makes the CBP API upload the archive to the cloud storage that is attached to the job's cloud resource, and register the location of the file to the CBP job (created in step 2). This means that the input archive's data is not transferred from the DCI Bridge to cloud storage through the CBP service, but the API call uploads the job's input archive directly to the cloud storage from the DCI Bridge service.
5. Once all the properties are set and the input archive is uploaded, the job is submitted to the CBP service.

According to our experiences, handling jobs with the above input and output archiving methods is very efficient for jobs with many input or output files. For example, the whole lifecycle of a short job (running for a few seconds) that produced 100 output files lasts for almost 15 minutes without, and for 3 minutes with output archive usage. The metrics are the same for short jobs that have similar number of input files. This behavior follows from the way that the CBP manages the data related to the jobs: first, the input files have to be added to the job, which results in uploads to the selected resource's storage through a redirection from the CBP service. Once the virtual machine is started, the files are transferred from the storage to the virtual machine. Output file handling happens similarly to input file handling, but in the opposite direction.

Status of submitted jobs is periodically updated using the CBP Java API. The returned CloudBroker-specific job status is matched against the relevant OGSA-BES job status. Once a job reaches a terminal status, its output files are handled as follows by the DCI Bridge:

1. The output archive file is fetched into a temporary directory on the DCI Bridge service's machine using the CBP Java API. Just like in case of the input file archive, this API call redirects the caller to the cloud Storage where the file resides, so the portal will fetch the results directly from the cloud Storage, and not through the CBP service used.
2. The content of the output archive is decompressed into the job's working directory on the DCI Bridge service.
3. Any files used by the job are removed from the CBP, and thus from the cloud storage as well.
4. The output files are uploaded to their specified target URL (in case of jobs submitted from WS-PGRADE, this is the file storage service of gUSE) as written in the JSDL document.
5. Finally, DCI Bridge queries and records the costs of the job so they can be presented later for the users through the web interface.

Beside the above, the CBP plugin of DCI Bridge is able to resubmit the failed jobs at most as many times as specified in the DCI Bridge's administration inter-

face for the plugin. Job resubmission is performed in the following cases: if the job's exit status is not zero or if any of the output files is missing. Job aborts are performed with the help of the CBP Java API as well. Once a job has been cancelled, all of its files are removed from the cloud storage and from the CBP's registry as well.

7.5 Integration Features

The integrated WS-PGRADE/gUSE and CBP architecture offers easy access to cloud infrastructures from scientific workflows. Thanks to the use of CBP, every cloud infrastructure supported by CBP is available for use by WS-PGRADE/gUSE users. This includes cloud infrastructures based on OpenNebula or OpenStack, and Eucalyptus, Amazon, and IBM clouds. As can be seen, not only academic (usually built on OpenNebula and OpenStack), but also commercial clouds are supported as well. In this latter case, if access to the cloud infrastructure is not free of charge, the extended set of WS-PGRADE portlets are capable of displaying a cost estimate before, and the actual costs occurred after the jobs' and workflows' execution. This is very important in commercial environments.

The integration completely supports the SaaS concept offered by the CBP. As presented earlier, WS-PGRADE/gUSE users can select from the available set of software applications deployed on the CBP for execution. Additionally to the SaaS concept, the integrated solution offers an IaaS-like operation mode as well (given that an appropriate wrapper software is deployed on the CBP), with which WS-PGRADE/gUSE users can run their own applications in the available cloud infrastructures. This feature provides a convenient way to test workflows during their development phase; once the workflow is ready, the included applications can be deployed on the cloud infrastructures and accessed following the SaaS concept.

Finally, all the customization features of WS-PGRADE/gUSE can exploit the integration features as well: both the Remote API and the ASM API can be used together with the CloudBroker Platform integration. Additionally, the robot credential concept is also supported by the integration. These enable the exploitation of cloud infrastructures for real e-scientists in a transparent way.

7.6 SCI-BUS Multicloud Access

Both the SCI-BUS and the CloudSME [CloudSME] projects are operating reference WS-PGRADE/gUSE deployments configured to use different CBP services, which are connected to different cloud infrastructures.

In the SCI-BUS project, the reference gateway is connected to the Public CBP service and the SCI-BUS CBP service. The Public service is devoted for production-level operation, and offers access to production-level cloud infrastructures, and properly tested software applications. On the other hand, the SCI-BUS CBP service is used to develop workflows and add new cloud infrastructures to the

SCI-BUS infrastructure. Once a cloud infrastructure or software is found to be stable on the SCI-BUS CBP service, it is moved onto the Public CBP service for production-level operation.

The CloudSME project follows the same approach, but with multiple WS-PGRADE deployments: a production one and a testbed one, along with a production and testbed CBP service. The production gateway is only connected to the production CBP service, whereas the testbed gateway is connected both to the production and testbed CBP services. The production CBP service offers access to the production-level cloud infrastructures and applications, whereas the testbed CBP service serves as a development (or integration) platform for new cloud infrastructures and applications.

7.7 Conclusions

In this chapter we presented the integration of WS-PGRADE/gUSE and the CloudBroker Platform (CBP) that enables the exploitation of cloud infrastructures from a user-friendly science gateway.

The integration offers access to most major academic (OpenNebula- or OpenStack-based) and commercial (Amazon, IBM) cloud infrastructures. Not only is the SaaS concept supported by the CBP offered, but also, with the help of a wrapper software, an IaaS-like operation mode is available for the users as well.

The different customization methods offered by WS-PGRADE/gUSE (Remote API and ASM API) can also make use of the integrated architecture easily. Additionally, the robot credential concept can also be used together with CBP services in WS-PGRADE/gUSE. These two features together enable creating e-science gateways for real end-users, which make use of cloud infrastructures in a transparent way.