# EGI-Engage

# Deployment of a gCube release with Federated Cloud support

**D4.5**

| | |
|---|---|
| **Date** | 04 March 2016 |
| **Activity** | JRA2.3 |
| **Lead Partner** | INFN |
| **Document Status** | FINAL |
| **Document Link** | https://documents.egi.eu/document/2662 |


## Abstract

This document reports on the integration of Federated Cloud resources in the D4Science platform, built on top of the gCube framework. The document introduces the D4Science platform, how it is currently exploited by scientific communities and motivates the need for the integration. The most relevant usage scenarios are analysed and corresponding requirements are identified. The extension to the gCube framework is defined in terms of overall architecture and description of single components. A number of enhancements have been selected for future evaluation and realization.




The EGI-Engage project is co-funded by the European Union (EU) Horizon 2020 program under Grant number 654142 http://go.egi.eu/eng

## COPYRIGHT NOTICE



## DELIVERY SLIP

| | *Name* | *Partner/Activity* | *Date* |
|---|---|---|---|
| **From:** | Paolo Fabriani | ENG/WP4 | 3 Feb 2016 |
| **Moderated by:** | Małgorzata Krakowian | EGI.eu/NA1 | |
| **Reviewed by** | Kostas Koumantaros<br>Diego Scardaci<br>Vincenzo Spinoso | GRNET/PMB<br>INFN/<br>INFN/WP5WP3 | 15.02.2016<br>17.02.2016<br>15.02.2016 |
| **Approved by:** | AMB and PMB | | 4.03.2016 |

## DOCUMENT LOG

| *Issue* | *Date* | *Comment* | *Author/Partner* |
|---|---|---|---|
| **v. 1** | 28 Jan 2016 | First internal review | M. Viljoen/EGI.eu<br>G. Giammatteo/ENG<br>P. Fabriani/ENG |
| **v. 2** | 2 Feb 2016 | Second internal review | M. Viljoen/EGI.eu<br>G. Giammatteo /ENG,<br>P. Fabriani/ENG,<br>F. Sinibaldi/CNR,<br>P. Pagano/CNR |
| **v. 3** | 3 Feb 2016 | Version ready for external review | P. Fabriani/ENG |
| **v. 4** | 26 Feb 2016 | Version after external and PMB review | P. Fabriani/ENG<br>G. Giammatteo/ENG<br>N. A. Galante/ENG |
| **FINAL** | 29 Feb 2016 | Final version | P. Fabriani/ENG |

## TERMINOLOGY

A complete project glossary is provided at the following page: http://www.egi.eu/about/glossary/

# Contents

# 1 Executive Summary

D4Science is a research e-infrastructure powered by the gCube system offering a number of services for seamless access and analysis to a wide spectrum of data - including biological and ecological data, geospatial data, statistical data and semi-structured data from multiple authoritative data providers and information systems. D4Science relies on a physical infrastructure counting more than 100 hosting nodes provided by the parties operating the infrastructure. Infrastructure usage, however, is far from being uniform over time, and peaks in the demand for computing capacity are often observed, highlighting the limits and drawbacks of a lack in elasticity.

This document shortly reports on the analysis of usage scenarios and their related requirements as well as architectural and implementation choices made to deliver a gCube release enhanced with EGI Federated Cloud support. The achieved integration makes the D4Science platform capable to closely follow scientific communities' needs in terms of computational resources. Cloud Bursting, in fact, appears to be the natural deployment model to face the above scenario. By allowing programmatic and dynamic creation, configuration and decommissioning of cloud resources on external infrastructures (either research-oriented like EGI Federated Cloud or commercial ones like Amazon EC2 and Microsoft Azure), benefits are in terms of i) exploitation of infrastructural resources on a per-use basis reducing capital expenditure in physical resources ii) capability to quickly scale in/out quickly in response to needs and ii) reduction of manual interventions of infrastructure managers.

With the EGI Federated Cloud support in place, D4Science made a step towards an efficient and effective management of its infrastructure. Some further exploitation of EGI services could bring additional benefits; in particular i) Occopus could be evaluated to support elasticity in terms of automatic provisioning and decommissioning of cloud resources across external cloud infrastructures; ii) depending on the EGI Federated Cloud release plan, the integration with the EGI accounting system would bring more accurate and coherent accounting information; iii) automating the creation of Virtual Appliances upon release of relevant gCube components and registration to EGI AppDB would speed up the timely availability of updated appliances; iv) integration with the EGI Fedcloud Information System (BDII) for the retrieval of up-to-date information about FedCloud services, sites, OS and resource templates to better support an automated VM lifecycle management.

# 2 Introduction

D4Science is an infrastructure powered by the gCube[1] system offering a number of services - currently integrating more than 500 software components - and Virtual Research Environments[2] (VREs) for seamless access and analysis to a wide spectrum of data including biological and ecological data, geospatial data, statistical data and semi-structured data from multiple authoritative data providers and information systems.

D4Science is a Hybrid Data Infrastructure connecting +2000 scientists in 44 countries; integrating +50 heterogeneous data providers; executing +13,000 models & algorithms/month; providing access to over a billion quality records in repositories worldwide, with 99.7% service availability. D4Science hosts +40 Virtual Research Environments to serve the biological, ecological, environmental, mining, and statistical communities worldwide. D4Science relies on a physical infrastructure counting more than 100 hosting nodes provided by the parties operating the infrastructure (namely CNR[3], UoA[4], FAO[5], CITE[6] and ENG[7] [8]).

D4Science Infrastructure usage - and thus resource needed - is far from being constant along time and spikes in the demand for computing capacity are often observed in the typical usage of the infrastructure. Cloud bursting appears as the natural deployment model to face this scenario. D4Science is implementing such model by relying on either research infrastructures (e.g. EGI Federated Cloud) and commercial public clouds (e.g. Amazon EC2); with respect to the former, this document shortly reports on the analysis of usage scenarios and their related requirements as well as architectural and implementation choices made to deliver a gCube release with FedCloud support.

[1] The gCube Framework - https://www.gcube-system.org

[2] L.Candela, D.Castelli, P.Pagano, Virtual Research Environments: an overview and a research agenda, Data Science Journal, Volume 12, 10 August 2013

[3] Consiglio Nazionale delle Ricerche - http://www.isti.cnr.it

[4] National and Kapodistrian University of Athens - http://en.uoa.gr

[5] Food and Agriculture Organization of the United Nations - http://www.fao.org

[6] Communication & Information Technologies Experts - http://www.cite.gr

[7] Engineering Ingegneria Informatica S.p.A. - http://www.eng.it

[8] Currently, ENG hosts part of the testing infrastructure

| | |
|---|---|
| **Tool name** | Federated Hosting Node Manager |
| **Tool url** | N/A |
| **Tool wiki page** | N/A |
| **Description** | The service enables D4Science infrastructure and VRE managers to respond to variations in the need for computational resources by joining/disposing additional resources from external infrastructures (i.e. FedCloud). |
| **Value proposition** | The D4Science relies on a physical infrastructure counting more than 100 hosting nodes provided by involved partners. Resource usage, however is far from being constant along time and spikes in the demand for computing power exceed the available infrastructure capacity. The service implements the cloud bursting model as an approach to face demand peaks. |
| **Customer of the tool** | Providers of the D4Science infrastructure |
| **User of the service** | D4Science infrastructure managers and VRE managers |
| **User Documentation** | https://wiki.gcube-system.org/gcube/GCube_Documentation |
| **Technical Documentation** | https://wiki.gcube-system.org/gcube/GCube_Documentation |
| **Product team** | Engineering Ingegneria Informatica (ENG),<br>Consiglio Nazionale delle Ricerche (CNR) |
| **License** | European Union Public Licence (EUPL version 1.1) |
| **Source code** | http://svn.research-infrastructures.eu/d4science/gcube/trunk |

# 3 Usage Scenarios and Requirements

The management of VREs resources is partially supported by the gCube system that currently allows VRE Managers to select what services, datasets and resources (i.e. computation and storage) in the D4Science infrastructure are visible and exploitable to the VRE members. VRE Managers can only select a subset of resources (identified according to the sharing and usage policies rules in place for each resource; i.e. resources contributed to the infrastructure to serve a specific VREs cannot be selected nor are visible to other VREs) already existing and registered in the infrastructure. Adding new resources to the infrastructure is not supported dynamically by the gCube system and it is done manually by Infrastructure Managers who are in charge, among other duties[9], of setting-up, configure, manage and monitor all resources belonging to the D4Science infrastructure.

The integration of the gCube system with FedCloud and, in general, with research/commercial cloud providers, aims to add elasticity to the D4Science infrastructure by allowing programmatic and dynamic creation, configuration and decommissioning of cloud resources on external infrastructures. The benefits of this integration facility for the gCube system are three: to exploit infrastructural resources on a per-use basis reducing capital expenditure in physical resources, to scale in/out quickly in response to specific needs, and to reduce manual interventions of Infrastructure Managers.

In order to give an idea of how the integration facility with cloud providers will fit in the VRE management workflow, two possible usage scenarios are provided in the context of a data analysis service implemented by the gCube system. The service is able to execute data analysis (e.g. signal forecasting) against one or more input datasets. The analysis is executed, when possible, in parallel on a set of gCube nodes equipped with an execution engine called SmartExecutor. The capability to perform analysis on data, together with the availability of large and heterogeneous datasets, is one of the most prominent and commonly exploited features that D4Science offers to scientific communities and which requires, at variable rates, a large part of the infrastructure resources.

**Scenario A** - the VRE Manager, according to the demand of data analysis executions in the VRE, decides the number of SmartExecutor nodes needed in the VRE. Through the VRE administration portal, the VRE Manager chooses (from a list of templates) to create the required number of SmartExecutors. She/he chooses on which infrastructure the resources should be created and their characteristics in terms of computation and/or storage capacity. The gCube system, via the integration facility here described, will automatically create the new nodes, register them to the D4Science infrastructure and make them available in the VRE.

[9] https://wiki.d4science.org/index.php?title=Role_Infrastructure_Manager

**Scenario B** - in this more advanced scenario, the gCube system intelligently adapt the number of SmartExecutor nodes to the actual demand. In the VRE administration portal, the VRE Manager decides the minimum and maximum number of SmartExecutor expected for the VRE. Then, the gCube system via the integration facility here described, automatically scales-in/out the pool of SmartExecutors (by creating/destroying nodes) predicting the demand for SmartExecutors by analysing VRE accounting and monitoring data.

It is possible to recognize the following functionalities that should be available in the integration facilities in order to realize the scenarios described:

- **Registration and management of cloud providers**. Access credentials and cloud resources templates on a per-VRE basis. Credentials must be securely stored in the infrastructure;
- **Cloud resource lifecycle management**. Creation, configuration and decommissioning of cloud resources must be supported. The integration facility should also allow to use templates for the configuration of cloud resources (e.g. SmartExecutor Template, cpu and memory characteristics);
- **Accounting** and **monitoring**. Accounting and monitoring data on usage and workload of resources created on the cloud must be collected and made available in the infrastructure. This is both a requisite to comply with D4Science infrastructure policies and for the automatic scale-in/out;
- **User interface**. The user interface should be integrated in the VRE administration portal since it already contains all the other VRE administration tools and because VRE Managers are already familiar with it. Since the portal is based on combinable modules (i.e. portlets), that technology can be used to easily meet this requirement;

In addition to the above mentioned functional requirements, the solution should have modular architecture in order to support the integration of new cloud providers in a simple and pluggable way.

# 4 Architecture

This section presents the architecture of the integration solution. An overview of the architecture is provided below:

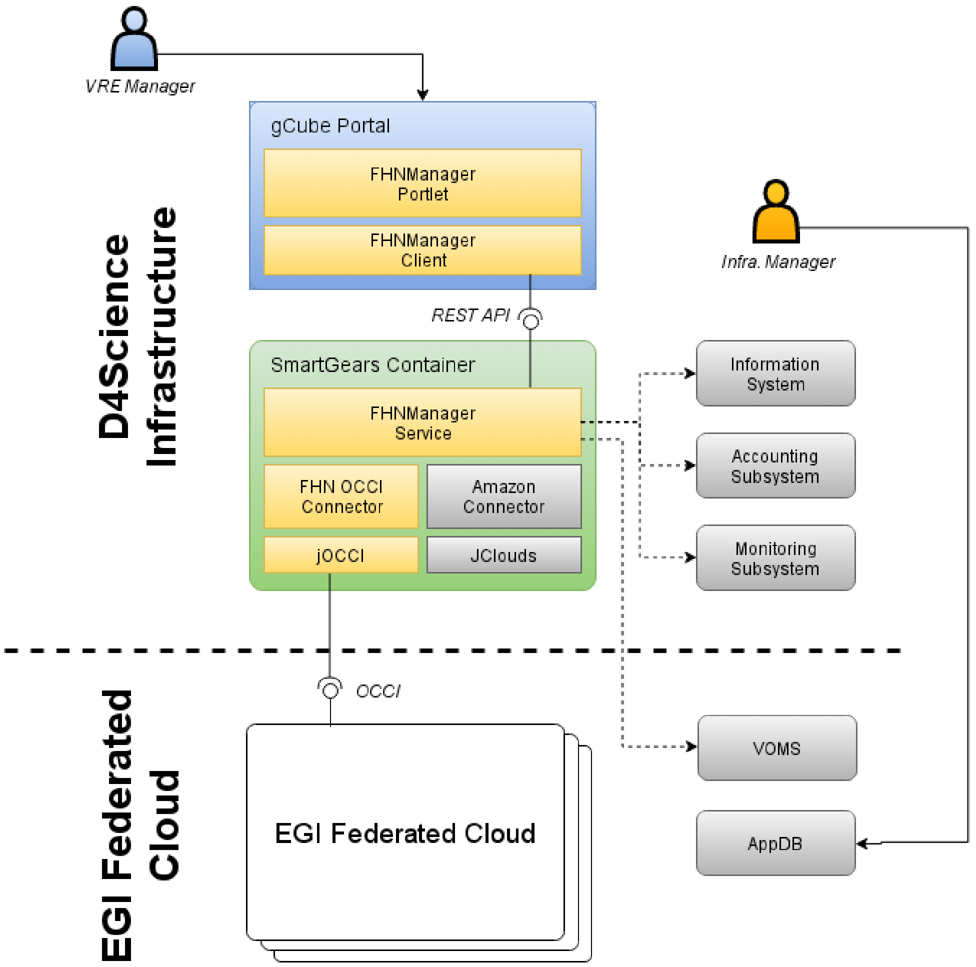



Figure 1 Integration of the EGI Federated Cloud into the D4Science Infrastructure

The architecture highlights a number of integration-specific components and their interaction among them as well as with existing components, belonging either to the D4Science infrastructure or to the EGI infrastructure. All of them are shortly described hereafter:

- **Cloud Libraries** are third-party software providing language-specific APIs and data model to easily interact with clouds from within applications (jOCCI and JClouds in Figure 1). Although they usually support a number of different clouds and cloud standards, there's no universal coverage for any of them; nor the API and data model they expose is uniform across libraries.

- **Connectors** are built on top of cloud libraries in order to abstract the specifics of their APIs and data models and expose a uniform interface to the upper layers. Connectors do not interact with any other service nor are expected to persist any information.
- The **Federated Hosting Node Manager** (FHNM) is the core part of the integration and is the place where all the business logic resides. It's the gateway for all the operations related to the management of external cloud infrastructures, via the most-appropriate connector/library; it manages connectors to the available clouds; it gathers accounting and status data and publishes them to the specific D4Science services.
- The **Federated Hosting Node Manager Portlet** provides infrastructure and VRE administrators with a dashboard and a control panel to easily monitor and manage resources in external infrastructures. It enables administrators to register cloud infrastructures and credentials associated with them as well as virtual appliances and service profiles.

The following services have not received modifications or enhancements to enable the integration with EGI Federated Cloud. Only exception is the D4Science Accounting system, whose datamodel has been enriched with manage cloud-related usage records.

- The **D4Science Information System** collects, holds and provides all the information related to the D4Science infrastructure. In particular, for the purpose of the integration of FedCloud, it holds the list of available cloud sites along with credentials to access them, the list of running cloud resources and their status and the list of virtual appliances available for instantiation.
- The **D4Science Accounting System** tracks the usage of all D4Science resources, including external ones. Usage records are fed to the accounting system either by querying external accounting systems - where available - or by tracking relevant events (e.g. create, start, stop, destroy VMs) at FHNM-level.
- The **D4Science monitoring system** tracks various aspects of the whole D4Science infrastructure. Data collected is harvested and analysed to produce alarms and/or take countermeasures in case of problems with one or more resources;
- The **Virtual Organization Membership Service** (VOMS) takes part in the architecture by issuing authorization attributes to be embedded in X.509 proxy certificates, needed to access FedCloud sites.
- **EGI FedCloud sites** and **AppDB** respectively host cloud resources instantiated through the components above and maintains the set of available virtual appliances.

From a deployment point of view, components related to the D4Science-FedCloud integration are essentially assembled in two packages that are deployed independently: a) the Federated Hosting

Node Manager, embedding cloud libraries and connectors, is deployed in a SmartGears[10] container and b) the FHNManagerPortlet is deployed in the gCube portal.

[10] SmartGears is the standard container for gCube services based on Apache Tomcat

# 5 Service Description

For the implementation of the solution described in the document, it has been chosen to use the Java technology for two main reasons: a) it is the same technology of the gCube system and it has made easier the integration in the D4Science infrastructure and b) there is high availability for third-party libraries implementing cloud standards.

## 5.1 Datamodel

The datamodel has been designed with the aim of representing the information related to cloud providers, nodes, templates, accounting and monitoring in a homogenous way independently from the cloud provider exposed API and technology. *Connectors* and third-party *Cloud Libraries* ensured a proper translation between the internal datamodel and the specific cloud provider model. An UML class diagram representing the main entities is reported in Appendix I

## 5.2 FHNManager

The FHNManager is a web service implemented in Java and running in a gCube SmartGears container. It exposes a REST API that allows to access data on nodes and cloud providers as well as to execute operations to create, destroy, start and stop nodes. The complete API specification is reported in Appendix II.

In order to account the usage of cloud provider resources by the D4Science infrastructure, the service relies on the gCube accounting framework provided by the infrastructure: new usage records have been defined to track changes in the lifecycle of the resources on the cloud. Through the connectors, the service also supports the fetching of accounting data from the cloud provider (when available).

Concerning the monitoring data, the service implements the support to fetch it from the cloud provider (if available) as well as collect data directly from the resource via the gCube monitoring framework.

The current implementation of the service - and thus the portlet - does not support the registration of new cloud providers along with their associated credentials. However, since the registration entries for cloud providers are maintained in the D4Science Information System, VRE/Infrastructure managers can still use the editing functionalities provided by the Information system itself, although the editing capabilities provided are not tailored to the specific task.

Since the FHNManager is deployed in a SmartGears container, it is automatically registered to the D4Science infrastructure and it is automatically authorized to exploit infrastructure capabilities (Information, Accounting and Monitoring). In addition, to properly generate and use X.509 proxy certificates, the host must also be equipped with VOMS clients (i.e. voms-proxy-init) and configured with trusted root certification authorities as distributed by EUGridPMA[11].

## 5.3 FHNManagerPortlet

The FHNManagerPortlet is a portlet offering facilities to exploit the FHNManager capabilities. Target users of this portlet are both infrastructure and VRE Managers that will use it in order to monitor and control the virtual machines created via FHNManager service.

Built on top of gCube framework and according to Liferay specifications, the portlet uses this technology to discover and connect to the FHNManager service, gather information on the current status of the federated infrastructure, modify it through service's API and sharing information with other users through the portal's social facilities implemented by gCube.

Portlet main use cases are:

- CRUD operations on FHNManager data model entities
- start/stop of federated hosting nodes
- report generation and sharing

The portlet implementation is based on the GWT framework[12] and Java 7.

The portlet is deployed in the D4Science portal, and exposed to the VREs configured to exploit FHNManager facilities. Access to the portlet is controlled by portal policies, enforced by means of the infrastructure capabilities involving the Information system, Accounting and Authorization systems.

## 5.4 Third-party dependencies

The developed software directly rely on the following set of external dependencies:

| Dependency | Usage | License |
|---|---|---|
| Datamodel and Connectors | | |
| jOCCI | java client library for clouds exposing the OCCI interface | Apache License Version 2.0 |
| Jclouds | java client library for accessing various cloud infrastructures | Apache License Version 2.0 |

[11] https://dist.eugridpma.info

[12] http://www.gwtproject.org

| FHNManager | | |
|---|---|---|
| Jersey | Java framework to to create REST APIs | CDDL Version 1.1<br>GPL Version 2 |
| FHNManagerPortlet | | |
| Liferay 6.0 CE | Enterprise portal framework | LGPL 2.1 |
| GWT | Java framework for portlet development | Apache License Version 2.0 |

## 5.5 EGI FedCloud Membership

D4Science-FedCloud integration is being tested in the “fedcloud.egi.eu” VO. Production-level operations will be moved to the “d4science.org” VO, registered on the “vomsmania.cnaf.infn.it” VOMS server and in production since December 2015[13].

## 5.6 Source code

The source code of the developed components is available on the gCube code repository.

Datamodel and Common:

http://svn.research-infrastructures.eu/d4science/gcube/trunk/vo-management/fhnmanager-api

Connector:

http://svn.research-infrastructures.eu/d4science/gcube/trunk/vo-management/occi-library

FHNManager:

http://svn.research-infrastructures.eu/d4science/gcube/trunk/vo-management/fhnmanager-service

FHN Client Library:

http://svn.research-infrastructures.eu/d4science/gcube/trunk/vo-management/fhnmanager-client

FHNManagerPortlet:

http://svn.researchinfrastructures.eu/d4science/gcube/trunk/portlets/admin/fhn-manager-portlet

## 5.7 gCube release and testing process

The software developed to enable the EGI Federated Cloud support in gCube follows the development process and guidelines defined for the whole gCube platform[14]. The platform source

[13] https://ggus.eu/index.php?mode=ticket_info&ticket_id=117484

[14] https://wiki.gcube-system.org/gcube/Software_Integration_and_Distribution:_Overview

code, kept on the project source code repository[15] is continuously built upon each and every change using the ETICS tool[16]. Furthermore, on a daily basis, all components of both the development and release candidate versions are integrated together in a single build[17]. Both for partial and full builds, the outcome of the process (build and unit test logs and packaged artifacts) is available to developers to react to integration issues. A project issue tracker[18] is used to support the resolution of major integration and testing issues.

Being a big distributed platform, the gCube integration and release processes rely on dedicated development and testing environments. In particular:

- an unstable development infrastructure (named ‘devNext’) hosts snapshot versions of each component as soon as they integrates correctly in nightly builds;
- a development infrastructure (named ‘devSec’) is a stable development environment hosting components that proved to work on ‘devNext’. Developers do extensive testing on their own components here;
- a pre-production infrastructure (named ‘preprod’) hosts candidate components for the upcoming gCube release. Here, automated deployment testing is performed as well as functional testing via extensive user validation through the system portal interfaces. Components successfully passing this staging phase are included in the gCube stable release.

At the time of writing, the components are under integration and are planned to be included in the next gCube release (ver. 3.11.0) scheduled by March 2016. Produced packages will be available on the public gCube distribution site[19].

## 5.8 Further documentation

User documentation for the FHNManagerPortlet can be found on the gCube User Guide[20]

FHNManager, connectors and libraries’ API documentation can be found on the gCube Developer’s Guide[21]

[15] http://svn.research-infrastructures.eu/public/d4science/gcube/

[16] https://etics.esl.eng.it/etics-gui

[17] http://eticsbuild2.research-infrastructures.eu/BuildReport/home/AllBuilds/org.gcube.HEAD/latest

[18] http://support.d4science.org/

[19] https://www.gcube-system.org/software-releases

[20] https://wiki.gcube-system.org/gcube/User%27s_Guide

[21] https://wiki.gcube-system.org/gcube/Developer%27s_Guide

Deployment and configuration details for the service and the portlet are available on the gCube Administrator's Guide[22]

## 5.9 License

The FHNManagerPortlet, the FHNManager and the connectors developed within the scope of this task are released under the terms of the European Union Public Licence[23] (EUPL version 1.1), like the rest of the gCube software.

[22] https://wiki.gcube-system.org/gcube/Administrator%27s_Guide

[23] https://joinup.ec.europa.eu/community/eupl/og_page/european-union-public-licence-eupl-v11

# 6 Conclusions and future enhancements

At the current stage of the integration of EGI Federated Cloud into D4Science, VM lifecycle management is fully supported by OCCI capabilities and load peaks can be easily managed by administrators by offloading computation to FedCloud. The adoption of the jOCCI library simplified considerably the interaction with FedCloud sites. The main difficulty faced during the usage of OCCI was the unavailability of structured data for resource templates: number of cores and memory are provided as human-oriented descriptions, making difficult to programmatically parse and use them. As a workaround, this information was extracted from running instances created from a given resource template. A better solution would be to expose these information via the REST API (if compliant with OCCI) or retrieve them from the FedCloud Information System (BDII).

The fedcloud.egi.eu Virtual Organisation, which supported the development and testing stages, showed sporadic unavailability; this is however perfectly acceptable and in line with the expectations of a testbed environment. The usage of the EGI VOMS for the authorization was straightforward and no issues were found. EGI AppDB provides adequate support for the management of gCube-specific Virtual Appliances.

A number of enhancements is already foreseen and could be eventually put in place within the lifetime of the EGI-Engage project, also in collaboration with the BlueBRIDGE project[24]:

- integration with the EGI Federated Cloud accounting facilities - Relying on an authoritative source of usage data, would avoid D4Science from producing duplicate and potentially incoherent accounting. In this direction, requirements and use cases were shared with the EGI accounting team. Although there's no short-term plan to provide support for this use case, EGI will evaluate it, also considering similar requirements coming from INDIGO DataCloud.
- integration with the EGI Information System - Sites supporting D4Science are, at the current stage, manually listed in the D4Science Information system along with other details (e.g. their status, available images, corresponding credentials, etc.); if not regularly updated, such information becomes quickly out-of-date and might cause unexpected behaviours. By integrating with the EGI Fedcloud Information System (BDII), not only the above problems would be solved, but further useful information would be available to support an automated VM lifecycle management (e.g. discovery of OS and resource templates along with their properties in a structured way).
- automate the creation of Virtual Appliances upon release of relevant gCube components and registration to EGI AppDB via the AppDB REST API. This would speed up the timely

[24] http://www.bluebridge-vres.eu/

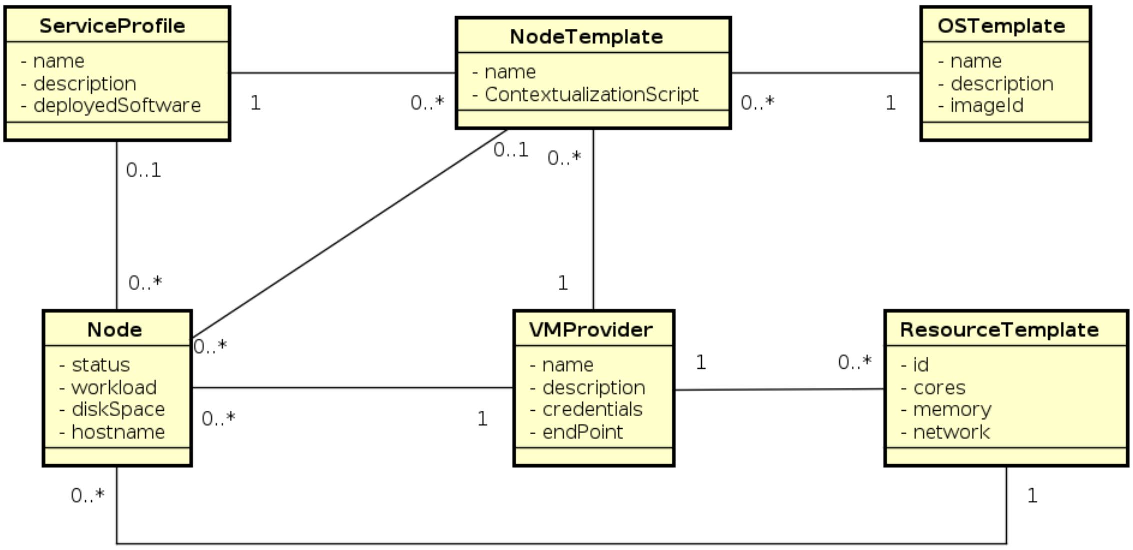
availability of updated appliances on AppDB and would pave the way for the creation of multiple gCube VAs thus enabling the realization of further gCube scenarios on FedCloud.

- further explore the opportunity of adopting Occopus[25] to support elasticity in terms of automatic provisioning and decommissioning of cloud resources across external cloud infrastructures. Different parameters are expected to drive the elastic behaviour, including current resources load, established quotas, pricing models, performance and QoS indexes, etc.
- support for multiple external computing infrastructures (e.g. Amazon EC2, Microsoft Azure) besides EGI FedCloud, as well as managing D4Science internal resources through D4Science itself. Infact, although D4Science fully relies on cloud resources, the management of them is done with external IaaS tools, making D4Science actually unaware of the underlying cloud.

[25] http://occopus.lpds.sztaki.hu

# Appendix I. Domain model

This appendix reports an UML class diagram representing the domain model with the core entities modelled in the integration facility. For each entity in the diagram, a brief description of each one of them is provided.




**VMProvider** - represents an external infrastructure that provides a programmatic interface to manage virtual machines. This entity holds all the information to access the cloud provider (e.g. endpoints, credentials).

**ResourceTemplate** - represents a set of hardware characteristics that the user specifies when a new node is created: number of cpu/cores, amount of memory, available bandwidth, etc.

**Node** - represents a gCube node created on a cloud provider and running a gCube software. The service maintain the list of all the nodes created on the managed cloud providers with information on the status, usage and workload of the node.

**ServiceProfile** - represents the information to identify a gCube software or service registered in the infrastructure. This entity is used to describe what is running on a gCube cloud node.

**NodeTemplate** - holds the information to create a new gCube node on a given cloud provider. For instance, references to the Virtual Appliances on FedCloud AppDB are kept in this entity.

# Appendix II. REST API

This appendix briefly document the REST API exposed by the FHNManager service implemented.

| | |
|---|---|
| URL | GET **/nodes**?vmProvider={vmProviderId}&serviceProfile={serviceProfileId} |
| Description | Returns a list of nodes filtered by following parameters:<br>• *vmProvider*: the id of the VMProvider that hosts the node<br>• *serviceProfile*: the id of the Service Profile running on the node<br>Filters are optional and can be omitted |

| | |
|---|---|
| URL | GET /**nodes/{id}** |
| Description | Returns the node identified by *id* |

| | |
|---|---|
| URL | POST **/nodes**?cloneFrom={id} |
| Description | Creates a new node. Following parameters must be provided in the request body:<br>• *serviceProfile*: the id of the Service Profile that will run in the new node<br>• *resourceTemplate:* the id of the Resource Template to use to create the virtual machine<br>• *vmProvider*: the id of the VMProvider that will host the node<br>If *clonedFrom* is provided, the parameters needed will be cloned frome the existing node identified by *id* |

| | |
|---|---|
| URL | UPDATE /**nodes/{id}/start** |
| Description | Starts the node identified by *id* |

| | |
|---|---|
| URL | UPDATE /**nodes/{id}/stop** |
| Description | Stops the node identified by *id* |

| | |
|---|---|
| URL | DELETE /**nodes/{id}** |

| | |
|---|---|
| Description | Deletes the node identified by *id* |

| | |
|---|---|
| URL | GET **/vmproviders**?serviceProfile={serviceProfileId} &resourceTemplate = {resourceTemplateId} |
| Description | Returns a list of VMProviders filtered by following parameters:<br>• *serviceProfile*: the id of ServiceProfiles available at the VMProvider<br>• *resourceTemplate:* the id of the Resource Template available on the VMProvider<br>Filters are optional and can be omitted |

| | |
|---|---|
| URL | GET /**vmproviders/{id}** |
| Description | Returns the VMProvider identified by id |

| | |
|---|---|
| URL | GET **/resourcetemplates**?serviceProfile={serviceProfileId}&vmProvider = {vmProviderId} |
| Description | Returns a list of ResourceTemplates filtered by the following parameters:<br>• *serviceProfile*: the id of the ServiceProfile available for the ResourceTemplate<br>• *vmProvider*: the id of the VMProvider offering the ResourceTemplate |

| | |
|---|---|
| URL | GET /**resourcetemplates/{id}** |
| Description | Returns the ResourceTemplate identified by *id* |

| | |
|---|---|
| URL | GET /**serviceprofiles/{id}** |
| Description | Returns the list of ServiceProfiles |