ИПМ им.М.В.Келдыша РАН • Электронная библиотека

Препринты ИПМ • Препринт № 168 за 2018 г.

ISSN 2071-2898 (Print)
ISSN 2071-2901 (Online)

Коваленко В.Н., Коваленко Е.И.

Новый этап развития грида: грид из облаков

Рекомендуемая форма библиографической ссылки: Коваленко В.Н., Коваленко Е.И. Новый этап развития грида: грид из облаков // Препринты ИПМ им. М.В.Келдыша. 2018. № 168. 22 с. doi:10.20948/prepr-2018-168
URL: http://library.keldysh.ru/preprint.asp?id=2018-168

Ордена Ленина
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ
имени М.В. Келдыша
Российской академии наук

В.Н. Коваленко, Е.И. Коваленко

Новый этап развития грида: грид из облаков

Москва — 2018

Коваленко В.Н., Коваленко Е.И.

Новый этап развития грида: грид из облаков

За последние несколько лет в Европе создан новый тип глобально распределённой среды вычислительных ресурсов, в которой сочетаются технологии грида и облачного компьютинга. Основная цель этой инициативы – расширить круг прикладных специалистов, использующих распределённые ресурсы для масштабных вычислений. В статье описывается программная архитектура и средства для поддержки пользовательской деятельности в этой среде.

Ключевые слова: грид, облачные технологии, распределенные вычисления, управление образами ВМ, управление приложениями, научный портал

Kovalenko Victor Nikolaevich, Kovalenko Evgeniia Ivanovna

New stage of grid development: grid from clouds

Over the past few years, a new type of globally distributed computing resource environment has been developed in Europe, combining grid and cloud computing technologies. The main goal of this initiative is to expand the range of application specialists, which make use of distributed resources for large–scale calculations. The article describes the software architecture and tools to support user activities in this environment.

Key words: grid, cloud technologies, distributed computing, VM image management, application management, science portal

1. Введение

За двадцатилетие развития грид-компьютинг получил широкую известность как способ решения задач из различных научно-технических областей вычислительными методами, предъявляющими повышенные требования к количеству необходимых компьютерных ресурсов. Способ основан на концепции, согласно которой обеспечивается унифицированный доступ к множеству распределённых в глобальной сети ресурсов и тем самым создаются условия для выполнения вычислений на их совокупности. В соответствии с этой концепцией построены международные грид-инфраструктуры из сотен тысяч компьютеров, ресурсы которых коллективно используются десятками тысяч пользователей.

Грид-компьютинг имеет очевидные достижения, выразившиеся в решении научных задач первостепенной важности. Однако опыт эксплуатации грид-инфраструктур выявил проблемные стороны программных средств, которые применяются для их создания и поддерживают пользовательскую деятельность. К проблемным вопросам относятся: эффективность использования ресурсов, обслуживание прикладного программного обеспечения устанавливаемого на компьютерах инфраструктуры, организация процесса вычислений, которая накладывает ограничения на возможные типы приложений.

Решение этих проблем даёт другое направлении распределённых вычислений – Облачный компьютинг. В облачном компьютинге большое количество ресурсов образуется, в отличие от грида, централизованно – путём их объединения в ресурсную среду (Облако) с общим управлением. Однако разработанные в этом направлении технологии могут быть использованы и в рамках глобально распределённых вычислений, соответствующих концепции грида. Распределённую архитектуру, в которой совмещаются интеграционный принцип грида, а в качестве ресурсных составляющих грид-инфраструктур выступают облака, можно назвать Облачным гридом.

Облачный грид создан и функционирует в рамках Европейской грид-инфраструктуры EGI (European Grid Infrastructure). В работе рассматриваются принцип интеграции компьютерных ресурсов в облачный грид, программная архитектура, обеспечивающая его функционирование, а также средства, которыми располагают пользователи для доступа к ресурсам и управления вычислениями.

2. Грид: достижения и проблемы

2.1. Концепция и основные результаты грида

Концепция грида возникла как ответ на вызовы, которые связаны с проведением компьютерных расчётов для решения достаточно многочисленного класса “больших” задач, поставленных в различных научно-технических областях. Задачи, названные большими, характеризуются тем, что

для их решения необходимы объёмы ресурсов, которые превосходят возможности компьютерных систем, имеющихся в отдельных учреждениях или корпоративных центрах коллективного пользования.

Многие годы возрастающие потребности прикладной вычислительной деятельности обеспечивались совершенствованием характеристик и наращиванием состава аппаратной базы, за счёт чего планка больших задач повышалась. Грид предложил новый подход, дополняющий это направление роста и позволяющий решать большие задачи на существующей аппаратной базе. Суть подхода состоит в том, что такие задачи могут решаться не на отдельных установках, а на совокупности ресурсов многих компьютерных систем. Концепция грида, изначально сформулированная в работе [1], определяет способ образования общего поля (грид-инфраструктуры) из распределённых в глобальной сети ресурсов и обеспечения к ним удалённого доступа из приложений. Интеграция ресурсов в грид-инфраструктуру не меняет, естественно, совокупный состав и общие объёмы ресурсов, однако создаются условия, когда любая их часть или даже все ресурсы могут использоваться одной задачей.

Концепция грида имеет весьма общий характер: в качестве ресурсов рассматриваются собственно компьютеры, устройства хранения данных, программы и файлы, то есть фактически любые элементы, участвующие в компьютерной обработке. Доступ к ресурсам – это дистанционное выполнение содержательных операций, например: запуск и управление программами на удалённых компьютерных установках, размещение файлов на распределённых в сети устройствах хранения, программный доступ к удалённым файлам. В данной работе мы ограничиваемся вопросами организации и использования вычислительного грида, то есть компьютерными ресурсами.

Развитие грида продолжается уже около 20 лет. К главным итогам можно отнести то, что концепция грида получила практическое воплощение, а грид стал распространённым и труднозаменимым инструментом исследований во многих областях науки.

Широкую известность гриду принесло решение двух фундаментальных задач физики, связанных с определением границ применимости современных теорий о строении вещества, структуры пространства и времени. Первая задача – подтверждение существования бозона Хиггса решена к 2013 году [2], а в феврале 2016 года объявлено об обнаружении гравитационных волн [3], предсказанных Эйнштейном. Решение обеих задач опирается на эксперименты, проведённые на уникальных физических установках. Поиски бозона Хиггса велись на Большом адронном коллайдере (БАК) – самом мощном ускорителе заряженных частиц, построенном в ЦЕРНе (Европейский Центр ядерных исследований), а гравитационных волн – на двух лазерно-интерферометрических обсерваториях гравитационных волн LIGO (Laser Interferometer Gravitational-Wave Observatory).

Характерная черта обоих экспериментов – порождение огромных потоков данных, например, годовой объём данных, получаемых на БАК, достигает 30 петабайтов (https://home.cern/about/computing). Решающую роль для получения конечных результатов сыграло использование грид-инфраструктур [4], которые, во-первых, агрегировали необходимые компьютерные ресурсы обработки и ресурсы хранения данных, и, во-вторых, позволили принять участие в их анализе коллективам специалистов разных стран мира.

В настоящее время в мире созданы и функционируют в производственном режиме несколько масштабных грид-инфраструктур. Крупнейшими из них являются EGI (European Grid Infrastructure) (https://www.egi.eu) и OSG (Open Science Grid) (https://www.opensciencegrid.org).

EGI – это федеративная европейская инфраструктура, объединяющая свыше 350 вычислительных центров из 24 стран, включая Российскую федерацию. По данным на 2017 год количество интегрированных в EGI ресурсов составляет 800 000 процессорных ядер и 290 петабайт дисковой памяти. Эти ресурсы активно используются: обслуживаются 45000 зарегистрированных пользователей, обрабатывается более 1,7 миллиона вычислительных заданий в день (https://accounting.egi.eu). В отличие от EGI, OSG является грид-инфраструктурой одной страны – США, но их масштабы сравнимы: OSG включает 126 ресурсных центров с общим количеством ресурсов примерно в два раза меньшем, чем в EGI.

Использование грида далеко не ограничивается физикой и уникальными задачами. В число научных областей применения грида входят естественные науки (математика, науки о Земле, биология, химия, астрономия), а также сельскохозяйственные, инженерные, медицинские, гуманитарные (история, археология, литература) науки. Информация о задачах, которые решаются с помощью грида, публикуется в периодических отчётах и на web-сайтах грид-инфраструктур [5,6]. Наиболее интересные приложения освещаются в сетевом издании Science Node (https://sciencenode.org). Имеется статистика количества публикаций по результатам работ с использованием грид-инфраструктур: пользователями EGI в открытом доступе с 2008 года опубликовано 20150 работ, из них 3300 работ за 2016 год [7].

2.2. Принципы организации и программное обеспечение грида

К принципам, определяющим концепцию грида, мы относим следующие положения.

- Грид-инфраструктура образуется из множества глобально распределённых узлов, содержащих компьютерные ресурсы. Обычно узлами выступают не отдельные компьютеры, а достаточно мощные аппаратные комплексы – компьютерные кластеры. Каждый узел имеет собственную систему управления с сетевыми интерфейсами, через которые поступают запросы на выполнение поддерживаемых узлом функций. Предполагается, что узлы гетерогенны, то есть могут различаться как по интерфейсам систем

управления, так и по программно-аппаратным платформам ресурсов.

- Глобальный доступ ко всему множеству ресурсов, интегрированных в грид-инфраструктуру, обеспечивается за счёт унификации – определения общих, независимых от конкретных систем управления узлами, протоколов и форматов удалённых запросов для выполнения операций над ресурсами. Интерпретация унифицированных запросов возлагается на компоненту программного обеспечения (ПО) грида, играющую роль шлюза для доступа в локальную среду ресурсов. Шлюз устанавливается на каждый узел, включённый в грид-инфраструктуру, и преобразует поступающие запросы из унифицированной формы в запросы соответствующих систем управления.
- Ресурсы грид-инфраструктуры разделяются между пользователями. Необходимую координацию использования ресурсов осуществляют неформальные органы управления гридом – виртуальные организации. Виртуальная организация представляет собой объединение пользователей, которые работают в одной или смежных прикладных областях и решают схожие задачи с помощью общего прикладного обеспечения. В плане разделения ресурсов функция виртуальной организации – заключение соглашений с провайдерами ресурсов и определение политики их предоставления пользователям. Грид применяется в научно-исследовательской сфере, поэтому ресурсы выделяются провайдерами в грид-инфраструктуру бесплатно, обычно стимулом для них является заинтересованность в решении определённых задач.

Концепция грида реализована в ряде программных комплексов, которые обеспечивают создание грид-инфраструктур и поддерживают их функционирование. В настоящее время наиболее востребованы различные версии комплексов Globus Toolkit, gLite и UNICORE. Несмотря на определённые различия, состав реализованных в них функций аналогичен и включает: управление вычислениями и данными, обеспечение безопасности, информационное обслуживание, мониторинг ресурсов и учёт их использования. Не претендуя на обзор всего многообразия имеющихся средств, ограничимся характеристикой компонент, относящихся к управлению вычислениями.

Единицей вычислительной деятельности в гриде, построенном с помощью этих комплексов, является задание. Задание формулируется пользователем в виде стандартизированного описания, в котором задаются требования к ресурсам (платформа, число и характеристики процессорных единиц, время выполнения и пр.), а также исполняемый файл, входные и выходные файлы. Управление заданиями осуществляется посредством пользовательского интерфейса, с помощью которого задание может быть послано для выполнения на конкретный ресурсный узел грида. Выбор исполнительного ресурса может также осуществляться автоматически через посредство брокера, который распределяет задания исходя из текущей загрузки узлов и ресурсных требований заданий. Примером брокера является Workload Management System

(WMS) [8] комплекса gLite. Помимо запуска заданий в состав операций управления входит также опрос состояния обработки, снятие задания и получение результирующих файлов.

Среди серверных компонент ПО грида важное место занимает упомянутый выше шлюз в локальную среду ресурсов (в комплексе Globus Toolkit, например, он называется GRAM – Globus Resource Allocation Manager). Основная функция шлюза – выполнение запросов по управлению заданиями в локальной среде узла. Выполнение не сводится к преобразованию запроса в команды локальной системы управления ресурсами. Так, при обработке запроса на запуск задания шлюз, исходя из описания задания, размещает в локальной среде узла исполняемый файл и входные файлы данных, а также передаёт на рабочее место пользователя результирующие файлы. Концепция грида предполагает, что в инфраструктуру могут включаться узлы с разными системами управления, в качестве которых выступают системы пакетной обработки в кластерах: Condor, Torque, PBS, LSF, LoadLeveler, Slurm.

В шлюзе реализуются также механизмы инфраструктуры безопасности грида AAI (Authentication and Authorization Infrastructure) [9], которая обеспечивает аутентификацию, авторизацию и защиту передаваемых данных в условиях глобально распределённой программной среды, состоящей из различных административных доменов. Базу AAI составляют протокол SSL (Secure Sockets Layer), цифровые сертификаты X.509 и шифрование с помощью пары публичный – приватный ключ. Благодаря AAI обеспечивается аутентификация в гриде в целом: однократно пройдя аутентификацию, пользователь получает доступ потенциально ко всем узлам. Прошедшие авторизацию задания выполняются на ресурсах под отдельными учётными записями и защищены средствами операционной системы (ОС). Шлюз также реализует механизм делегирования прав доступа. Делегирование заключается в том, что одна программа может безопасно передать права, с которыми она выполняется, другой программе.

2.3. Проблемные стороны традиционного грида

Разработанные к настоящему времени комплексы ПО с достаточной полнотой поддерживают создание, эксплуатацию и вычислительную деятельность в гриде, о чём свидетельствует наличие широкомасштабных постоянно действующих грид-инфраструктур и полученные в них результаты. Однако практика выявила и проблемы.

Платформа исполнительных ресурсов. Прикладное ПО разрабатывается в расчёте на определённую программно-аппаратную платформу: архитектуру компьютеров, ОС и её версию. Хотя концепция грида допускает включение в инфраструктуру компьютеров с разными платформами, круг возможных исполнительных ресурсов для приложения ограничивается его требованиями к платформе. На практике это приводит к тому, что провайдеры ресурсов грида ориентируются на платформы, которые требуются для решения важнейших

задач из отдельных прикладных областей, а возможности постановки в гриде иных задач остаются весьма ограниченными.

Среда выполнения приложений. Помимо определённой платформы прикладным программам, как правило, необходима своя рабочая среда, включающая системные и прикладные библиотеки, а также, возможно, приложения общего назначения, например, web-сервер. В реалиях современного грида деятельность по обслуживанию приложений возлагается на провайдеров ресурсов. Виртуальная организация заключает соглашения с кругом провайдеров, которые имеют компьютеры с нужной архитектурой и готовы их предоставлять для задач, решаемых виртуальной организацией. На провайдеров ложится серьёзная работа по установке и сопровождению рабочей среды для каждого обслуживаемого приложения на всех выделяемых под него компьютерах. Неудивительно, что большая часть ресурсов грида занимается крупными прикладными задачами, решаемыми в рамках хорошо организованных виртуальных организаций. Этот вывод подтверждается статистикой портала грида EGI (https://accounting.egi.eu): до 96% объёма ресурсов потребляется четырьмя виртуальными организациями ядерной физики – atlas, cms, alice, lhcb, занимающихся обработкой данных БАК.

Типы приложений. В гриде поддерживается единственная форма вычислительной деятельности – пакетная обработка заданий. Это обусловлено тем, что доступ к ресурсам осуществляется через посредство систем управления кластерами. Режим пакетной обработки накладывает ограничения на тип обслуживаемых в гриде приложений. Во-первых, прежде чем задание начинает выполняться, оно находится в состоянии ожидания в очереди кластера, так что время старта приложения трудно предсказуемо. Во-вторых, заказанные в задании ресурсы выделяются приложению только на ограниченное время. В-третьих, приложение не получает IP-адреса, и взаимодействие с ним во время выполнения невозможно. Эти свойства пакетной обработки делают грид малопригодным для таких перспективных типов приложений как интерактивные, потоковые и постоянно действующие сервисы.

2.4. Решение проблем традиционного грида с помощью облачных технологий

Перечисленные проблемы традиционного грида получили решение в другом направлении распределённых вычислений – Облачном компьютинге [10]. Облачный компьютинг, также как и грид, ставит цель интеграции большого количества ресурсов и обеспечение к ним удалённого доступа, однако предлагает иной способ организации ресурсов и иную форму их использования.

В вопросе организации ресурсов грид основывается на распределённом подходе: ресурсная инфраструктура создаётся путём программной интеграции автономных ресурсных узлов с собственными системами управления.

Облачный компьютинг основан на одноуровневой организации ресурсной инфраструктуры в виде облака. Ресурсы облака управляются одной системой, которая помимо прочего обеспечивает к ним непосредственный удалённый доступ.

Начиная с 2006 года, ведущими компаниями информационной индустрии [11-13] созданы облака, которые содержат от 50000 до 80000 серверов. Компания Amazon, например, к 2014 году имела 20 облаков, включающих по консервативной оценке 1,5 миллиона серверов [14]. То есть по объему накопленных ресурсов облака, по крайней мере, сравнимы, если не превосходят современные грид-инфраструктуры.

Покажем, каким образом с помощью технологий, применяемых в облаках, решаются проблемы грида. Базовой облачной технологией является виртуализация компьютеров, которая позволяет устанавливать на одном реальном компьютере несколько виртуальных машин (ВМ) с разными операционными средами и с собственным набором логических ресурсов (процессорных, оперативной памяти, устройств хранения). Компьютеры в облаке работают под управлением гипервизора, реализующего виртуализацию, а ВМ устанавливается в виде гостевой системы. Современные гипервизоры (Xen, VirtualBox, VMware, KVM, QEMU и др.) способны работать на различных аппаратных архитектурах и поддерживать широкий спектр ОС гостевых систем, включая различные варианты Linux и Windows. Таким образом, виртуализация компьютеров даёт решение проблемы платформы исполнительных ресурсов.

Проблема рабочей среды решается за счёт того, что пользователь облака получает в распоряжение собственную ВМ, характеристики которой отвечают его требованиям. Понятие ВМ в облаке имеет тот же смысл, что и понятие ВМ на реальном компьютере, с тем отличием, что ВМ устанавливается на ресурсах облака. Важно, что ВМ подготавливается пользователем в виде Образа, представляющего собой файл, в котором в специальном формате содержится полностью сконфигурированная программная среда. Помимо ОС в образ могут быть включены пользовательские приложения, инструментальные средства, прикладные библиотеки, серверы приложений. Файл образа передаётся в облако, и на его основе производится автоматическая установка ВМ.

При установке ВМ в облаке пользователь имеет возможность определить требования к ресурсам, которые должна получить ВМ. В составе характеристик ресурсов: архитектура, число ядер, производительность ЦПУ, объём оперативной памяти. Могут быть заказаны также сетевые ресурсы, включая доменное имя и диапазон IP-адресов, благодаря чему серверные приложения, установленные в ВМ, становятся доступными в глобальной сети.

Одним из основных свойств облачного компьютинга является гарантированное выделение ресурсов (в рамках контракта пользователя с провайдером) и быстрый старт ВМ (время порядка минуты) [15]. В сочетании с наличием глобального сетевого доступа это создаёт основу для решения

третьей проблемы традиционного грида – возможности выполнения интерактивных, потоковых приложений и постоянно действующих сервисов.

Функции удалённого создания, управления ВМ и выделения для них ресурсов в облаке реализуются во всех облачных платформах в виде сервисов, соответствующих модели обслуживания Инфраструктура Как Сервис (IaaS – Infrastructure as a Service). Однако поддержку собственно вычислительной деятельности, сопоставимой с имеющейся в гриде, эта модель не предусматривает. Доступ к ВМ становится возможным лишь по удалённым протоколам ОС, например SSH, либо за счёт предусмотренных в ней программных средств.

Для Облачного компьютинга определены ещё две модели [10]: Платформа Как Сервис (PaaS – Platform as a Service) и ПО Как Сервис (SaaS – Software as a Service), которые также как и модель IaaS рассматриваются как стандартные. PaaS является моделью обслуживания, в которой провайдер предоставляет потребителю готовую платформу, которая позволяет ему разрабатывать и выполнять приложения в облаке. Примером реализации PaaS может служить Google App Engine (https://cloud.google.com/appengine). В модели SaaS провайдер устанавливает в облаке приложения, например, почтовый сервер или систему моделирования, с которыми работают удалённые пользователям, обычно через web-браузер. PaaS и SaaS поддерживаются во многих облачных платформах, однако в их реализациях используются частные решения, ориентированные на специфические средства разработки приложений и имеющие ограничения на языки программирования. В этом их отличие от сервисов IaaS, для которых определены и стандартизованы протокол и прикладной интерфейс, что создаёт возможности для разработки интероперабельных средств управления ресурсами и виртуальными машинами.

3. Грид на основе облаков

Известны два способа использования облачных технологий для решения больших задач. Первый способ – создание крупномасштабных облаков. Современные публичные облака содержат десятки тысяч компьютеров. Хотя они ориентированы на рядовые задачи и обслуживание массового пользователя, сопоставимые по объёму ресурсов облака могут быть созданы и в интересах научно-технической сферы. Второй способ – интеграция облаков различных провайдеров в соответствии с концепцией грида. Оба способа имеют свои достоинства. В облаке, например, за счёт применения скоростных линий связи могут поддерживаться высокопроизводительные вычисления. С другой стороны, принцип интеграции грида открывает дополнительные возможности по масштабированию, делая доступным суммарный объём ресурсов, содержащихся во множестве облаков.

Как показывает детальный анализ [16], технологии грида и облаков дополняют друг друга и могут использоваться совместно. Архитектуру, совмещающую преимущества грида и облаков, можно назвать Облачным

гридом. Облачный грид отличается от традиционного тем, что его составляющими служат не кластерные среды, а облака. В этой архитектуре роль облачных технологий – управление ресурсной базой в отдельных облаках и обеспечение удалённого доступа к ресурсам, а технологии грида решают задачи по поддержке деятельности в условиях глобально распределённой среды, объединяющей облака разных провайдеров.

3.1. Облачный грид Европейской грид-инфраструктуры

Облачный грид существует: с мая 2014 года в составе Европейской грид-инфраструктуры EGI наряду с традиционным гридом функционирует облачный грид EGI Federated Cloud (EGI FC) [17]. В настоящее время в EGI FC интегрированы облака 20 провайдеров с общим количеством ресурсов 6000 ядер ЦПУ и 300 Тбайт дисковой памяти. В числе провайдеров Россия представлена Объединенным институтом ядерных исследований [18,19].

Цель создания грид-инфраструктуры нового типа состояла в реализации описанных в п. 2.4 возможностей, которыми обладают облака, при условии сохранения программных технологий традиционного грида, поддерживающих деятельность в распределённой сетевой среде [20].

Базовым принципом интеграции в EGI FC является обеспечение непосредственного доступа ко всему множеству ресурсов содержащихся в инфраструктуре облаков в соответствии с моделью обслуживания IaaS. В отличие от кластеров традиционного грида платформы управления облаками поддерживают доступ к своим ресурсам по глобальной сети, но для образования общей инфраструктуры ресурсов необходима унификация интерфейсов доступа. Однако в организациях – фактических и потенциальных участниках EGI FC облака построены на основе различных платформ управления, которые различаются не только функциональными возможностями, но и по интерфейсам сервисов. В EGI FC выбран компромиссный подход: включение в грид разноплатформенных облаков допускается, но при условии, что они поддерживают открытые стандарты.

Серию стандартов Open Cloud Computing Interface (OCCI) (http://occi-wg.org/about/specification), которые обеспечивают интероперабельность между различными облачными платформами, разработала организация Open Grid Forum (OGF). Документ [21] определяет прикладные интерфейсы для взаимодействия с сервисами IaaS облачных платформ. Основные положения этого документа:

- доступ к ресурсам осуществляется путём создания ВМ;
- в операции создания ВМ определяются требования к ресурсам и расположение образа ВМ;
- управление ВМ осуществляется операциями: старт, стоп, рестарт, приостановка, сохранение образа ВМ.

К настоящему времени стандарты OCCI поддерживаются лишь тремя облачными платформами с открытым кодом: OpenNebula, OpenStack и Synnefo,

причём поддержка реализована в дополнительных компонентах: rOCCI (http://gwdg.github.com/rOCCI), которая разработана для OpenNebula, но может также быть использована для платформ CloudStack и Amazon Web Services (AWS), OOI (https://github.com/openstack/ooi) для OpenStack, snf-occi (https://github.com/itminedu/snf-occi) для Synnefo.

Соответствие стандартам OCCI не единственное условие для включения облака в инфраструктуру EGI FC. Провайдер облака должен также обеспечить интеграцию своей платформы с сервисом управления образами ВМ и сервисами поддержки функционирования EGI FC (рис. 1) [20].

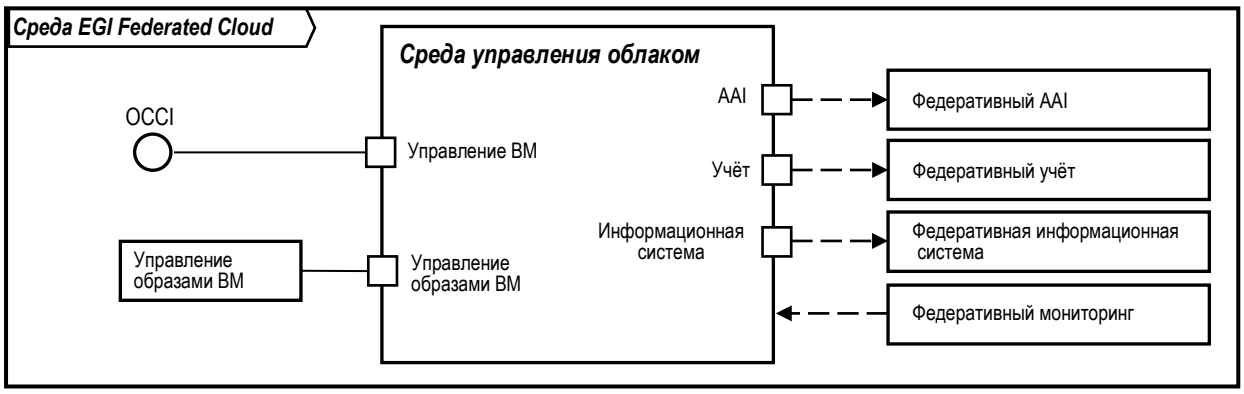

Рис. 1. Программная архитектура управления компьютерными ресурсами EGI FC.

Сервис управления образами ВМ – AppDB (Applications Database) (https://appdb.egi.eu) реализует централизованное хранение образов, а также поддерживает доступ к базе данных, в которой содержится содержательная информация, аннотирующая образы. Централизация хранения решает две задачи. Во-первых, создаются условия для коллективного использования образов: образы общего назначения становятся доступными всем пользователям EGI FC, а Образы, созданные под определённый прикладной проект, могут использоваться всеми его участниками (обычно они объединены в виртуальную организацию). Во-вторых, AppDB выполняет автоматическую доставку Образов для установки в различных облаках. Доставка осуществляется программно с помощью механизма подписки на уведомления об изменениях в составе образов, хранящихся в AppDB. Соответствующий механизм реализуется ПО vmcatcher (https://github.com/hepix-virtualisation/vmcatcher). Для интеграции с сервисом AppDB требуется установка vmcatcher в облаке и подписка на списки образов тех ВО, которые в нём обслуживаются.

Сервисы поддержки функционирования EGI FC аналогичны тем, которые используются в инфраструктуре традиционного грида EGI – EGI Core Infrastructure. В их составе сервисы: (1) контроля удалённого доступа к ресурсам на основе инфраструктуры безопасности грида AAI, (2) учёта

потребления ресурсов, (3) информационного обслуживания, (4) мониторинга работоспособности сервисов. Контроль доступа к ресурсам, осуществляемый сервисами (1), выполняется на основе сертификата X.509, расширенного атрибутами, которые определяют виртуальную организацию пользователя. Сервисы (2), (3) агрегируют данные соответствующего типа, получая эти данные от облаков, интегрированных в инфраструктуру. В поддерживаемых облачных платформах реализованы внутренние механизмы сбора данных, а для их поставки в сервисы федеративного объединения разработаны специальные компоненты, работающие на стороне провайдера.

3.2. Пользовательская деятельность в EGI FC по управлению виртуальными машинами

Ссылки на различные руководства по пользовательской деятельности в EGI FC размещены на странице [22], дальнейшее изложение опирается на эти материалы.

Основная пользовательская деятельность в гриде – вычислительная, заключающаяся в выполнении приложений на удалённых ресурсах. Вспомогательной работой – установкой на ресурсы программной среды – в традиционном гриде занимаются провайдеры, а в Облачном гриде на пользователей возлагаются задачи создания и управления ВМ.

ВМ создаются из образов, предпочтительной формой которых в EGI FC является Virtual Appliance (VA). В отличие от других формы распространения ПО в VA содержится полностью установленная и сконфигурированная программная среда, включающая ОС, прикладное обеспечение и пользовательские программы. Благодаря этому установка VA на сервер облака в качестве гостевой ВМ может происходить автоматически без участия или с минимальным участием пользователя.

Для создания VA рекомендуется применение инструментального средства Packer (https://www.packer.io). К его достоинствам можно отнести следующее.

- Packer может производить образы для многих платформ виртуализации, например для EC2, VMware, VirtualBox.
- Образ может быть получен как копия существующей ВМ или реального компьютера, либо сформирован “с нуля” путём установки и конфигурирования отдельных элементов программной среды ВМ. Во втором варианте Packer опирается на современные средства управления конфигурацией ПО (Chef, Puppet и др.).

Деятельность по созданию VA аналогична той, которую выполняют системные администраторы в локальной корпоративной среде, и она также требует владения соответствующим инструментарием. Практическое решение, которое используется в EGI FC, состоит в том, что в состав виртуальных организаций включаются специалисты со статусом разработчика. Их роль – создание VA для предметной области виртуальной организации и загрузка подготовленных VA в хранилище образов сервиса AppDB, благодаря чему они

становятся доступны прикладным специалистам. Такой подход оправдывает себя – в настоящее время в AppDB содержится более 150 VA разной степени общности и для разных научных областей, например VA, содержащие:

- операционные системы Ubuntu 14, Centos 6;
- среду программирования COMP Superscalar (COMPSs) разработки приложений для распределённых инфраструктур;
- портал BioVel для решения вычислительных задач в науках о биологическом разнообразии.

Что касается средств управления ВМ, то здесь базовый уровень составляют прикладной программный интефейс (jOCCI для языка Java и rOCCI для Ruby) и интерфейс командной строки (rOCCI-cli), которые реализуют операции доступа к сервисам отдельных облаков в соответствии со стандартом OCCI. Для прикладных пользователей имеется комплексное решение – Web-приложение VMOps Dashboard [23].

В VMOps Dashboard установка ВМ производится из образов, которые загружены в хранилище сервиса AppDB. Образы выбираются с помощью интерфейса поиска по значениям описывающих их атрибутов, в числе которых: ОС, аппаратная платформа, виртуальная организация, в которой разработан образ, предметная область, поддерживаемые языки программирования и пр. Интерфейс поддерживает также заказ ресурсов для установки ВМ. В форме заказа представлены список облаков, доступных пользователю, и панель для определения характеристик ресурсов, которые должны быть выделены для ВМ. Интерфейс VMOps Dashboard позволяет также выполнять операции запуска и остановки ВМ.

3.3. Вычислительная деятельность в EGI FC

Облачный грид EGI FC образуется на основе стандартов сервисов IaaS, функции которых ограничены управлением ВМ. Основная для научно-технической сферы пользовательская деятельность – выполнение приложений на ресурсах облаков этими сервисами не поддерживается, хотя пользователь и получает доступ к ВМ по сетевым протоколам. Управление вычислениями эффективно организовано в традиционном гриде на основе понятия задания, в котором определяются исполняемый файл, входные и выходные данные, параметры запуска и требования к ресурсам. Модель управления заданиями имеет абстрактный характер и применима для различных ресурсных инфраструктур, в том числе для облачного грида.

Однако, как показано в п. 2.4, для постановки в грид интерактивных, потоковых приложений и сервисов требуется изменить пакетный режим обработки заданий, применяемый в традиционном гриде. Изменение заключается в том, что в облачном гриде реализуется непосредственная доставка приложений на постоянно выделенные ресурсы облаков и их запуск без ожидания в очереди. Рассмотрим имеющиеся в EGI FC средства для

ведения вычислительной деятельности и обеспечивающие её механизмы управления заданиями.

Вычислительная деятельность в EGI FC поддерживается клиентским ПО в форме Научных порталов (Science Gateway – SG). Научный портал является web-приложением, в котором реализуются наиболее сложные механизмы управления заданиями – контроля доступа, доставки файлов на облачные ресурсы, запуска/остановки программ, опроса состояния выполнения и т.д. Очевидное достоинство такого подхода – пользователи избавляются от необходимости установки и обновления ПО для работы в гриде на своём рабочем месте.

Возможно более существенно, что научные порталы минимизируют участие прикладного пользователя в подготовке заданий для запуска приложений. В конечном счёте, задание представляет собой текстовый файл, в котором описываются программные объекты приложения, их взаимодействие, а также определяется адреса исполняемых файлов и файлов данных. Важным свойством научных порталов является то, что для работы с приложением знания реализационных деталей не требуется, обычно для его запуска пользователь должен определить лишь расположение входных файлов и файлов результатов. К настоящему времени в EGI создано несколько десятков научных порталов для различных дисциплин [24]. Характерный пример – портал Chipster (https://chipster.csc.fi), относящийся к области биохимии и содержащий более 360 приложений по секвенированию генома и анализу белков живых организмов.

Разработка научных порталов “с нуля” является непростым делом, но она существенно упрощается за счёт применения инструментально-базовых программных средств – типовых порталов. Опора на типовые порталы имеет ряд преимуществ. Их разработка ведётся на долговременной основе специалистами в сфере системного ПО. Известные типовые порталы имеют модульную архитектуру, компоненты которой решают отдельные задачи, что облегчает их развитие. Как показывает практика, использование типовых порталов не только ускоряет разработку специализированных порталов, но и наделяет их необходимыми технологическими качествами.

Типовой портал состоит из двух частей. Базовая часть содержит программные компоненты, которые непосредственно включаются в состав специализированных порталов. В первую очередь, это компоненты, реализующие операции и механизмы, соответствующие абстрактной модели управления заданиями, которые универсальны и не зависят от предметной области. Инструментальная часть типового портала предназначается для его настройки на прикладную деятельность в рамках определённой предметной области – описания поддерживаемых приложений и создания интерфейсов для прикладных пользователей.

В EGI введены в действие два типовых портала: Catania Science Gateway Framework, на основе которого разработано 15 специализированных порталов

[25], и WS-PGRADE/gUSE [26] – 8 порталов. Далее рассматривается разработка научных порталов на примере последней системы.

3.4. Разработка научных порталов с помощью WS-PGRADE/gUSE

Являясь инструментально-базовым комплексом, WS-PGRADE/gUSE [27] содержит средства по:

- подготовке приложений для их выполнения в ресурсных инфраструктурах;
- созданию пользовательского интерфейса для запуска приложений;
- запуску приложений на удалённые ресурсы и управлению ими.

Использованные в WS-PGRADE/gUSE решения имеют весьма общий характер. Во-первых, поддерживается выполнение приложений в инфраструктурах нескольких типов – облаках, гридах и кластерах. Во-вторых, WS-PGRADE/gUSE ориентирован на многокомпонентные приложения – потоки, которые состоят из нескольких программ, связанных друг с другом по данным, а элементарные приложения, состоящие из единственной программы, рассматриваются как частный случай.

Подготовка приложения включает два этапа: 1) описания структуры потока и 2) конфигурирования. Структура приложения описывается ориентированным графом, который строится с помощью графического редактора (рис. 2). Узлы графа соответствуют выполняемым в потоке программам. Узлам приписываются порты: входной порт соответствует получаемому программой файлу, выходной порт – файлу, который в ней порождается. Ребро графа, соединяющее выходной порт одного узла с входным портом другого, определяет передачу данных между программами.

На этапе конфигурирования узлам и портам построенного абстрактного графа присваиваются конкретные значения, которые вводятся в Web-формах графического интерфейса. Конфигурация узла определяется программой и ресурсом для её выполнения. WS-PGRADE/gUSE поддерживает два типа программ: исполняемые файлы и web-сервисы. Кроме того, в узле может быть задано выполнение ранее определённого и сохранённого потока. Исполняемый файл может находиться либо на пользовательском компьютере, с которого производится запуск потока, либо он может храниться в репозитарии приложений, к которому имеет доступ WS-PGRADE/gUSE. При выполнении потока исполняемый файл доставляется на ресурс, определённый в конфигурации узла. Помимо исполняемых файлов в потоке могут быть задействованы web-сервисы. Они вызываются удалённо, описываются адресом (URL) и именем вызываемого метода.

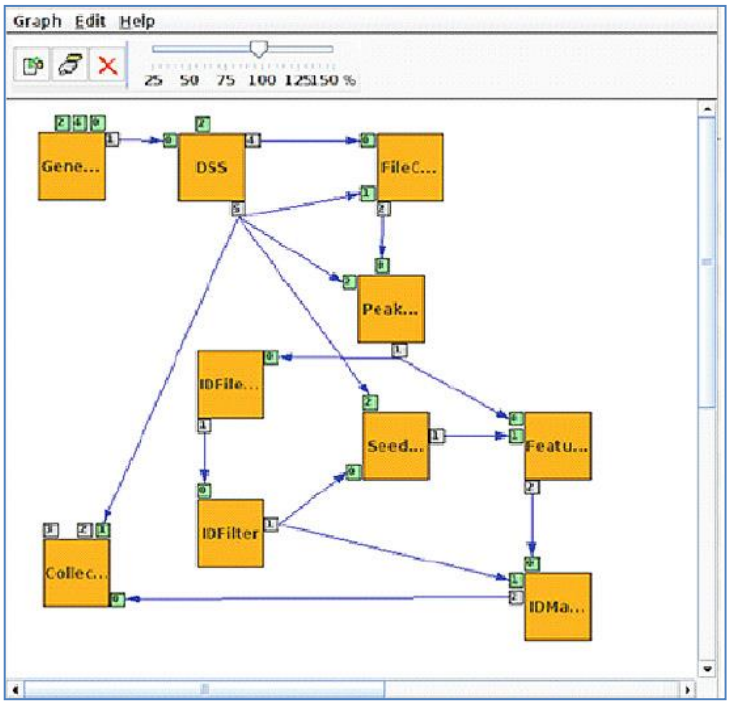

Рис. 2. Графический редактор для построения графа потокового приложения [27]

Каждая программа потока может выполняться на отдельном исполнительном ресурсе. Выбор ресурса производится по списку доступных для использования инфраструктур, в частности, облаков (рис. 3).

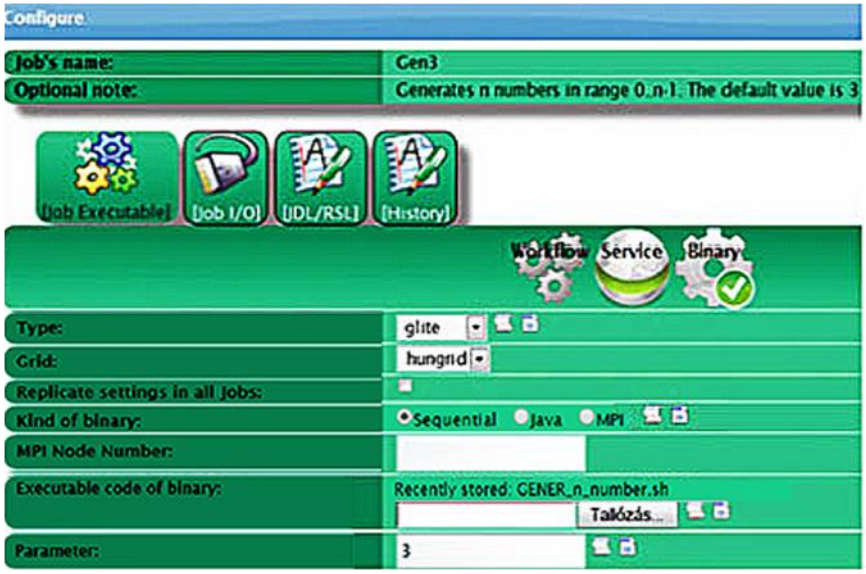

Рис. 3. Панель конфигурирования потока

В графе потока узлам приписываются порты, обозначающие файлы, которые обрабатываются (входные порты) в соответствующей программе или создаются в ней (выходные порты). Конфигурирование порта заключается в том, что ему присваивается имя файла, под которым с ним работает программа,

описанная в узле. Отметим, что потоки могут создаваться из программ, которые разрабатывались независимо. Возможность их совместного выполнения поддерживается механизмом интерпретации рёбер: для согласования имён файлов при передаче файла из выходного порта одного узла во входной порт следующего производится его переименование.

Обычно в графе имеются входные порты, у которых нет входящих рёбер. Это соответствует ситуации, когда входной файл расположен на компьютере, с которого запускается выполнение потока, или файл, размещённый в некоторой внешней системе хранения. Такие входные файлы автоматически доставляются на исполнительный ресурс программы. Аналогичная ситуация с выходными портами без исходящих рёбер, с тем отличием, что результирующие файлы не передаются на компьютер запуска, а временно помещаются на сервер портала, откуда они могут быть получены пользователями.

Графический интерфейс конфигурирования потока используется и для разработки пользовательского интерфейса запуска приложений. Если поток полностью сконфигурирован, его можно выполнить. Однако есть второй вариант: поток может быть сконфигурирован частично и сохранён в репозитарии приложений в виде шаблона. На основе шаблонов поддерживается уровень обслуживания пользователей, соответствующий модели SaaS. Прикладной пользователь, получая шаблон из репозитария, может его редактировать теми же средствами, что и разработчик шаблона, добавляя недостающие описания. Например, в шаблоне могут быть зафиксированы программы, определена передача данных и адреса исполнительных ресурсов. В этом случае для запуска потока пользователю остаётся задать имена файлов входных данных и результатов.

Предусмотрен также инструментарий для разработки оригинального пользовательского интерфейса, который заменяет или дополняет собственный графический интерфейс WS-PGRADE/gUSE, но может использовать его сервисы.

3.5. Выполнение приложений на удалённых ресурсах

При выполнении приложения WS-PGRADE/gUSE преобразует граф потока в набор заданий, которые в определённой последовательности должны быть выполнены на соответствующих ресурсах. Задание строится для каждого узла и представляет собой текстовый файл, содержащий информацию об исполнительной программе, входных и выходных данных.

Архитектура WS-PGRADE/gUSE разработана с целью обеспечения выполнения заданий в ресурсных средах разных типов. Управление заданиями в ней выделено в отдельную компоненту – web-сервис DCI Bridge. Интерфейсы DCI Bridge соответствуют стандарту OGSA Basic Execution Service 1.0 (BES) и реализуют запросы по запуску задания, опросу состояния выполнения, передаче входных и результирующих файлов. Описания заданий DCI Bridge

получает в стандартной форме языка JSDL (Job Submission Description Language) [28].

Реализация операций управления заданиями зависит от типа исполнительных ресурсов, поэтому DCI Bridge содержит расширяемый набор модулей – плагинов, отвечающих за взаимодействие с ресурсами определённого типа. Плагины для сред типа грида и кластера реализуют операции управления, обращаясь к имеющимся в этих средах системам управления. Поддерживаются гриды на основе различных версий gLite, Globus Toolkit, ARC, UNICORE, а также кластеры под управлением PBS, LSF, SGE, MOAB.

Платформы облаков не имеют средств управления заданиями, поэтому для них в DCI Bridge разработано собственное, но достаточно универсальное решение. Оно основано на том, что в облаке устанавливается дополнительная компонента DCI Bridge Slave, функция которой – запуск и управление заданиями на ВМ облака. Плагин для облаков включает ещё одну компоненту – DCI Bridge Master, которая устанавливается на сервере DCI Bridge. Получая описание задания, DCI Bridge Master стартует ВМ, используя для этого сервисы IaaS, и далее передаёт DCI Bridge Slave запросы управления заданием. Таким способом становится возможным прямой запуск заданий на облачные ресурсы, отличный от запуска посредством систем пакетной обработки.

4. Заключение

Изначально облака ориентировались на массового потребителя, но со временем в них начали применяться передовые технологии, представляющие интерес для науки. В аппаратном аспекте это, например, высокопроизводительные сети (с задержкой в несколько микросекунд) и ускорители, в том числе специализированные (тензорные процессоры TPU, программируемые интегральные схемы FPGA), в программном аспекте – микросервисы [29].

Представляется, однако, что подход грида сохраняет своё значение в научной сфере. Для науки, по-видимому, всегда будет нужно больше ресурсов, чем есть в наличии, а интеграционная концепция грида делает их доступными независимо от расположения. Кроме того, грид позволяет использовать ресурсы разных типов и масштаба, что также важно для решения некоторых задач.

EGI FC является важным шагом в деле создания гибридной архитектуры, в которой сочетаются достоинства грида и облаков. В настоящее время программный стек EGI FC обеспечивает функционирование распределённой среды ресурсов, реализуя базовую модель управления ВМ, которая соответствует имеющимся стандартам (OCCI), и с достаточной полнотой поддерживает пользовательскую деятельность, основанную на этой модели. Развитие программных средств EGI FC продолжается, и в качестве первоочередных задач рассматривается [20]:

- расширение состава поддерживаемых облачных платформ, в том числе коммерческих;
- разработка механизма миграции ВМ между провайдерами;
- стандартизация и расширение функциональных возможностей сервисов OCCI для копирования ВМ во время выполнения и динамического изменения выделенных для них ресурсов;
- обеспечение брокерских услуг, автоматизирующих выбор провайдера при выполнении конкретных приложений.

5. Библиографический список

[1]. Ian Foster, Carl Kesselman, Steven Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations // International Journal of High Performance Computing Applications. Volume 15, Issue 3, August 2001. pp. 200–222. URL: http://toolkit.globus.org/alliance/publications/papers/anatomy.pdf

[2]. New results indicate that new particle is a Higgs boson. URL: https://home.cern/about/updates/2013/03/new-results-indicate-new-particle-higgs-boson

[3]. Observation of Gravitational Waves from a Binary Black Hole Merger. Abbott B. P. et al. // Phys. Rev. Lett. 116, 061102. Published 11 February 2016. URL: https://dcc.ligo.org/public/0122/P150914/014/LIGO-150914%3ADetection_of_GW150914.pdf

[4]. How grid computing helped CERN hunt the Higgs. URL: https://sciencenode.org/feature/how-grid-computing-helped-cern-hunt-higgs.php

[5]. EGI: Research stories. URL: https://www.egi.eu/use-cases/research-stories

[6]. OSG News. URL: http://opensciencegrid.org/news

[7]. OpenAIRE. URL: https://www.openaire.eu/egi-stats

[8]. The gLite Workload Management System. Cecchi Marco et al. // Journal of Physics: Conference Series 219 062039, 2010. URL: http://iopscience.iop.org/article/10.1088/1742-6596/219/6/062039

[9]. Foster I., Kesselman C., Tsudik G., Tuecke S. A Security Architecture for Computational Grids // Proc. 5th ACM Conference on Computer and Communications Security Conference, pp. 83-92, 1998. URL: http://toolkit.globus.org/ftppub/globus/papers/security.pdf

[10]. Mell Peter, Grance Timothy. The NIST Definition of Cloud Computing // National Institute of Standards and Technology: U.S. Department of Commerce. URL: https://doi.org/10.6028%2FNIST.SP.800-145

[11]. Amazon Elastic Compute Cloud (Amazon EC2). URL: http://aws.amazon.com/ec2

[12]. Google App Engine. URL: https://cloud.google.com/appengine

[13]. Microsoft Azure. URL: https://azure.microsoft.com

[14]. 5 Numbers That Illustrate the Mind-Bending Size of Amazon's Cloud. URL: https://www.bloomberg.com/news/2014-11-14/5-numbers-that-illustrate-the-mind-bending-size-of-amazon-s-cloud.html
[15]. Ming Mao, Marty Humphrey. A Performance Study on the VM Startup Time in the Cloud // IEEE 5th International Conference on Cloud Computing, CLOUD 2012. URL: http://doi.org/10.1109/CLOUD.2012.103
[16]. Ian Foster, Yong Zhao, Ioan Raicu, Shiyong Lu. Cloud Computing and Grid Computing 360-Degree Compared // Grid Computing Environments Workshop. 12-16 Nov. 2008. URL: https://arxiv.org/pdf/0901.0131
[17]. EGI Federated Cloud. URL: https://www.egi.eu/federation/egi-federated-cloud
[18]. Многофункциональный информационно-вычислительный комплекс ОИЯИ. URL: http://micc.jinr.ru/
[19]. JINR cloud infrastructure evolution. Baranov A.V., Balashov N.A., Kutovskiy N.A. et al. // Phys. Part. Nuclei Lett. (2016) 13: 672. URL: https://doi.org/10.1134/S1547477116050071
[20]. Enol Fernández-del-Castillo, DiegoScardaci, Álvaro López García. The EGI Federated Cloud e-Infrastructure // Procedia Computer Science Volume 68, 2015, pp. 196-205. URL: https://doi.org/10.1016/j.procs.2015.09.235
[21]. Thijs Metsch, Andy Edmonds, Boris Parak. Open Cloud Computing Interface – Infrastructure // Open Grid Forum. September 19, 2016. URL: https://www.ogf.org/documents/GFD.224.pdf
[22]. Federated Cloud user support. URL: https://wiki.egi.eu/wiki/Federated_Cloud_user_support
[23]. Federated Cloud AppDB VMOps Dashboard. URL: https://wiki.egi.eu/wiki/Federated_Cloud_AppDB_VMOps_Dashboard
[24]. EGI: Available science gateways. URL: http://wwwtest.egi.eu/services/support/science-gateways/science_gateways_for_users.html
[25]. The Catania Science Gateway Framework in the ReCaS Environment. Barbera R., Bruno R., Fargetta M., La Rocca G. // High Performance Scientific Computing Using Distributed Infrastructures, 2017, pp. 473-483. URL: https://doi.org/10.1142/9789814759717_0039
[26]. WS-PGRADE/gUSE. URL: https://sourceforge.net/projects/guse/
[27]. Kacsuk P. Science gateways for distributed computing infrastructures: Development framework and exploitation by scientific user communities // Springer International Publishing, 2014. URL: https://doi.org/10.1007/978-3-319-11268-8
[28]. Job Submission Description Language (JSDL) specification, version 1.0. Anjomshoaa A. et al. // Open Grid Forum, 2006. URL: http://www.ogf.org/documents/GFD.56.pdf
[29]. Dennis Gannon. The State of the Cloud for Science-2018 // Technical Report, June 2018. URL: https://esciencegroup.com/2018/06/11/the-state-of-the-cloud-for-science-2018/

Оглавление