УДК 004.272.2:004.75

В.В. КАЗИМИР[*], О.А. ПРІЛА[*], М.С. КРИЩЕНКО[*]

АРХІТЕКТУРА ПОБУДОВИ ВІРТУАЛЬНОГО ОТОЧЕННЯ ДЛЯ ГРІД-ЗАСТОСУВАНЬ

[*] Чернігівський державний технологічний університет, м. Чернігів, Україна

Анотація. *У статті розглянуто проблематику конфігурації грід-ресурсів для виконання прикладних обчислювальних задач у грід-середовищі. Запропоновано технологію автоматизованого налаштування оточення для виконання завдань у грід-середовищі з використанням технології віртуалізації. Представлена архітектура побудови віртуального оточення для грід-застосувань.*
Ключові слова: *грід-обчислення, віртуалізація, архітектура.*

Аннотация. *В статье рассмотрена проблематика конфигурирования грид-ресурсов для выполнения прикладных вычислительных задач в грид-среде. Предложена технология автоматизированной настройки окружения для выполнения задач в грид-среде с использованием технологии виртуализации. Представлена архитектура построения виртуального окружения для грид-применений.*
Ключевые слова: *грид-вычисления, виртуализация, архитектура.*

Abstract. *The article considers the problems of configuring of grid resources to perform computational tasks in grid environment. Moreover it is suggested the technology of automated configuration environment to perform tasks in grid environment using virtualization technology. The architecture of a virtual environment for grid applications is introduced in the article.*
Keywords: *grid computations, virtualization, architecture.*

1. Вступ

Грід-обчислення представляють собою форму розподілених обчислень, в якій віртуальний суперкомп'ютер представлений у вигляді слабкозв’язаних гетерогенних обчислювальних ресурсів, з'єднаних за допомогою глобальної мережі і використовуваних для вирішення обчислювальних задач великої розмірності. Використання потужностей розподілених ресурсів замість нарощування потужності локального ресурсу є економічно вигідним рішенням.

Грід-технології швидко розвиваються і широко використовуються для досліджень у різних галузях науки [1, 2]. Однак складність підготовки обчислювальних задач і конфігурації середовища для виконання задачі на віддаленому комп'ютері залишається актуальною проблемою. Під середовищем розуміємо вимоги до програмного забезпечення (ПЗ), яке буде встановлено на обчислювальний ресурс для виконання задач.

Поточні проблеми використання розподіленого грід-середовища полягають у:

– складності адміністрування обчислювальних ресурсів грід-середовища;

– проблемі розгортання ліцензійного програмного забезпечення на віддалених грід-ресурсах та використанні відповідного ПЗ для виконання задач;

– несумісності версій операційної системи (ОС), під управлінням якої працює обчислювальний грід-ресурс, і операційної системи, необхідної для виконання прикладної обчислювальної задачі користувачів грід-середовища.

Платформи проміжного ПЗ EMI [3] не підтримують автоматизованої конфігурації середовища для виконання задачі. Існує проблема несумісності версії ОС з апаратними характеристиками грід-ресурсів. Необхідність виконувати обчислювальні задачі, використовуючи програмне забезпечення під ОС Microsoft Windows, у той час як проміжне ПЗ грід використовує ОС Linux. Виконання обчислювальних задач у грід-середовищі з використанням ліцензійного ПЗ та виконання задач з правами адміністратора сьогодні не є можли-

ISSN 1028-9763. Математичні машини і системи, 2017, № 3

вим у грід-середовищі. Це накладає суттєві обмеження на використання ресурсів грід-середовища та зменшує їх доступність.

Використання технології віртуальної машини дозволить сформувати віртуальний образ (VI) з налаштуваннями і використовувати його при запуску обчислювальних задач на грід-ресурсах, що також є вирішенням проблеми використання ліцензованого ПЗ при вирішенні задач у грід-середовищі. Важливим аспектом широкого впровадження грід-технологій є забезпечення необхідного рівня якості обслуговування користувача, що для некомерційного середовища найчастіше визначається гарантованим часом успішного завершення обчислень. Також важливим аспектом є забезпечення ефективного розміщення сховищ образів віртуальних машин та алгоритмів планування, спрямованих на мінімізацію комунікаційних витрат.

Метою статті є розробка технології розгортання віртуальних середовищ у грід інфраструктури та архітектури побудови середовища автоматизованого обчислювального вузла, що зробить процес підготовки задачі більш гнучким і збільшить доступність ресурсів та послуг, які використовуються у грід-середовищі.

2. Аналіз технологій використання віртуальних середовищ у грід-інфраструктурі

Авторами було проведено дослідження існуючих технологій для використання віртуальних середовищ у грід-інфраструктурі, а саме: Nimbus [5], skifGrid [6], Rainbow [7] та пакет програмного забезпечення "Віртуальний контейнер" [8]. Існуючі технології не дають повного вирішення проблеми, оскільки не забезпечують такі можливості:

– завантаження віртуального образу на віддалений ресурс;

– збереження VI в розподіленому репозиторії, включаючи засоби реплікації;

– вибрати віртуальне середовище для виконання задачі;

– виконання обчислювальних задач різних типів [9] та різних прикладних галузей;

– оптимізація планування з урахуванням особливостей використання віртуалізації у розподіленому середовищі;

– високорівневий інтерфейс користувача;

– розширюваність.

Нами було розглянуто проблему налаштування середовища для виконання обчислювальних задач у грід-середовищі [4]. Запропоновано технологію автоматизованої конфігурації грід-середовища на основі віртуалізації. Реалізація розробляється на основі фреймворку для розробки високорівневих грід-застосувань, що забезпечує API для виконання базових грід-операцій [4].

3. Сервіси фреймворку для розробки грід-застосувань

Використання технології віртуальної машини дозволить сформувати віртуальний образ з налаштуваннями і використовувати його для запуску обчислювальних задач на грід-ресурсах, що також є рішенням проблеми використання ліцензованого ПЗ при вирішенні завдань у грід-середовищі.

Практична реалізація виконується на базі існуючого фреймворку для розробки грід-додатків [10] з використанням технології, що розробляється. Користувачам грід-середовища мають надаватися такі сервіси (рис. 1).

Перегляд доступних ресурсів означає можливість перегляду та фільтрування списків елементів зберігання (SE), обчислювальних елементів (CE) та послуг віртуальної організації (VO).

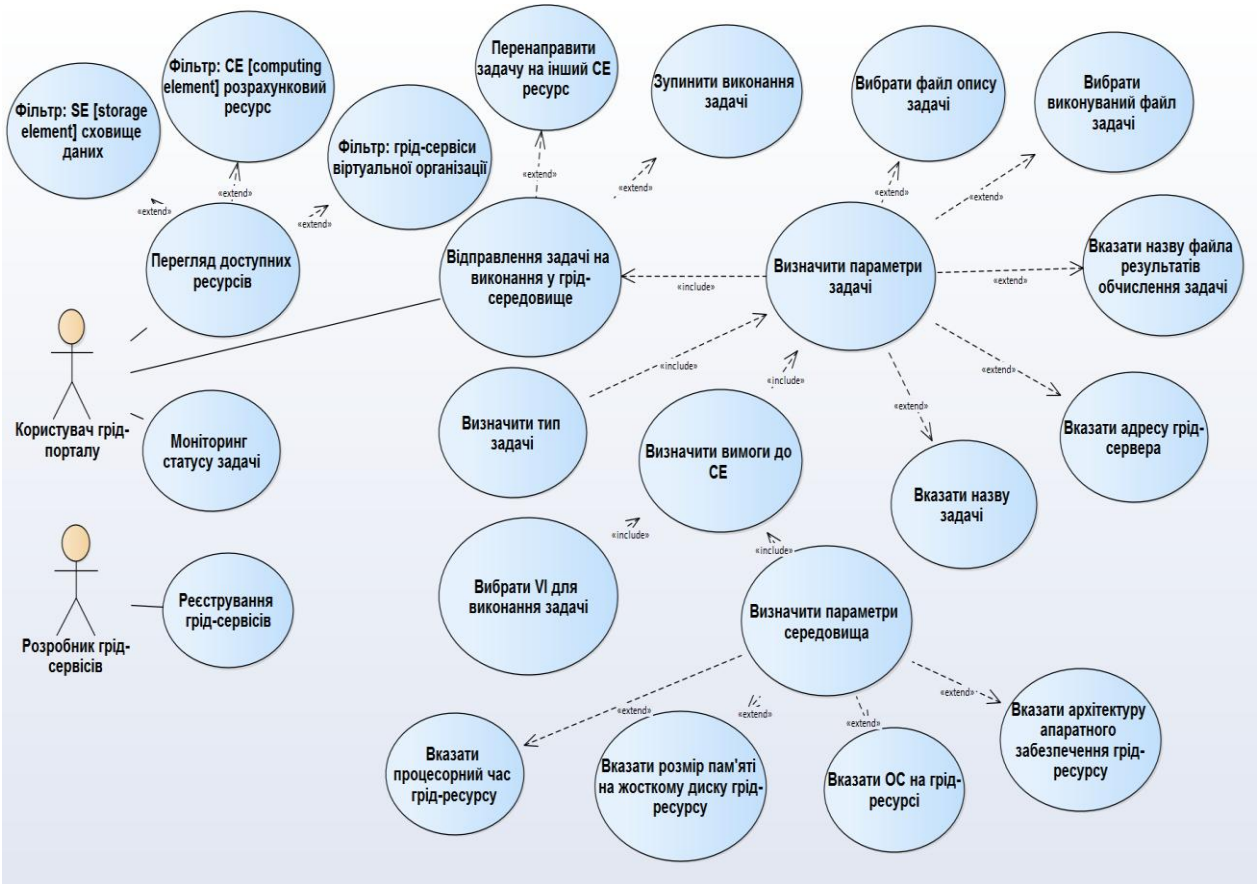

Рис. 1. Сервіси фреймворку для розробки грід-застосувань

Фреймворк забезпечує можливість відправлення задачі для виконання у грід-середовище. Сервіс для відправлення задачі включає в себе можливості повторного запуску задачі на інший CE та зупинки виконання задачі. Перед відправленням користувачеві необхідно визначити набір параметрів задачі, а саме:

– вибрати виконуваний файл задачі. Виконуваний файл задачі – виконуваний файл, який буде проводити розрахункову роботу на грід-ресурсі;

– вказати назву файла результатів обчислення задачі. Результати виконання грід-задачі автоматизовано завантажуються у вигляді файла з назвою, яку вказав користувач;

– вказати адресу грід-ресурсу – адрес грід-ресурсу, на якому буде виконуватись задача;

– вказати назву задачі;

– визначити тип задачі. Користувачу надається вибір програмних додатків, які можна буде використати при виконанні задачі на відповідних грід-ресурсах;

– визначити вимоги до CE.

При необхідності користувач може детально вказати параметри середовища (архітектуру середовища, ОС, RAM, HDD, CPU time). У випадку виконання грід-задачі з використанням віртуального середовища на CE ресурсах повинен бути завантажений VI, якщо до цього він не був завантажений при виконанні попередніх задач. Параметри задачі у внутрішньому форматі специфікації відправляються до метапланувальника грід.

Грід-фреймворком надається можливість відслідковувати статус задачі на всіх етапах її виконання.

Автоматизоване виконання грід-задач з використанням віртуального середовища на CE ресурсі забезпечується грід-сервісами. Реєстрацією грід-сервісів на CE ресурсах займається розробник грід-сервісів.

4. Технологія використання віртуалізації у грід-середовищі

Нижче представлена технологія використання віртуалізації у грід-середовищі (рис. 2).

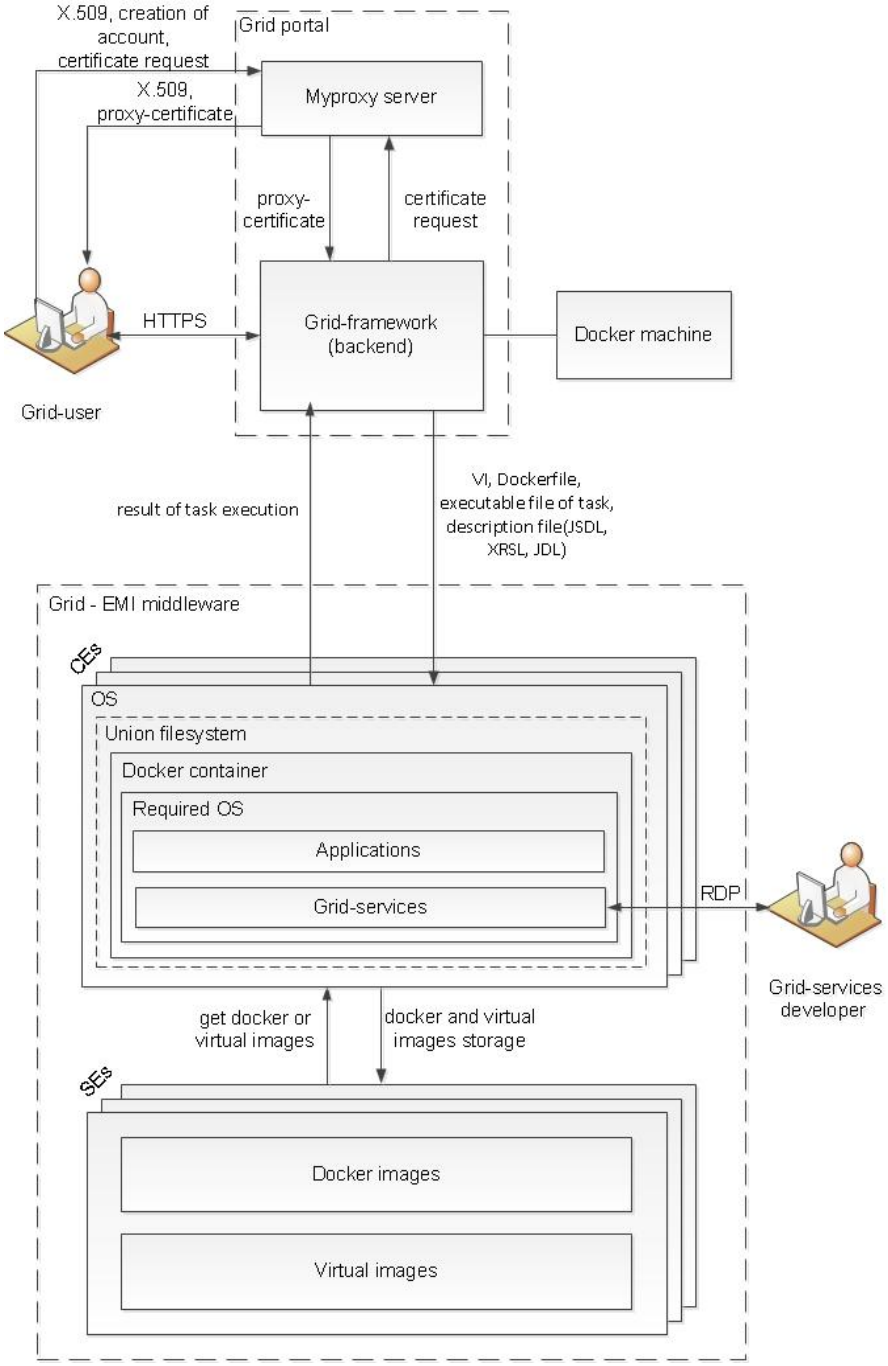

Рис. 2. Технологія використання віртуалізації у грід-середовищі

Під Grid-user мається на увазі користувач грід-порталу, який може використовувати різні операційні системи та браузери для доступу. Grid-user можуть бути користувачами, які мають права на використання грід-ресурсів організації. MyProxy server – онлайн репозиторій для зберігання сертифікатів та видачі проксі-сертифікатів. Сервер Grid-framework – web-сервер, на якому розміщений портал.

Проксі-сертифікат для запуску завдання створюється на стороні сервера засобами сервісів MyProxy [11]. Для використання довгострокових проксі-сертифікатів, що необхідно при запуску завдань зі значним терміном виконання, використовуються також сервіси MyProxy.

Грід-фреймворком надається повноцінна підтримка грід-задач різного типу. Наприклад, ARC Nordugrid [12] і gLite [13], які включені як одні з основних постачальників проміжного ПЗ грід в EMI. В українській національній грід-інфраструктурі широко використовуються такі формати специфікації грід-задач: JSDL [14], XRSL [15] та JDL [16].

Запити HTTPS від Grid-user включають ряд параметрів у форматі JSON [17]. Основні з них такі:

– Name – назва грід-задачі;

– Description – короткий опис грід-задачі;

– Executable – параметр, який передає виконуваний файл задачі;

– Server – параметр, який містить посилання на CE;

– VirtualImage – параметр, потрібний для вибору та передачі віртуального образу до CE та SE ресурсів.

Клієнтська сторона системи дозволяє користувачам грід-порталу використовувати різні платформи, включаючи мобільні пристрої. При використанні параметра VirtualImage до грід-задачі прикріплюється віртуальний образ, якщо вибраний VI завантажується вперше, а також Dockerfile [18].

Подібно віртуальній машині докер запускає свої процеси у власній, заздалегідь налаштованій операційній системі. Але при цьому всі процеси Docker працюють на фізичному грід-сервері, ділячи всі процесори і всю доступну пам'ять з усіма іншими процесами, запущеними фреймворком. Підхід, який використовується Docker, називається контейнеризацією.

Docker machine – інструмент, що дозволяє встановлювати та управляти docker container на грід-ресурсах.

Dockerfile містить набір інструкцій з аргументами. Dockerfile використовує звичайний DSL [19] з інструкціями для побудови образів docker. Після цього виконується команда docker build для побудови нового docker image з інструкціями dockerfile. Інструкції обробляються зверху вниз. Кожна інструкція додає новий шар в образ і фіксує зміни.

Docker image – образ, збудований на основі ОС, з якого створюється контейнер. Кожен docker image складається з набору рівнів. Docker використовує Union File System для поєднання цих рівнів в один образ.

Union File System або UnionFS – це файлова система, яка працює, створюючи рівні, що робить її дуже легкою і швидкою [20]. Docker використовує UnionFS для створення блоків, з яких будується контейнер. Docker може використовувати декілька варіантів UnionFS, включаючи AUFS [21], btrfs [22], vfs [23] і DeviceMapper [24].

Union file system складається з шарів (layers), накладених один на одного. Деякі шари захищені від запису. Наприклад, усі наші контейнери використовують загальні, захищені від запису, шари, в яких знаходяться незмінні файли операційної системи.

Для змінюваних файлів кожен із контейнерів буде мати власний шар. Docker використовує такий підхід не тільки для ОС, але і для будь-яких загальних частин контейнерів, створених на основі загальних «предків» їх образів.

Під час виконання грід-задачі, на основі вибраного VI, у docker контейнері створюється віддалена ОС з потрібними для користувача налаштуваннями та додатками. Результати виконання задачі будуть завантажені автоматично на клієнтську частину користувача. Grid-service developer має RDP [25] доступ до віддаленої ОС для розробки грід-сервісів. Під грід-сервісами розуміється сценарій автоматизованого виконання задачі в середині ОС docker контейнера.

5. Розширена архітектура фреймворку для створення грід-додатків

На рис. 3 представлена архітектура фреймворку для створення грід-додатків із підтримкою сучасного проміжного ПЗ грід [3], яка була розширена такими модулями: модуль розміщення образу та модуль планування з урахуванням використання VI. А також був розширений модуль формування і зчитування формату специфікації.

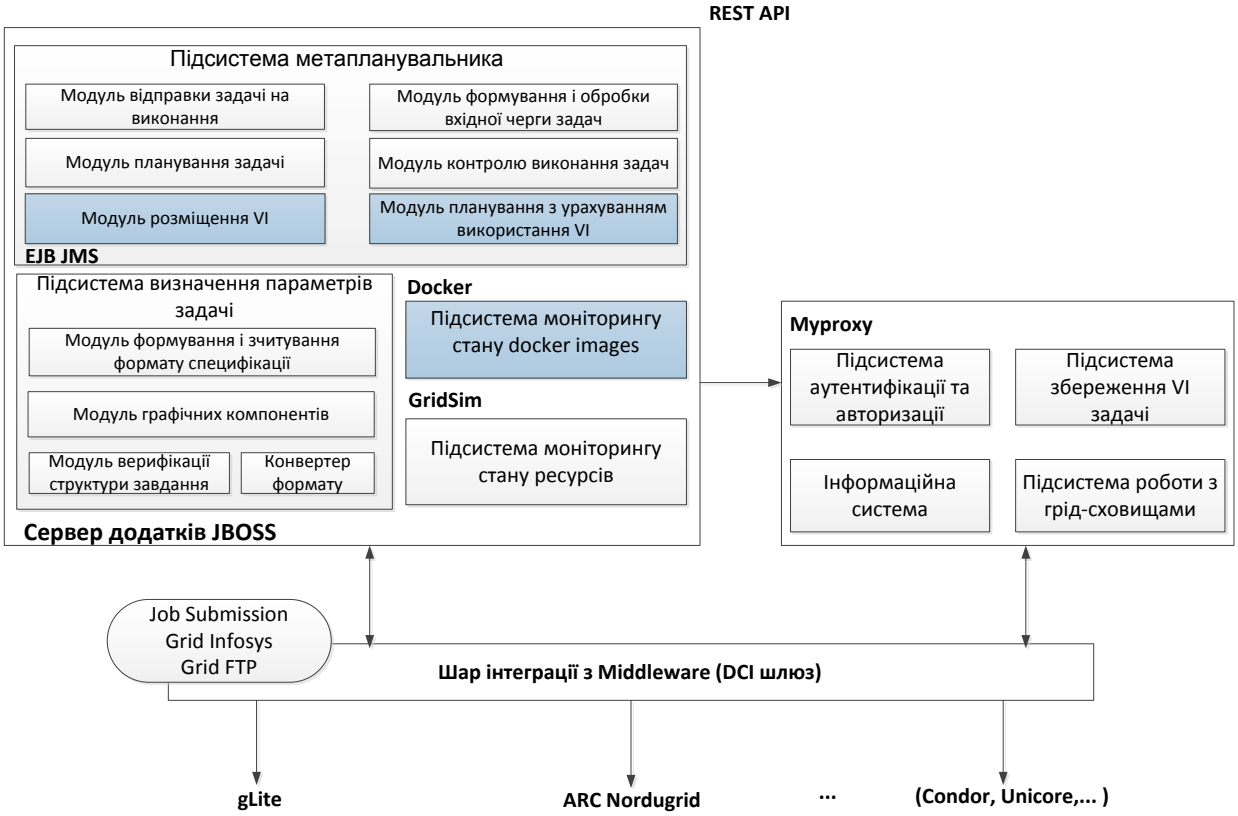

Рис. 3. Архітектура побудови віртуального оточення для грід-застосувань

Модуль розміщення образу визначає обчислювальний ресурс SE [4] для зберігання віртуального образу. Використовується при проектуванні схеми реплікації VI.

Сервіси модуля планування з урахуванням використання VI реалізують автоматичний пошук потрібного VI і вибір обчислювального ресурсу CE [4] для розгортання образу і виконання задачі.

Підсистема моніторингу стану docker images відслідковує стан грід-задач, які були виконані на вибраному CE ресурсі. Механізм реалізації процесу розгортання на віддалених обчислювальних ресурсах реалізується за допомогою програмної платформи Docker.

На рис. 4 представлено інтерфейс користувача фреймворку для розробки високорівневих грід-застосувань.

Реалізація серверної частини фреймворку виконувалась мовою програмування Java. Клієнтська частина реалізована з використанням JavaScript. Як веб-сервер використовується nginx [26].

Для оцінки ефективності підходу була розроблена модель процесу виконання завдань у грід-середовищі з використанням технології віртуалізації на базі симулятора GridSim [27].

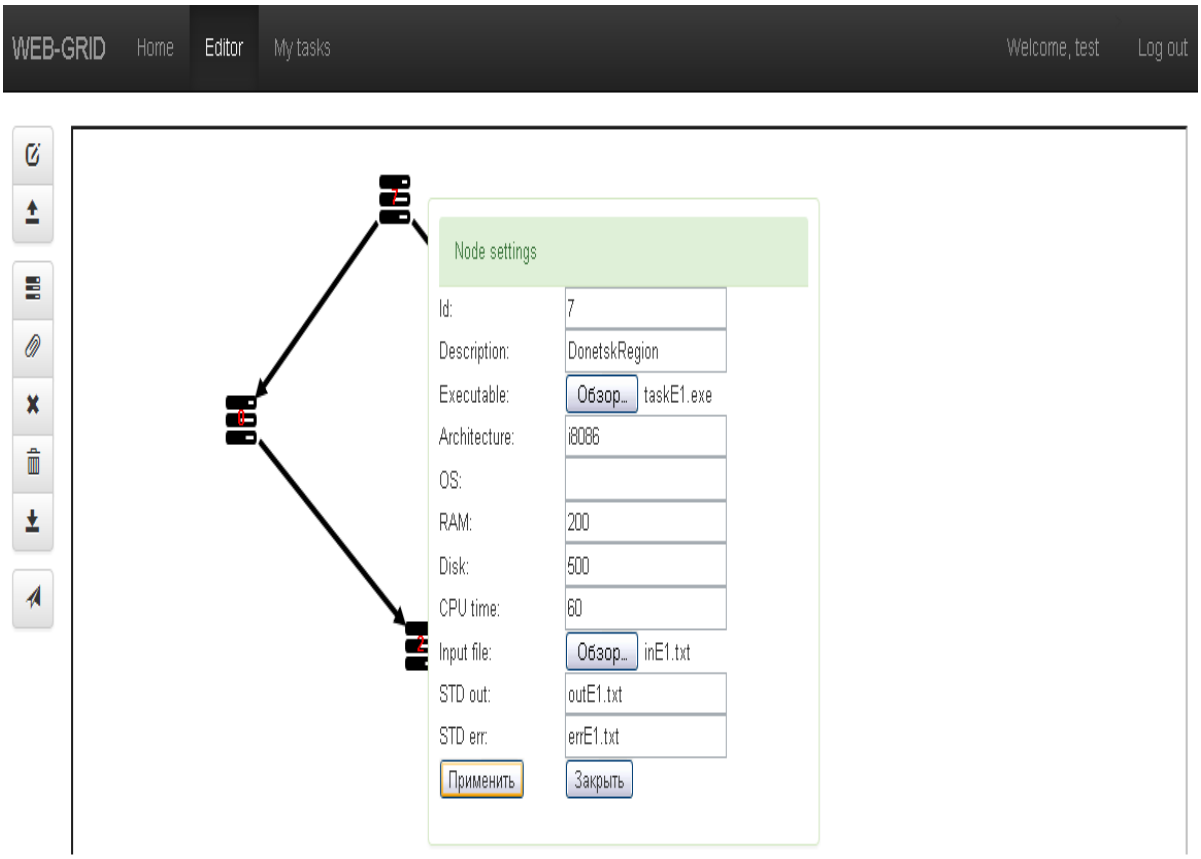

Рис. 4. Інтерфейс користувача фреймворку для розробки високорівневих грід-застосувань

Експерименти по оцінці часу на пересилку VI і даних про задачу з віртуальної машини на грід-ресурс проводилися в реальному середовищі за допомогою кросплатформної консольної клієнт-серверної утиліти iperf [28].

За отриманими результатами можна сказати, що часові витрати на виконання задачі у грід-середовищі з використанням віртуалізації (з встановленим VI на віддаленому ресурсі) не перевищують 5% у порівнянні з витратами на виконання задачі без використання віртуалізації у грід-середовищі. Однак відношення між часом виконання задачі у грід-середовищі без використання технології віртуалізації і з використанням, але без встановленого VI на віддалений обчислювальний ресурс, значно більше [4]. Витрати на пересилку VI можуть бути компенсовані за допомогою розробки схеми реплікації сховищ віртуальних образів, а також алгоритму планування у грід-середовищі з урахуванням використання засобів віртуалізації.

6. Висновок

У статті запропоновано підхід для забезпечення доступності грід-ресурсів та сервісів через використання технології віртуалізації. Запропоноване рішення дозволяє грід-користувачам налаштувати необхідне середовище для виконання задач. Такий підхід підвищує рівень доступності грід-середовища.

Технологія автоматизованої побудови віртуального оточення на розподілених віддалених обчислювальних вузлах реалізується засобами інструментарію Docker [29]. Високорівневий інтерфейс користувача для підготовки, відправлення та формування необхідного середовища для виконання грід-задачі реалізується як розширення попередньо розробленого фреймворку для виконання базових грід-операцій [10].

Ефективна архітектура для розподільного зберігання віртуальних образів та алгоритми планування виконання грід-задачі з використанням віртуального оточення потребує подальшого дослідження. Ефективність запропонованої технології була оцінена шляхом проведення експериментів на основі імітаційної моделі, а також у реальному грід-середовищі [4].

СПИСОК ЛІТЕРАТУРИ

1. Vishnevsky V. Web-services of Medgrid project / V. Vishnevsky, M. Volzheva, O. Prila // Ukrainian Journal of Telemedicine and medical telematics. – 2012. – Vol. 10, N 2. – P. 4 – 5.
2. Kazymyr V. The technology of using the grid environment for ECG signals distributed storage, visualization and processing / V. Kazymyr, O. Prila., M. Kryshchenko // Proc. «Information Technologies in Innovation Business Conference, ITIB 2015». – Kharkiv, 2015 – P. 19 – 23.
3. European Middleware Initiative [Електронний ресурс]. – Режим доступу: https://www.dcache.org/manuals/EMI_FACT-SHEET-1_4.pdf.
4. Building of the virtual environment for grid applications / V. Kazymyr, D. Melnychenko, O. Prila [et al.] // Information Models and Analyses". – 2016. – Vol. 5, N 1. – P. 37 – 48.
5. Nimbus is cloud computing for science [Електронний ресурс]. – Режим доступу: http://www.nimbusproject.org/.
6. SkifGrid wiki [Електронний ресурс]. – Режим доступу: http://grid.basnet.by/projects/skifgrid/wiki.
7. Medical Grid-system for population research in the field of cardiology with electrocardiogram database [Електронний ресурс]. – Режим доступу: http://medgrid.immsp.kiev.ua/.
8. Virtual computing environments: the use of polygons on the grid / V. Volohov, D. Varlamov, N. Surkov [et al.] // Herald of South Ural State University. Series: Mathematical modeling and programming. – 2009. – N 17. – P. 24 – 35.
9. Prila O. The framework for high level grid applications development / O. Prila // Problems of programming. – 2014. – N 1. – P. 31 – 39.
10. Kazymyr V. Grid workflow design and management system / V. Kazymyr, O. Prila, V. Rudyi // International Journal "Information Technologies & Knowledge". – 2013. – Vol. 7, N 3. – P. 241 – 255.
11. MyProxy, Credential Management Service [Електронний ресурс]. – Режим доступу: http://grid.ncsa.illinois.edu/myproxy/protocol/.
12. Advanced Resource Connector middleware for lightweight computational Grids / M. Ellert, M. Grønager, A. Konstantinov (eds.) [et al.] // Future Generation Computer Systems. – 2007. – Vol. 23. – P. 219 – 240.
13. gLite – Lightweight Middleware for Grid Computing [Електронний ресурс]. – Режим доступу: http://grid-deployment.web.cern.ch/grid-deployment/glite-web/.
14. Job Submission Description Language (JSDL) Specification / A. Anjomshoaa, M. Drescher, D. Fellows [et al.] // Global Grid Forum, 2005. – Version 1.0, GFD-R.136. – 71 p.
15. Extended Resource Specification Language, Reference Manual for ARC versions 0.8 and above, Nordugrid-Manual-4. – 2013. – P. 13 – 28.
16. Job description language attributes specification for the gLite Workload Management System, WMS-JDL.doc. – 2011. – P. 7 – 10, 38 – 40.
17. Introducing JSON [Електронний ресурс]. – Режим доступу: http://www.json.org/.
18. Docker Overview [Електронний ресурс]. – Режим доступу: https://docs.docker.com/engine/understanding-docker/.
19. Using Jenkins Job DSL for Job Lifecycle Management [Електронний ресурс]. – Режим доступу: https://blog.codecentric.de/en/2015/10/using-jenkins-job-dsl-for-job-lifecycle-management/.
20. Unionfs: A Stackable Unification File System [Електронний ресурс]. – Режим доступу: http://unionfs.filesystems.org/.
21. Aufs [Електронний ресурс]. – Режим доступу: http://aufs.sourceforge.net/aufs.html.
22. Btrfs [Електронний ресурс]. – Режим доступу: https://wiki.archlinux.org/index.php/Btrfs.
23. The Virtual File System [Електронний ресурс]. – Режим доступу: http://www.science.unitn.it/~fiorella/guidelinux/tlk/node102.html.
24. Device-mapper Resource Page [Електронний ресурс]. – Режим доступу: http://www.sourceware.org/dm/.

25. Remote Desktop Protocol [Електронний ресурс]. – Режим доступу: https://msdn.microsoft.com/en-us/library/aa383015.aspx.
26. Nginx [Електронний ресурс]. – Режим доступу: https://nginx.org/ru/.
27. Buyya R. GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing / R. Buyya, M. Manzur // The Journal of Concurrency and Computation: Practice and Experience (CCPE). – 2002. – Vol. 14, Is. 13 – 15. – P. 1179 – 1219.
28. iPerf – The network bandwidth measurement tool [Електронний ресурс]. – Режим доступу: https://iperf.fr/iperf-doc.php.
29. Docker Toolbox [Електронний ресурс]. – Режим доступу: https://www.docker.com/products/docker-toolbox.

Стаття надійшла до редакції 13.07.2017