O ANALIZĂ A ECOSISTEMULUI OPENSHIFT

Mădălina Zamfir **Ştefan Preda**

madalina@ici.ro stefanalex@ici.ro

Institutul Național de Cercetare-Dezvoltare în Informatică, ICI Bucureşti

Rezumat: Articolul prezină o analiză a ecosistemului OpenShift, considerat o abordare reprezentativă a soluțiilor Platform ca Service (PaaS) şi un sistem open source extrem de utilizat în prezent. Au fost descrise avantajele arhitecturii şi componentele principale ale noii versiuni a sistemului OpenShift v3 şi paşii de instalare şi administrare a unei aplicații pe platforma OpenShift în mod local, pentru a arăta modul de lucru cu funcționalitățile acesteia de pe un PC sau un notebook.

Cuvinte cheie: OpenShift, Kubernetes, Docker, Vagrant.

Abstract: In this article we performed an analysis on OpenShift ecosystem, considered a representative approach to Platform as a Service (PaaS) solutions and an extremely used open source system in present. The article describes the advantages of the architecture and the main components of the new version of OpenShift v3 system, the installation steps of the system and the management steps of an application on a local OpenShift platform, in order to present its functionality on a PC or notebook.

Keywords: OpenShift, Kubernetes, Docker, Vagrant.

1. Introducere

Proiectul [1] realizat în cadrul Programului Nucleu Tehnologii avansate şi servicii pentru dezvoltarea societății informaționale - TEHSIN a prezentat interesul organizațiilor în crearea unor infrastructuri de tip Cloud privat, pentru a beneficia de avantajele tehnologiilor de Cloud computing, cum ar fi: reducerea costurilor, utilizarea eficientă a resurselor (puterea de procesare, capacitatea de stocare), minimizarea riscurilor de securitate, în locul achiziției de servicii de la un Cloud public.

Cu ajutorul platformei OpenShift de la Red Hat [2], aplicațiile într-un mediu cloud sunt create, găzduite şi scalate rapid, punându-se accentul pe codul acestora. Platforma a fost considerată în 2015 unul dintre cele mai bune instrumente pentru dezvoltatori, profesionişti IT şi pentru domeniul afacerilor.

Pentru diferite instituții, cercetători şi studenți, Cloud-ul privat de tip open source aduce multe avantaje, deoarece aceştia îl pot implementa cu resurse restrânse (şi pe un laptop) pentru realizarea experimentelor lor.

2. Arhitectura şi componentele principale ale sistemului OpenShift v3

Pe platforma OpenShift Online pot fi găzduite şi dezvoltate aplicațiile într-un Cloud public, care automatizează procesul de furnizare şi de administrare a aplicațiilor la cerere. Limbajele de programare utilizate sunt Java, Ruby, Node.js, Python, PHP, Scala, Perl. Pot fi adăugate alte limbaje şi baze de date sau componente de tip middleware, necesare dezvoltatorilor.

Pe platforma OpenShift Enterprise, dezvoltatorii pot crea rapid aplicații la cerere folosind instrumentele preferate [3]. Executarea aplicațiilor de către clienți este realizată pe un Cloud privat sau public sau pe o infrastructură de Cloud hibridă.

Pe platforma de aplicații OpenShift Origin [1] dezvoltatorii construiesc, testează, lansează pe server şi execută aplicațiile care sunt executate în containere de tip Docker, în timp ce proiectul Kubernetes [4] asigură partea de programare şi administrare, metodele de implementare, orchestrare şi de rețea. Datorită containerelor grupate numite "pod"-uri de tip Kubernetes şi care se comportă ca o singură maşină virtuală, creşte numărul aplicațiilor care pot fi încărcate în OpenShift, construite ca micro-servicii care comunică împreună cu ajutorul interfețelor. Accesul la resurse este administrat prin intermediul proiectului.

Arhitectura OpenShift v3 conține componentele infrastructurii, concepte nucleu şi concepte

suplimentare. Datorită arhitecturii bazate pe straturi, aplicațiile au o configurație flexibilă, astfel că o bază de date poate fi reutilizată de două containere de tip web sau poate fi afișată direct din rețea. Componentele infrastructurii cuprind infrastructura Kubernetes, Image Registry şi Web Console.

Infrastructura Kubernetes este o platformă open-source pentru administrarea aplicațiilor depozitate în containere şi furnizează mecanismele pentru lansarea acestora în execuție, întreținerea şi scalarea aplicațiilor [4]. Sunt autentificați utilizatorii care au credențiale [1]. Dezvoltatorii (clienții sistemului) şi administratorii pot fi autentificați prin certificate SSL. Interfața poate fi apelată printr-un program client, cu ajutorul comenzii `oc` sau de la consola web printr-un browser. Componentele infrastructurii (de exemplu, nodurile) folosesc certificatele client generate de sistem, care conțin identitățile acestora. Autorizarea este tratată în politicile OpenShift, care definesc acțiuni precum "creează pod" sau "lista de servicii", pe care le grupează în roluri într-un document de politici. Rolurile sunt adresate utilizatorilor sau grupurilor printr-un identificator de utilizator sau de grup. Când un utilizator sau un serviciu aşteaptă o anumită acțiune, mecanismul de politici verifică unul sau mai multe roluri atribuite utilizatorului (de exemplu administrator de cluster sau administrator de proiect).

OpenShift poate utiliza orice server care să implementeze interfața fişierelor de tip registru Docker ca sursă de imagini (Image Registry), incluzând Hub-ul Docker, fişierele de tip registru privați executați de părți terțe şi fişierele de tip registru OpenShift integrate.

Web Console este o interfață utilizator accesibilă dintr-un browser web. Dezvoltatorii pot folosi consola pentru vizualizarea, căutarea şi administrarea conținuturilor proiectelor. Un proiect permite comunității de utilizatori să-şi organizeze şi administreze conținutul, separat față de alte comunități. Utilizatorii primesc acces la proiecte din partea administratorilor sau au dreptul de a crea proiecte.

Conceptele nucleu [1] fac referire la containerele şi imaginile care reprezintă blocuri pentru dezvoltarea aplicațiilor, la "pod"-urile şi serviciile care permit containerelor să comunice unele cu altele şi cu conexiunile de tip proxy, la proiectele şi utilizatorii care furnizează spațiul pentru comunitățile de utilizatori, pentru organizarea şi administrarea conținutului, la implementările care oferă suport pentru dezvoltarea de software.

Containerele reprezintă unitățile de bază pentru aplicațiile din OpenShift. Fiecare container furnizează un singur serviciu (numit "micro-serviciu"), cum ar fi un server web sau o bază de dată.

Containerele de tip Docker [1] sunt bazate pe imagini Docker. O imagine Docker este un fişier care include toate cerințele pentru executarea unui singur container Docker, precum şi a metadatelor care descriu necesitățile şi capacitățile sale. Prin implementarea aceleiaşi imagini în containere multiple, pe gazde multiple, OpenShift poate furniza scalare orizontală pentru un serviciu împachetat într-o imagine.

"Pod"-urile reprezintă unul sau mai multe containere implementate împreună pe o gazdă. Fiecare "pod" are alocată o adresă IP internă şi containerele din cadrul "pod"-urilor pot partaja capacitatea de stocare locală şi din rețea. "Pod"-urile au un ciclu de viață în care sunt definite, apoi se execută pe un nod până când containerele lor se opresc sau sunt şterse.

Serviciul Kubernetes identifică un set de replici ale "pod"-urilor pentru a modifica conexiunile pe care le primeşte de la acestea. Serviciile au atribuită o adresă IP şi o pereche de porturi care, atunci când sunt accesate, împuternicesc un anume "pod".

Concepte suplimentare fac referire la faptul că, Kubernetes [1] asigură ca "pod"-urile să se poată conecta între ele în rețea şi alocă fiecărui "pod" o adresă de IP de la o rețea internă. Astfel, toate containerele din cadrul unui "pod" se comportă ca şi când ar fi pe aceeaşi gazdă. Dând fiecărui "pod" propria adresă de IP, înseamnă că acestea pot fi tratate ca şi gazde fizice sau maşini virtuale pentru alocarea de porturi, de rețea, descoperirea de servicii, optimizarea folosirii resurselor, configurarea aplicației şi migrarea aplicațiilor.

Nu este recomandată crearea de legături între "pod"-uri şi comunicarea între ele direct, folosind adresa de IP, ci se recomandă crearea unui serviciu şi apoi interacțiunea cu serviciul.

3. Vagrant

Cu ajutorul instrumentului de virtualizare Vagrant [1], [5] sunt create şi configurate medii de dezvoltare performante, portabile şi reproductibile. Acesta oferă opţiuni multiple pentru furnizarea unei maşini, de la simple script-uri, la sisteme complexe de management al configurării. Maşinile sunt provizionate cu VirtualBox, VMware, AWS sau alţi furnizori. Vagrant este un proiect open source rezultat datorită contributorilor pe GitHub.

Vagrant aduce beneficii atât programatorilor, cât şi inginerilor de sistem şi designer-ilor: un mediu de dezvoltare pe proiect (fişiere de configurare diferite pentru fiecare proiect), acelaşi fişier de configurare pentru mediile de dezvoltare - Vagrantfile (uşor de configurat şi de modificat).

Instalarea Vagrant presupune alegerea executabilului potrivit pentru platforma pe care se doreşte instalarea şi de asemenea instalarea instrumentului VirtualBox şi se face conform figurii 1:

```
$ vagrant
Usage: vagrant [-v] [-h] command []
    -v, --version                    Print the version and exit.
    -h, --help                       Print this help.
Available subcommands:
     box           manages boxes: installation, removal, etc.
     destroy       stops and deletes all traces of the vagrant machine
     halt          stops the vagrant machine
     help          shows the help for a subcommand
     init          initializes a new Vagrant environment by creating a
     package       packages a running vagrant environment into a box
     plugin        manages plugin s: install, uninstall, update, etc.
     provision     provisions the vagrant machine
     reload        restarts vagrant machine, loads new Vagrantfile
     resume        resume a suspended vagrant machine
     ssh           connects to machine via SSH
     ssh-config    outputs OpenSSH valid configuration to connect to
           the machine
     status        outputs status of the vagrant machine
     suspend       suspends the machine
     up            starts and provisions the vagrant environment
```

Figura 1

În continuare, se iniţializează instrumentul Vagrant în noul proiect, prin rularea comenzii din figura 2:

```
# Every Vagrant virtual environment requires a box to build off of.
  config.vm.box = "precise32"
  # The url from where the 'config.vm.box' box will be fetched if it
  # doesn't already exist on the user's system.
  config.vm.box_url = "http://files.vagrantup.com/precise32.box"
```

Figura 2

Fişierul Vagrantfile generat automat în directorul proiectului conform figurii 3, are rolul de a selecta scheletul pentru maşina virtuală şi furnizorul de virtualizare, de a configura parametrii maşinii virtuale, de a configura reţeaua, de modificare a setărilor SSH, de sincronizare a directoarelor locale cu cele de pe maşina virtuală, de furnizare a maşinii virtuale:

```
$ vagrant init

A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please
Read the comments in the Vagrantfile as well as documentation
on`vagrantup.com` for more information on using Vagrant.
```

Figura 3

Directiva `config.vm.box="precise32"` descrie tipul maşinii virtuale. ”Box” reprezintă scheletul pe care maşinile virtuale Vagrant vor fi construite.

Docker este un instrument similar cu Vagrant şi se pot face câteva observaţii referitoare la cele două instrumente: Vagrant este mai bun deoarece păstrează codul sursă şi informaţiile de lansare în execuţie în acelaşi loc, este stabil şi poate fi folosit în producţie; de asemenea, Vagrant este mai bun pentru că poate fi integrat cu Linux, Windows şi Mac OS X; Docker este mai bun pe partea de furnizare; Docker poate fi integrat doar cu maşinile de tip Ubuntu; Docker nu este recomandat în producţie deoarece este încă în faza de dezvoltare [1].

VirtualBox este o aplicaţie de virtualizare între platforme [7] şi este folosit în situaţii precum:

- lansarea în execuţie a mai multor sisteme de operare simultan (VirtualBox permite utilizatorului să lanseze în execuţie în acelaşi timp mai multe sisteme de operare, astfel putând executa programe software pentru un sistem de operare, pe alt sistem de operare (de exemplu, software Windows pe Linux sau Mac));
- instalări mai uşoare ale programelor software;
- testarea şi recuperarea în caz de erori: odată instalată maşina virtuală, poate fi considerată un container care poate fi îngheţat, copiat, salvat (copie de rezervă) şi transportat între maşinile gazdă. Operaţia de virtualizare poate reduce semnificativ costurile cu electricitatea şi hardware-ul.

4. Instalarea şi administrarea unei aplicaţii în OpenShift

Primul pas în cazul instalării şi administrării unei aplicaţii cu OpenShift Origin v3 în mod local, constă în instalarea şi pornirea maşinii virtuale. Maşina Virtuală pe care este instalată platforma OpenShift Origin se poate porni din Oracle VM VirtualBox sau din linia de comandă, conform figurii 4:

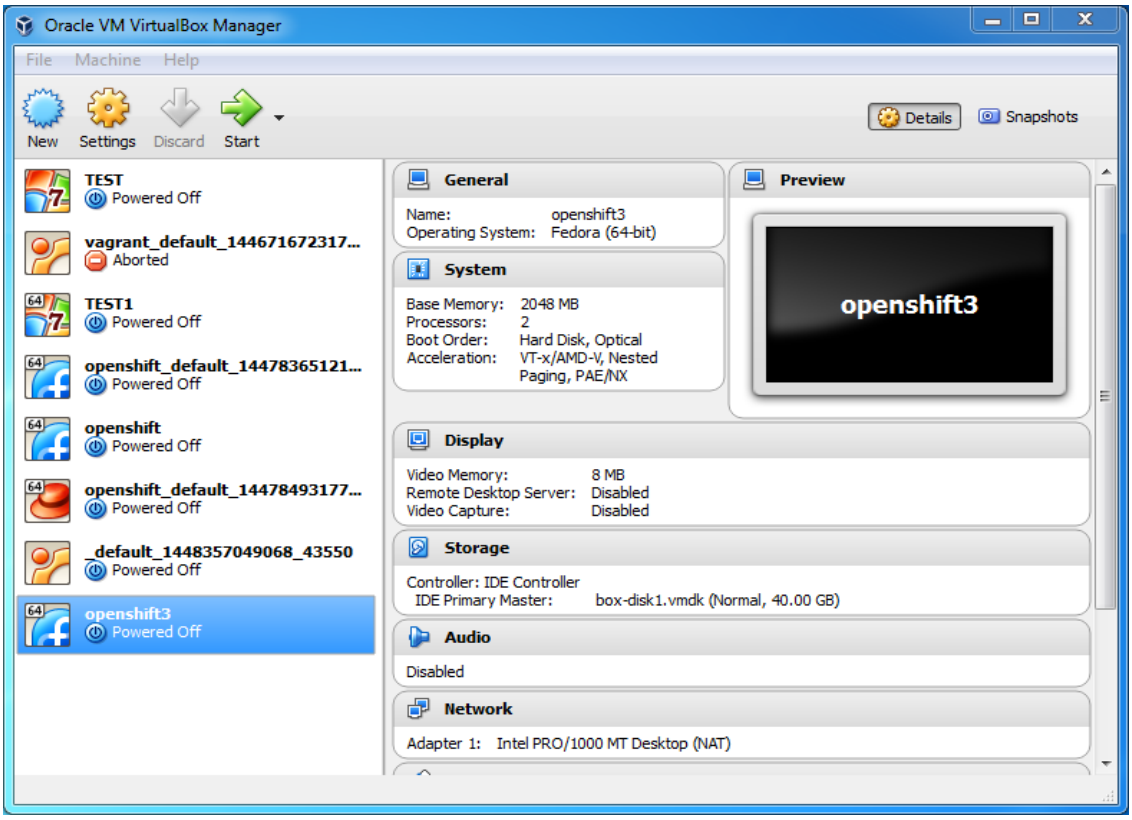

Figura 4

Din linia de comandă, cu comanda `vagrant up` (folosind furnizorul VirtualBox 5.0) executată din directorul în care a fost instalat OpenShift, obținem rezultatele din figurile 5 şi 6:

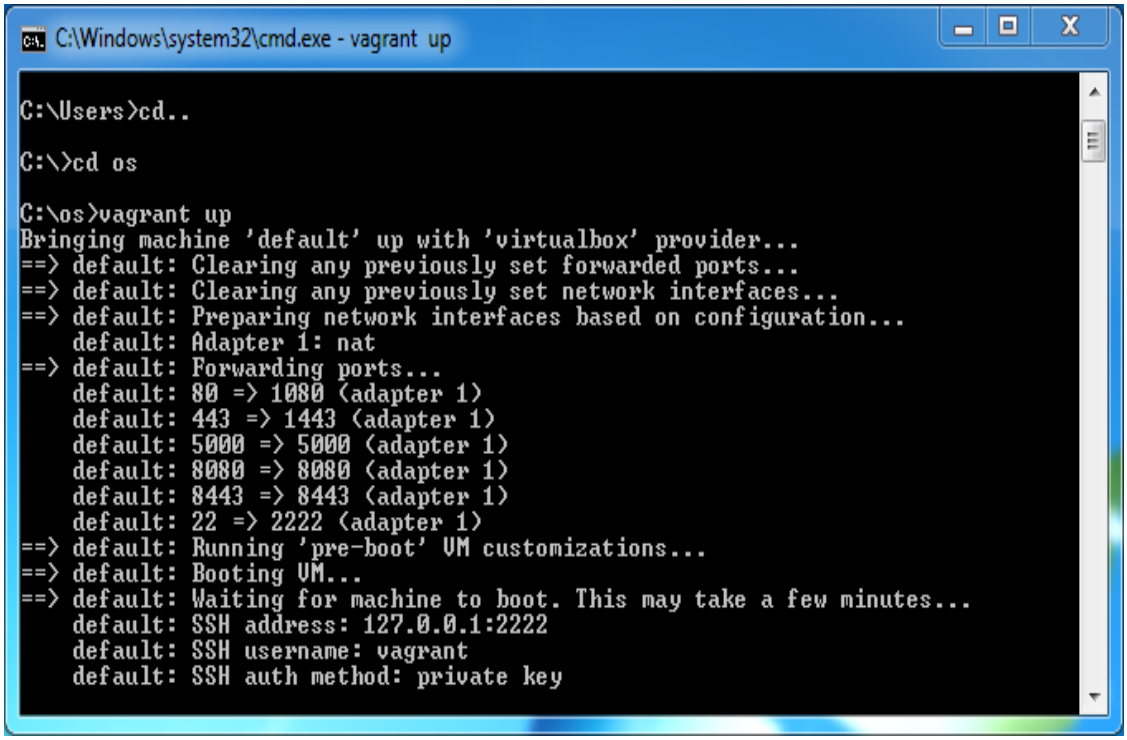

Figura 5

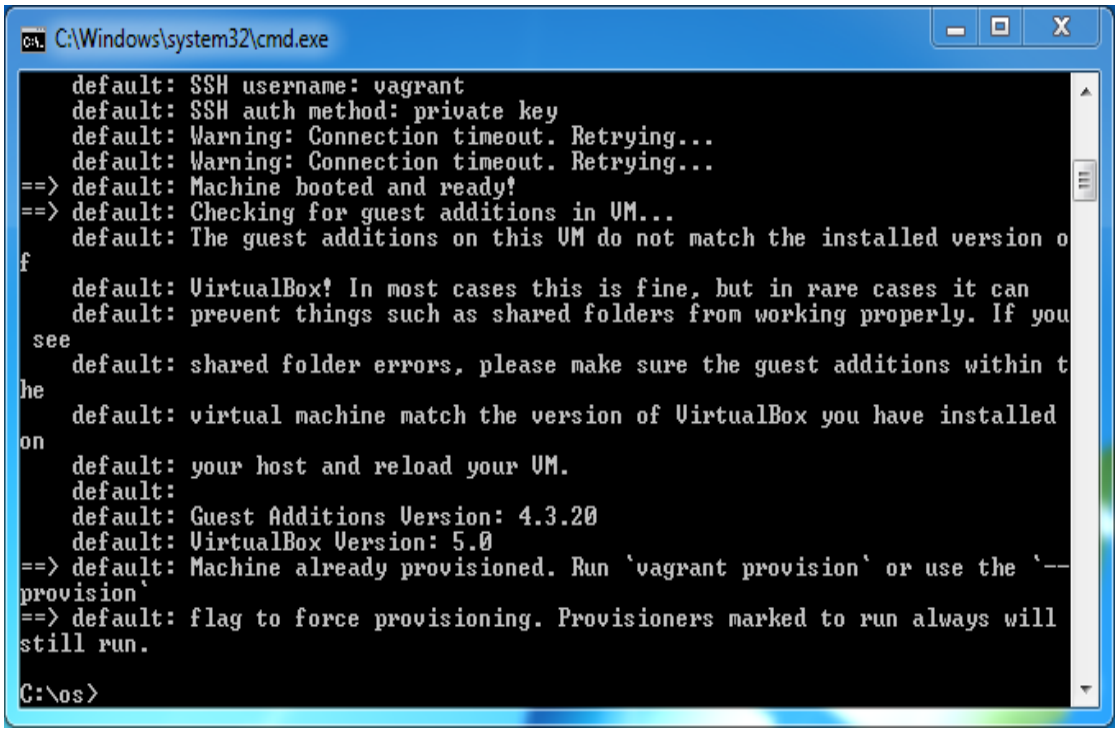

Figura 6

Se accesează din browser https://localhost:8443 şi se obține pagina de conectare a platformei, în care se introduc numele de utilizator şi parola specificate în tutorialul de instalare, aşa cum se observă în figura 7:

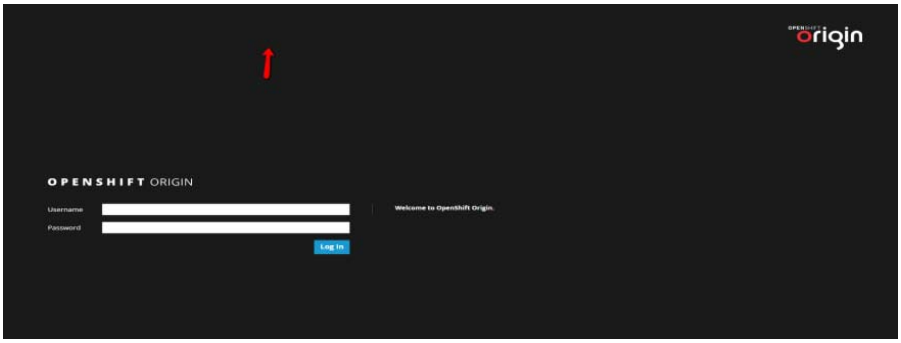

Figura 7

Din consola OpenShift Origin se poate crea un nou proiect, aşa cum se observă în figura 8:

Figura 8

Există două metode în care se creează un proiect nou: din browser şi din linie de comandă. Noua aplicație este descărcată de pe Internet, este arhivată, iar script-urile sale sunt instalate automat în proiect.

A fost creat proiectul myApache, din linie de comandă, care conține un serviciu Apache şi un serviciu de mysql. De asemenea, s-a creat o aplicație folosind un repository de tip git remote:

```
C:\os\oc project myapache
C:\os\oc new_app_https://github.com/openshift/ruby_hello_world.git#beta4
```

Cu comanda `C:\os\oc` status se află rapid programele instalate în proiect, IP-urile lor, locația de unde au fost descărcate, dacă rulează sau nu, ca în figurile 9 şi 10:

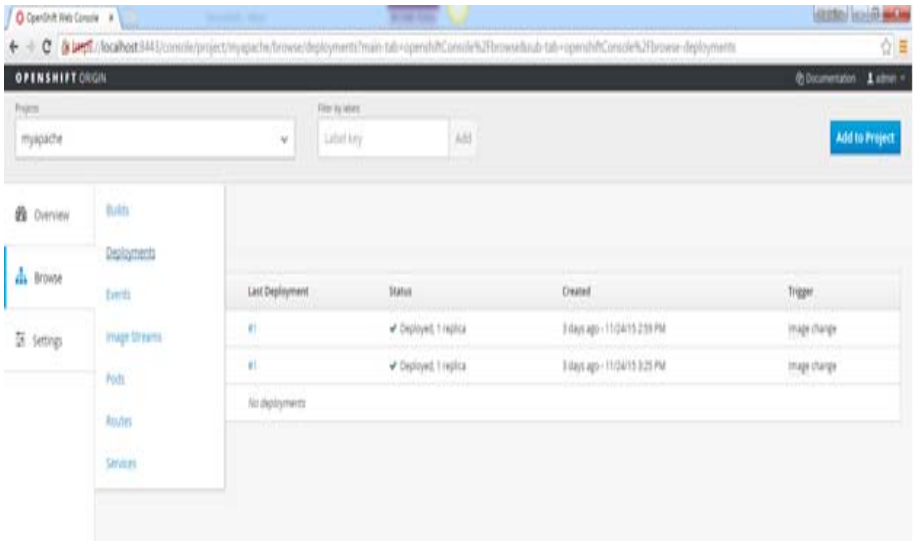

Figura 9

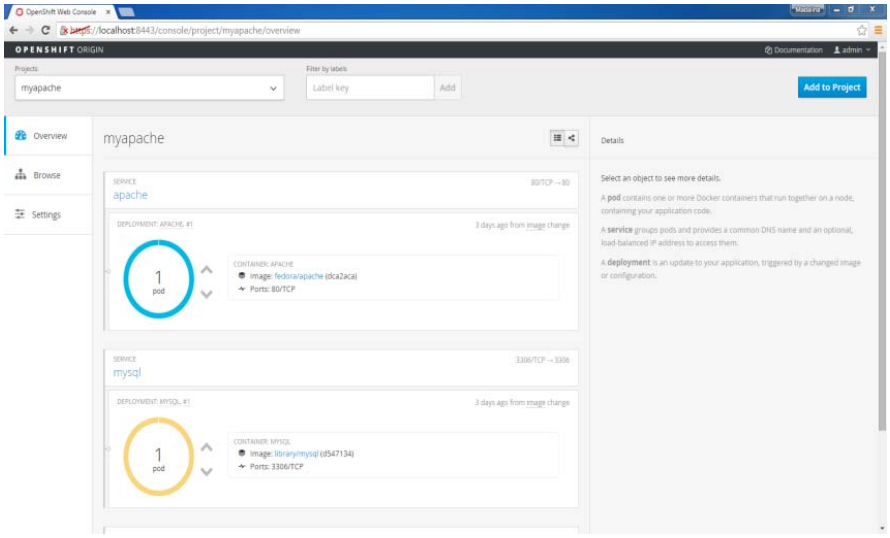

Figura 10

În secțiunea Browse, Image Streams se observă fluxurile de imagini instalate (proiectul), conform figurilor 11, 12 şi 13:

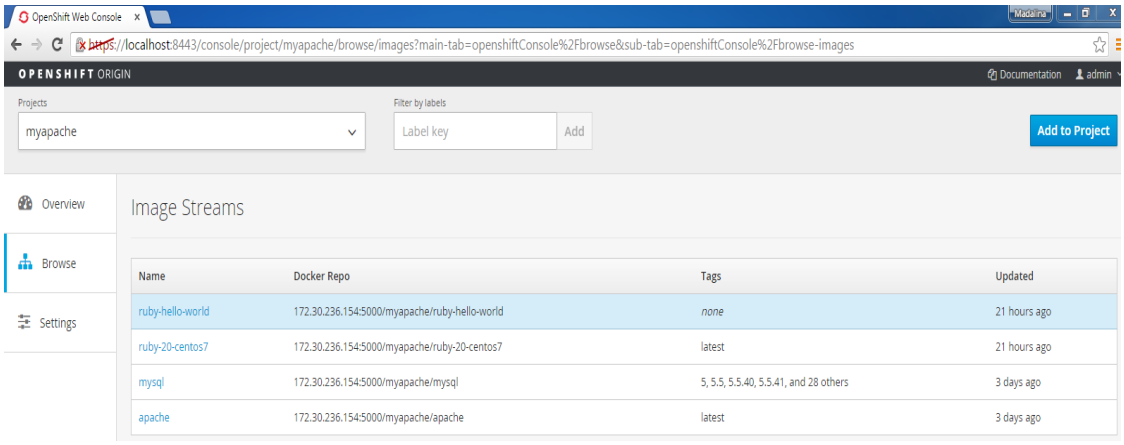

Figura 11

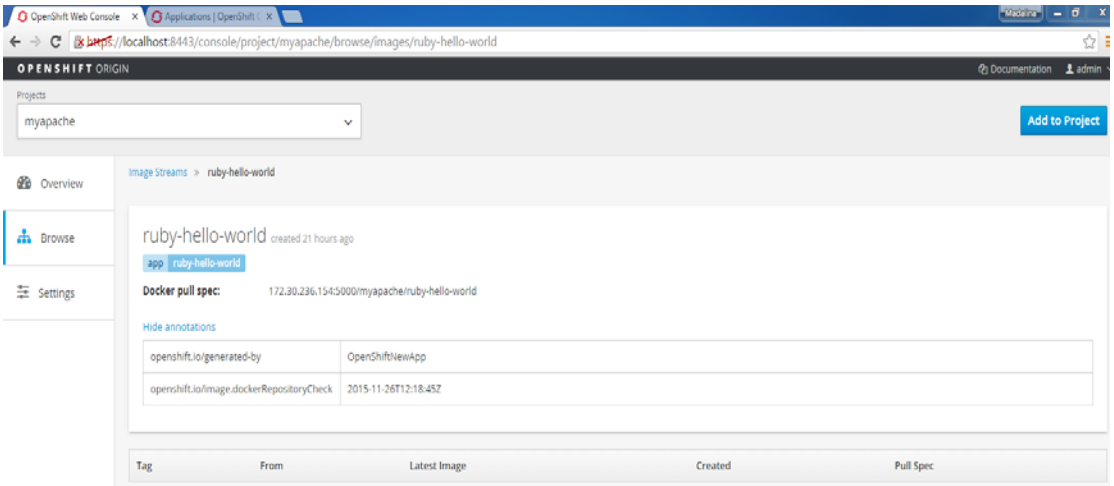

Figura 12

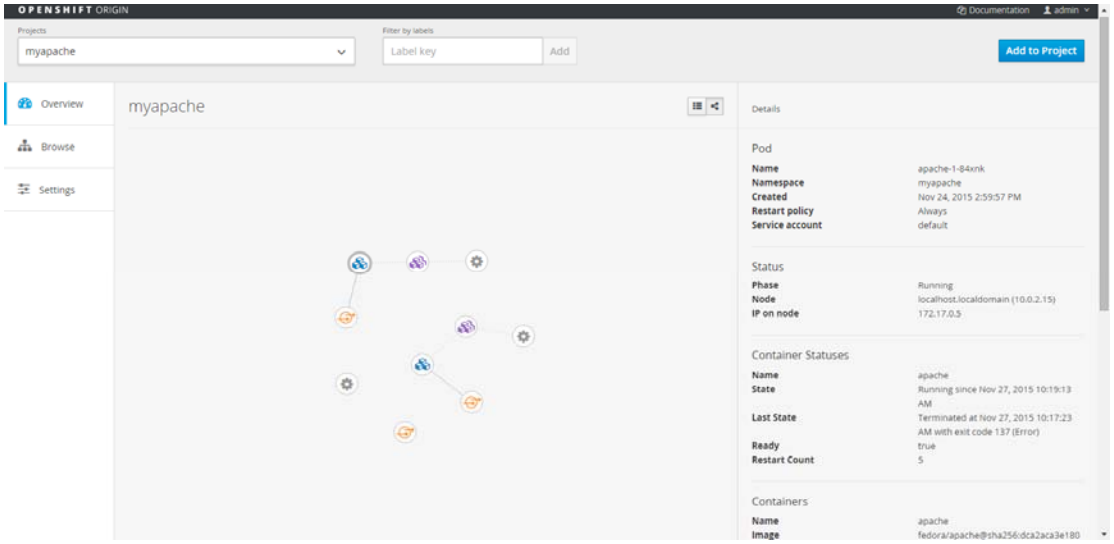

Figura 13

5. Concluzii

Datorită diversității tutorialelor disponibile a fost aleasă pentru experimentare soluția cea mai cea mai eficientă - OpenShift, care poate fi gândită ca un prim pas în ajutorul cercetătorilor din diferite domenii, în utilizarea unei soluții de tip PaaS. Au fost prezentate rezultatele unui experiment de instalare a platformei OpenShift în mod local. Funcționalitatea sistemului OpenShift v3 poate fi extinsă la capacitatea de a genera imagini Docker direct din codul sursă al aplicațiilor pentru utilizator.

BIBLIOGRAFIE

1. **ZAMFIR, M.; NEAGU, G.; FLORIAN, V.; VREJOIU, M.; STANCIU, A.; PREDA, Ş.:** Experimentare platformă Open Source de servicii Cloud pentru activitatea de cercetare-dezvoltare. Faza de elaborare 2 - Instalare soluție Open Source şi experimentare servicii, raport de cercetare ICI, proiect PN 09230408, decembrie 2015.
2. https://www.openshift.com/about/index.html
3. https://enterprise.openshift.com/features/index.html
4. (http://kubernetes.io/v1.1/docs/whatisk8s.html)
5. https://docs.vagrantup.com/v2/why-vagrant/index.html
6. http://www.todaysoftmag.ro/article/730/vagrant-pentru-incepatori
7. https://www.virtualbox.org/manual/ch01.html