Creazione Cloudy Shell ed installazione software di base

Si suppone che sia stata creata la VM che ospiterà la Cloudify Shell (es. pr01cloudifysh01).

Entrare nella VM e modificare **/etc/hosts** aggiungendo:

```
127.0.0.1 <hostname>
```

Importare il file cloudify-2.7.2reply.zip

Nella VM appena creata, sotto la cartella **/usr**, importare (ad esempio mediante client FTP FileZilla) la cartella **cloudify-2.7.2reply**, presente nella cartella *CONFIGURAZIONI → Cloudify.* La cartella **cloudify-2.7.2reply** si compone di 3 file compressi:

- **1-cloudify-2.7.2reply.zip**
- **2-cloudify-2.7.2reply.zip**
- **3-cloudify-2.7.2reply.zip**

Installare unzip:

```
# apt-get install unzip
```

Estrarre il file compressi di Cloudify direttamente nella cartella `/usr/cloudify-2.7.2reply`:

```
# unzip /usr/cloudify-2.7.2reply/1-cloudify-2.7.2reply.zip -d /usr/cloudify-2.7.2reply/
# unzip /usr/cloudify-2.7.2reply/2-cloudify-2.7.2reply.zip -d /usr/cloudify-2.7.2reply/
# unzip /usr/cloudify-2.7.2reply/3-cloudify-2.7.2reply.zip -d /usr/cloudify-2.7.2reply/
```

Dalla cartella `/usr/cloudify-2.7.2reply` rimuovere i 3 file compressi, di modo che appaia come di seguito:

```
bin/
clouds/
config/
deploy/
lib/
logs/
notice.txt
policy/
README.txt
```

```
recipes/
START_HERE.htm
tools/
```

Installare java 6

Installare java 6 da repository Webupd8 mediante i seguenti passi:

```
# sudo add-apt-repository ppa:webupd8team/java
# sudo apt-get update
# sudo apt-get install oracle-java6-installer
```

Modificare **/etc/profile** aggiungendo:

```
export JAVA_HOME="/usr/lib/jvm/java-6-oracle"
export PATH="$JAVA_HOME/bin:$PATH"
```

Quindi riavviare la macchina (oppure eseguire `# source /etc/profile`)

Configurare il cloud driver

Nella directory **/usr/cloudify-2.7.2reply/clouds** sono presenti i Cloud Driver per le diverse infrastrutture IaaS supportate da Cloudify.

In Appendice, si riportano le configurazioni effettuate per lo IaaS OpenStack, distinguendo sulla base della configurazione di rete (“con Floating IP”, e “senza Floatign IP”).

Si fa notare che per ciascun Project OpenStack nel quale verrà installa la VM Cloudify Manager, è necessario che sia creata una cartella del tipo:

```
openstack-icehouse-<nome_project>
```

nella directory **/usr/cloudify-2.7.2reply/clouds**

Ora, si suppone che:

- esiste il project (es. “<nome_project>”), nel quale Cloudify dovrà allocare le VM dei servizi IaaS/PaaS della piattaforma PaaS;
- di questo project si conoscono la **username** e la **password** dell’utente;
- in questo project è stato creato un keypair con il nome “**key-<nome_project>**” la cui chiave privata (key-<nome_project>.pem) è detenuta dagli amministratori della piattaforma PaaS.

Creazione Cloud Driver

Posizionarsi in **/usr/cloudify-2.7.2reply/clouds** ed eseguire:

```
# cp openstack-havana openstack-icehouse-<nome_project>
```

dove `<nome_project>` è il nome del project utente.

N.B: il cloud driver per havana è identico a quello di Icehouse e Juno.

Posizionarsi in **/usr/cloudify-2.7.2reply/clouds/openstack-icehouse-<nome_project>** e modificare i file `openstack-havana-cloud.properties` **e** `openstack-havana-cloud.groovy` come da Appendice.

In **/usr/cloudify-2.7.2reply/clouds/openstack-icehouse-<nome_project>/upload** copiare la chiave privata del keypair (key-<nome_project>.pem).

Keystone in HTTPS

Se il servizio Keystone di OpenStack è in HTTPS, ed il certificato del server Keystone non è validato, è necessario memorizzare tale certificato nel registro dei certificati di java (cacert), dal momento che Cloudify non sarà in grado di validarlo. Questo va fatto sia nella **VM Cloudify Shell** che nella **VM Cloudify Manager** e nelle **VM Cloudify Application**.

Cloudify SHELL

Per far cio, nella cloudify shell eseguire:

keytool -import -trustcacerts -file <PATH_DEL_cacert.pem> -keystore /usr/lib/jvm/java-6-oracle/jre/lib/security/cacerts -alias <nome_generico>

Quindi inserire la password “changeit” e poi rispondere “yes”

OSSERVAZIONE: E' necessario inserire tutto il PATH del certificato (path assoluto)

NB: cacert.pem è il nome del certificato di keystone. Il nome del certificato può essere anche diverso da cacert (es. certificato.pem) e può essere anche .crt (es. something.crt)

OSSERVAZIONE: il **keystore** potrebbe essere diverso a seconda della distribuzione java usata. Per verificarlo, basta fare una ricerca di "cacerts" oppure un "which java". Ad esempio, con le installazioni di openjdk, potrebbe essere:

/usr/local/java/jdk1.7.0_45/jre/lib/security/cacerts

/usr/local/java/jdk1.7.0_45/jre/lib/security/cacerts

VM Management e Application

I seguenti passi consentono di memorizzare il certificato nel keystore della VM di Management e Application

1. In **/usr/cloudify-2.7.2reply/clouds/openstack-icehouse-<nome_project>/upload** copiare certificato del server keystone
2. Modificare il file **/usr/cloudify-2.7.2reply/clouds/openstack-icehouse-<nome_project>/upload/bootstrap-management.sh** come riportato in Appendice.

Network

Rete "senza Floating IP" – rete pre-esistente pubblica

Per poter usare una rete flat pre-esistente ad indirizzamento pubblico, leggere quanto presente in Appendice, nel Cloud Driver relativo alla rete "senza Floating IP".

Rete "con Floating IP"

Per poter usare una rete flat pre-esistente ad indirizzamento privato, ma consentire alle VM di ricevere un Floating IP, leggere quanto presente in Appendice, nel Cloud Driver relativo alla rete "con Floating IP".

Utilizzare utente Ubuntu invece di root

Per poter eseguire operazioni nelle VM Manager ed Application come utente "ubuntu", nel Cloud Driver `nome_driver.properties` è necessario commentare la sezione "Dati per utente root" e decommentare la sezione "Dati per utente ubuntu".

Sorgenti da storage ad oggetti della piattaforma PaaS

Nel processo di installazione della VM Manager e delle VM Application, vengono scaricati ed installati gli eseguibili di Cloudify e Java.

Si fa in modo che tali eseguibili vengano scaricati non da Internet ma da repository interni a alla piattaforma PaaS, allo scopo di avere un repository il più possibile sotto controllo ed evitare malfunzionamenti legati alla rete Internet. Per questo motivo, i file **jdk-6u32-linux-x64.bin, jdk-6u32-linux-i586.bin**, **cloudify-2.7.2reply.tar.gz** (presenti nella cartella CONFIGURAZIONI → Cloudify) devono essere caricati in un container SWIFT del tenant "orchestrator". Il nome del container dovrà essere **publicAppRep.**

Per far ciò, modificare il file **bootstrap_management.sh** del Cloud Driver (vedere Appendice), per scaricare i pacchetti dallo storage ad oggetti (SWIFT) della piattaforma PaaS.

Queste le variabili da valorizzare in fase di configurazione del Cloud Driver:

```
# Link al repository pubblico della piattaforma PaaS
CLOUDIFY_SWIFT_LINK="<endpoint_SWIFT_PaaS>/publicAppRepo/cloudify-2.7.2reply.tar.gz"
JDK_64_SWIFT_LINK="<endpoint_SWIFT_PaaS>/publicAppRepo/jdk-6u32-linux-x64.bin"
JDK_32_SWIFT_LINK="<endpoint_SWIFT_PaaS>/publicAppRepo/jdk-6u32-linux-i586.bin"
```

Maggiori dettagli in Appendice.

Operazioni sulla Cloudify Manager

A questo punto, tutto è stato configurato per poter eseguire la Cloudify Manager in un project OpenStack per il quale è stato creato il relativo Cloud Driver.

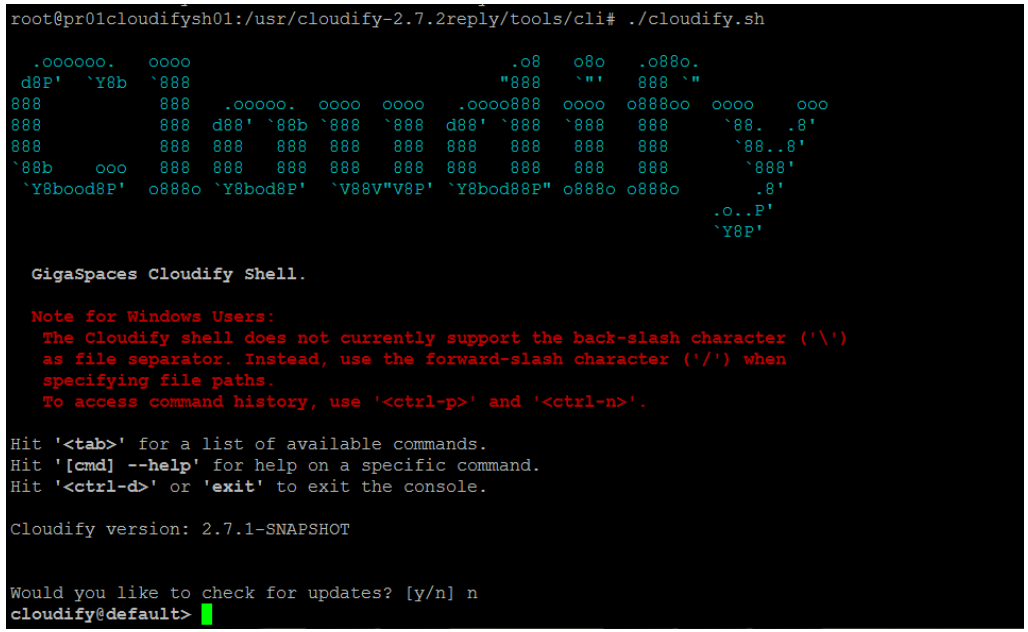
Accedere alla Cloudify Shell, ed eseguire lo script:

/usr/cloudify-2.7.2reply/tools/cli/cloudify.sh

N.B: se viene proposta la ricerca di nuove versioni (Would you like to check for updates? [y/n]), rispondere **n**.

Si accederà così alla shell di Cloudify:

cloudify@default>



Digitando help, si potrà visualizzare la lista dei comandi eseguibili dalla Cloudify Shell.

Avvio della VM Manager

Il comando **bootstrap-cloud**, in particolare, consente di avviare il processo di creazione di una Cloudify Manager all'interno del project, previa opportuna configurazione del relativo Cloud Driver. Supponendo che il Cloud Driver del project sia openstack-icehouse-<nome_progetto>, eseguire:

cloudify@default> bootstrap-cloud --verbose openstack-icehouse-<nome_progetto>

Seguiranno una serie di chiamate alle API REST di OpenStack per la verifica delle credenziali e per l'allocazione delle risorse.

Distruzione della VM Manager

Bisogna prima connettersi ad una VM Manager:

cloudify@default> connect <IP_VM_Manager>

e poi eseguire il seguente comando:

cloudify@default> teardown-cloud openstack-icehouse-<nome_progetto>

Seguiranno una serie di chiamate alle API REST di OpenStack per la disallocazione delle risorse.

Reboot della Vm Manager senza perdita dei dati

Nel Cloud Driver viene specificato un path, nelle VM di Management, in cui verranno salvate le informazioni sullo stato dei deploy.

- **persistencePath="/opt/vmmanagerstate" (se si suna un utente root);**
- **persistencePath="/home/ubuntu/vmmanagerstate" (se si suna un utente ubuntu);**

Può capitare che in seguito ad errori nella VM Manager, sia necessario riavviare la stessa VM senza perdere i dati sui deploy. Di seguito i passi da eseguire:

- entrare nella Cloudify Shell e connettersi alla VMManager:

  cloudify@default> connect <IP_VM_Manager>

- eseguire:

  cloudify@default> shutdown-managers -timeout 10 --verbose

- eseguire l'avvio usando il parametro "-use-existing"

cloudify@default> bootstrap-cloud --verbose -use-existing openstack-icehouse-<nome_project>

Security Group creati da Cloudify

A titolo informativo, si riportano i Security Group creati automaticamente da Cloudify:

cloudify-manager-management

Manage Security Group Rules: cloudify-manager-management

Security Group Rules

+ Add Rule | Delete Rules

Direction	Ether Type	IP Protocol	Port Range	Remote	Actions
Egress	IPv4	Any	-	0.0.0.0/0 (CIDR)	Delete Rule
Egress	IPv6	Any	-	::/0 (CIDR)	Delete Rule
Ingress	IPv4	TCP	22 (SSH)	0.0.0.0/0 (CIDR)	Delete Rule
Ingress	IPv4	TCP	4174	cloudify-manager-cluster	Delete Rule
Ingress	IPv4	TCP	6666	cloudify-manager-cluster	Delete Rule
Ingress	IPv4	TCP	7000	cloudify-manager-cluster	Delete Rule
Ingress	IPv4	TCP	7001	cloudify-manager-cluster	Delete Rule
Ingress	IPv4	TCP	7002	cloudify-manager-cluster	Delete Rule
Ingress	IPv4	TCP	7003	cloudify-manager-cluster	Delete Rule
Ingress	IPv4	TCP	7010 - 7110	cloudify-manager-cluster	Delete Rule
Ingress	IPv4	TCP	8099	0.0.0.0/0 (CIDR)	Delete Rule
Ingress	IPv4	TCP	8100	0.0.0.0/0 (CIDR)	Delete Rule

Displaying 12 items

cloudify-manager-cluster

Manage Security Group Rules: cloudify-manager-cluster

Security Group Rules

+ Add Rule | Delete Rules

Direction	Ether Type	IP Protocol	Port Range	Remote	Actions
Egress	IPv4	Any	-	0.0.0.0/0 (CIDR)	Delete Rule
Egress	IPv6	Any	-	::/0 (CIDR)	Delete Rule

Displaying 2 items

cloudify-manager-agent

Manage Security Group Rules: cloudify-manager-agent

Security Group Rules

+ Add Rule | Delete Rules

Direction	Ether Type	IP Protocol	Port Range	Remote	Actions
Egress	IPv6	Any	-	::/0 (CIDR)	Delete Rule
Egress	IPv4	Any	-	0.0.0.0/0 (CIDR)	Delete Rule
Ingress	IPv4	TCP	7000	cloudify-manager-cluster	Delete Rule
Ingress	IPv4	TCP	7001	cloudify-manager-cluster	Delete Rule
Ingress	IPv4	TCP	7002	cloudify-manager-cluster	Delete Rule
Ingress	IPv4	TCP	7010 - 7110	cloudify-manager-cluster	Delete Rule

Osservazioni: modifiche ai file di configurazione

Aumentare la verbosità dell'output del processo di installazione

Modificare il file **/usr/cloudify-2.7.2reply/config/gs_logging.properties** aggiungendo:

openstack.wire.level = FINE
org.cloudifysource.esc.driver.provisioning.openstack.level = ALL

Modifica intervalli temporale di client e server

$CFY_HOME/config/gs.properties

Modificare:

- `-Dsun.rmi.dgc.client.gcInterval=360000000` in `-Dsun.rmi.dgc.client.gcInterval=3600000000`
- `-Dsun.rmi.dgc.server.gcInterval=360000000` in `-Dsun.rmi.dgc.server.gcInterval=3600000000`

$CFY_HOME/bin/setenv.sh

Modificare:

- `RMI_OPTIONS="-Dsun.rmi.dgc.client.gcInterval=36000000 -Dsun.rmi.dgc.server.gcInterval=36000000 ….”`

in `RMI_OPTIONS="-Dsun.rmi.dgc.client.gcInterval=3600000000 -Dsun.rmi.dgc.server.gcInterval=3600000000 ….”`

Appendice

Cloud Driver – rete "con Floating IP"

Nella cartella in **/usr/cloudify-2.7.2reply/clouds/openstack-icehouse-<nome_project>,** modificare come riportato i seguenti file:

openstack-havana-cloud.properties

```
// Credentials - Enter your cloud provider account credentials here
user="<username>"                // enter the Horizon console user name
apiKey="<password>"       // enter the Horizon console password
tenant="<nome-project>"
keyFile="<nome_keypair>.pem"
keyPair="<nome_keypair>"

// For instance: "https://<IP>:5000/v2.0/"
openstackUrl="<keystone_endpoint>:5000/v2.0"

//Dati del Cloud Driver
nomeCloudDriver="openstack-icehouse-$tenant"
prefissoMachineAgent="cfy-agn-"
//Questo prefisso viene usato anche per la rete manager e per i security groups
prefissoMachineManager="cfy-mngt-"

//Nome della(e) rete(i) private pre-esistenti da usare
nomeRetePrivataShared="PaaS-management-net"
nomeRetePrivataProject="rete-privata"
//nomeAltraRete="xxxx"


////////////////////
// Immagini Virtuali
////////////////////

//Ubuntu precise - consente accesso ssh come root
imageIdUbuntu12="<Region>/<imageID>"


availabilityZone="nova"


////////////////////
// Compute Template
////////////////////
```

```
//Tipo_di_template_per_VM_CFY_MNG
//cfyTemplate="XCLOUDIFY_UBUNTU14"
cfyTemplate="XCLOUDIFY_UBUNTU12"

//Dati per utente root
nomeDirectoryRemota="/root/gs-files"
nomeUtenteTemplate="root"
nomeDirectoryLocale="upload"
persistencePath="/opt/vmmanagerstate"

//Dati per utente ubuntu
//nomeDirectoryRemota="/home/ubuntu/gs-files"
//nomeUtenteTemplate="ubuntu"
//nomeDirectoryLocale="upload"
//persistencePath="/home/ubuntu/vmmanagerstate"

/////////
//Flavor
/////////

//small
smallGold="<Region>/<flavorID>"
smallSilver="<Region>/<flavorID>"
smallBronze="<Region>/<flavorID>"
smallMachineMemory=<memory_size>


//medium
mediumGold="<Region>/<flavorID>"
mediumSilver="<Region>/<flavorID>"
mediumBronze="<Region>/<flavorID>"
mediumMachineMemory=<memory_size>


//large
largeGold="<Region>/<flavorID>"
largeSilver="<Region>/<flavorID>"
largeBronze="<Region>/<flavorID>"
largeMachineMemory=<memory_size>
```

openstack-havana-cloud.groovy

```
/***************
 * Cloud configuration file for the openstack-havana cloud
 */
cloud
{
       //name = "openstack-icehouse-sielte-Demo"
     name = nomeCloudDriver

       /********
        * General configuration information about the cloud driver implementation.
        */
       configuration
       {
           // Optional. The cloud implementation class. Defaults to the build in jclouds-based provisioning driver.
              className "org.cloudifysource.esc.driver.provisioning.openstack.OpenStackCloudifyDriver"
              networkDriverClassName "org.cloudifysource.esc.driver.provisioning.network.openstack.OpenstackNetworkDriver"
              storageClassName "org.cloudifysource.esc.driver.provisioning.storage.openstack.OpenstackStorageDriver"

           // Optional. The template name for the management machines. Defaults to the first template in the templates section below.
           //managementMachineTemplate "XCLOUDIFY_UBUNTU12"
              managementMachineTemplate cfyTemplate

              // Optional. Indicates whether internal cluster communications should use the machine private IP. Defaults to true.
              connectToPrivateIp true

              bootstrapManagementOnPublicIp false

              //remoteUsername "root"
              //remotePassword "Your Password Here"

              persistentStoragePath persistencePath

       }


       /*************
        * Provider specific information.
        */
       provider
       {
              provider "openstack-nova"
```

```
            // Optional. The HTTP/S URL where cloudify can be downloaded from by newly started machines. Defaults to downloading
the
            // cloudify version matching that of the client from the cloudify CDN.
            // Change this if your compute nodes do not have access to an internet connection, or if you prefer to use a
            // different HTTP server instead.
            // IMPORTANT: the default linux bootstrap script appends '.tar.gz' to the url whereas the default windows script
appends '.zip'.
            // Therefore, if setting a custom URL, make sure to leave out the suffix.
            // cloudifyUrl "http://repository.cloudifysource.org/org/cloudifysource/2.7.1-6300-RELEASE/gigaspaces-cloudify-2.7.1-
ga-b6300.zip"

            // Mandatory. The prefix for new machines started for servies.
            machineNamePrefix prefissoMachineAgent
            // Optional. Defaults to true. Specifies whether cloudify should try to deploy services on the management machine.
            // Do not change this unless you know EXACTLY what you are doing.

            //
            managementOnlyFiles ([])

            // Optional. Logging level for the intenal cloud provider logger. Defaults to INFO.
            sshLoggingLevel "WARNING"

            // Mandatory. Name of the new machine/s started as cloudify management machines. Names are case-insensitive.
            managementGroup prefissoMachineManager

            // Mandatory. Number of management machines to start on bootstrap-cloud. In production, should be 2. Can be 1 for dev.
            numberOfManagementMachines 1

            //commento - Giuseppe
            //reservedMemoryCapacityPerMachineInMB 1024

      }

      /*************
       * Cloud authentication information
       */
      user
      {
            // Optional. Identity used to access cloud.
            user "${tenant}:${user}"
            apiKey apiKey
```

```
}

        /*******************
         * Cloud storage configuration.
         */
        cloudStorage
        {
                    templates ([
                           SMALL_BLOCK : storageTemplate
                           {
                                                    deleteOnExit false
                                                    partitioningRequired true
                                                    size 1
                                                    path "/storage"
                                                    namePrefix "cloudify-storage-volume"
                                                    deviceName "/dev/vdc"
                                                    fileSystemType "ext4"
                                                    custom (["openstack.storage.volume.zone":availabilityZone])
                           }
                    ])
        }



    /*******************
     * Cloud networking configuration.
     */
/*
    cloudNetwork
    {

        // Details of the management network, which is shared among all instances of the Cloudify Cluster.
        management
        {
               networkConfiguration
               {
                      //name  "private-net-Demo"
                name  nomeReteManager
                      subnets ([
                             subnet {
                                    name nomeSottoreteManager
```

```
                                range rangeSottoreteManager
                                options ([ "gateway" : gatewaySottoreteManager ])
                            //tra le opzioni c'è anche dnsNameServers null
                               }
                        ])
                        custom ([ "associateFloatingIpOnBootstrap" : "false" ])
                }
        }



        // Templates for networks which applications may use
        // Only service isntances belonging to an application will be attached to this network.
        templates ([
               "APPLICATION_NET" : networkConfiguration
               {
                      name  "app-net"
                      subnets {
                             subnet {
                                    name "app-subnet"
                                    range "160.0.0.0/24"
                                    options { gateway "null" }
                            //tra le opzioni c'è anche dnsNameServers "null"
                               }
                      }
                      custom ([ "associateFloatingIpOnBootstrap" : "false" ])
               }
        ])
}

*/

       cloudCompute
       {

              /***********
               * Cloud machine templates available with this cloud.
               */
              templates ([
                  XCLOUDIFY_UBUNTU12 : computeTemplate
                     {
                         imageId imageIdUbuntu12
                           machineMemoryMB largeMachineMemory
```

```
        hardwareId largeGold
            remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        //options (["keyPairName" : keyPair])
            options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])

        // Optional. Use existing networks.
        computeNetwork { networks ([ nomeRetePrivataShared, nomeRetePrivataProject ]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext
a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])

    },
    small_gold_UBUNTU12 : computeTemplate
                {
        imageId imageIdUbuntu12
        machineMemoryMB smallMachineMemory
        hardwareId smallGold
            remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        //options (["keyPairName" : keyPair])
```

```
                    options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])

            // Optional. Use existing networks.
            computeNetwork { networks ([ nomeRetePrivataProject, nomeRetePrivataShared ]) }

            // when set to 'true', agent will automatically start after reboot.
            autoRestartAgent true

            // Optional. Overrides to default cloud driver behavior.
            // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext
a
            overrides ([
                    "openstack.endpoint": openstackUrl
            ])

            // enable sudo.
            privileged true

            // optional. A native command line to be executed before the cloudify agent is started.
            initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

            //optional - set the availability zone, required to match storage
            custom (["openstack.compute.zone":availabilityZone])
    },
    small_silver_UBUNTU12 : computeTemplate
         {
            imageId imageIdUbuntu12
            machineMemoryMB smallMachineMemory
            hardwareId smallSilver
                remoteDirectory nomeDirectoryRemota
            localDirectory nomeDirectoryLocale
            keyFile keyFile
            username nomeUtenteTemplate

            //options (["keyPairName" : keyPair])
                options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])

            // Optional. Use existing networks.
            computeNetwork { networks ([ nomeRetePrivataProject, nomeRetePrivataShared ]) }

            // when set to 'true', agent will automatically start after reboot.
            autoRestartAgent true
```

```
                // Optional. Overrides to default cloud driver behavior.
                // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext
a
                overrides ([
                        "openstack.endpoint": openstackUrl
                ])

                // enable sudo.
                privileged true

                // optional. A native command line to be executed before the cloudify agent is started.
                initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

                //optional - set the availability zone, required to match storage
                custom (["openstack.compute.zone":availabilityZone])
        },
        small_bronze_UBUNTU12 : computeTemplate
             {
              imageId imageIdUbuntu12
                   machineMemoryMB smallMachineMemory
              hardwareId smallBronze
                  remoteDirectory nomeDirectoryRemota
              localDirectory nomeDirectoryLocale
              keyFile keyFile
              username nomeUtenteTemplate

             //options (["keyPairName" : keyPair])
                  options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])

              // Optional. Use existing networks.
              computeNetwork { networks ([ nomeRetePrivataProject, nomeRetePrivataShared ]) }

              // when set to 'true', agent will automatically start after reboot.
              autoRestartAgent true

              // Optional. Overrides to default cloud driver behavior.
              // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext
a
              overrides ([
                      "openstack.endpoint": openstackUrl
              ])

              // enable sudo.
```

```
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
    },
    medium_gold_UBUNTU12 : computeTemplate
         {
        imageId imageIdUbuntu12
        machineMemoryMB mediumMachineMemory
        hardwareId mediumGold
            remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        //options (["keyPairName" : keyPair])
            options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])

        // Optional. Use existing networks.
        computeNetwork { networks ([ nomeRetePrivataProject, nomeRetePrivataShared ]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext
a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
    },
```

```
medium_silver_UBUNTU12 : computeTemplate
      {
       imageId imageIdUbuntu12
       machineMemoryMB mediumMachineMemory
       hardwareId mediumSilver
           remoteDirectory nomeDirectoryRemota
       localDirectory nomeDirectoryLocale
       keyFile keyFile
       username nomeUtenteTemplate

       //options (["keyPairName" : keyPair])
           options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])

       // Optional. Use existing networks.
       computeNetwork { networks ([ nomeRetePrivataProject, nomeRetePrivataShared ]) }

       // when set to 'true', agent will automatically start after reboot.
       autoRestartAgent true

       // Optional. Overrides to default cloud driver behavior.
       // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext
a
       overrides ([
               "openstack.endpoint": openstackUrl
       ])

       // enable sudo.
       privileged true

       // optional. A native command line to be executed before the cloudify agent is started.
       initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

       //optional - set the availability zone, required to match storage
       custom (["openstack.compute.zone":availabilityZone])
},
medium_bronze_UBUNTU12 : computeTemplate
       {
       imageId imageIdUbuntu12
       machineMemoryMB mediumMachineMemory
       hardwareId mediumBronze
              remoteDirectory nomeDirectoryRemota
       localDirectory nomeDirectoryLocale
       keyFile keyFile
```

```
        username nomeUtenteTemplate

        //options (["keyPairName" : keyPair])
              options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])

        // Optional. Use existing networks.
        computeNetwork { networks ([ nomeRetePrivataProject, nomeRetePrivataShared ]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext
a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
},
large_gold_UBUNTU12 : computeTemplate
        {
        imageId imageIdUbuntu12
        machineMemoryMB largeMachineMemory
        hardwareId largeGold
              remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        //options (["keyPairName" : keyPair])
              options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])

        // Optional. Use existing networks.
        computeNetwork { networks ([ nomeRetePrivataProject, nomeRetePrivataShared ]) }
```

```
                // when set to 'true', agent will automatically start after reboot.
                autoRestartAgent true

                // Optional. Overrides to default cloud driver behavior.
                // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext
a
                overrides ([
                        "openstack.endpoint": openstackUrl
                ])

                // enable sudo.
                privileged true

                // optional. A native command line to be executed before the cloudify agent is started.
                initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

                //optional - set the availability zone, required to match storage
                custom (["openstack.compute.zone":availabilityZone])
        },
        large_silver_UBUNTU12 : computeTemplate
                {
                imageId imageIdUbuntu12
                machineMemoryMB largeMachineMemory
                hardwareId largeSilver
                       remoteDirectory nomeDirectoryRemota
                localDirectory nomeDirectoryLocale
                keyFile keyFile
                username nomeUtenteTemplate

                //options (["keyPairName" : keyPair])
                       options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])

                // Optional. Use existing networks.
                computeNetwork { networks ([ nomeRetePrivataProject, nomeRetePrivataShared ]) }

                // when set to 'true', agent will automatically start after reboot.
                autoRestartAgent true

                // Optional. Overrides to default cloud driver behavior.
                // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext
a
                overrides ([
                        "openstack.endpoint": openstackUrl
```

```
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
    },
    large_bronze_UBUNTU12 : computeTemplate
        {
        imageId imageIdUbuntu12
        machineMemoryMB largeMachineMemory
        hardwareId largeBronze
               remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        //options (["keyPairName" : keyPair])
               options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])

        // Optional. Use existing networks.
        computeNetwork { networks ([ nomeRetePrivataProject, nomeRetePrivataShared ]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext
a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"
```

```
                        //optional - set the availability zone, required to match storage
                        custom (["openstack.compute.zone":availabilityZone])
                    }
            ])

    }

    /*****************
     * Optional. Custom properties used to extend existing drivers or create new ones.
     */
    //custom ([:])
      // Se si usa il persistent path, e' necessario usare questo comando custom
      custom (["org.cloudifysource.clearRemoteDirectoryOnStart" : true])
}
```

Cloud Driver – rete "senza Floating IP"

openstack-havana-cloud.properties

```
// Credentials - Enter your cloud provider account credentials here
user="<username>"                // enter the Horizon console user name
apiKey="<password>"       // enter the Horizon console password
tenant="<nome_project>"
keyFile="<nome_keypair>.pem"
keyPair="<nome_keypair>"


// For instance: "https://<IP>:5000/v2.0/"
openstackUrl="<keystone_endpoint>:5000/v2.0"

//Dati del Cloud Driver
nomeCloudDriver="openstack-icehouse-$tenant"
prefissoMachineAgent="cfy-agn-"
//Questo prefisso viene usato anche per la rete manager e per i security groups
prefissoMachineManager="cfy-mngt-"
//Numero di Macchine di Management
numberOfMngtMachines="1"

//Rete esterna
nomeReteEsterna="public-net"
//Nome della(e) rete(i) private pre-esistenti da usare
//nomeAltraRete="xxxx"
```

```
//////////////////
// Immagini Virtuali
//////////////////


//Immagine PaaS-Ubuntu-12.04-x86_64 - consente accesso ssh come root
imageIdUbuntu12="<Region>/<imageID>"

availabilityZone="nova"


//////////////////
// Compute Template
//////////////////
//Tipo_di_template_per_VM_CFY_MNG
//cfyTemplate="XCLOUDIFY_UBUNTU14"
cfyTemplate="XCLOUDIFY_UBUNTU12"

//Dati per utente root
nomeDirectoryRemota="/root/gs-files"
nomeUtenteTemplate="root"
nomeDirectoryLocale="upload"
persistencePath="/opt/vmmanagerstate"

//Dati per utente ubuntu
//nomeDirectoryRemota="/home/ubuntu/gs-files"
//nomeUtenteTemplate="ubuntu"
//nomeDirectoryLocale="upload"
//persistencePath="/home/ubuntu/vmmanagerstate"

/////////
//Flavor
/////////

//small
smallGold="<Region>/<flavorID>"
smallSilver="<Region>/<flavorID>"0"
smallBronze="<Region>/<flavorID>"
smallMachineMemory=<memory_size>
```

```
//medium
mediumGold="<Region>/<flavorID>"
mediumSilver="<Region>/<flavorID>"
mediumBronze="<Region>/<flavorID>"
mediumMachineMemory=<memory_size>


//large
largeGold="<Region>/<flavorID>"
largeSilver="<Region>/<flavorID>""
largeBronze="<Region>/<flavorID>"
largeMachineMemory=<memory_size>
```

openstack-havana-cloud.groovy

```
/***************
 * Cloud configuration file for the openstack-havana cloud
 */
cloud
{
	name = nomeCloudDriver

	/********
	 * General configuration information about the cloud driver implementation.
	 */
	configuration
	{
		className "org.cloudifysource.esc.driver.provisioning.openstack.OpenStackCloudifyDriver"
		networkDriverClassName "org.cloudifysource.esc.driver.provisioning.network.openstack.OpenstackNetworkDriver"
		storageClassName "org.cloudifysource.esc.driver.provisioning.storage.openstack.OpenstackStorageDriver"

		managementMachineTemplate cfyTemplate
		// Optional. Indicates whether internal cluster communications should use the machine private IP. Defaults to true.
		connectToPrivateIp true
	  //eventualmente configurare questi dati
		bootstrapManagementOnPublicIp false
		//remoteUsername nomeUtenteTemplate
		//remotePassword "Your Password Here"

		// Optional. Path to folder where management state will be written. Null indicates state will not be written.
	  persistentStoragePath persistencePath
	}

	/*************
```

```
        * Provider specific information.
        */
       provider
       {
              provider "openstack-nova"

              // Optional. The HTTP/S URL where cloudify can be downloaded from by newly started machines. Defaults to downloading
the
              // cloudify version matching that of the client from the cloudify CDN.
              // Change this if your compute nodes do not have access to an internet connection, or if you prefer to use a
              // different HTTP server instead.
              // IMPORTANT: the default linux bootstrap script appends '.tar.gz' to the url whereas the default windows script
appends '.zip'.
              // Therefore, if setting a custom URL, make sure to leave out the suffix.
              // cloudifyUrl "http://repository.cloudifysource.org/org/cloudifysource/2.7.1-6300-RELEASE/gigaspaces-cloudify-2.7.1-
ga-b6300.zip"

              // Mandatory. The prefix for new machines started for servies.
              // machineNamePrefix "cloudify-agent-" -- Giuseppe
              machineNamePrefix prefissoMachineAgent
              // Optional. Defaults to true. Specifies whether cloudify should try to deploy services on the management machine.
              // Do not change this unless you know EXACTLY what you are doing.

              //
              managementOnlyFiles ([])

              // Optional. Logging level for the intenal cloud provider logger. Defaults to INFO.
              sshLoggingLevel "WARNING"

              // Mandatory. Name of the new machine/s started as cloudify management machines. Names are case-insensitive.
              //managementGroup "cloudify-manager-"
              managementGroup prefissoMachineManager

              // Mandatory. Number of management machines to start on bootstrap-cloud. In production, should be 2. Can be 1 for dev.
              numberOfManagementMachines numberOfMngtMachines

              //reservedMemoryCapacityPerMachineInMB reservedMemCapacityPerMachineInMB

       }

       /*************
        * Cloud authentication information
        */
```

```
	user
	{
		// Optional. Identity used to access cloud.
		user "${tenant}:${user}"
		apiKey apiKey



	}

		/********************
		 * Cloud storage configuration.
		 */
		cloudStorage
		{
				templates ([
					SMALL_BLOCK : storageTemplate
					{
											deleteOnExit false
											partitioningRequired true
											size 1
											path "/storage"
											namePrefix "cloudify-storage-volume"
											deviceName "/dev/vdc"
											fileSystemType "ext4"
											custom (["openstack.storage.volume.zone":availabilityZone])
					}
			])
		}

	/********************
	 * Cloud networking configuration.
	 */

/* --- Decommento per poter utilizzare una rete esistente pubblica
	cloudNetwork
	{

		// Details of the management network, which is shared among all instances of the Cloudify Cluster.
		management
		{
			networkConfiguration
			{
```

```
                        name  "Cloudify-Management-Network"
                        subnets ([
                                subnet {
                                        name "Cloudify-Management-Subnet"
                                        range "177.86.0.0/24"
                                        options ([ "gateway" : "177.86.0.111" ])
                                }
                        ])
                        custom ([ "associateFloatingIpOnBootstrap" : "true" ])
                }
        }

        // Templates for networks which applications may use
        // Only service isntances belonging to an application will be attached to this network.
        templates ([
                "APPLICATION_NET" : networkConfiguration
                {
                        name  "Cloudify-Application-Network"
                        subnets {
                                subnet {
                                        name "Cloudify-Application-Subnet"
                                        range "160.0.0.0/24"
                                        options { gateway "null" }
                                }
                        }
                        custom ([ "associateFloatingIpOnBootstrap" : "true" ])
                }
        ])
    }
*/

    cloudCompute
    {

        /***********
         * Cloud machine templates available with this cloud.
         */
        templates ([
       //computeTemplate per le VMs di Management
       //la vNIC abilitata di default e' la prima rete che compare nella sezione computeNetwork
       //l'altra vNIC va configurata e abilitata, ad esempio nel bootstrap_management.sh
                XCLOUDIFY_UBUNTU12 : computeTemplate
                {
```

```
        imageId imageIdUbuntu12
        machineMemoryMB largeMachineMemory
        hardwareId largeGold
        remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])
        //options (["keyPairName" : keyPair])

        // Optional. Use existing networks.
        computeNetwork { networks ([nomeReteEsterna]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
},
        small_gold_UBUNTU12 : computeTemplate
        {
        imageId imageIdUbuntu12
        machineMemoryMB smallMachineMemory
        hardwareId smallGold
        remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])
```

```
        //options (["keyPairName" : keyPair])

        // Optional. Use existing networks.
        computeNetwork { networks ([nomeReteEsterna]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
},
small_silver_UBUNTU12 : computeTemplate
        {
        imageId imageIdUbuntu12
        machineMemoryMB smallMachineMemory
        hardwareId smallSilver
        remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])
        //options (["keyPairName" : keyPair])

        // Optional. Use existing networks.
                    computeNetwork { networks ([nomeReteEsterna]) }
        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext a
```

```
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
},
small_bronze_UBUNTU12 : computeTemplate
        {
        imageId imageIdUbuntu12
        machineMemoryMB smallMachineMemory
        hardwareId smallBronze
        remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])
        //options (["keyPairName" : keyPair])

        // Optional. Use existing networks.
        computeNetwork { networks ([nomeReteEsterna]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"
```

```
        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
},
medium_gold_UBUNTU12 : computeTemplate
        {
        imageId imageIdUbuntu12
        machineMemoryMB mediumMachineMemory
        hardwareId mediumGold
        remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])
        //options (["keyPairName" : keyPair])

        // Optional. Use existing networks.
        computeNetwork { networks ([nomeReteEsterna]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
},
medium_silver_UBUNTU12 : computeTemplate
        {
        imageId imageIdUbuntu12
        machineMemoryMB mediumMachineMemory
        hardwareId mediumSilver
```

```
        remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])
        //options (["keyPairName" : keyPair])

        // Optional. Use existing networks.
        computeNetwork { networks ([nomeReteEsterna]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
},
medium_bronze_UBUNTU12 : computeTemplate
        {
        imageId imageIdUbuntu12
        machineMemoryMB mediumMachineMemory
        hardwareId mediumBronze
        remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])
        //options (["keyPairName" : keyPair])

        // Optional. Use existing networks.
```

```
        computeNetwork { networks ([nomeReteEsterna]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
},
large_gold_UBUNTU12 : computeTemplate
        {
        imageId imageIdUbuntu12
        machineMemoryMB largeMachineMemory
        hardwareId largeGold
        remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])
        //options (["keyPairName" : keyPair])

        // Optional. Use existing networks.
        computeNetwork { networks ([nomeReteEsterna]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext a
        overrides ([
                "openstack.endpoint": openstackUrl
```

```
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
        custom (["openstack.compute.zone":availabilityZone])
},
large_silver_UBUNTU12 : computeTemplate
        {
        imageId imageIdUbuntu12
        machineMemoryMB largeMachineMemory
        hardwareId largeSilver
        remoteDirectory nomeDirectoryRemota
        localDirectory nomeDirectoryLocale
        keyFile keyFile
        username nomeUtenteTemplate

        options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])
        //options (["keyPairName" : keyPair])

        // Optional. Use existing networks.
        computeNetwork { networks ([nomeReteEsterna]) }

        // when set to 'true', agent will automatically start after reboot.
        autoRestartAgent true

        // Optional. Overrides to default cloud driver behavior.
        // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext a
        overrides ([
                "openstack.endpoint": openstackUrl
        ])

        // enable sudo.
        privileged true

        // optional. A native command line to be executed before the cloudify agent is started.
        initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

        //optional - set the availability zone, required to match storage
```

```
                custom (["openstack.compute.zone":availabilityZone])
        },
        large_bronze_UBUNTU12 : computeTemplate
                {
                imageId imageIdUbuntu12
                machineMemoryMB largeMachineMemory
                hardwareId largeBronze
                remoteDirectory nomeDirectoryRemota
                localDirectory nomeDirectoryLocale
                keyFile keyFile
                username nomeUtenteTemplate

                options (["keyPairName" : keyPair,"securityGroups" : ["zabbix"] as String[]])
                //options (["keyPairName" : keyPair])

                // Optional. Use existing networks.
                computeNetwork { networks ([nomeReteEsterna]) }

                // when set to 'true', agent will automatically start after reboot.
                autoRestartAgent true

                // Optional. Overrides to default cloud driver behavior.
                // When used with the default driver, maps to the overrides properties passed to the ComputeServiceContext a
                overrides ([
                        "openstack.endpoint": openstackUrl
                ])

                // enable sudo.
                privileged true

                // optional. A native command line to be executed before the cloudify agent is started.
                initializationCommand "#!/bin/sh\necho Inserire qui eventuali comandi per la VM `hostname`"

                //optional - set the availability zone, required to match storage
                custom (["openstack.compute.zone":availabilityZone])
        }
         ])

}

/*****************
 * Optional. Custom properties used to extend existing drivers or create new ones.
 */
```

```
       //custom ([:])
         // Se si usa il persistent path, e' necessario usare questo comando custom
         custom (["org.cloudifysource.clearRemoteDirectoryOnStart" : true])
}
```

upload

bootstrap-management.sh

```
#! /bin/bash

########################################################################
# This script starts a Gigaspaces agent for use with the Gigaspaces
# Cloudify. The agent will function as management depending on the value of $GSA_MODE
#
# Parameters the should be exported beforehand:
#      $LUS_IP_ADDRESS - Ip of the head node that runs a LUS and ESM. May be my IP. (Required)
#   $GSA_MODE - 'agent' if this node should join an already running node. Otherwise, any value.
#   $NO_WEB_SERVICES - 'true' if web-services (rest, webui) should not be deployed (only if GSA_MODE != 'agent')
#   $NO_MANAGEMENT_SPACE - 'true' if cloudifyManagementSpace should not be deployed (only if GSA_MODE != 'agent')
#   $NO_MANAGEMENT_SPACE_CONTAINER - 'true' if container for cloudifyManagementSpace should not be started(only if GSA_MODE !=
'agent')
#   $MACHINE_IP_ADDRESS - The IP of this server (Useful if multiple NICs exist)
#      $WORKING_HOME_DIRECTORY - This is where the files were copied to (cloudify installation, etc..)
#      $GIGASPACES_LINK - If this url is found, it will be downloaded to $WORKING_HOME_DIRECTORY/gigaspaces.zip
#      $GIGASPACES_OVERRIDES_LINK - If this url is found, it will be downloaded and unzipped into the same location as cloudify
#      $CLOUD_FILE - Location of the cloud configuration file. Only available in bootstrap of management machines.
#      $GIGASPACES_CLOUD_IMAGE_ID - If set, indicates the image ID for this machine.
#      $GIGASPACES_CLOUD_HARDWARE_ID - If set, indicates the hardware ID for this machine.
#      $AUTO_RESTART_AGENT - If set to 'true', will allow to perform reboot of agent machine.
#      $PASSWORD - the machine password.
########################################################################

echo "####################################################################"
echo "---Inizio esecuzione boootstrap_management.sh nella VM $HOSTNAME..."
echo "####################################################################"

certName="<nome_certificato.estensione"
echo "Nome certificato: $certName"

echo "--Aggiunta ad /etc/hosts quanto segue:"
echo 127.0.0.1 $HOSTNAME | sudo tee -a /etc/hosts
```

```
echo "---Setting Link al repository pubblico SWIFT della Piattaforma PaaS"
# Link al repository pubblico della piattaforma PaaS
CLOUDIFY_SWIFT_LINK="https://<endpoint_SWIFT>/v1/AUTH_<projectID>/publicAppRepo/cloudify-2.7.2reply.tar.gz"
JDK_64_SWIFT_LINK="https://<endpoint_SWIFT>/v1/AUTH_<projectID>/publicAppRepo/jdk-6u32-linux-x64.bin"
JDK_32_SWIFT_LINK="https://<endpoint_SWIFT>/v1/AUTH_<projectID>/publicAppRepo/jdk-6u32-linux-i586.bin"



# some distro do not have which installed so we're checking if the file exists
if [ -f /usr/bin/wget ]; then
      DOWNLOADER="wget"
elif [ -f /usr/bin/curl ]; then
      DOWNLOADER="curl"
fi


# args:
# $2 the error code of the last command (should be explicitly passed)
# $3 the message to print in case of an error
#
# an error message is printed and the script exists with the provided error code
function error_exit {
      echo "$3 : error code: $2"
      exit ${2}
}

# args:
# $1 the error code of the last command (should be explicitly passed)
# $2 the message to print in case of an error
# $3 the threshold to exit on
#
# if (last_error_code [$1]) >= (threshold [$4]) (defaults to 0), the script
# exits with the provided error code [$2] and the provided message [$3] is printed
function error_exit_on_level {
      if [ ${1} -ge ${4} ]; then
            error_exit ${2} ${3}
      fi
}

# args:
# $1 the name of the script. must be located in the upload folder.
function run_script {
```

```
    FULL_PATH_TO_SCRIPT="$WORKING_HOME_DIRECTORY/$1.sh"
    if [ -f $FULL_PATH_TO_SCRIPT ]; then
        chmod +x $FULL_PATH_TO_SCRIPT
        echo Running script $FULL_PATH_TO_SCRIPT
        $FULL_PATH_TO_SCRIPT
        RETVAL=$?
        if [ $RETVAL -ne 0 ]; then
          error_exit $RETVAL "Failed running $1 script"
        fi
    fi
}

# args:
# $1 download description.
# $2 download link.
# $3 output file.
# $4 the error code.
function download {
      echo Funzione download: Downloading $1 from $2
      if [ "$DOWNLOADER" = "wget" ];then
            Q_FLAG="-q"
            O_FLAG="-O"
            LINK_FLAG=""
      elif [ "$DOWNLOADER" = "curl" ];then
            Q_FLAG="--silent"
            O_FLAG="-o"
            LINK_FLAG="-O"
      fi
      echo "Viene eseguito: $DOWNLOADER $Q_FLAG $O_FLAG $3 $LINK_FLAG $2"
      $DOWNLOADER $Q_FLAG $O_FLAG $3 $LINK_FLAG $2 || error_exit $? $4 "Failed downloading $1"
}

function download_INSECURE {
       echo Funzione download_INSECURE: Downloading $1 from $2
       if [ "$DOWNLOADER" = "wget" ];then
              Q_FLAG="-q"
              O_FLAG="-O"
              LINK_FLAG=""
              NO_CECK_CERT="--no-check-certificate"
       elif [ "$DOWNLOADER" = "curl" ];then
              Q_FLAG="--silent"
              O_FLAG="-o"
              LINK_FLAG="-O"
```

```
                NO_CECK_CERT="--insecure"
        fi
        echo "Eseguo: $DOWNLOADER $NO_CECK_CERT $Q_FLAG $O_FLAG $3 $LINK_FLAG $2"
        $DOWNLOADER $NO_CECK_CERT $Q_FLAG $O_FLAG $3 $LINK_FLAG $2 || error_exit $? $4 "Failed downloading $1"
}

echo "--Loading Cloudify Environment..."
SCRIPT=`readlink -f $0`
SCRIPTPATH=`dirname $SCRIPT`
echo script path is $SCRIPTPATH

if [ -f ${SCRIPTPATH}/cloudify_env.sh ]; then
       ENV_FILE_PATH=${SCRIPTPATH}/cloudify_env.sh
else
       if [ -f ${SCRIPTPATH}/../cloudify_env.sh ]; then
              ENV_FILE_PATH=${SCRIPTPATH}/../cloudify_env.sh
       else
              error_exit 100 "Cloudify environment file not found! Bootstrapping cannot proceed!"
       fi

fi

source ${ENV_FILE_PATH}

echo "##############################"
echo "# Priviliged Script execution#"
echo "##############################"
echo CLOUDIFY_OPEN_FILES_LIMIT is $CLOUDIFY_OPEN_FILES_LIMIT

if [ -f ${WORKING_HOME_DIRECTORY}/break.bin ]
then
       echo "Exiting due to break file detected"
       exit 0
fi

function privilegedActions {
       echo Executing priviliged bootstrap actions
       if [ ! -z $CLOUDIFY_OPEN_FILES_LIMIT ]
       then
              echo setting hard and soft open files ulimit to $CLOUDIFY_OPEN_FILES_LIMIT
              ulimit -HSn $CLOUDIFY_OPEN_FILES_LIMIT
              echo Finished setting open files limit
```

```
	fi
	if [ -f ${WORKING_HOME_DIRECTORY}/privileged-script.sh ]
	then
		echo executing privileged script
		source ${WORKING_HOME_DIRECTORY}/privileged-script.sh
	fi

	echo finished priviliged actions
}


# first check if we are in an advanced step of priviliged bootstrap
if [ ! -z $PRIVILEGED_BOOTSTRAP_USER ]
then
	echo In second phase of privileged bootstrap
	# phase 2
	privilegedActions
	targetUser="$PRIVILEGED_BOOTSTRAP_USER"
	export PRIVILEGED_BOOTSTRAP_USER=
	echo "export PRIVILEGED_MARKER=on;${WORKING_HOME_DIRECTORY}/bootstrap-management.sh" | sudo -u $targetUser -s
	exit $?
else
	if [ ! -z $PRIVILEGED_MARKER ]
	then
		# finished privileged phase of bootstrap
		export PRIVILEGED_MARKER=
		echo finished privileged phase of bootstrap
	else

		if [ ! -z $CLOUDIFY_OPEN_FILES_LIMIT ] || [ -f "privileged-script.sh" ]
		then
			# phase 1 - begin priviliged bootstrap process
			echo In first phase of privileged bootstrap
			if [ `whoami` = "root" ]
			then
				# just run the privileged actions now
				priviligedActions
			else
				# verify passwordless sudo privileges for current user
				if [ "$GIGASPACES_AGENT_ENV_PRIVILEGED" = "true" ]; then
					sudo -n ls > /dev/null || exit 1
					export PRIVILEGED_BOOTSTRAP_USER=`whoami`
					sudo -E ${WORKING_HOME_DIRECTORY}/bootstrap-management.sh
```

```
                                exit 0
                        else
                                # not a password-less sudoer - bootstrap must fail
                                exit 115
                        fi
                fi
        else
                echo Standard bootstrap process will be used
        fi
fi

echo "# Execute pre-bootstrap customization script (if exists)"
run_script "pre-bootstrap"



echo "--Setting URL JDK per Cloudify..."
JAVA_32_URL="http://repository.cloudifysource.org/com/oracle/java/1.6.0_32/jdk-6u32-linux-i586.bin"
JAVA_64_URL="http://repository.cloudifysource.org/com/oracle/java/1.6.0_32/jdk-6u32-linux-x64.bin"


#LINK a SWIFT
if [ -z "$GIGASPACES_AGENT_ENV_JAVA_URL" ]; then
        ARCH=`uname -m`
        echo Machine Architecture -- $ARCH
        if [ "$ARCH" = "i686" ]; then
                export GIGASPACES_AGENT_ENV_JAVA_URL=$JDK_32_SWIFT_LINK
        elif [ "$ARCH" = "x86_64" ]; then
                export GIGASPACES_AGENT_ENV_JAVA_URL=$JDK_64_SWIFT_LINK
        else
                echo Unknown architecture -- $ARCH -- defaulting to 32 bit JDK
                export GIGASPACES_AGENT_ENV_JAVA_URL=$JDK_32_SWIFT_LINK
        fi
fi


if [ "$GIGASPACES_AGENT_ENV_JAVA_URL" = "NO_INSTALL" ]; then
        echo "JDK will not be installed"
else
        echo Previous JAVA_HOME value -- $JAVA_HOME
        export GIGASPACES_ORIGINAL_JAVA_HOME=$JAVA_HOME
```

```
        #download "JDK" $GIGASPACES_AGENT_ENV_JAVA_URL $WORKING_HOME_DIRECTORY/java.bin 101
        download_INSECURE "JDK" $GIGASPACES_AGENT_ENV_JAVA_URL $WORKING_HOME_DIRECTORY/java.bin 101
        chmod +x $WORKING_HOME_DIRECTORY/java.bin
        echo -e "\n" > $WORKING_HOME_DIRECTORY/input.txt
        rm -rf ~/java || error_exit $? 102 "Failed removing old java installation directory"
        mkdir ~/java
        cd ~/java

        echo Installing JDK
        $WORKING_HOME_DIRECTORY/java.bin < $WORKING_HOME_DIRECTORY/input.txt > /dev/null
        mv ~/java/*/* ~/java || error_exit $? 103 "Failed moving JDK installation"
        rm -f $WORKING_HOME_DIRECTORY/input.txt
    export JAVA_HOME=~/java
    rm -f $WORKING_HOME_DIRECTORY/java.bin || error_exit $? 136 "Failed deleting java.bin from home directory"
fi

export EXT_JAVA_OPTIONS="-Dcom.gs.multicast.enabled=false"


# verifico se si tratta di una VM Management o Application
# esaminando se il nome dell'host contiene la parola mngt
#ed eseguo le relative operazioni
if [[ $HOSTNAME == *"mngt"* ]]; then
echo "--WORKING_HOME_DIRECTORY per la VM di Management= $WORKING_HOME_DIRECTORY"
# WORKING_HOME_DIRECTORY = /root/gs-files/upload per utente root
# WORKING_HOME_DIRECTORY = /home/ubuntu/gs-files/upload per utente ubuntu
export USER_HOME_DIRECTORY="$WORKING_HOME_DIRECTORY/../.."
echo "La USER_HOME_DIRECTORY e'  = $USER_HOME_DIRECTORY"

echo "$HOSTNAME is VM manager: Esiste la cartella $USER_HOME_DIRECTORY/gs-files/upload"
echo "|--> Aggiungo certificato al keystore - per le VM di management...."
sudo $USER_HOME_DIRECTORY/java/bin/keytool -import -trustcacerts -file $USER_HOME_DIRECTORY/gs-files/upload/$certName -keystore
$USER_HOME_DIRECTORY/java/jre/lib/security/cacerts << _EOF_
changeit
yes
_EOF_
echo "Imposto un crontab per la rimozione dei file di log 5 minuti dopo la mezzanotte..."
echo "5 0 * * * $USER_HOME_DIRECTORY/gs-files/upload/removelog.sh" | sudo tee -a /var/spool/cron/crontabs/root
sudo chmod +x $USER_HOME_DIRECTORY/gs-files/upload/removelog.sh

else

echo "--WORKING_HOME_DIRECTORY per la VM Application= $WORKING_HOME_DIRECTORY"
```

```
# WORKING_HOME_DIRECTORY = /root/gs-files per utente root
# WORKING_HOME_DIRECTORY = /home/ubuntu/gs-files per utente ubuntu
export USER_HOME_DIRECTORY="$WORKING_HOME_DIRECTORY/.."
echo "La USER_HOME_DIRECTORY e'  = $USER_HOME_DIRECTORY"

echo "$HOSTNAME is VM Application: Non esiste la cartella di upload in $USER_HOME_DIRECTORY/gs-files/upload"
echo "|--> Aggiungo certificato al keystore - per le VM di Application...."
sudo $USER_HOME_DIRECTORY/java/bin/keytool -import -trustcacerts -file $USER_HOME_DIRECTORY/gs-files/$certName -keystore
$USER_HOME_DIRECTORY/java/jre/lib/security/cacerts << _EOF_
changeit
yes
_EOF_
echo "Imposto un crontab per la rimozione dei file di log 5 minuti dopo la mezzanotte..."
echo "5 0 * * * $USER_HOME_DIRECTORY/gs-files/removelog.sh" | sudo tee -a /var/spool/cron/crontabs/root
sudo chmod +x $USER_HOME_DIRECTORY/gs-files/removelog.sh
fi


# Download ed installazione Cloudify Agent
echo "Procedo al download ed installazione di cloudify nella VM allocata..."
if [ ! -z "$CLOUDIFY_SWIFT_LINK" ]; then

   download_INSECURE "cloudify installation" $CLOUDIFY_SWIFT_LINK $WORKING_HOME_DIRECTORY/gigaspaces.tar.gz 104

else
   download "cloudify installation" $GIGASPACES_LINK.tar.gz $WORKING_HOME_DIRECTORY/gigaspaces.tar.gz 104

fi

#if [ ! -z "$GIGASPACES_LINK" ]; then
#      download "cloudify installation" $GIGASPACES_LINK.tar.gz $WORKING_HOME_DIRECTORY/gigaspaces.tar.gz 104
#fi
#
#if [ ! -z "$GIGASPACES_OVERRIDES_LINK" ]; then
#            download "cloudify overrides" $GIGASPACES_OVERRIDES_LINK.tar.gz $WORKING_HOME_DIRECTORY/gigaspaces_overrides.tar.gz 105
#fi

# Todo: Check this condition
if [ ! -d "~/gigaspaces" -o $WORKING_HOME_DIRECTORY/gigaspaces.tar.gz -nt ~/gigaspaces ]; then
       rm -rf ~/gigaspaces || error_exit $? 106 "Failed removing old gigaspaces directory"
       mkdir ~/gigaspaces || error_exit $? 107 "Failed creating gigaspaces directory"

       # 2 is the error level threshold. 1 means only warnings
```

```
	# this is needed for testing purposes on zip files created on the windows platform
	tar xfz $WORKING_HOME_DIRECTORY/gigaspaces.tar.gz -C ~/gigaspaces || error_exit_on_level $? 108 "Failed extracting cloudify
installation" 2
	rm -f $WORKING_HOME_DIRECTORY/gigaspaces.tar.gz error_exit $? 134 "Failed deleting gigaspaces.tar.gz from home directory"

	# Todo: consider removing this line
	chmod -R 777 ~/gigaspaces || error_exit $? 109 "Failed changing permissions in cloudify installation"
	mv ~/gigaspaces/*/* ~/gigaspaces || error_exit $? 110 "Failed moving cloudify installation"

	if [ ! -z "$GIGASPACES_OVERRIDES_LINK" ]; then
		echo Copying overrides into cloudify distribution
		tar xfz $WORKING_HOME_DIRECTORY/gigaspaces_overrides.tar.gz -C ~/gigaspaces || error_exit_on_level $? 111 "Failed
extracting cloudify overrides" 2
		rm -f $WORKING_HOME_DIRECTORY/gigaspaces_overrides.tar.gz error_exit $? 135 "Failed deleting
gigaspaces_overrides.tar.gz from home directory"
	fi
fi

# if an overrides directory exists, copy it into the cloudify distribution
if [ -d $WORKING_HOME_DIRECTORY/cloudify-overrides ]; then
	cp -rf $WORKING_HOME_DIRECTORY/cloudify-overrides/* ~/gigaspaces
fi

# UPDATE SETENV SCRIPT...
echo Updating environment script
cd ~/gigaspaces/bin || error_exit $? 112 "Failed changing directory to bin directory"

sed -i "2i . ${ENV_FILE_PATH}" setenv.sh || error_exit $? 113 "Failed updating setenv.sh"
sed -i "2i export NIC_ADDR=$MACHINE_IP_ADDRESS" setenv.sh || error_exit $? 113 "Failed updating setenv.sh"
sed -i "2i export LOOKUPLOCATORS=$LUS_IP_ADDRESS" setenv.sh || error_exit $? 113 "Failed updating setenv.sh"
sed -i "2i export PATH=$JAVA_HOME/bin:$PATH" setenv.sh || error_exit $? 113 "Failed updating setenv.sh"
sed -i "2i export JAVA_HOME=$JAVA_HOME" setenv.sh || error_exit $? 113 "Failed updating setenv.sh"


# Privileged mode handling
if [ "$GIGASPACES_AGENT_ENV_PRIVILEGED" = "true" ]; then
	# First check if sudo is allowed for current session
	export GIGASPACES_USER=`whoami`
	if [ "$GIGASPACES_USER" = "root" ]; then
		# root is privileged by definition
		echo Running as root
	else
		sudo -n ls > /dev/null || error_exit_on_level $? 115 "Current user is not a sudoer, or requires a password for sudo" 1
```

```
	fi

	# now modify sudoers configuration to allow execution without tty
	grep -i ubuntu /proc/version > /dev/null
	if [ "$?" -eq "0" ]; then
			# ubuntu
			echo Running on Ubuntu
			if sudo grep -q -E '[^!]requiretty' /etc/sudoers; then
				echo creating sudoers user file
				echo "Defaults:`whoami` !requiretty" | sudo tee /etc/sudoers.d/`whoami` >/dev/null
				sudo chmod 0440 /etc/sudoers.d/`whoami`
			else
				echo No requiretty directive found, nothing to do
			fi
	else
			# other - modify sudoers file
			if [ ! -f "/etc/sudoers" ]; then
					error_exit 116 "Could not find sudoers file at expected location (/etc/sudoers)"
			fi
			echo Setting privileged mode
			sudo sed -i 's/^Defaults.*requiretty/#&/g' /etc/sudoers || error_exit_on_level $? 117 "Failed to edit sudoers
file to disable requiretty directive" 1
	fi

fi

# Execute per-template command
if [ ! -z "$GIGASPACES_AGENT_ENV_INIT_COMMAND" ]; then
	echo Executing initialization command
	cd $WORKING_HOME_DIRECTORY
	eval "$GIGASPACES_AGENT_ENV_INIT_COMMAND"
fi

cd ~/gigaspaces/tools/cli || error_exit $? 118 "Failed changing directory to cli directory"

# Removing old nohup.out
if [ -f nohup.out ]; then
	echo Removing old nohup.out
	rm nohup.out
fi

if [ -f nohup.out ]; then
   error_exit 114 "Failed to remove nohup.out, it might be used by another process"
```

```
fi

# START AGENT ALONE OR WITH MANAGEMENT
START_COMMAND_ARGS="-timeout 30 --verbose"
if [ "$GSA_MODE" = "agent" ]; then
       ERRMSG="Failed starting agent"
       START_COMMAND="start-agent"
else
       ERRMSG="Failed starting management services"
       START_COMMAND="start-management"
       START_COMMAND_ARGS="${START_COMMAND_ARGS} -cloud-file ${CLOUD_FILE}"
       if [ "$NO_WEB_SERVICES" = "true" ]; then
              START_COMMAND_ARGS="${START_COMMAND_ARGS} -no-web-services"
       fi
       if [ "$NO_MANAGEMENT_SPACE" = "true" ]; then
              START_COMMAND_ARGS="${START_COMMAND_ARGS} -no-management-space"
       fi
       if [ "$NO_MANAGEMENT_SPACE_CONTAINER" = "true" ]; then
              START_COMMAND_ARGS="${START_COMMAND_ARGS} -no-management-space-container"
       fi
fi

echo "Execute post-bootstrap customization script (if exists)"
run_script "post-bootstrap"

if [ "$AUTO_RESTART_AGENT" = "true" ]; then
       # Add agent restart command to scheduled tasks.
       cat <(crontab -l) <(echo "@reboot export EXT_JAVA_OPTIONS=$EXT_JAVA_OPTIONS; nohup ~/gigaspaces/tools/cli/cloudify.sh
$START_COMMAND $START_COMMAND_ARGS") | crontab -
fi

./cloudify.sh $START_COMMAND $START_COMMAND_ARGS

RETVAL=$?

if [ $RETVAL -ne 0 ]; then
       echo start command failed, exit code is: $RETVAL
       # exit codes that are larger than 200 are not specified by Cloudify. We use the 255 code to indicate a custom error.
       if [ $RETVAL -gt 200 ]; then
              RETVAL=255
       fi
       error_exit $? $RETVAL "$ERRMSG"
fi
```

```
echo "---Fine esecuzione boootstrap_management.sh nella VM $HOSTNAME..."
echo "################################################################"

exit 0
```

post-bootstrap.sh

```
#! /bin/bash

echo "############################################"
echo "------Esecuzione di  post-bootstrap script..."

echo "Setting Time Zone"
timeZone="Europe/Rome"

echo "Setting DNS zone"
DNS_ZONE="dominio.it"

echo "Setting Server Zabbix Infrastruttura"
zabbix_metrics="zabbix-metrics.$DNS_ZONE"
zabbix_watcher="zabbix-watcher.$DNS_ZONE"
zabbix_iaas="zabbix-iaas.$DNS_ZONE"

echo "Setting # Puppet"
puppetMaster=" puppet-master.$DNS_ZONE"


echo "Imposto la sincronizzazione dell'ora quotidiana. Time Zone: $timeZone"
sudo touch /etc/cron.daily/ntpdate
echo ntpdate ntp1.inrim.it  | sudo tee -a /etc/cron.daily/ntpdate
sudo chmod 755 /etc/cron.daily/ntpdate
echo $timeZone | sudo tee /etc/timezone
sudo dpkg-reconfigure --frontend noninteractive tzdata

echo "--Configurazione Zabbix Agent INFN..."
wget http://repo.zabbix.com/zabbix/2.2/ubuntu/pool/main/z/zabbix-release/zabbix-release_2.2-1+precise_all.deb
sudo dpkg -i zabbix-release_2.2-1+precise_all.deb
sudo apt-get -q update
sudo apt-get install zabbix-agent

# verifico se si tratta di una VM Management o Application
# esaminando se il nome dell'host contiene la parola manager
if [[ $HOSTNAME == *"mngt"* ]]; then
       echo "$HOSTNAME is VM manager: Zabbix Agent autoconfiguring with zabbix iaas..."
```

```
            sudo sed -i -e "s/^Server=.*/Server=${zabbix_iaas}/" /etc/zabbix/zabbix_agentd.conf
        sudo sed -i -e "s/^ServerActive=.*/ServerActive=${zabbix_iaas}/" /etc/zabbix/zabbix_agentd.conf

else
      echo "$HOSTNAME is VM Application: Zabbix Agent autoconfiguring with zabbix metrics and watcher..."
        sudo sed -i -e "s/^Server=.*/Server=${zabbix_metrics},${zabbix_watcher}/" /etc/zabbix/zabbix_agentd.conf
        sudo sed -i -e "s/^ServerActive=.*/ServerActive=${zabbix_metrics},${zabbix_watcher}/" /etc/zabbix/zabbix_agentd.conf

fi

sudo sed -i -e "s/^Hostname=.*/Hostname=${HOSTNAME//./_}/" /etc/zabbix/zabbix_agentd.conf
sudo service zabbix-agent restart

# echo "--Configurazione Puppet Agent ..."
# sudo rm -f /var/lib/puppet/ssl/certs/$HOSTNAME.$DNS_ZONE.pem
# sudo puppet agent -t
# echo "-Puppet: registro l'agent sul Puppet Master"
# sudo puppet agent --server $puppetMaster --onetime --no-daemonize --verbose

echo "------Fine post-bootstrap script"
echo "################################"

exit 0
```

removelog.sh

```
#!/bin/bash

echo "--------Removing Cloudify logs file..."
#rm -r /root/gigaspaces/logs/
today_date="$(date +"%Y-%m-%d")"
echo "Today date is: $today_date"
echo "Remove file before date:  $today_date"
grep -rLZ 2015-04-06 /root/gigaspaces/logs/ | while IFS= read -rd '' x; do rm "$x"; done


exit 0
```