P.O.N. RICERCA E COMPETITIVITA' 2007-2013 – Avviso n.° 84/Ric. del 2 marzo 2012 -
Asse II: “Sostegno all’Innovazione”
Obiettivo Operativo: “Azioni Integrate per lo sviluppo sostenibile e per lo sviluppo della società dell’Informazione”
“Smart Cities and Communities and Social Innovation”
Codice Progetto: **PON04a2_A - “PRISMA”**

PRISMA Platform Status Report

Workshop CCR

Catania - 29 Maggio 2014

Marica Antonacci - Giacinto Donvito
INFN-BARI
Progetto PRISMA

collaborazione PRISMA: INFN-BA, INFN-CT, INFN-NA

Outline

- Havana in production
  - configurazione e deployment dei servizi
  - funzionalità abilitate e componenti installati
- Implementazione dell'HA dei servizi core
- Sperimentazione con Heat
  - possibilità e problemi
- Architettura di PaaS+IaaS di PRISMA
- Architettura di Monitoring di PRISMA
- Attività future
- Conclusioni

PRISMA-INFN-BARI major upgrade

Nel mese di marzo abbiamo “stravolto” il nostro testbed di produzione per migliorare la qualità e l’affidabilità dei servizi erogati.

- **Nuovo hardware**: server con 256GB RAM, 32 CPU Intel(R) Xeon(R) CPU E5-2650 0 @ 2.00GHz, 7 dischi da 4TB, 2 NIC (1Gbit/s e 10Gbit/s)

- **Filesystem** GlusterFS 3.3 in replica 2 —> 3.4 in replica 3
  - Usato come backend di Nova, Cinder (accanto al backend Ceph), Glance

- Migrazione da Grizzly ad **Havana** (su Ubuntu 12.04LTS kernel 3.8)

- Implementazione dell’**High-Availability** dei servizi “core”

- Installazione del **cluster Ceph**: 3 OSD, 3 MON e 2 MDS
  - Integrazione con Openstack: al momento in produzione solo come backend di Cinder. Abbiamo però testato anche l'integrazione con Glance e Nova (sia con Ceph-fs sia con driver RBD).

La migrazione dei servizi e delle VM

- Main goal: migrazione delle VM e dei dati persistenti (block device) con il minimo downtime

- Gli step principali della migrazione:

  - ***migrazione del database keystone***: il dump del database di produzione è stato riversato su un'istanza di keystone grizzly installata su una VM; quindi, è stato fatto l'update di questo keystone ad havana. Il comando *keystone db_sync* consente di aggiornarne lo schema. Il dump del nuovo database (in versione havana) così ottenuto è stato importato nel nuovo testbed.

  - ***migrazione delle VM***: sul testbed di produzione sono stati eseguiti gli snapshot delle VM che sono poi stati caricati come immagini nel server Glance del nuovo testbed. A questo punto, è stato possibile re-istanziare tutte le VM. Fixed-IP sono stati usati nel caso delle VM per cui era necessario mantenere invariato l'IP.

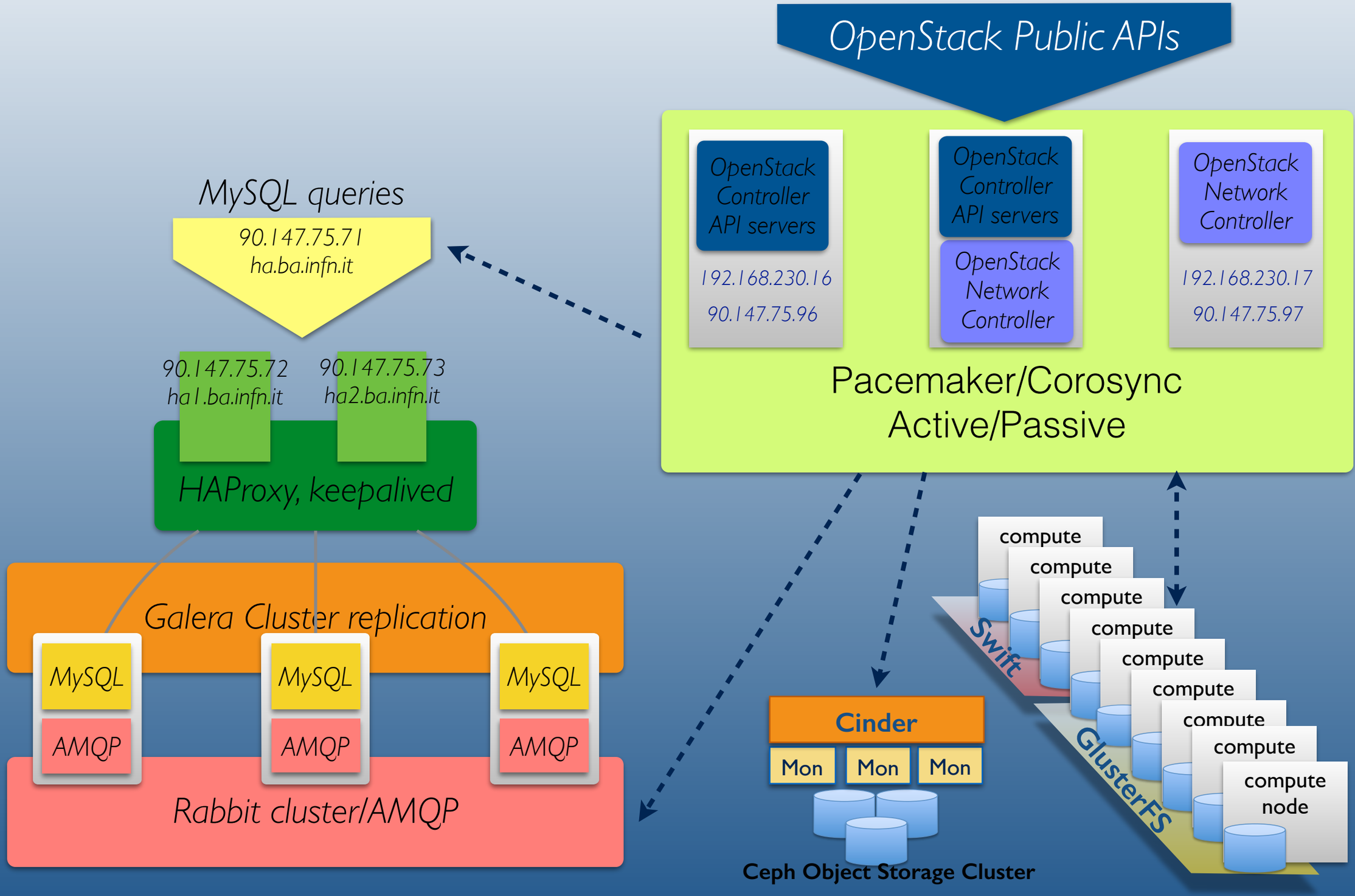
  - ***migrazione dei volumi***: i file utilizzati come loopback device sono stati copiati nel nuovo filesystem distribuito (GlusterFS) e agganciati ai nuovi host.

Strategia di HA nel testbed INFN-BARI

- Eliminare i SPOF del deployment Openstack
- Mysql Galera cluster + HA Proxy load balancing
- Messaging server: Rabbitmq cluster. Configurazione “active-active mirrored queues”
- Servizi di Openstack: configurazione active/passive tramite l’uso di Pacemaker/ Corosync
- Swift: proxy server ridondato. L’endpoint del servizio è rappresentato dall’alias DNS RR
- Shared Filesystem: glusterFS in replica 3 ospita le immagini delle VM running

- *Logical* deployment: 2 controller node & 2 network node (configurazione active/passive)
- *Physical* deployment: 3 host per i servizi di Openstack, 3 host per il cluster Mysql e per il cluster RabbitMQ, 2 VM per i load-balancer del cluster Mysql

Testbed set-up: panoramica



Funzionalità supportate

- **Compute Services (Nova)**:
  - ☑ **Live migration** basata su shared storage (GlusterFS)
    Open issue: conflitto nell'uso delle porte TCP nel range 49152-65535 usate da libvirt per la live migration e la base-port usata dal server GlusterFS 3.4 (bug di libvirt)
    Workaround (temporanea): abilitata la **tunnelled** live-migration.
    INEFFICIENTE
  - ☑ **Live VM snapshot** (new)
  - ☑ **VM resizing** (new)
  - ☑ **Quote per utente** (new): comandi `nova-quota-(show|update|delete)`
  - ☑ **Policy di accesso** che limitano le operazioni dell'utente sulle risorse del tenant: modificato il file policy.json di nova

NEW

```
"context_is_admin":  "role:admin",
"admin_or_owner":  "is_admin:True or project_id:%(project_id)s",
"admin_or_user": "is_admin:True or user_id:%(user_id)s",

"default": "rule:admin_or_user",

"cells_scheduler_filter:TargetCellFilter": "is_admin:True",

"compute:create": "",
"compute:create:attach_network": "",
"compute:create:attach_volume": "",
"compute:create:forced_host": "is_admin:True",
"compute:get": "rule:admin_or_owner",          1
"compute:get_all": "",
"compute:get_all_tenants": "",
"compute:unlock_override": "rule:admin_api",
```

OLD

```
{
    "context_is_admin":  "role:admin",
    "admin_or_owner":  "is_admin:True or project_id:%(project_id)s",
    "default": "rule:admin_or_owner",

    "cells_scheduler_filter:TargetCellFilter": "is_admin:True",

    "compute:create": "",
    "compute:create:attach_network": "",
    "compute:create:attach_volume": "",
    "compute:create:forced_host": "is_admin:True",

    "compute:get_all": "",
    "compute:get_all_tenants": "",
    "compute:unlock_override": "rule:admin_api",
```

[1] policy necessaria per mantenere la compatibilità con occi ed evitare errore generico nella dashboard

Funzionalità supportate (cont.)

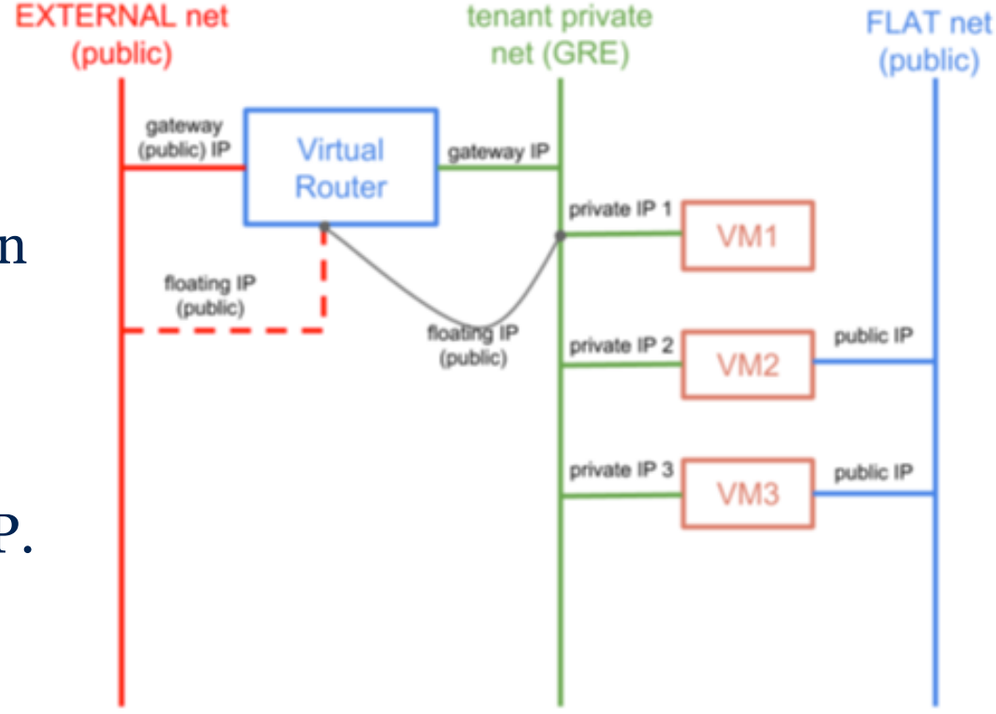
Nova (cont.)

- **interfacce** supportate: **Ec2** e **OCCI** (su https)
- **overbooking** abilitato:

```
compute_scheduler_driver=nova.scheduler.filter_scheduler.FilterScheduler
scheduler_default_filters= RetryFilter, AvailabilityZoneFilter,
AggregateCoreFilter,  RamFilter, ComputeFilter, ComputeCapabilitiesFilter,
ImagePropertiesFilter
cpu_allocation_ratio = 4.0
ram_allocation_ratio = 1.2
```

Networking (Neutron):

- Configurazione mista: rete **GRE** (plugin OVS) con incapsulamento e tunneling affiancata a rete **FLAT** con IP pubblici.
  STP enabled:
  `ovs-vsctl set bridge <bridge name> stp_enable=true`
- Load balancer (**LBaaS**). Testato anche con floating IP.
- VPN as a Service (**VPNaaS**). Testato anche
  - tra 2 regioni (geograficamente distribuite)
  - tra 2 istanze indipendenti di Openstack



Funzionalità supportate (cont.)

- **Cinder (Block Storage)**:
  - ☑ Due backend configurati:
    - ✓**LVM ISCSI** con loopback device su **GlusterFS**
    - ✓**Ceph** (preferred)
  - ☑ **Backup** su Object-Storage (Swift)
  - ☑ **Rate-limiting**. Testata la possibilità di definire dei parametri di QoS - IOPS o bandwidth – che vengono passati all'hypervisor quando il volume viene agganciato alla VM.
  - ☑ **Block device encryption**. Testata la configurazione di default con key manager che supporta una singola chiave statica condivisa tra tutti i volumi.
    - Blueprint per rendere l'interfaccia del key manager interoperabile con Barbican - proposed for Juno release.
  - ☑ **Trasferimento di volumi** tra tenant
  - ☑ **Migrazione di volumi** tra backend diversi (only admin)

Funzionalità supportate (cont.)

- **Swift (Object Storage)**:
  - ☑ configurato per utilizzare l'autenticazione di Keystone
  - ☑ 3 storage node, un'unica zona con replica 3
  - ☑ 2 proxy server configurati nel DNS in RR con supporto SSL e interfaccia **CDMI** abilitata
  - ☑ 1 proxy server su http per supportare l'interfaccia Amazon **S3**.
    - Testato tramite client CLI di Amazon **s3curl** (http://s3.amazonaws.com/doc/s3-example-code/s3-curl.zip)
  - ☑ abilitate le **quote per account** (max 1TB):
    configurato il middleware *account_quotas* che consente l'utilizzo del *metadata x-account-meta-quota-bytes* che può essere modificato solo da un utente con ruolo ResellerAdmin con il comando:

```
# swift --os-tenant-id=<TENANT_ID> post -m quota-bytes:<QUOTA_IN_BYTES>
```

Funzionalità supportate (cont.)

- **Identity Service:**
  - ☑ Keystone con supporto SSL
  - ☑ Autenticazione X.509 tramite modulo **keystone-VOMS**
- Dashboard (Horizon) con supporto SSL abilitato
- **Image Service:**
  - ☑ Glance configurato con default backend store = ‘file’ su GlusterFS. Backend alternativi testati: swift, ceph. Possibilità di scelta usando l’opzione `--store`
  - ☑ vmcatcher/vmcaster per la gestione di liste di immagini virtuali “endorsed” (utilizzato prevalentemente all’interno della EGI FC)
  - ☑ Il nostro sito mette a disposizione le immagini relative alle principali distribuzioni: Ubuntu, Scientific Linux, CentOS, Debian, etc.
  - ☑ Tutte le immagini hanno Cloud-init pre-installato per la contestualizzazione delle VM che avviene nel nostro caso tramite configuration drive
- **Ceilometer**: configurato con mongodb
  - Test accurato di tutte le metriche fornite e della funzionalità di alarming
  - Utilizzo in Heat (p.e. per l'autoscaling)

IaaS automation: Heat

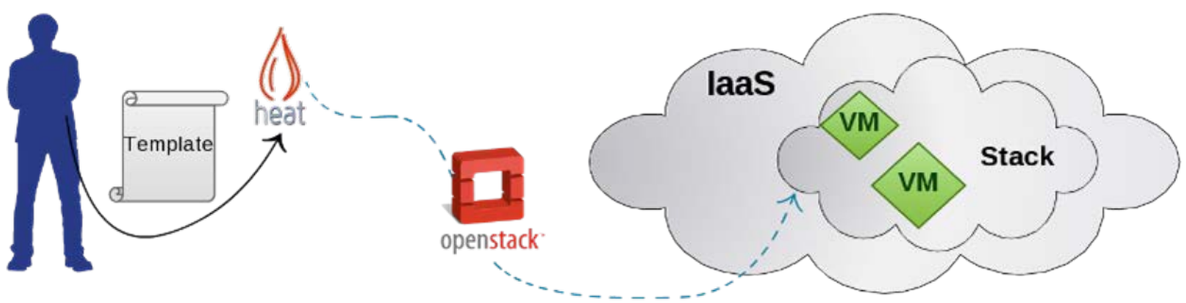
- **Template-driven orchestration engine**: automatizza il deploy di risorse Openstack in modo dichiarativo

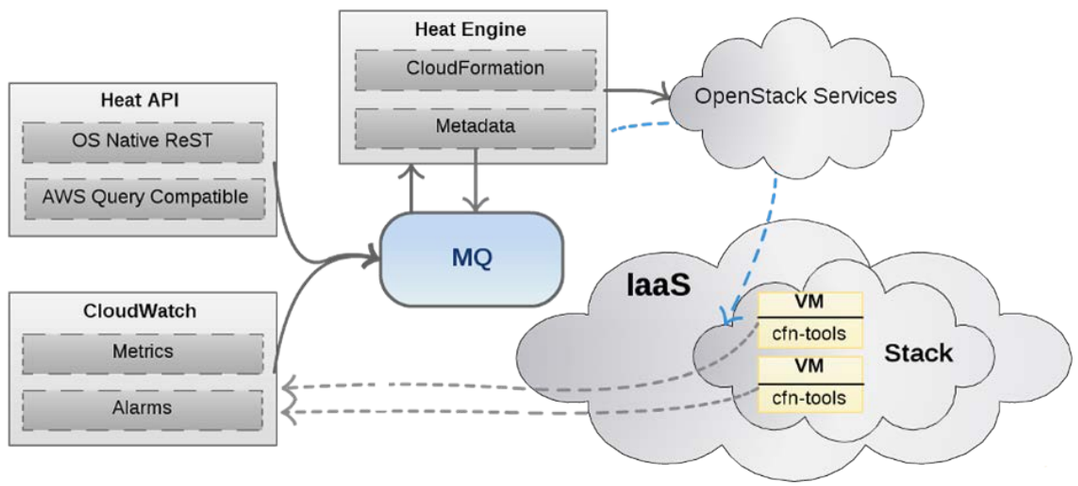
- Heat è integrato con tutti i progetti Openstack e implementa diverse API (HOT, AWS Cloudformation, OpenStack-native ReST API, CloudFormation-compatible Query API)

- Ogni template (file JSON o YAML) descrive un set di risorse (stack) - VM, volumi, reti, sotto-reti, router, porte, floating ip, security groups, etc. - che Heat andrà a creare

  - Esempi di template per diversi casi d'uso sono disponibili su github

- Implementazione di funzionalità avanzate attraverso la creazione di allarmi (Ceilometer) che possono innescare azioni, p.e. l'Autoscaling





Heat: vantaggi

- Deploy di risorse cloud completamente automatico e ripetibile
- Più semplice fornire infrastrutture “on demand”
- Template sostituisce la scrittura di scripting
- Sfruttare la “potenza” di Openstack senza conoscere nel dettaglio servizi, API e tools che utilizza
- Aumentare la modularità definendo template (anche innestati)
- Flessibilità:
  - attraverso l’uso di parametri
  - attraverso l’uso di file di Environment: utile per modificare a runtime il comportamento di un template (valori attuali dei parametri, credenziali extra, risorse non di default)

Il livello PaaS può interfacciarsi in maniera trasparente con il livello IaaS attraverso l’uso di Heat che semplifica e automatizza il deployment di risorse IaaS

Heat: attività svolte e in corso

- Heat non è ancora in produzione nel nostro testbed Havana perché abbiamo riscontrato una serie di problemi legati alla configurazione https di keystone
- In ambiente di test (Havana), abbiamo verificato il flusso di esecuzione di alcuni stack (use case di base) dopo aver applicato patch al codice.
- Al più presto passeremo ad Icehouse per continuare i test

Use case di base

- Interazione con i servizi di Openstack:
  - ✓Nova + Cinder + Neutron
  - ✓Neutron (floatingIP)
  - ✓Ceilometer (cpu_util)
  - ✓Swift (?)

Use case avanzati (funzionali al livello PaaS)

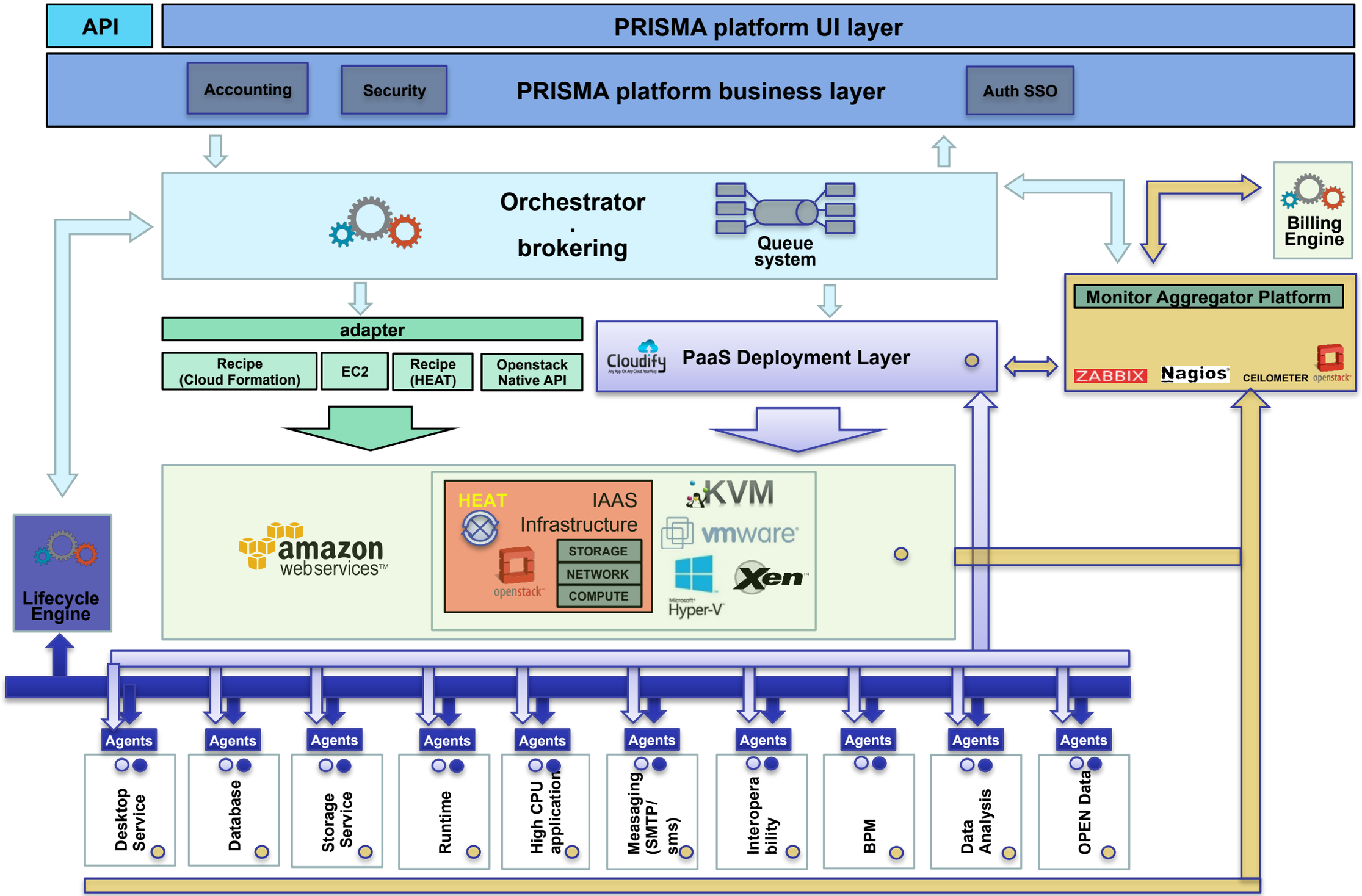
✓ Installazione SW (MySQL e Apache)

✓ Load balancing (OS::Neutron::LoadBalancer)

✓ Autoscaling (OS::Heat::AutoScalingGroup)

✓ VPNaaS (OS::Neutron::VPNService, OS::Neutron::IKEPolicy, OS::Neutron::IPsecPolicy, etc.)

Architettura IaaS + PaaS



IaaS+PaaS: soluzioni

- **PaaS deployment layer**: realizza l'orchestrazione del deploy delle applicazioni sulle macchine virtuali delle piattaforme IaaS e la gestione del loro ciclo di vita. Sarà basato su **Cloudify**

- **Lifecycle Management**: gestisce il ciclo di vita delle configurazioni e delle versioni delle applicazioni installate sulle macchine virtuali. Sarà basato su **Puppet**

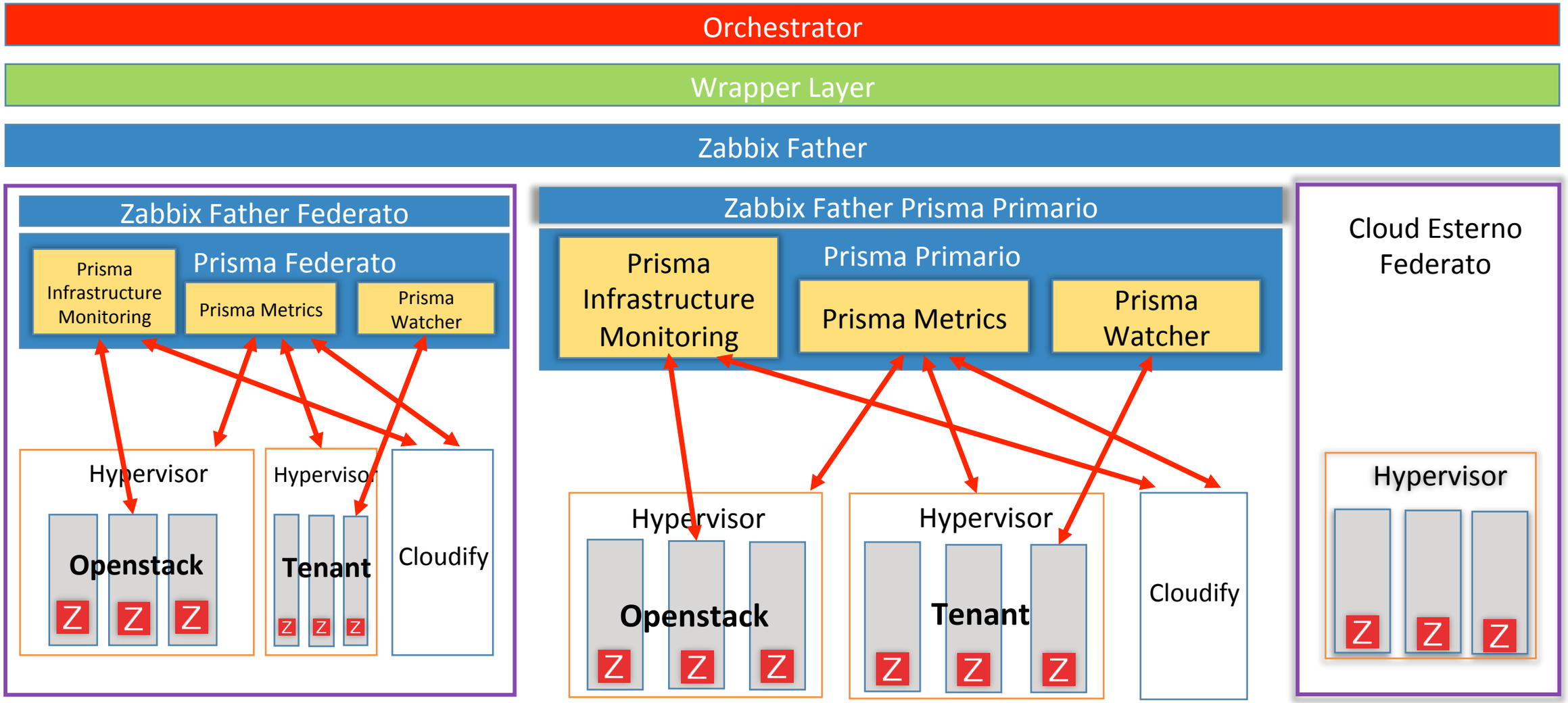
- **Sistema di orchestrazione di livello IaaS**: semplifica l'utilizzo dello IaaS da parte dello strato superiore PaaS. Per Openstack verrà sfruttato il componente Heat, per altri tipi di piattaforme verranno integrati i relativi eventuali adapter di orchestrazione.

- **Monitor aggregator platform**: aggrega i dati di monitoring provenienti dai diversi servizi consentendo di conoscere in ogni momento lo stato di salute di ogni componente hardware o software. Sarà basato su **Zabbix**

- **Sistema ternario di monitoring**: realizzato tramite 3 livelli indipendenti di Zabbix

  - monitoraggio delle istanze IaaS e PaaS ai fini del metering;
  - monitoraggio della parte infrastrutturale della piattaforma (server sui quali PRISMA viene eseguito)
  - costruzione di un servizio attivabile a richiesta dall'utente denominato PRISMA Watcher che si propone di misurare le performance delle singole istanze istantaneamente.

Architettura di monitoring



Il **wrapper**, gestito ad un livello più alto dall'Orchestrator, interroga il Server Zabbix Father.

Il **Server Zabbix Father** raccoglie le informazioni dai Server Child.

Lo strato Wrapper ha anche il compito di implementare la logica di billing della piattaforma: esso sarà in grado di recuperare i dati di metering dal Server Zabbix Father ed associare a questi il corrispondente costo da imputare all'utente, tale modulo verrà sviluppato separatamente dall'architettura di monitoring e verrà quindi richiamato tramite API REST

Use-case supportati a Bari

Il testbed PRISMA-INFN-BARI supporta use-case differenti ed è utilizzato nell'ambito della EGI Federated Cloud Task Force come sito "certificato" dall'inizio dell'anno e dichiarato come PRODUCTION READY nell'EGI FC 2014 (19-23 May) mettendo a disposizione al momento le seguenti risorse:

- 300 Virtual CPU, 600GB RAM e circa 50TB di Storage

❑ Use-case supportati:

✓IaaS per PRISMA

✓servizi PaaS di PRISMA

✓EGI Cloud Task Force

✓attività scientifiche delle comunità non-HEP supportate a Bari

✓"desktop as a service" per i dottorandi e ricercatori di UNIBA

✓alcuni servizi critici della comunità Bio

✓attività di test e sviluppo condotte a Bari

Prossimi step

- **IaaS**: Upgrade ad ***Icehouse***
  - è partita una attività che mira a supportare l'installazione del testbed in modo automatico con **Puppet**. Tenendo conto della complessità intrinseca del sistema, è necessario supportare vari servizi e configurazioni avanzate su ogni host

- **PaaS**: work-in-progess sullo sviluppo di portale e orchestratore

Contributo INFN:

- Implementazione dell'IdP "di PRISMA"
- Contributo al disegno dell'architettura complessiva
- Implementazione del SSO sul portale (SAML2)
- Configurazione dei servizi di backend: MySQL over SSL, etc.
- Fornitura di Storage as a Service (owncloud, etc)
- "Applicazioni High CPU demanding" as a Service
- Supporto tramite Heat alle richieste di tipo "IaaS"

Prossimi step (cont.)

- Attività di testing della soluzione di scheduling avanzato di OpenStack realizzata a Padova:
  - per il momento in fase di installazione in un testbed di test separato
- **Documentazione** di configurazioni e procedure. Al momento sul wiki di progetto; work-in-progress per renderla disponibile nel wiki INFN

Conclusioni

- Dopo la migrazione ad havana e la riconfigurazione completa del cluster, l'affidabilità generale è molto migliorata
- In particolare la replica tre di GlusterFS è stata cruciale
- Manteniamo l'approccio di "un solo testbed per tutti gli use-case". Con lo stesso testbed forniamo le risorse per use-case molto diversi.
- L'utenza sta aumentando in modo costante
- Previsto un upgrade hw a brevissimo per aumentare le risorse utilizzabili (circa altri 200 Cores)
- Stiamo testando Icehouse in un ambiente di test controllato