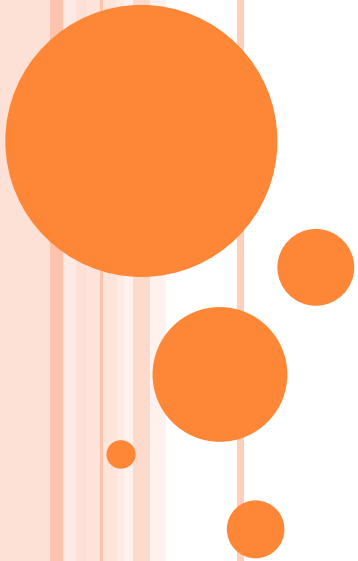


KÖLTSÉGHATÉKONYSÁG CLOUD ALAPÚ RENDSZEREKBEN - ERŐFORRÁSALLOKÁCIÓ PRIVÁT IAAS CLOUDOK ESETÉN

Hartung István

BME Irányítástechnika és Informatika Tanszék

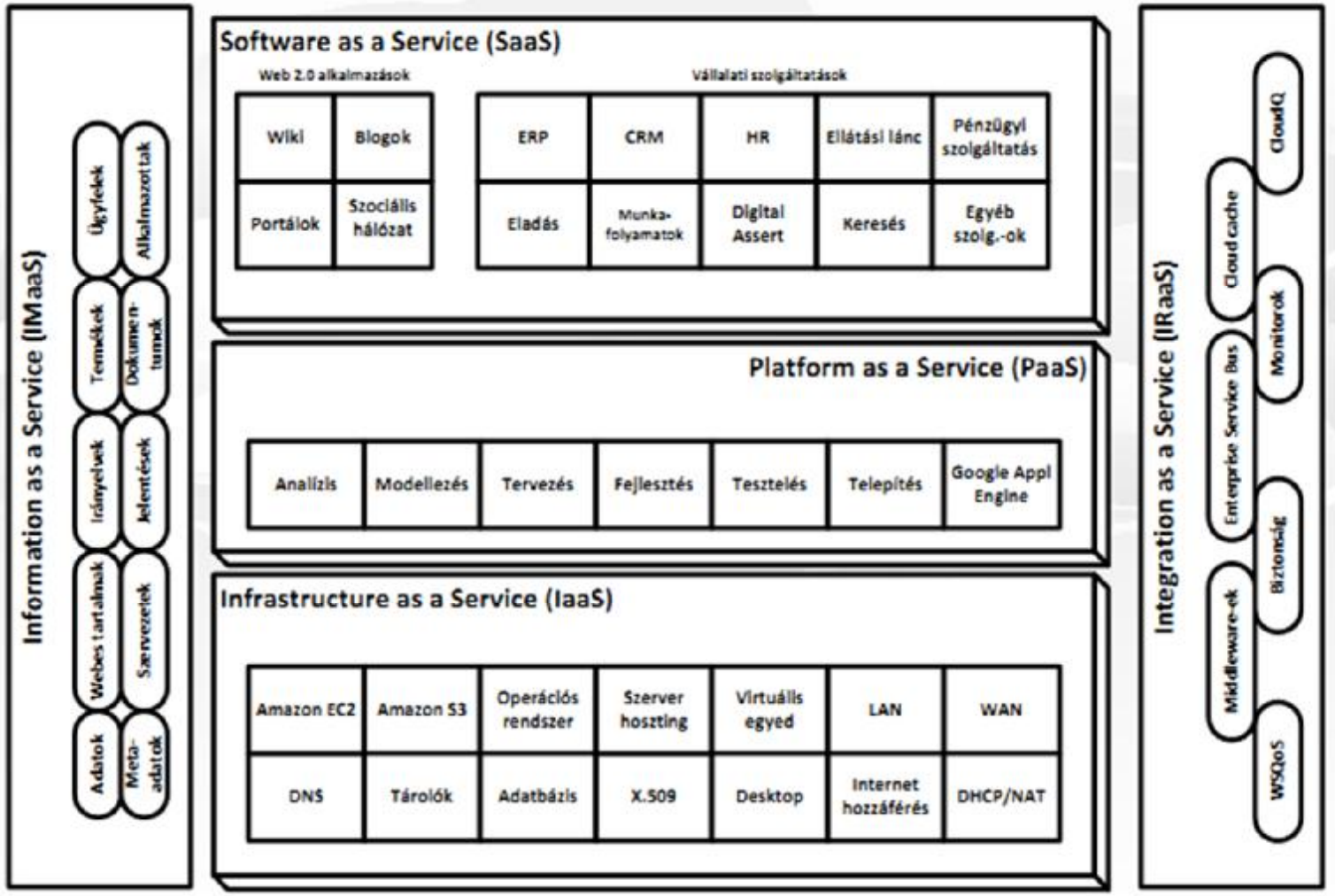
Tematika

- Cloud definíció, típusok, megvalósítási modellek
- Nyílt forráskódú IaaS cloudok architektúrája és erőforrásallokációs algoritmusaik
- Eucalyptus
- OpenStack
- CloudStack
- Jövőbeli munka

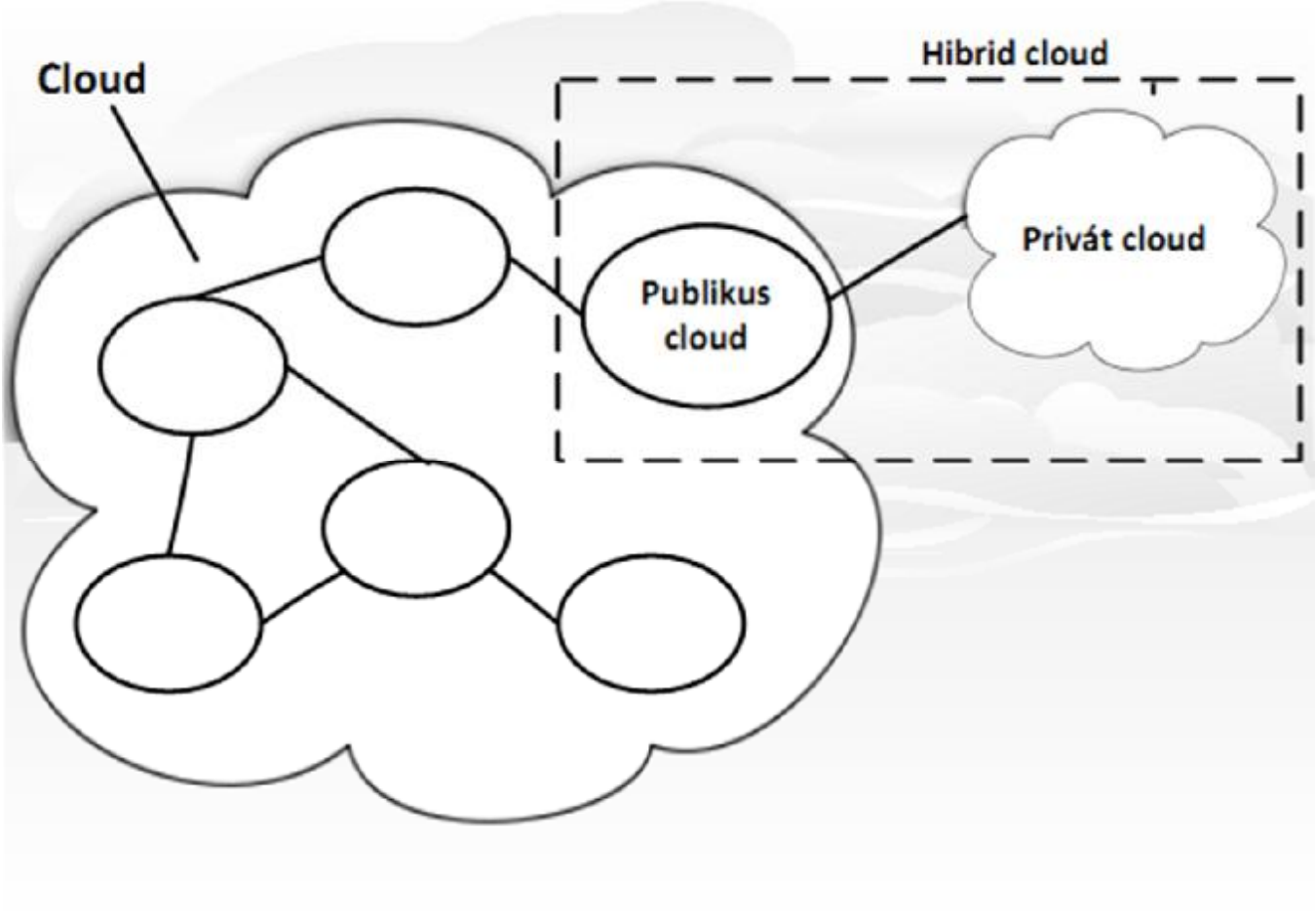
Cloud technológia

- Infrastruktúrát, platformot vagy szolgáltatást nyújt
- Feliratkozás alapú
- Pay-per-use
- Rugalmas, végtelen erőforrások illúziója
- Önkiszolgáló interfészek
- Virtualizált, absztrakt erőforrások, valódi fizikai hardver a felhasználótól rejtve marad

CLOUD TÍPUSOK



Megvalósítási modellek



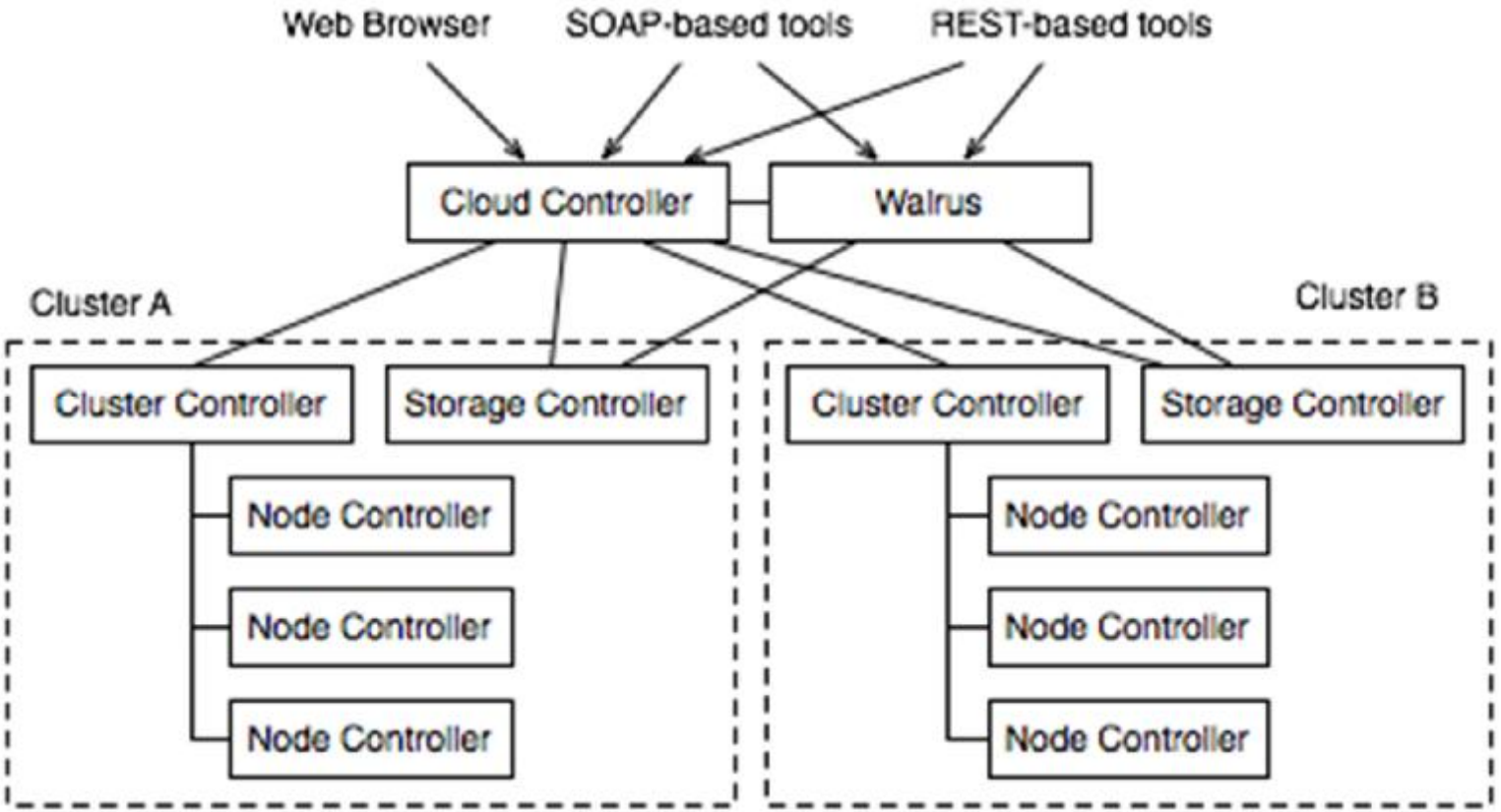
IaaS cloud szolgáltatások

- Infrastruktúrát nyújt: virtuális gép, lemez, adatbázis
- Amazon Web Services: piacvezető publikus IaaS szolgáltató
  - Amazon Elastic Compute Cloud (EC2)
  - Amazon Simple Storage Serice (S3)
  - Amazon Relational Database Service (RDS)
- Privát szolgáltatások:
  - Amazon Virtual Private Cloud (VPC)
  - Eucalyptus
  - OpenStack
  - CloudStack

EUCALYPTUS

- Privát cloud szolgáltatás (képességei szerint hibriddé is válhat)
- Nyílt forráskódú, C-ben íródott
- API kompatibilis Amazon EC2-vel és S3-mal
- Moduláris felépítés, WSDL interfészek
- Minimálisan két gép szükséges

EUCALYPTUS – ARCHITEKTÚRA



EUCALYPTUS – ERŐFORRÁSOK

- Virtuális gépek igényelhetőek
- Felső szintű döntések: *Cloud Controller*
- Valódi erőforrásallokáció: *Cluster Controller*
- Vizsgált paraméterek:
  - CPU magok száma
  - Diszk kapacitás
  - Memóriaméret
- Három allokációs módszer:
  - Round-robin
  - Greedy
  - Powersave

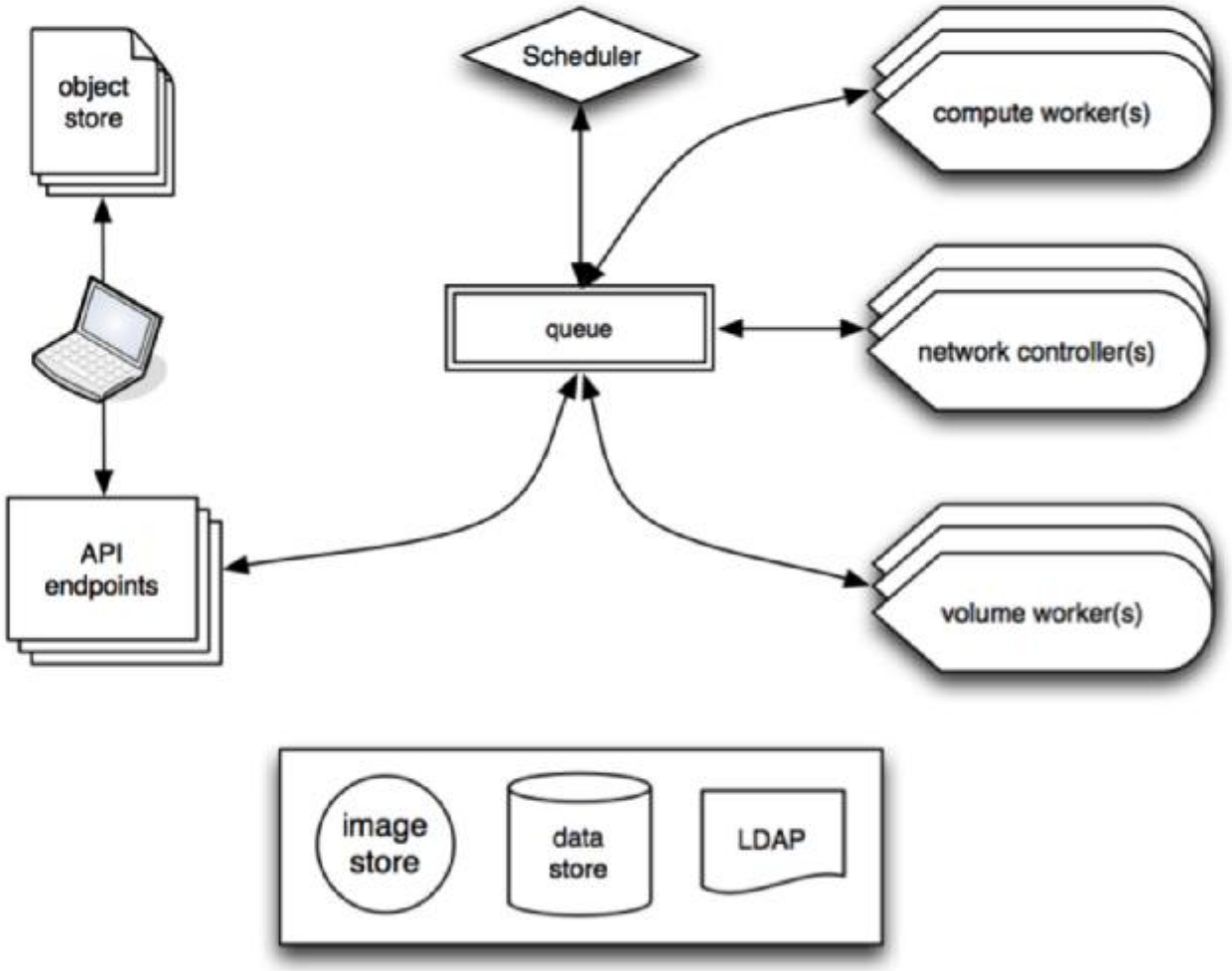
EUCALYPTUS – ERŐFORRÁSALLOKÁCIÓ

- *Round Robin*
  - Mindig a legrégebben vizsgált nem alvó hosztot vizsgálja meg, hogy elegendő-e az erőforrás, ha igen, akkor le is foglalja.
  - Beregisztrálja a lefoglalt erőforrásokat
- *Mohó (Greedy, First Fit)*
  - Az első lehetséges éber hosztot adja vissza, ahol van elegendő erőforrás
- *Energiatakarékos (Powersave)*
  - Ugyanaz, mint a mohó
  - Ha nincs éber, akkor felébreszt egy hosztot
  - Alvó állapotba tud helyezni végpontot

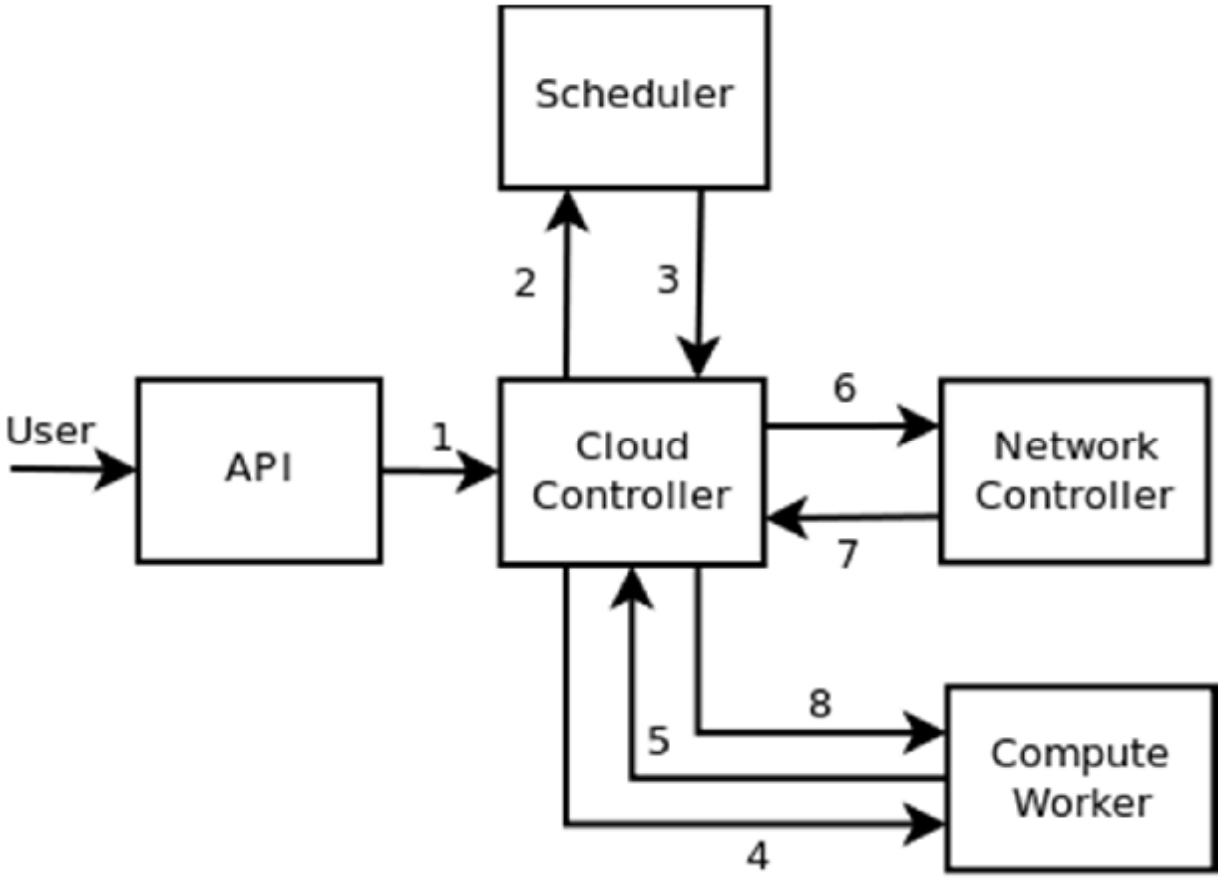
OpenStack

- Privát cloud szolgáltatás
- Nyílt forráskódú, Pythonban íródott
- Webes interfészen keresztül érhető el
- Nem tartalmaz virtualizációs technológiát, csak drivereket
- Három fő komponens:
  - Nova: IaaS megvalósítás
  - Glance: objektum tároló szolgáltatás
  - Swift: virtuális gép tároló szolgáltatás

OpenStack – Architektúra



OpenStack – Erőforrás igénylés



OpenStack – Erőforrásallokáció

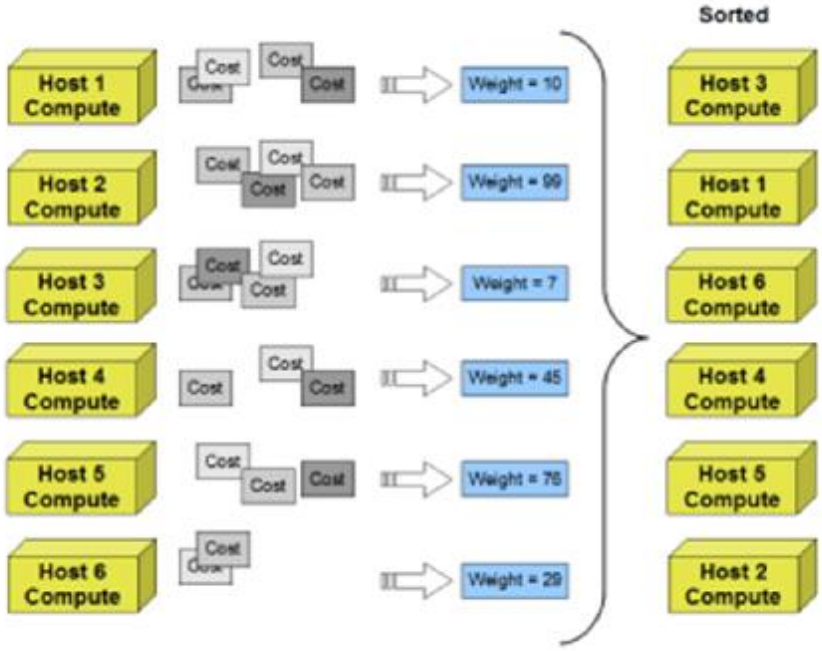
- Közös adatbázis, de minden komponens csak a saját adatait éri el
- Egy OpenStack telepítés egy Zóna
  - Különálló API node, Scheduler, adatbázis és üzenetsor
  - Egymásba ágyazhatóak
  - Publikus API-n keresztül érik csak el egymást, nincs megosztott információ
  - Csak közvetlen gyermeket ismerik, kéréseket továbbíthatják
- Az ütemező kapcsolatban van egy *ZoneManager* objektummal, ami összegyűjti a gyermekek adatait
- Három üzemmód: *Egyszerű*, *Esély*, *Zóna*

OpenStack – Ütemezési algoritmusok

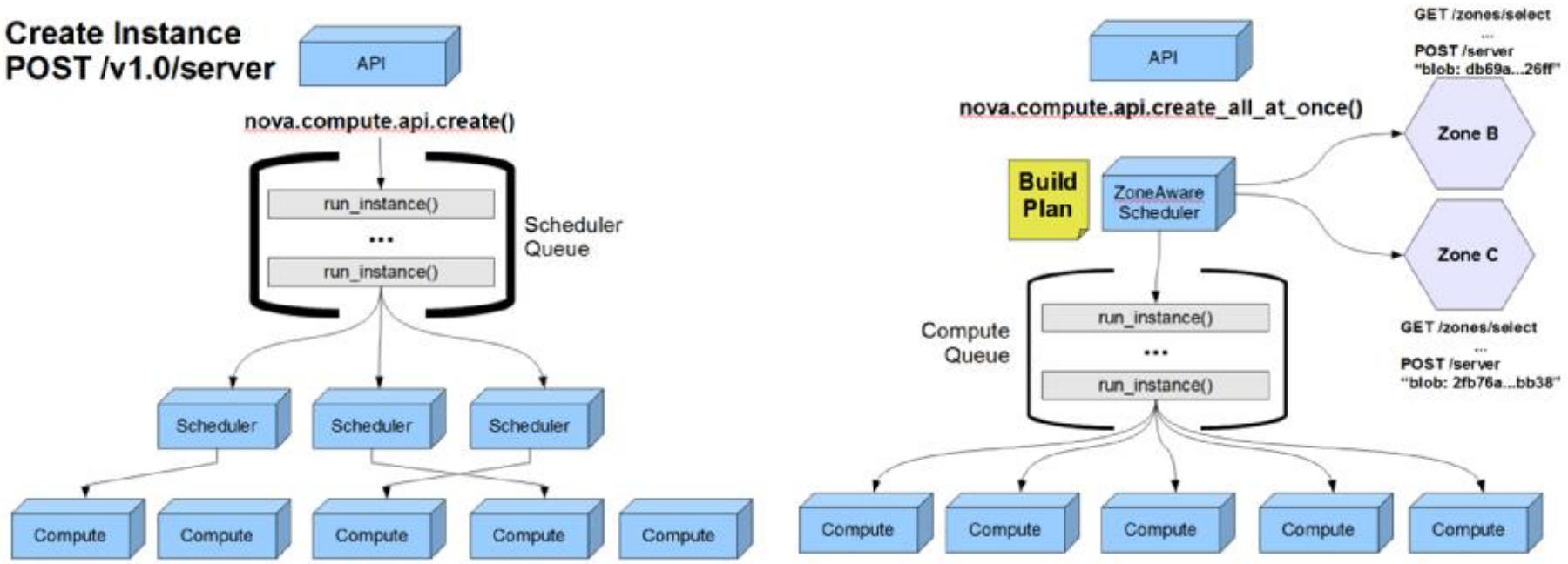
- *BaseScheduler*
  - Ősosztály, amely elfedi a gyermek zónákból való választást az egyes ütemezők elől
- *Egyszerű (Simple)*
  - A legkevésbé terhelt hosztot választja
- *Esély (Chance)*
  - Véletlenszerűen választ hosztot
  - Gyermek zónából is választhat
- *Zóna (Availability Zone)*
  - Adott zónából választ csak hosztot véletlenszerűen
  - Gyermek zónát figyelmen kívül hagyja

OpenStack – Költségek számítása

- A fizikai gépek terheltségének számításához
- Minden hoszt minden erőforrásához egy egész szám hozzárendelése
- Ezek összege adja a súlyt, ami alapján sorrendezni lehet



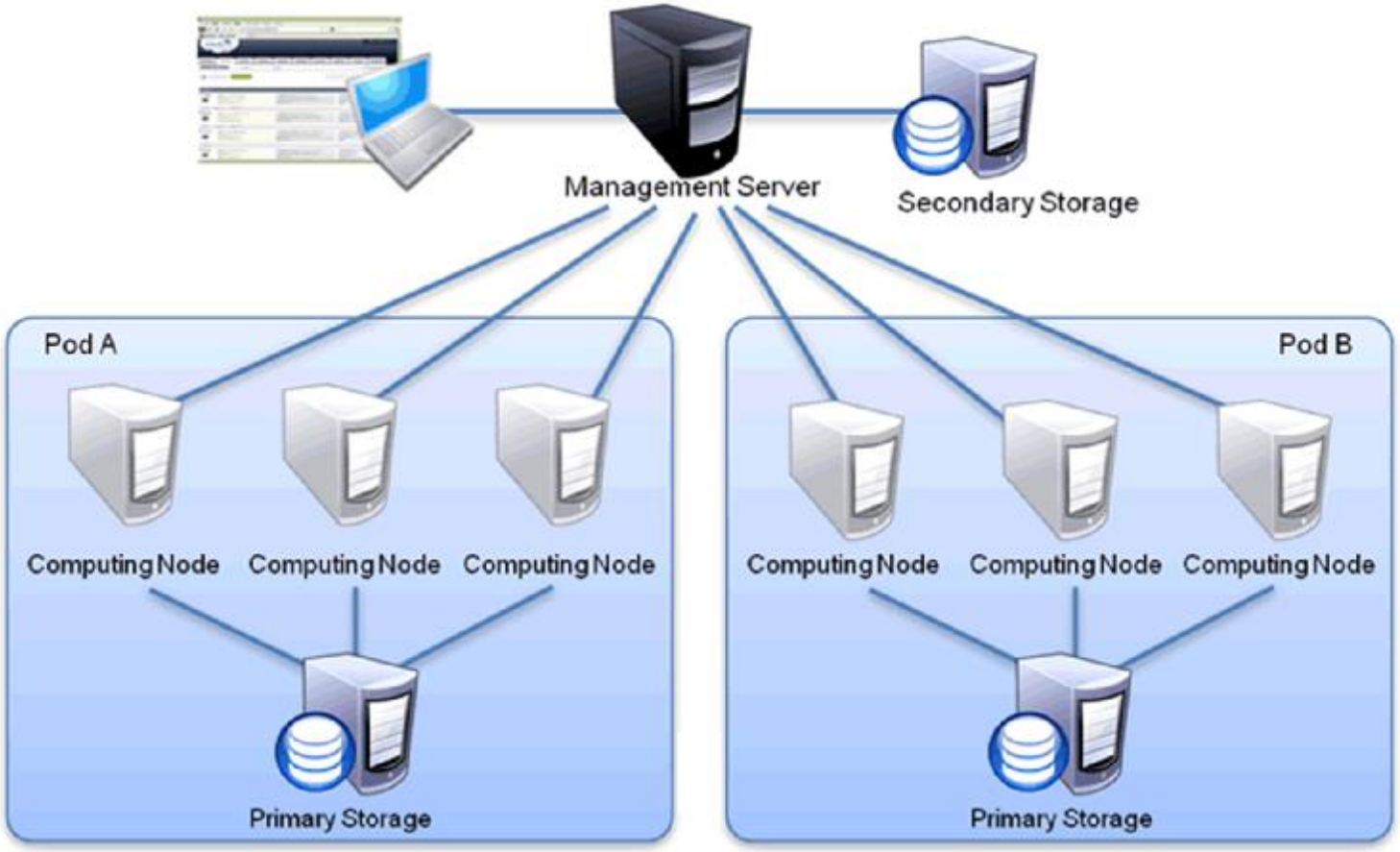
OpenStack – Allokációs kérések kötegelése



CloudStack

- Privát IaaS cloud szolgáltatás
- Nyílt forráskódú, Javaban íródott
- Webes interfészen keresztül érhető el
- Nem tartalmaz virtualizációs technológiát, csak drivereket
- Komponens alapú architektúra, nagyon könnyen bővíthető
- Egyedüli korlát a közös adatbázis

CloudStack – Architektúra



CloudStack – Erőforrásallokáció

- Vizsgált erőforrások
  - CPU: magok száma és teljesítmény
  - Memóriaméret
- Három algoritmus:
  - *TestingAllocator*
    - Véletlenszerűen választott erőforrás, semmi ellenőrzés
  - *RandomAllocator*
    - Véletlenszerűen választott, elegendő erőforrás
  - *FirstFitAllocator*
    - Mohó algoritmus
- Könnyen bővíthető

Célok

- Mindhárom cloud más előnyökkel bír:
  - Eucalyptus: Powersaver üzemmód
  - OpenStack: Kérések kötegelése, terv kiépítés
  - CloudStack: Könnyen bővíthető architektúra
- Algoritmus kidolgozása:
  - Gyors lefutás
  - Terhelés kiegyenlítése
  - Költségcsökkentés – minél kevesebb fizikai hardver
- Ellentmondó szempontok

Összefoglalás – Jövőbeli célok

- Mindhárom cloud rendszer vizsgálata megtörtént
- Ütemezési algoritmusok kezdetlegesek, nem nyújtanak valódi előnyt ipari környezetben
- Tanszéken sok szerveres CloudStack telepítése és konfigurációja megtörtént
- Algoritmusok kutatása, implementációja, tesztelése a különböző szempontok kielégítéséhez
- A fenti eredményeket a TÁMOP - 4.2.2.B-10/1--2010-0009 projekt támogatta.

Köszönöm a figyelmet!

Kérdések?