Referencia architektúrák használati koncepciója az ELKH Cloudon

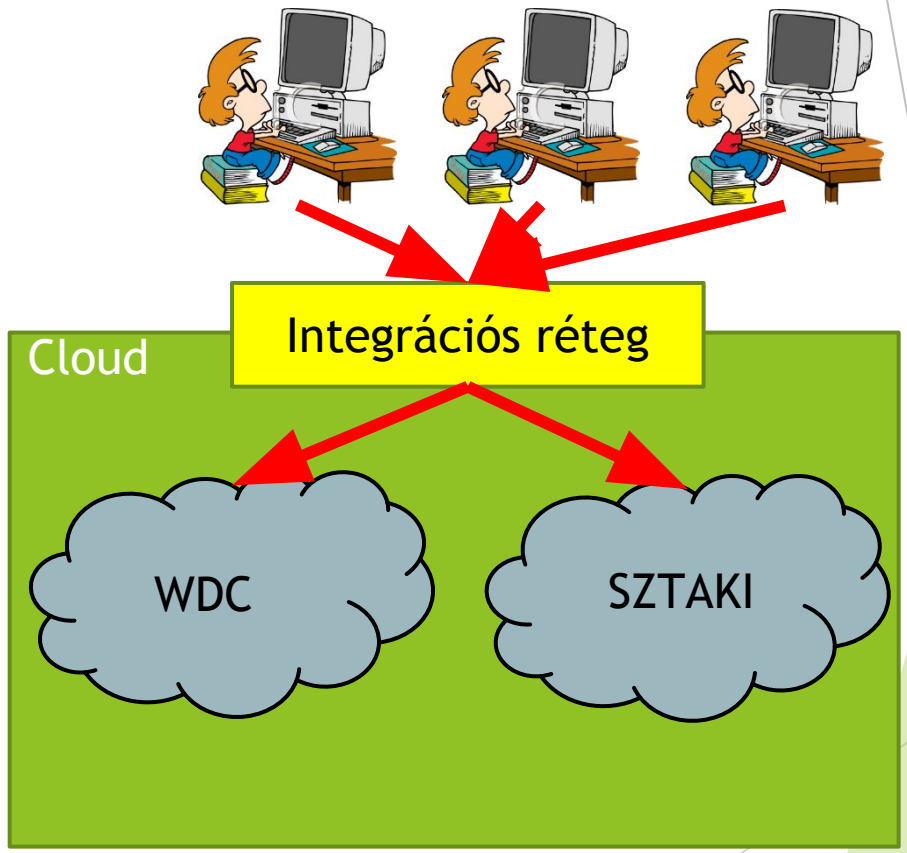
Dr. Kacsuk Péter
ELKH Cloud projekt
szakmai vezetője

Az MTA/ELKH Cloud története dióhéjban

- 2016. okt. 1: Az MTA Cloud, mint teljes körű Infrastructure as a Service (IaaS) szolgáltatás elindul
- 2020. július 1: Elindul az ELKH Cloud projekt, melynek célja a Cloud korszerűsítése és jelentős kapacitásbővítése
- 2020. okt. 1: a cloud hivatalos átnevezése **ELKH Cloud** névre
- 2021. július 1: Az új ELKH Cloud tervezett átadása a magyar kutatói közösség számára

Az MTA és ELKH Cloud kapacitásának összehasonlítása

	MTA	ELKH
vCPU (max)	1368	4000
GPU core	12	76
vGPU (max)	12	2060
RAM (TB)	3,25	11
SSD storage (TB)	0	153
HDD storage (TB)	527	1500
Tensor GPU teljesítmény (PFLOPS)	~0	7.16
Lebegőpontos GPU teljesítmény (PFLOPS)	~0	0.89
Hálózati sávszélesség (Gbps)	10	100



Az ELKH Cloud jelenlegi kihasználtsága

- Az elindított projektek száma: 150
- A befejezett projektek száma: 62
- Web lapon elérhető ezekről minden lényeges információ: https://cloud.mta.hu/projektek

Projektek

Összes Futó Befejezett

11 - 20 | 150 projekt

Projekt neve	Intézmény	Vezető
Adatintegrációs biomatemetikai modellezés	Enzimológiai Intézet	Dr. Pálhalmi János
Komputációs statisztikai modellezés transzkriptomikai, proteomikai és epigenomikai adatbázisok alapján.		
alicebp	Részecske- és Magfizikai Intézet	Bíró Gábor
Az ALICE Budapest Csoportjának a projektje: Monte Carlo szimulációk fejlesztése és futtatása, adatkiértékelés.		
ALMA rádiótávcsővel végzett megfigyelések kiértékelése	Konkoly Thege Miklós Csillagászati Intézet	Mező György
Az Atacama Large Millimeter/submillimeter Array (ALMA) http://www.almaobservatory.org/en/home/ rádiótávcsővel végzett megfigyelések kiértékelése. Kóspál Ágnes Lendület csoportja http://www.konkoly.hu/DRG/ és ERC		

Az ELKH Cloud koncepciója

- **Az ELKH Cloud egy e-infrastruktúra keretrendszer**, ami
  - Nem felhasználó-orientált, hanem **projekt-orientált**
    - Elsődlegesen projektek regisztrálnak
    - A felhasználóknak nem a cloudra, hanem a projekthez kell regisztrálniuk
  - Egy projekt **kvótát** kap és ezen belül olyan e-infrastruktúrát épít fel a felhőben, amilyet csak akar
    - Pl. csinálhat Hadoop v. Spark klasztert, de azt csak az ő felhasználói érik el

Az ELKH Cloud koncepciója

- e-infrastruktúrát felépíteni a felhőben nem könnyű, ezért az ELKH Cloud projekt
  - A típikus, gyakran használt e-infrastruktúrák kiépítéséhez **referencia architekturákat** biztosít, amikből a kívánt infrastruktúra gyorsan felépíthető (részleteket ld. a mai előadásokban)
  - A nem típikus e-infrastruktúrák felépítéséhez közvetlen segítséget nyújt a felhasználóknak, majd az így létrejött e-infrastruktúrából referencia architekturát csinál
- A referencia architekturák tárolására és elérésére repozitóriumot biztosítunk, ahonnan minden ELKH felhasználó elérheti és használhatja ezeket

Jelenlegi helytelen gyakorlat

- A projekt nagy munkával 2-3 hónap alatt összeállítja azt a szoftver környezetet, amit a felhőn használni szeretne
- De nincs leírás, dokumentáció arról, hogy ez hogyan történt, ezért egy esetleges lebontás után, megint sok időt igényelne a a szoftver környezet újbóli felállítása
- E miatt a projektek félnek lebontani a a szoftver környezetet és akkor is futtatják, amikor semmi szükségük nincs rá. Ez akár hónapokon keresztül történő felesleges futtatás, ami feleslegesen elveszi az erőforrásokat más felhasználóktól

Korszerű gyakorlat, amit szeretnénk meghonosítani

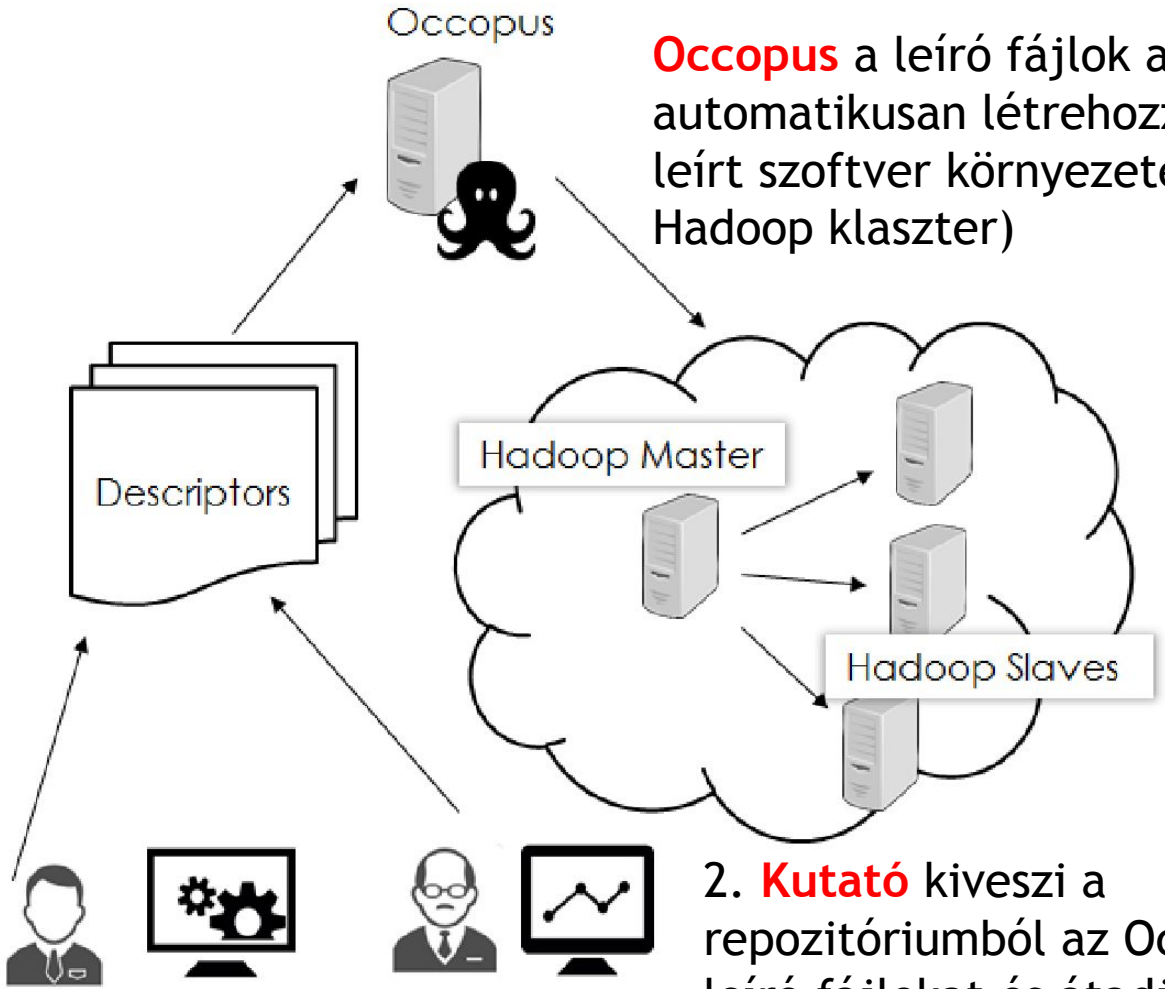
ELKH Cloud

- A szoftver környezet kialakítása egy felhő orkesztrátor segítségével történik.
- A szoftver környezet kialakításának lépéseit un. leírók (descriptorok) formájában tároljuk repozitóriumban
- A leírók alapján a felhő orkesztrátor képes csaknem automatikusan, gyorsan és reprodukálható módon felépíteni a felhőben a szükséges szoftver környezetet
- Ha tehát felhő orkesztrátort használunk, akkor sokkal egyszerűbben és gyorsabban (20-30 perc alatt és nem 1-2 hónap alatt) tudjuk felállítani a szükséges szoftver környezetet
- Ezért bármikor leállíthatjuk a szoftver környezetet, ha hosszabb időre nincs rá szükség

Felhő orkesztrátorok

- Sokféle felhő orkesztrátor létezik
  - Occopus (SZTAKI fejlesztés)
  - Terraform
  - Stb.
- Jelenleg az Occopust támogatjuk, de folyamatosan vonjuk be a Terraform használatát is
- A következő előadás fogja részletezni az Occopust

Felhő orkesztrátorok használati elve szoftver környezet kiépítésére



Occopus a leíró fájlok alapján automatikusan létrehozza a leírt szoftver környezetet (pl. Hadoop klaszter)

1. **Intézeti informatikus** legyártja és tárolja egy repozitóriumban az Occopus leíró fájlokat

2. **Kutató** kiveszi a repozitóriumból az Occopus leíró fájlokat és átadja az occopusnak

Referencia architektúrák

- Az **általánosan használt szoftver környezetekhez** az ELKH Cloud fejlesztő csapat előre legyártotta a szükséges Occopus leírokat
- Így ezekkel az intézeti informatikusoknak nem kell foglalkozni
- Ráadásul, ezek jó példák arra, hogyan kell az Occopus leírókat létrehozni

Felhasználó

edu Belépés

magyar English

Címlap

Felhasználást segítő szolgáltatások

Az ELKH Cloud projekt koncepciójának megfelelően a felhasználást segítő szolgáltatások referencia architektúra formájában vannak megadva. A projektek ezekből építhetik fel azokat a szolgáltatásokat, amiket az alábbi listából kiválogattak.

A szolgáltatások telepítésének és elindításának lépései a következők:

0. Lépés: Előkészítés (Üres Ubuntu virtuális gép indítása a saját projekten belül)

Az 1-8 lépések leírását ld. az Occopus cloud orchestrator indítása referencia architektúra leírásában

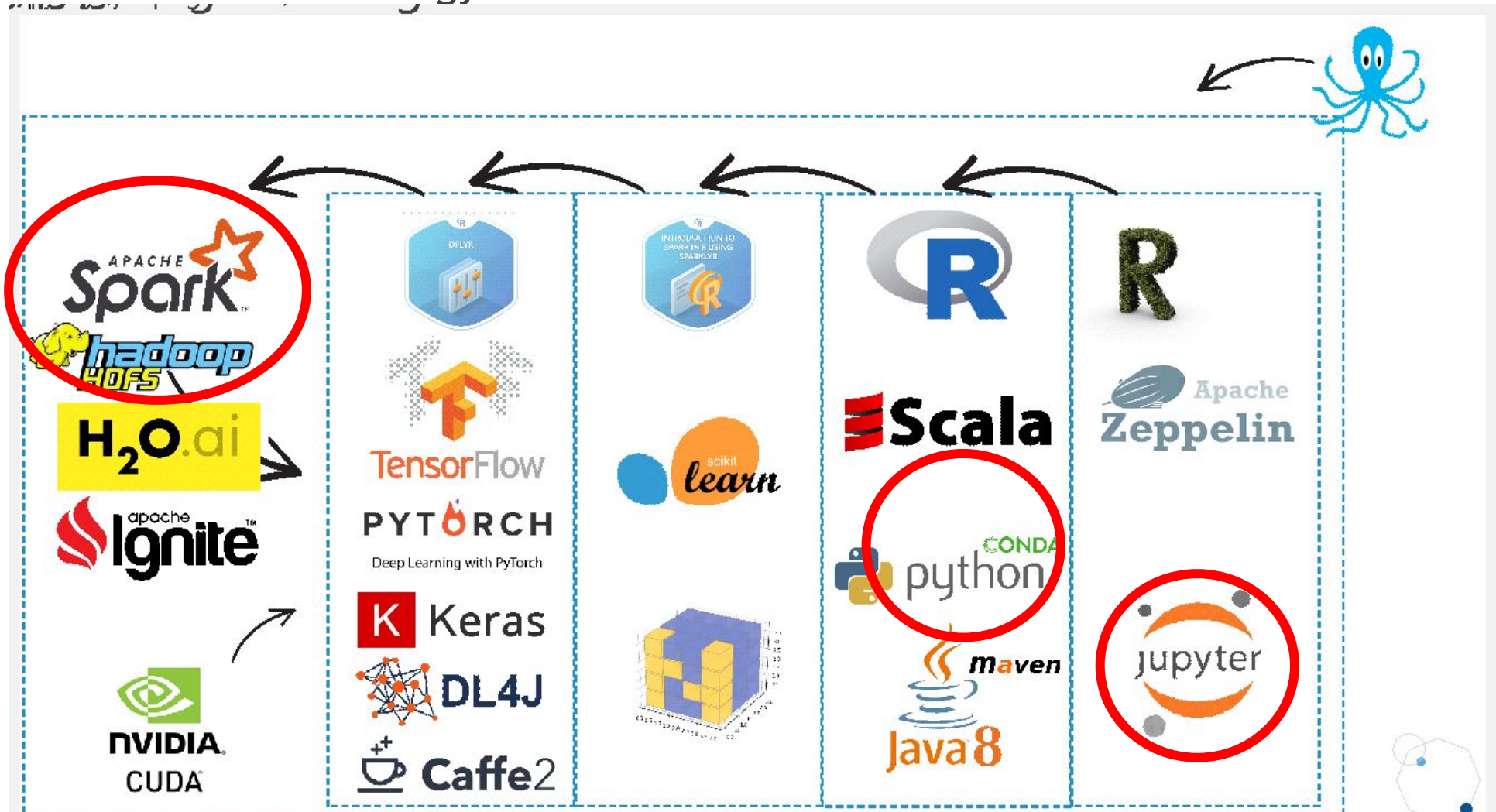
1. Lépés: Occopus telepítés/konfigurálás a virtuális gépen
2. Lépés: Leírók letöltése a virtuális gépre
3. Lépés: Tűzfalszabályok létrehozása ELKH Cloud OpenStack felületén
4. Lépés: Occopus leírók személyre szabása a virtuális gépen
5. Lépés: Occopus aktiválása
6. Lépés: Leírók importálása Occopus számára
7. Lépés: Szolgáltatás kiépítése
8. Lépés: A kiépített szolgáltatás használata

A rendelkezésre álló referencia architektúrák és leírásuk

- Occopus cloud orchestrator indítása
- JupyterLab
- DataAvenue
- Cloud alkalmazásokat támogató portál indítása
- Flowbster - Autodock Vina
- CQueue klaszter
- Docker-Swarm klaszter kiépítése *(Frissítés: ELKH Cloud - Microsoft Azure hibrid felhő támogatással)*
- Kubernetes klaszter
- Apache Hadoop klaszter kiépítése
- Apache Spark klaszter RStudio stack-el
- Apache Spark klaszter Python stack-el *(Frissítés: ELKH Cloud - Microsoft Azure hibrid felhő támogatással)*
- TensorFlow, Keras, Jupyter Notebook stack
- TensorFlow, Keras, Jupyter Notebook GPU stack *(Frissítés: ELKH Cloud - Microsoft Azure hibrid felhő támogatással)*
- Horovod klaszter
- Kafka klaszter
- Slurm klaszter

További indok, hogy miért kellenek referencia architektúrák

Spark és Hadoop referencia architektúrák Big Data és egyszerűbb ML alkalmazásokhoz



Összefoglalás: A referencia architektúrák előnyei

- Olyan szoftvereket tartalmaznak, amik képesek egymással együttműködni
- Együttműködésük alaposan le van tesztelve
- A legnépszerűbb, legmegbízhatóbb, sokak által használt szoftvereket tartalmazzák, ami segít a másokkal való együttműködésben
- Segítségükkel a kívánt szoftver környezet
  - Automatikusan
  - Gyorsan
  - Reprodukálhatóan

  épül fel.

További információk

- Web: https://science-cloud.hu/
- GYIK: https://science-cloud.hu/gyakran-ismetelt-kerdesek
- E-mail: info@science-cloud.hu

Köszönöm a figyelmet

Kérdések?