

# Cloud Computing

## Originile conceptului

**CONF.DR. CRISTIAN KEVORCHIAN**  
FACULTATEA DE MATEMATICA SI INFORMATICA  
UNIVERSITATEA BUCURESTI

# Clustere de Calculatoare

- În anii '90 este promovată ideea agregării a sute de mașini văzute ca un tot în scopul obținerii unui raport preț/performanță optim.
- Aplicații paralele erau dezvoltate pentru a fi implementate în cluster.
- Cluster-ul de servere (ferma de servere) reunește o familie de mașini (noduri) care operează împreună(peste un LAN) și care sunt văzute drept o singură entitate de calcul în scopul:
  - Creșterii puterii de calcul și a disponibilității
  - Load balancing
  - Simplificării managementului resurselor(CPU,memorie disk și bandă în rețea)

## Tipuri de clustere

**Practic, există 3 tipuri de clustere:**

- Fail-over
- Load-balancing
- HPC (High Performance Computing)

Cele mai implementate  
fiind:

**Failover Cluster**

**Load - balancing Cluster.**

# Tipuri de noduri

---

- Cluster-ele includ următoarele tipuri de noduri:
  - Noduri destinate calculului-sunt cele mai numeroase(16,32,64,128 sau 256)
  - Noduri de stocare(pina la 10 noduri)
  - Noduri front-end(unul sau mai multe)
  - Noduri adiționale
  - Noduri destinate sistemelor de masurare și analiza.

## Nodurile interopereaza in rețele

---

Rețelele destinate calculului(computing network)

---

Rețelele destinate administrării si controlului(încărcarea imaginilor de sistem, culegerea de informații din cluster, stocarea măsurătorilor, etc

---

Teoretic toate nodurile beneficiază de egalitate în utilizarea benzii pentru că prioritară este latența redusă. In general viteza era de 1-10 Gb/s, iar latentă era la nivelul câtorva nano secunde.

---

Rețeaua de control fiind o rețea clasică Ethernet cu viteze cuprinse între 100 Mb/s până la 1 Gb/s.

---

Programele executate paralel în cluster utilizeazău frecvent bibliotecile **MPI(Message Passing Interface)**.

# Grid Computing

---

- Termenul grid a fost introdus la sfârșitul anilor '90 de Ian Foster și Carl Kesselman și reprezintă agregarea și partajarea puterii de calcul într-o arhitectură de calcul distribuită.
- Conceptul este o reluare a celui de metacomputing introdus și utilizat în anii '80.
- Specific conceptului de grid este faptul că permite o utilizare transparentă a resurselor de calcul în condițiile de distribuire la nivel global fără a conta locația unde sunt plasate resursele de calcul.
- Conceptul este revizuit în jurul anului 2000. Noțiunea de serviciu ia locul celei de resursă.

# Sisteme grid

- Două mari familii de sisteme au fost dezvoltate:
  - **HPG(High-Performance Grid)** – oferă putere de calcul mare prin intermediul unor sisteme de calcul cu mare putere de procesare sau prin **agregarea de clustere**.
  - Peer-to-peer Grid- obținerea de putere de calcul prin disponibilizarea resurselor neutilizate de la nivelul nodurilor.
  - Comunitățile științifice sau de business pot crea structuri de calcul de tip grid bazate pe servere standardizate plasate într-o rețea.


# GRID5000

- Platformă grid experimentală, creată cu scopul de a studia algoritmi și protocoale pentru calcul paralel și distribuit intensiv dar și pentru aplicații P2P.
- Grid5000 se caracterizează prin: 8 locații, 31 de clustere, 828 noduri cu 12328 coruri și Ethernet 10Gb
- Reconfigurare automată a procesoarelor
- Peste 500 de utilizatori susținuți de o puternică echipă tehnică.
- Aceste resurse de calcul sunt utilizate pentru:
  - Rezerva o partiție din GRID5000(grup de mașini)
  - Instalarea imaginilor software pe fiecare din nodurile rezervate
  - Execuția experimentului
  - Colectarea rezultatelor
  - Pe toată perioada rezervării utilizatorul are total control asupra resurselor rezervate.




# Anatomia unei aplicatii distribuite.


Simularea digitală și analiza complexă a unor volume mari de date caracterizează aplicațiile mediului "High-Performance Distributed Computing"




Simularea digitala a unui sistem fizic(masina, avion, reactor nuclear, etc) constă din rezolvarea unor ecuații ce modelează comportamentul acestora.



Divizarea și integrarea comportamentului diferitelor sisteme componente implică simularea comportamentului acestora in timp, iar ecuațiile asociate sunt rezolvate iterativ.



Optimizările impun factorizarea problemei în componente calculate independet în paralel.



Pentru un job distribuit fiecare instanță este executată pe un calculator diferit. Fiind executate în paralel este necesar schimbul de date între acestea.  
Map/Reduce

# Modelul de programare MapReduce

---

Model de procesare masivă a datelor pe un număr mare de calculatoare

---

Funcțiile Map și Reduce ale MapReduce pot fi utilizate în clustere de dimensiuni mari cu sarcini de procesare la nivelul petabaitilor de date în ore(ex.sortare).

---

Dependențele între task-urile elementare sunt reprezentate printr-un graf aciclic(DAG-Directed Acyclic Graph)

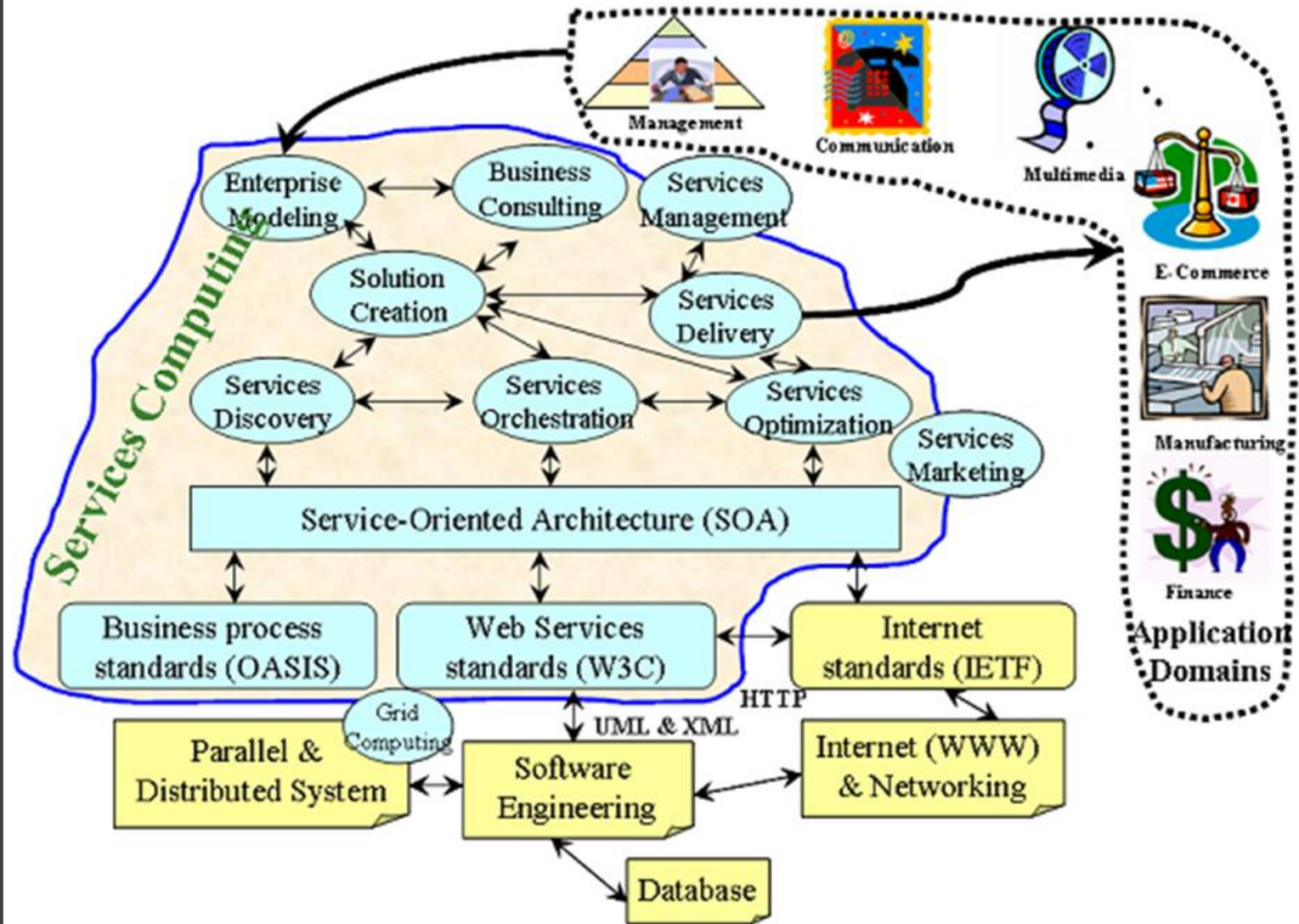
# Simularea stării la momentul Big Bang

- LHC(Large Hadron Collider) de la CERN este un instrument global partajat de un număr de peste 2000 de fizicieni aparținând la peste 150 de universitati din 34 de țări.
- Volumul de date este de ordinul petabaitilor/an(15 petabaiti/an)
- 100000 de procesoare sunt implicate în procesele de calcul
- Fiecare utilizator are alocat un anumit segment al resurselor de calcul
- Există un grid global(LCG-Large Collider Grid). 10 centre de calcul sunt distribuite la nivel continental.

# Service Computing(SC), IEEE 2003)

- "Services Computing" este un concept aflat la intersecția științei cu tehnologia creat pentru a reduce decalajul dintre "Serviciile asociate Businessului" și Serviciile IT.
- Suita de tehnologii include: **SOA, Cloud Computing**, Modelarea Proceselor de Business, Transformare și Integrare.
- Obiectivul SC este de a eficientiza interoperarea dintre serviciile IT și tehnologiile de calcul cu serviciile de business.

Problematika acoperită de SC (conform IEEE) pornește din zona consultanței pentru afaceri și “enterprise modelling” până la servicii Web și cloud computing

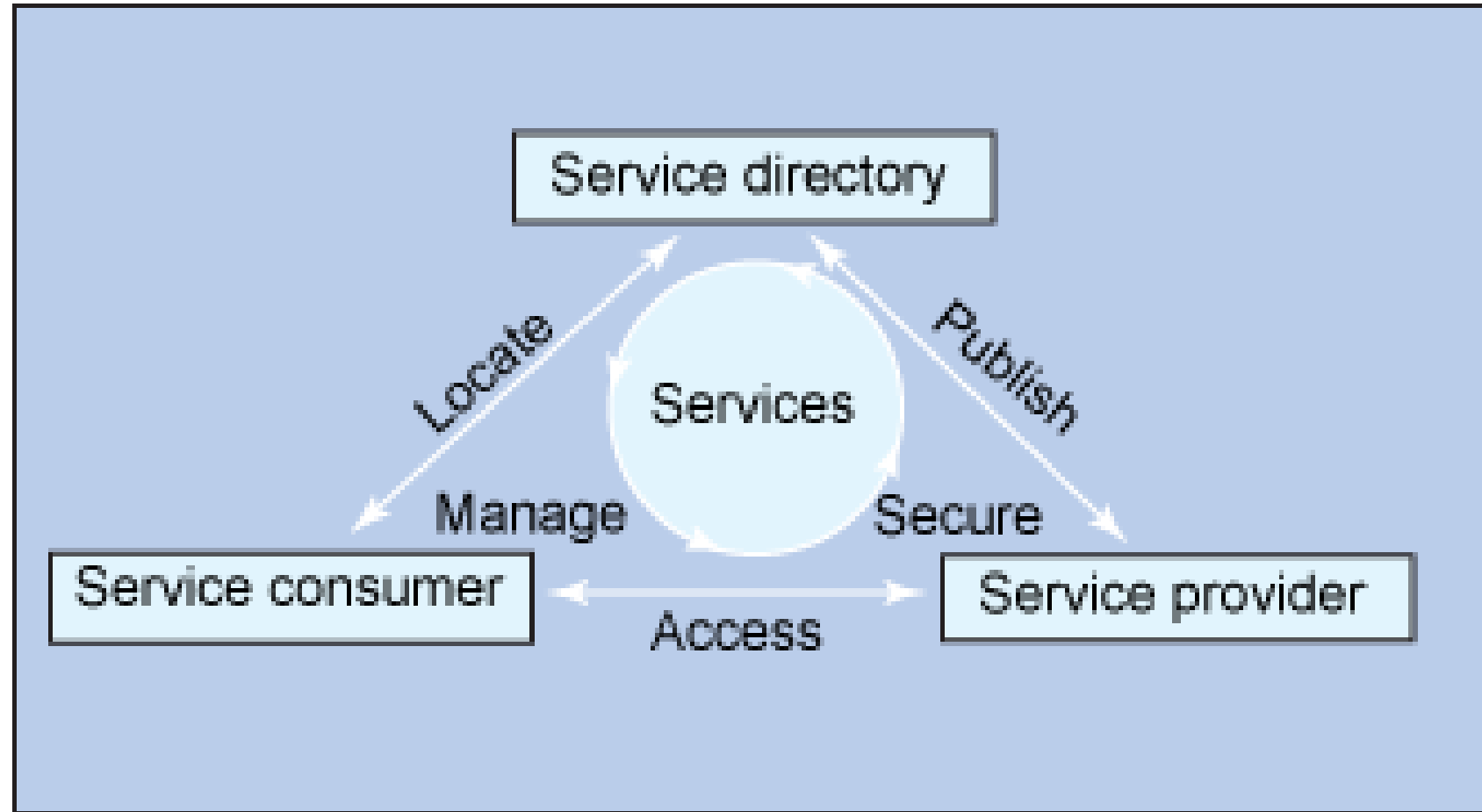


# SOA vs. Calcul Distribuit

- Calculul Distribuit este un model de calcul în care componentele unui system software sunt partajate peste o rețea de calculatoare, iar comunicarea și coordonnarea lor bazându-se pe protocoale de mesagerie(ex. HTTP, RPC, MSMQ)
- SOA este un model de arhitectură în calcul distribuit având asociată o paradigmă de software în care serviciile sunt reprezentate de aplicații executate la nivelul nodurilor computationale.
- Pe scurt, componentele unui serviciu sunt:
  - **UDDI**(Universal Description Discovery and Identification)-repository de servicii în vederea identificării lor
  - **Furnizorul de servicii** – publicarea WSDL(Web Services Description Language) și referențierea în UDDI.
  - Consumatorul de servicii

# Schema unui serviciu

(sursa: M. Fowler)



## SOA - Avantaje

Serviciile Web sunt independente de platformă și independente de limbaj, deoarece utilizează XML. Aceasta înseamnă că aplicațiile care consumă serviciul pot fi programate în C#, Java, C++, PHP și rulează sub Windows, în timp ce serviciul Web poate fi programat în Java și să ruleze atât sub Linux cât și sub Windows.

Majoritatea serviciilor Web utilizează HTTP pentru transmiterea mesajelor (cum ar fi cererea de servicii și răspunsul). Acesta este un avantaj major dacă doriți să construiți o aplicație la scară largă, deoarece majoritatea proxy-urilor și firewall-urilor de pe Internet nu vor afecta traficul HTTP (spre deosebire de CORBA (**Common Object Request Broker Architecture**), care de obicei ridică probleme de securitate).



Overhead. Transmiterea tuturor datelor în XML nu este în mod evident la fel de eficientă ca utilizarea unui cod binar. Avem câștig în portabilitate, și pierderi în eficiență. "Cheltuiala" este de obicei acceptabilă pentru majoritatea aplicațiilor, dar aproape sigur nu veți găsi niciodată o aplicație critică în timp real care utilizează serviciile Web.

Slabă versatilitate. Serviciile Web nu sunt foarte versatile, deoarece permit doar unele forme de bază ale invocării serviciului. CORBA de exemplu, oferă programatorilor o mulțime de servicii de asistență (cum ar fi persistența, notificările, gestionarea ciclului de viață, tranzacțiile etc.). Din fericire, există o mulțime de specificații de servicii Web emergente (inclusiv WSRF, OASIS) care ajută la îmbunătățirea serviciilor Web.

## SOA Dezavantaje

Cloud Computing este  
o paradigmă  
CaaS(**Computing as a  
Service**) operațională  
peste Internet

## Computing-ul livrat ca bun de larg consum

- Computing-ul se transformă în baza unui model bazat pe servicii care se distribuie similar utilităților (electricitate, gaz curent, tv, etc)
- În 1969 **Leonard KLEINROCK** (ARPANET) afirma:  
**"As of now, computer networks are still in the infancy, but as they grow up and become sophisticated, we will probably see the spread of 'computer utilities' which, like present electric and telephone"**
- Modelul a fost aplicat începând din 2007 și cunoscut sub numele de "utility computing", care ulterior a fost adoptat drept cloud computing.

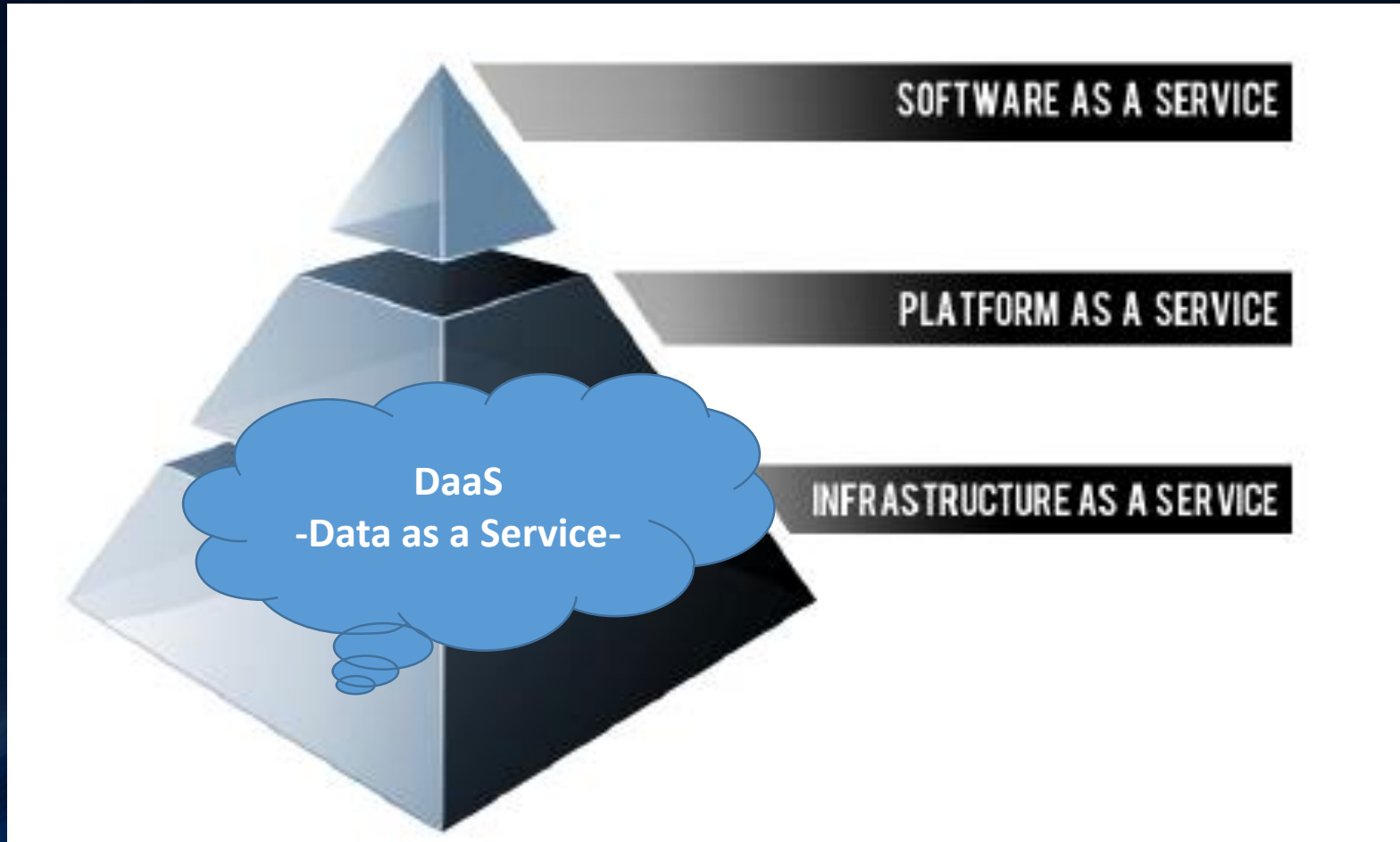
## Paradigma Cloud Computing

- Cloud computing, clasificat drept o nouă paradigmă pentru provizionare dinamică a serviciilor de calcul bazate pe tehnologii de virtualizare pentru consolidarea și utilizarea eficientă a resurselor.
- Cele prezentate pot fi reduse la:

“Nu are importanță unde se găsesc serverele și cine asigură managementul acestora, unde sunt stocate documentele sau unde sunt instalate aplicațiile, dar vreau să-mi fie disponibile 7x24 și să le pot accesa de pe orice device conectat la Internet”
- Cloud computing este un model care permite accesul permanent la un grup de resurse de calcul configurabile (de exemplu, rețele, servere, soluții de stocare, aplicații și servicii), care pot fi gestionate cu un efort minim.

# Cloud Computing Architecture

Cloud Computing = SaaS+PaaS+IaaS+DaaS



# Cloud Computing = SaaS+PaaS+IaaS+DaaS

- **Software as a Service (SaaS)**
  - Din perspectiva utilizatorului
  - Aplicațiile sunt localizate în cloud într-o arhitectură multitenant.
  - Experiența utilizării funcționalităților expuse de aplicație este **consumată** peste Internet
- Exemple
  - Office 365, GoogleApps, SalesforceIQ(CRM pentru IMM-uri), Oracle ERP Cloud

# Cloud Computing = SaaS+PaaS+IaaS+DaaS

- **Platform as a Service (PaaS)**

- Din perspectiva dezvoltatorului
- Furnizorii de cloud oferă dezvoltatorilor care nu doresc utilizarea cloud-ului privat o platformă-bazată-pe-Internet cu ajutorul căreia să creeze, instaleze și să întrețină servicii software.

- **Exemple**

- ORACLE Java Cloud Services, SQL Azure, GAE(Google App Engine-Python și MTV-ul Django), Heroku PaaS(Java, Scala, GO,Ruby, Clojure), SAP Hana Cloud Platform
- Piețe de app AppExchange(SalesForce.com), Mechanical Turk(Amazon)

# Cloud Computing = SaaS+PaaS+IaaS+DaaS

- **Infrastructure as a Service (IaaS)**

- Furnizorii de cloud construiesc rețele de centre de date la nivel global
  - hardware, networking, storage, sisteme distribuite, scalabilitate, instalații electrice etc
- Datacenter as a service
- Utilizatorii de cloud închiriază servicii de storage, calcul și mentenanță de la furnizorii de cloud (pay-as-you-go; similar utilităților)

- **Apache Mesos aduce valoare adăugată în IaaS (chiar în bar-metal cloud)**

- Fiind “cluster manager” poate consolida resursele acestuia tratându-l drept o resursă unitară. De exemplu resursa CPU poate ajunge (în cloud) la sub 7%.



# IaaS vs. Containerizare

Docker permite orchestrarea resurselor de calcul furnizate de o familie de servere.

Permite dezvoltatorilor să construiască containere în care să pună aplicații cu posibilitatea deployment-ului în AWS

Containere ușor de generat cu scriptul Dockerfile

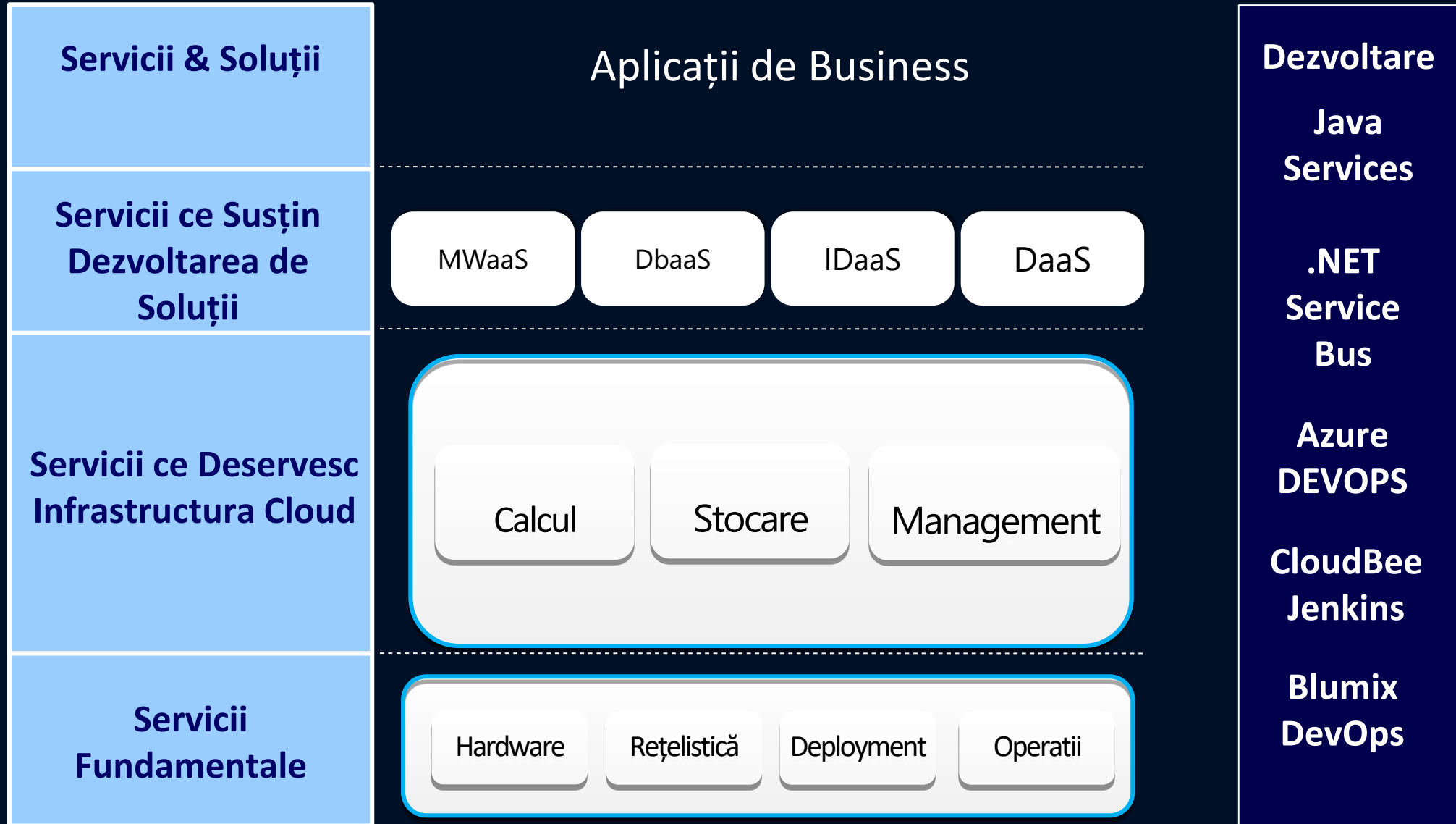
Scalare automată

**Cloud Computing = SaaS+PaaS+IaaS+DaaS**

**Date → Informații → Cunoștințe → KDE (Knowledge and Data Engineering)**

- Data Mining și Knowledge Discovery
- Obținerea de cunoștințe și utilizarea lor ca suport de decizie
- Datele furnizate de aplicații și servicii vor face obiectul unor instrumente avansate de analiză (ML).

# Platforma pentru Cloud Computing



# Avantajele utilizării Cloud Computing

- Reduce atât cheltuielile de capital cât și cele operaționale
- Reduce cheltuielile legate de construcția și exploatarea centrelor de date
- Elimină provizionarea pentru acest tip de cheltuieli.
- Se poate porni de la o soluție simplă și evolua la soluții complexe în funcție de dinamica business-ului.
- Simplificarea proceselor de instalare și mentenanță a aplicațiilor
- Același model de programare pentru aplicații mobile, client-sever sau cloud.
- Integrare cu actualele aplicații și servicii

Vă mulțumesc!  
ck@fmi.unibuc.ro