Simulateurs de Cloud computing

E.I. Djebbar
Département de Génie des systèmes
Ecole Nationale Polytechnique d'Oran

ENP d'Oran -Maurice Audin-
Ingénierie et Management des Systèmes d'Information

PLAN

- Simulateurs de Cloud computing
- Concepts généraux

Cloud computing

- Le Cloud computing est la prestation de services informatiques sur Internet. Ses services permettent à des individus et des entreprises d'utiliser les logiciels et les matériels qui sont gérées par les tiers sur les sites distants.

Services et avantages du Cloud

- Les services proposés par le Cloud comprennent le stockage de fichiers en ligne, sites de réseaux sociaux, webmail, et les applications en ligne.
- Le modèle de Cloud computing permet d'accéder à des ressources d'information et de l'informatique depuis n'importe quel endroit, où une connexion réseau est disponible.
- Les avantages du Cloud computing comprennent l'économie de coûts, la haute disponibilité et l'évolutivité facile.

Services de Cloud

- ***Software-as-a-Service (SaaS)*** : Le service proposé est une application qui est accessible par Internet. En effet, au lieu d'installer et de faire la maintenance des logiciels, il suffit à une connexion d'Internet pour accéder au service.

- ***Platform-as-a-Service (PaaS)*** : Ce modèle offre des plates-formes d'exploitation et de développement au consommateur. Le consommateur peut utiliser la plate-forme pour développer et exécuter ses propres applications, soutenues par une infrastructure fournie par le fournisseur de Cloud.

- ***Infrastructure-as-a-Service (IaaS)*** : C'est le modèle de service plus bas dans la pile technologie de Cloud, offrant des ressources d'infrastructure comme un service, tel que le stockage de données brutes, la puissance de traitement et la capacité du réseau.

Modèles de déploiement de Cloud

- ***Privé*** : Cloud privé est détenu et exploité par une seule société, et les services fournis par le Cloud sont utilisés par les différents secteurs d'activité dans la même entreprise.
- ***Public*** : Les Clouds publics sont détenus et exploités par les fournisseurs pour offrir un accès rapide à des ressources informatiques abordables à d'autres organisations ou individus.
- ***Hybride :*** Ce Cloud utilise la base de Cloud privé combiné avec les stratégies d'usage de services du Cloud public.
- ***Communauté*** : Le Cloud partagé qui est destiné à un ensemble limité d'organisations(universités).

Pourquoi la Simulation?

- Les chercheurs utilisent les simulateurs pour effectuer leur scénarios avant de les effectuer au sein d'un système distribué réel.

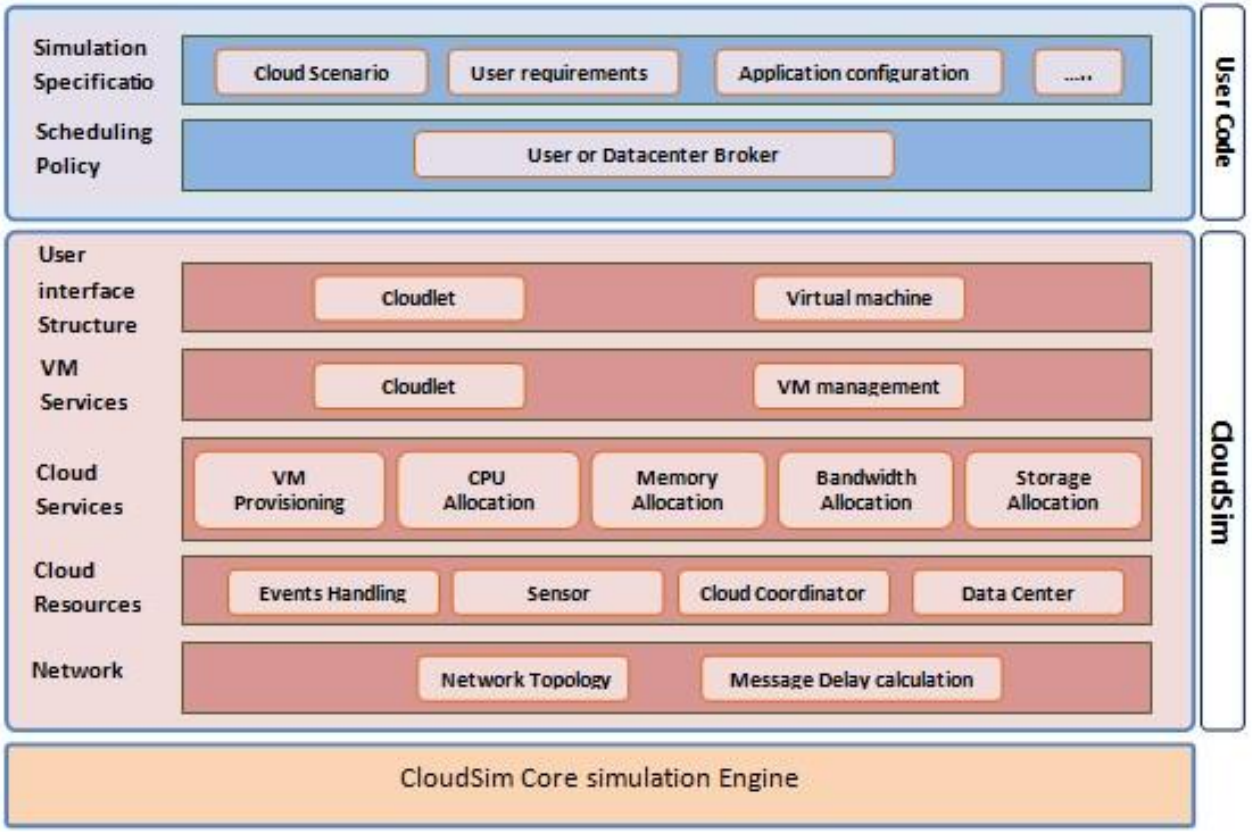
Outils de simulation

- Il y a plusieurs outils de simulation de systèmes distribués. Les plus connus sont GridSim, CloudSim, Simgrid. Dans ce qui suit, on va présenter le framework CloudSim, qui permet de la simulation de l'environnement du Cloud computing.

Cloudsim

- Ce framework modélise et simule l'environnement du Cloud computing et ses services,
- Il est réalisé en Java.

Architecture du CloudSim



Paramétrage de la simulation

- Chaque Cloud est constitué des Datacenters et dans le dernier, on trouve des hôtes et chaque hôte héberge les VMs.
- Pour faire la simulation, il faut définir une classe qui contient la fonction Main()
- On définit les paramètres de notre Cloud comme le nombre de Datacenter, hôtes, les caractéristiques de chaque hôte et VM comme la bande passante, CPU et RAM.

Principales classes

- Le simulateur ClouSim est composé de plusieurs classes.

- Parmi les classes fondamentales qui forment les blocs constitutifs du simulateur CloudSim:

Principales classes

- **Datacenter :** La classe Datacenter englobe un ensemble de machines physiques appelées hosts avec leurs caractéristiques telles que la mémoire, le nombre de cœurs ou encore le stockage.

- **Cloudlet :** La classe cloudlet modélise les tâches. Chaque Cloudlet a une longueur d'instruction pré assignée. Elle contient un nombre d’instructions et de données à exécuter et à transférer.

Principales classes

- **Datacenter Broker :** Cette classe représente un courtier (broker) agissant pour le compte d'un utilisateur. Il masque la gestion des machines virtuelles, comme la création de machines virtuelles, l'envoi de cloudlets à ces machines virtuelles et la destruction des machines virtuelles.

- **Host :** Cette classe représente les caractéristiques d’une machine physique. Elle encapsule des informations importantes, telles que le nombre de cœurs de traitement et la taille de la mémoire.

Principales classes

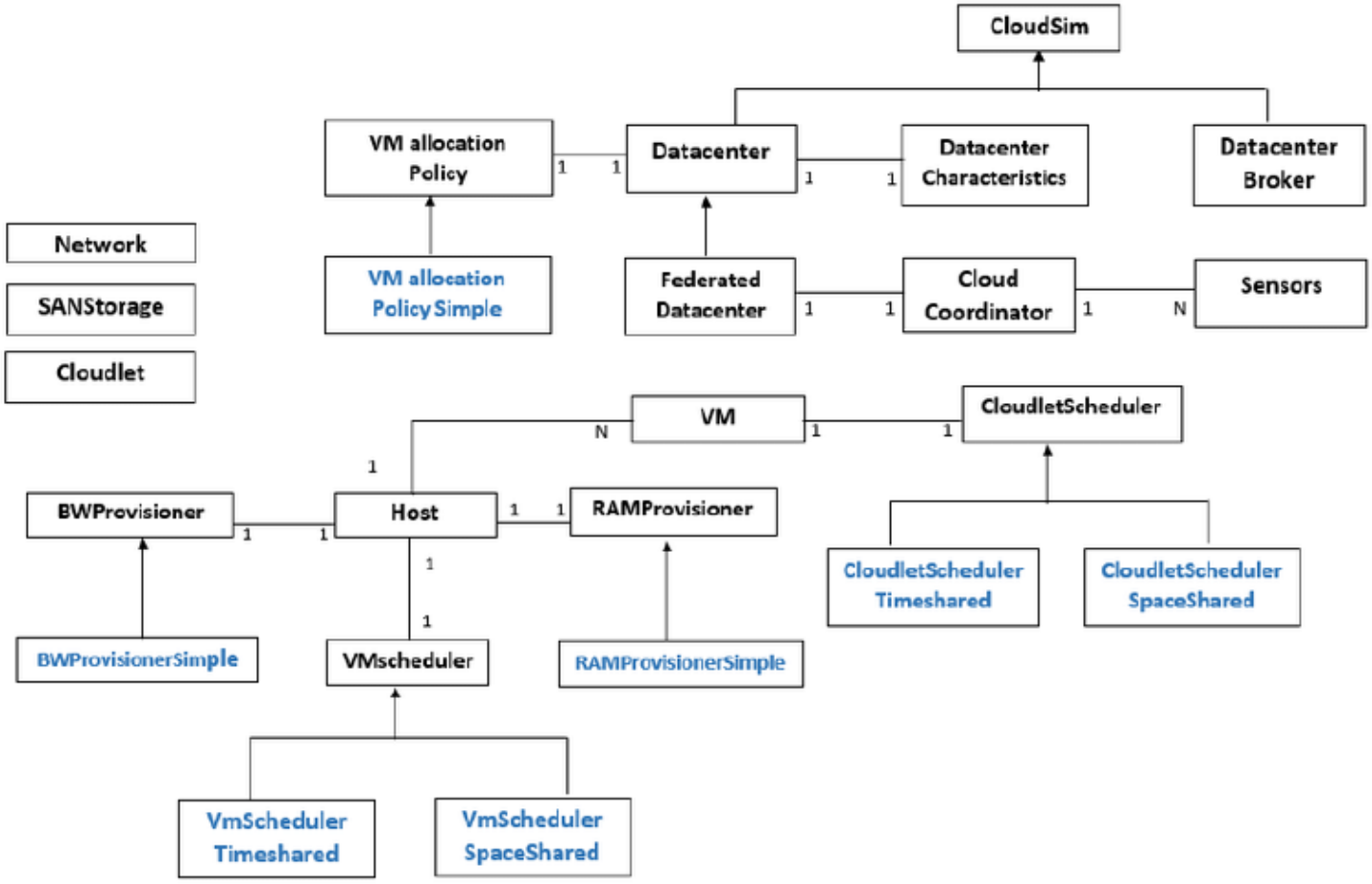
- **Datacenter Broker :** Cette classe représente un courtier (broker) agissant pour le compte d'un utilisateur. Il masque la gestion des machines virtuelles, comme la création de machines virtuelles, l'envoi de cloudlets à ces machines virtuelles et la destruction des machines virtuelles.

- **Host :** Cette classe représente les caractéristiques d’une machine physique. Elle encapsule des informations importantes, telles que le nombre de cœurs de traitement et la taille de la mémoire.

Principales classes

- **VM :** Cette classe modélise une machine virtuelle, qui est gérée et hébergée par un host. Chaque VM a accès à un composant qui stocke les caractéristiques liées à une VM (le processeur, mémoire, la taille de stockage).

Principales classes



La configuration de la VM

```
//VM description
    int vmid = 0;//vm id
    int mips = 250;//number of operations
    long size = 10000; //image size (MB)
    int ram = 512; //vm memory (MB)
    long bw = 1000;//vm bandwidth
    int pesNumber = 1; //number of cpus
    String vmm = "Xen"; //VMM name

//create VMs
     Vm vm1 = new Vm(vmid, brokerId, mips,
  pesNumber, ram, bw, size, vmm, new
  CloudletSchedulerTimeShared());
```

La configuration de l'hôte

```
//create host
    List<Host> hostList = new ArrayList<Host>();
    List<Pe> peList = new ArrayList<Pe>();
     int mips = 1000;
     peList.add(new Pe(0, new PeProvisionerSimple(mips))); // need to store Pe id and MIPS Rating
     int hostId=0;
     int ram = 2048; //host memory (MB)
     long storage = 1000000; //host storage
     int bw = 10000;

     hostList.add(
                    new Host(
                           hostId,
                           new RamProvisionerSimple(ram),
                           new BwProvisionerSimple(bw),
                           storage,
                           peList,
                           new VmSchedulerSpaceShared(peList)
                    )
             );
```

La configuration du Datacenter

```
String arch = "x86";      // system architecture
String os = "Linux";          // operating system
String vmm = "Xen";
double time_zone = 10.0;         // time zone this resource located
double cost = 3.0;              // the cost of using processing in this resource
double costPerMem = 0.05;                // the cost of using memory in this resource
double costPerStorage = 0.001;  // the cost of using storage in this resource
double costPerBw = 0.0;                  // the cost of using bw in this resource
LinkedList<Storage> storageList = new LinkedList<Storage>();    //we are not adding
SAN devices by now

  DatacenterCharacteristics characteristics = new DatacenterCharacteristics( arch, os,
vmm, hostList, time_zone, cost, costPerMem, costPerStorage, costPerBw);

  Datacenter datacenter = null;
        try {
              datacenter = new Datacenter(name, characteristics, new
VmAllocationPolicySimple(hostList), storageList, 0);
        } catch (Exception e) {
              e.printStackTrace();
        }
```

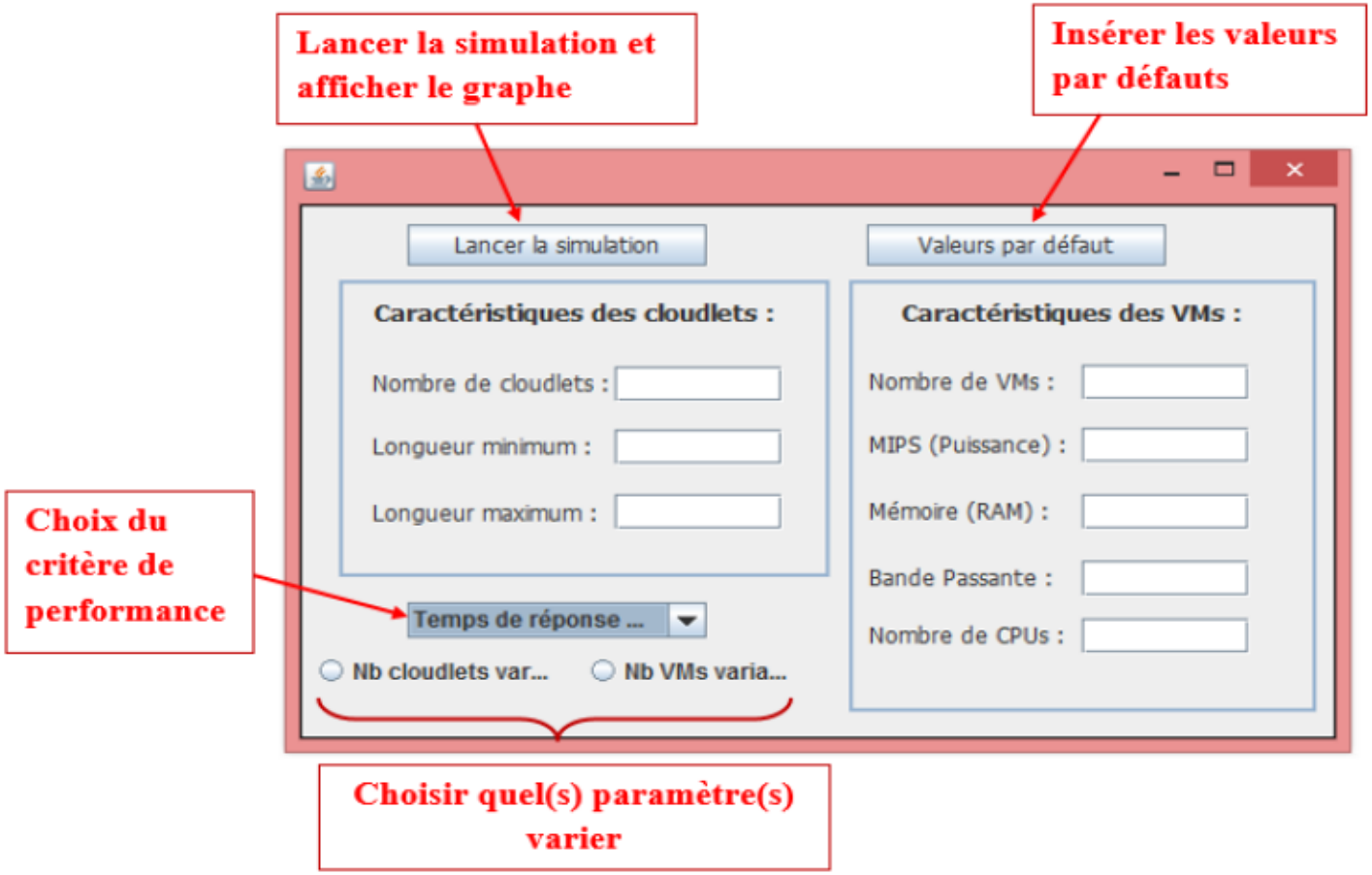
Remarque

- Dans CloudSim, il y a deux aspects importants, Broker et Cloudlet.
  - Broker gère la création de vms, la soumission aux VMs et la destruction de VMs et les Cloudlet sont les tâches à exécuter sur les machines virtuelles.
- La dernière version de CloudSim, nous permet de faire le réseau entre les hôtes dans un datacenter, aussi entre les datacenters en utilisant des switchs et des routeurs.

Etude de cas sur le simulateur Cloudsim

- TP à réaliser par les étudiants

Interface graphique de simulation



Politiques d'ordonnancement

- Il existe deux politiques qui sont denies dans le simulateur CloudSim :
  - La politique d'ordonnancement Space Shared (Espace partagé)
  - La politique d'ordonnancement Time Shared (Temps partagé)

Etape pour définir la politique SPACE SHARED

- l'ordonnanceur (Broker) planifie une tâche sur la machine virtuelle concernée à un instant donnée et après son achèvement, il lance une autre tâche sur la machine virtuelle.
- Cette même politique est utilisée pour programmer les machines virtuelles sur l'hôte.
- Cette politique suit la même procédure que l'algorithme du premier arrivée, premier servi (PAPS)

Etape pour définir la politique SPACE SHARED

- **Etape 1:** Les tâches acceptées sont disposées dans une file d'attente.
- **Etape 2:** La première tâche dans la le d'attente est lancée sur la machine virtuelle donnée.
- **Etape 3:** Apres la terminaison de la première tâche, la prochaine tâche dans la file d'attente sera considérée.

Etape pour définir la politique SPACE SHARED

- Etape 4: Si la le d'attente est vide, le Broker vérifie pour une éventuelle tâche.
- Etape 5: Répéter ensuite a partir de l’étape 1.
- Etape 6: Fin.

Etape pour définir la politique TIME SHARED

- L'ordonnanceur planifie toutes les tâches sur la machine virtuelle en même temps. Il partage le temps entre toutes les tâches et les planifie simultanément sur la machine virtuelle.
- Cette politique est également utilisée pour ordonnancer la machine virtuelle sur l'hôte.
- Le concept de l'algorithme d'ordonnancement Round-Robin (RR)

Etape pour définir la politique TIME SHARED

- Etape1: Les tâches acceptées sont disposées dans une file d'attente.
- Etape 2: Planifier les tâches simultanément sur la machine virtuelle.
- Etape 3: Si la file d'attente est vide, vérifier pour une éventuelle tâche.
- Etape 4: Si une nouvelle tâche arrive, répéter a partir de l'étape 2.
- Etape 5:Fin.