Cloud computing: Document de Synthèse

Nom et Prénom : ALVAREZ Josué, DIOP Mame Aminata.
Date : Janvier 2017.

I. Comprendre la notion du Cloud Computing

- **Expliquez la notion du Cloud Computing (max 10 lignes).**

C'est un ensemble de ressources informatiques mises à disposition d'un utilisateur lui permettant de les exploiter à distance sans se soucier des couches basses gérées par un prestataire de cloud.
L'utilisateur peut se servir de ces ressources sous différentes formes de services:
IAAS: Une infrastructure virtuelle est offerte au client lui donnant la possibilité d'installer des applications ou machines et donc se dispenser d'acheter des machines.
PaaS: Le fournisseur propose le système d'exploitation et l'infrastructure. L'utilisateur peut les exploiter directement ou ajouter ses propres applications.
SaaS: Le fournisseur offre des applications à l'utilisateur qui peut les installer sur ses propres machines ou les manipuler directement.

- **L'avantage du cloud computing sont :**

Le coût : une petite entreprise n'a pas forcément moyen d'investir dans une infrastructure et des serveurs qu'elle doit financer puis maintenir. Le cloud computing permet à ses usagers de ne pas se soucier des problèmes liés à la maintenance des infrastructure (pas besoin de personnel supplémentaire, etc…).
La flexibilité : une application peut avoir des charges différentes selon les périodes du jour / l'année. Le besoin en ressources n'est donc pas constant au fil du temps. Le cloud computing permet de gagner en flexibilité en donnant la possibilité d'augmenter ou de diminuer la capacité en fonction de la charge.

II. Utiliser une infrastructure de cloud

- **Positionnez OpenStack dans l'architecture Cloud.**

OpenStack est un produit de type IaaS (Infrastructure as a service). Il nous donne accès à une plateforme qui nous permet de déployer des machines virtuelles, et prends en charge, entre autres, les aspects de stockage, de gestion du réseau, et de gestion du parc de machines.

- **Expliquez les fonctionnalités offertes par OpenStack à un utilisateur Cloud et les manières avec lesquelles ce dernier peut interagir avec. Donnez la requête que vous avez utilisée pour la création d'une Instance via l'API REST.**

OpenStack donne la possibilité à ses utilisateurs d'accéder à plusieurs services (Compute (Nova), Network (Neutron), Storage etc…) via son client web léger ainsi qu'une API REST.

Nova donne la possibilité de gérer des images (sortes de templates de machines virtuelles) et de les instancier à la demande avec le gabarit (“flavor”) demandé. Le gabarit représente les ressources CPU, Mémoire RAM, Stockage qui seront allouées à la machine virtuelle. Il est aussi possible de configurer la manière dont les instances sont mises en réseau.
La requête suivante permet de créer une instance de machine virtuelle. Les paramètres de cette requête sont :

- image : l’identifiant de l’image à utiliser.
- flavor : l’identifiant du gabarit à utiliser.
- network : l’identifiant du réseau à utiliser.

Requête de configuration

```
{
"server": {
        "name": "My-Shiny-Server",
        "imageRef":
        "http://openstackIP:8774/v2/{{tenant}}/images/{{image}}",
        "flavorRef":
        "http://openstackIP:8774/v2/{[tenant]}/flavors/{{flavor}}",
         "networks": [
                {
                   "uuid": "{{network}}"
                }
         ],
        "security_groups": [
                {
                   "name": "default"
                }
          ],
         "metadata": {
                "My Server Name": "Create Instance via REST API"
            }
         }
}
```

III. Déployer et adapter de manière autonomique une plate-forme pour l’Internet des Objets sur le cloud

Déployer une architecture PaaS basée sur OM2M (TP 2)

- **Quelles sont les composants de l’architecture OM2M ?**

L’architecture OM2M est composée de deux parties principales : le noeud IN, le noeud MN. Chacun des ces noeuds possède une base de données qui peut être locale ou externe. Dans notre cas, la base de données du noeud IN est externe.

- **étant positionné comme un fournisseur de service Cloud, à qui et à quels fins la PF OM2M sera offerte ?**

OM2M peut être déployé en temps que Platform As A Service, pour des clients souhaitant gérer des parcs de capteurs et d'actionneurs. Ces clients peuvent ne pas avoir connaissance du fonctionnement d'OM2M.
L'intérêt est que des applications de plus haut niveau se chargent de proposer une interface "user-friendly" à l'utilisateur ; ces applications s'appuieraient sur le cloud OM2M.

- **Quel est l'intérêt de déployer le « IN-CSE » sur une infrastructure Cloud ?**

L'intérêt de déployer le noeud IN-CSE sur le cloud est la scalabilité : selon la charge et le nombre de noeuds MN associés, il faut pouvoir offrir une solution qui permet d'ajuster les ressources nécessaires au bon fonctionnement de la plateforme.

- **Dans la réalité, y'a-t-il un intérêt de déployer le « MN-CSE » sur une infrastructure Cloud ? pourquoi ? Même question pour la BD du « MN-CSE ».**

La charge d'un noeud MN est souvent très faible, de plus il s'agit d'une gateway pouvant fonctionner directement sur des devices connectés à des capteur, souvent par le biais d'un réseau sans fil. Dans ce cas, il n'y a aucun intérêt à externaliser le MN-CSE, puisqu'il faudrait tout de même rajouter un noeud pour récupérer les données des capteurs sur leur réseau.
Quand à la base de données du noeud MN-CSE, elle peut rester locale pour s'affranchir d'un trafic réseau potentiellement important.

Rendre autonomique une architecture PaaS (TP 3)

- **Expliquez le paradigme de l'Autonomic Computing (max 15 lignes).**

L'autonomic computing consiste à gérer les services d'un système informatique de façon autonome sans intervention humaine. Ce paradigme s'appuie sur le principe de self-management qui se décline en quatre grands principes : la self-configuration (configuration automatique des composants), le self-healing (détection et correction automatique des fautes), la self-optimization (contrôle automatique des ressources pour assurer une optimisation fonctionnelles selon des critères définis), la self-protection (protection d'attaques diverses).

Cette automatisation est centrée sur la collecte d'information, la planification (la périodicité d'une action par exemple), la réalisation d'action suite à des évènements et/ou le pilotage via une interface. Avec Frameself par exemple, ce contrôle automatique est géré par la boucle MAPE-K: Monitor, Analyze, Plan, Execute et Knowledge base. Le moniteur récupère les informations des capteurs. Ensuite il crée les symptômes qui vont être analysés par l'Analyzer et envoie les changements au Planner. Ce dernier planifie ce qu'il faut pour les changements. Enfin l'Executer convoque les actionneurs. pour récupérer les données du monde réel : capteur de présence, de luminosité, etc.

- **A quoi correspond un instantané ? Et quel est son usage ?**

Il s'agit d'une copie d'une machine virtuelle à un instant donné. L'utilité principale de l'instantané est qu'il permet de dupliquer à volonté une machine dans un état connu, et prête à fonctionner immédiatement avec des services déjà lancés.

- **Etant positionné comme un fournisseur de service Cloud, expliquez la démarche élaborée afin d'offrir votre Plate-forme comme service PaaS.**

Un service PaaS permet aux développeurs de développer des applications sur des machines virtuelles déjà configurées.
En tant que fournisseur de service Cloud, nous nous assurons que notre PAAS soit facile à utiliser par l'utilisateur. Nous gérons la préconfiguration de sorte que l'utilisateur n'aie pas à se soucier des problèmes de configuration. Nous nous occupons de la création d'une machine virtuelle, et nous préparons un script post-création qui s'exécutera après la création de la machine pour en effectuer la configuration, selon les besoins que le client aura exprimé.

- **Joindre les règles DRULES établies pour chaque composant de la boucle MAPE-K.**

symptomeinference.drl

```
import frameself.format.*;
import java.util.Date;

rule "add HighRequestsRate"
   when
     Event($id: id, category == "RequestsRate",  $value: value, Integer.parseInt(value) >=
2)
   then
                Symptom symptom = new Symptom();
                symptom.setCategory("HighRequestRate");
                symptom.setValue($value);
                symptom.setTimestamp(new Date());
                symptom.setExpiry(new Date(System.currentTimeMillis()+4000));
                insert(symptom);
end


rule "add LowRequestRate"
   when
     Event($id: id, category == "RequestsRate",  $value: value, Integer.parseInt(value) < 2)
   then
                Symptom symptom = new Symptom();
                symptom.setCategory("LowRequestRate");
                symptom.setValue($value);
                symptom.setTimestamp(new Date());
                symptom.setExpiry(new Date(System.currentTimeMillis()+4000));
                insert(symptom);
end


rule "add HighIN-CSENumber"
   when
     Event($id: id, category == "IN-CSENumber",  $value: value, Integer.parseInt(value)
>= 2)
   then
```

```
                Symptom symptom = new Symptom();
                symptom.setCategory("HighIN-CSENumber");
                symptom.setValue($value);
                symptom.setTimestamp(new Date());
                symptom.setExpiry(new Date(System.currentTimeMillis()+4000));
                insert(symptom);
end

rule "add LowIN-CSENumber"
   when
     Event($id: id, category == "IN-CSENumber",  $value: value, Integer.parseInt(value) <
2)
   then
                Symptom symptom = new Symptom();
                symptom.setCategory("LowIN-CSENumber");
                symptom.setValue($value);
                symptom.setTimestamp(new Date());
                symptom.setExpiry(new Date(System.currentTimeMillis()+4000));
                insert(symptom);
end
```

planninference.drl

```
import frameself.format.*;
import java.util.ArrayList;
import java.util.Date;

rule "add AddIN-Dup"
   when
     Rfc(category == "IncreaseIN-CSENumber")
   then
          ArrayList<Attribute> attributes = new ArrayList<Attribute>();
          attributes.add(new Attribute("state", "true"));
          Action action = new Action();
          action.setCategory("Banana");
          action.setName("IncreaseIN-CSENumber");
          action.setAttributes(attributes);
          action.setEffector(new Effector("Thing"));
          action.setTimestamp(new Date());
     insert(action);
end

rule "add RemoveIN-Dup"
   when
     Rfc(category == "DecreaseIN-CSENumber")
   then
          ArrayList<Attribute> attributes = new ArrayList<Attribute>();
          attributes.add(new Attribute("state", "true"));
```

```
            Action action = new Action();
            action.setCategory("Banana");
            action.setName("DecreaseIN-CSENumber");
            action.setAttributes(attributes);
            action.setEffector(new Effector("Thing"));
            action.setTimestamp(new Date());
      insert(action);
end
```

rfcinference.drl

```
import frameself.format.*;
import java.util.Date;


rule "add IncreaseIN-CSENumber rfc"
   when
      Symptom(category == "LowIN-CSENumber")
      Symptom(category == "HighRequestRate")
   then
                  Rfc rfc = new Rfc();
                  rfc.setCategory("IncreaseIN-CSENumber");
                  rfc.setTimestamp(new Date());
                  rfc.setExpiry(new Date(System.currentTimeMillis()+4000));
                  insert(rfc);
end

rule "add DecreaseIN-CSENumber rfc"
   when
      Symptom(category == "HighIN-CSENumber")
      Symptom(category == "LowRequestRate")
   then
                  Rfc rfc = new Rfc();
                  rfc.setCategory("DecreaseIN-CSENumber");
                  rfc.setTimestamp(new Date());
                  rfc.setExpiry(new Date(System.currentTimeMillis()+4000));
                  insert(rfc);
end
```