IA04 - Printemps 2016

Examen final – 3p

2 heures

Documents interdits

Exercice 1 (4 pts)

1. Donner la forme d'une opération dans le cas d'actions simples et séquentielles dans un monde passant d'un état s1 à un état s2.
2. Donner la forme d'une opération dans le cas d'actions générant des influences.
3. Donner deux principales caractéristiques d'un service REST. Quelles sont les verbes décrivant les méthodes les plus employées ?
4. Donner deux caractéristiques principales différentiant les services Web de type SOAP et ceux de type REST.
5. Donner quatre résultats principaux de la phase d'analyse de la méthode de conception de systèmes multi-agents JADE.

Exercice 2 (3 pts)

On considère un modèle de logique modale contenant les mondes A, B, C et D. La relation d'accessibilité est définie par : R(A,D), R(A,C), R(A,B), R(B,A), R(B,C), R(B,D), R(D,C).

1. Représenter graphiquement les mondes et l'accessibilité entre les mondes.
2. On considère une proposition p valide dans le monde A et dans le monde C. Indiquer dans quel(s) monde(s) $\Diamond p$ est valide. Pourquoi ?
3. Dans quel(s) monde(s) a-t-on $\Box p$ valide ? Pourquoi ?
4. A-t-on $\Diamond\Diamond p$ valide dans le monde A ? A-t-on $\Box\Diamond p$ valide dans le monde A ?

Exercice 3 (3 pts)

On donne les matrices de gain suivantes. Analyser chacune des situations et expliquer, s'il y a lieu, les stratégies dominantes, les équilibres de Nash et les cas où une stratégie collaborative serait plus intéressante.

(1) B / A	Stratégie personnelle	Stratégie collaborative
Stratégie personnelle	300 300	100 700
Stratégie collaborative	700 100	500 500

(2) B / A	Stratégie personnelle	Stratégie collaborative
Stratégie personnelle	10 10	30 100
Stratégie collaborative	100 30	50 50

Exercice 4 (4 pts)

On considère l'extrait de base de connaissance suivant basée sur l'ontologie FOAF :

```
:p1 a foaf:Person ;
    foaf:knows :p2 ;
    foaf:knows :p5 ;
    foaf:topic_interest :p8 .

// + n autres déclarations
 de personnes connaissant
 d'autres personnes
 et/ou s'intéressant à d'autres
 personnes
```

```
:gia04 a foaf:Group ;
       foaf:name
            "groupe UV IA04 de l'UTC" ;
       foaf:member :p1 ;
       foaf:member :p2 .

 // + p autres déclarations de groupes et de
 personnes membres de ces groupes
```

On interroge cette base de connaissance. Dans les requêtes demandées on ne spécifiera pas les valeurs des préfixes.

a. Ecrire une requête SPARQL donnant les couples de personnes qui se connaissent l'une l'autre.

b. Ecrire une requête SPARQL donnant les couples de personnes dont l'une connaît l'autre et qui ne sont pas membres d'un même groupe.

c. Ecrire une requête SPARQL donnant les personnes et les groupes dont sont membres les personnes auxquelles elles s'intéressent.

d. Ecrire une requête SPARQL donnant les groupes ayant au moins deux personnes qui se connaissent et dont le nom évoque une UV de l'UTC.

A faire sur une copie séparée.

Exercice 5 (6 pts) : Modélisation de systèmes complexes

Suite aux précipitations ayant marqué ce début d'année, la ville de Compiègne souhaite mettre en place un système multi-agents pour la prévention d'inondations, notamment en cas de crue de l'Oise. Il s'agit de réguler le niveau de l'Oise en ouvrant ou en fermant le barrage. On vous fournit les agents suivants :

- 2 agents permettent de mesurer le niveau de l'Oise grâce à des capteurs. L'un est placé en amont du barrage (pour l'agent nommé *OiseAmontAgent*), l'autre en aval (pour l'agent nommé *OiseAvalAgent*). La constante LIMIT_INONDATION contient la valeur correspondant au niveau critique de crue.
- 50 capteurs sont placés dans les rues de la ville les plus exposées à l'inondation lors d'une crue, et mesurent le niveau d'eau par rapport au sol. Grâce aux résultats fournis par ces capteurs, l'agent *CarteAgent* établit une cartographie des inondations. Plus précisément, il calcule le niveau d'inondation sur une échelle de 0 (pas d'inondation) à 10 (danger direct pour la population) sur les zones *amontCompiegne* et *avalCompiegne*, qui correspondent aux zones de la ville situées au nord et au sud du barrage.
- L'agent *BarrageAgent* permet d'ouvrir et de fermer le barrage pendant un laps de temps x (exprimé en minutes). Lorsque le barrage est ouvert, le niveau en amont diminue alors que le niveau en aval augmente. Lorsqu'il est fermé, c'est le niveau en amont qui augmente alors que celui en aval diminue.

Vous souhaitez développer l'agent *RegulationAgent*. Cet agent communiquera avec les agents *OiseAmontAgent*, *OiseAvalAgent*, *CarteAgent* et *BarrageAgent* pour réguler automatiquement le niveau de l'Oise par rapport aux critères suivants :

- La priorité, lorsque la situation est normale, est que le seuil critique de crue de l'Oise LIMIT_INONDATION ne soit atteint ni en amont, ni en aval de l'écluse.
- Si l'Oise est en crue (le seuil critique est dépassé en amont comme en aval de l'écluse), alors le système doit réguler le niveau de l'eau afin que le niveau d'alerte soit le plus proche possible entre les zones *amontCompiegne et avalCompiegne* (on privilégie des niveaux d'alerte de 5 en amont et aval plutôt que de 0 et 10, puisque 10 correspond à un danger important pour la population).

1. Définir l'ensemble des communications possibles entre le *RegulationAgent* et les autres agents du SMA. Pour chacune des communications, vous prendrez soin d'indiquer le performatif du message, ainsi qu'un exemple cohérent de contenu au format JSON.
2. Donner, en pseudo-code, les algorithmes des behaviours de l'agent *RegulationAgent* qui permettent au système de répondre aux besoins énoncés.