

IPST-CNAM
Systèmes répartis
NFP 214
Jeudi 11 Février 2010

Sans document
Durée : **2 heures**
Enseignants : LAFORGUE Jacques

1^{ère} Session SMB 214

(COURS)

CORRECTION

1. QCM (60 points)

Mode d'emploi :

Ce sujet est un QCM dont les questions sont de 3 natures :

- **les questions à 2 propositions**: dans ce cas une seule des 2 propositions est bonne.
 - +1 pour la réponse bonne
 - -1 pour la réponse fausse
- **les questions à 3 propositions** dont 1 seule proposition est bonne
 - + 1 pour la réponse bonne
 - -1/2 pour chaque réponse fausse
- **les questions à 3 propositions** dont 1 seule proposition est fausse
 - + 1/2 pour chaque réponse bonne
 - -1 pour la réponse fausse

Il s'agit de faire une croix dans les cases de droite en face des propositions.

On peut remarquer que cocher toutes les propositions d'une question revient à ne rien cocher du tout (égal à 0).

Si vous devez raturer une croix, faites-le correctement afin qu'il n'y ait aucune ambiguïté.

N'oubliez pas d'inscrire en en-tête du QCM, votre nom et prénom.

Vous avez droit à **4 points** négatifs sans pénalité.

NOM:	PRENOM:
------	---------

Un Middleware est, :		Q 1
1	dans une architecture client-serveur classique, une API assurant la communication entre le client et le serveur	X
2	dans une architecture 3-tiers Web, une API assurant la communication entre le client et le serveur	
3	dans une architecture répartie, un ORB assurant la communication entre le client et les serveurs	X

Dans une architecture répartie, il est toujours possible de centraliser les informations		Q 2
1	OUI	X
2	NON	

Le middleware est constitué de un ou plusieurs composants logiciels se trouvant "au-dessous" de l'appliquatif, "au-dessus" du système d'exploitation et "entre" deux logiciels ayant besoin de communiquer entre eux		Q 3
1	OUI	X
2	NON	

Une architecture 3-Tiers est une architecture qui est composée de trois APIs utilisées, respectivement, par le client, le middleware et le serveur. Elles assurent ainsi la communication entre le client et le middleware, puis entre le middleware et le serveur		Q 4
1	OUI	
2	NON	X

Ces 2 architectures sont des modèles d'architecture répartie N-tiers :		Q 5
1	OUI	
2	NON	X

		Q 6
1	Physiquement, un système réparti n'est pas un système centralisé	X
2	Du point de vue de l'utilisateur, un système réparti est souvent perçu comme un système centralisé	X
3	Physiquement, un système réparti est un système dont les données sont centralisées dans une base de données	

	En programmation distribuée, un objet distribué est un objet instancié côté serveur, puis envoyé au client via un socket de communication	Q 7
1	OUI	
2	NON	X

	En RMI (Remote Method Invocation) de Java,	Q 8
1	la classe d'appartenance d'un objet distribué, hérite de <code>UnicastRemoteObject</code> et implémente une interface qui décrit les méthodes distantes	X
2	la classe d'appartenance d'un objet distribué, hérite de <code>RemoteObject</code> et implémente l'interface <code>Remote</code>	

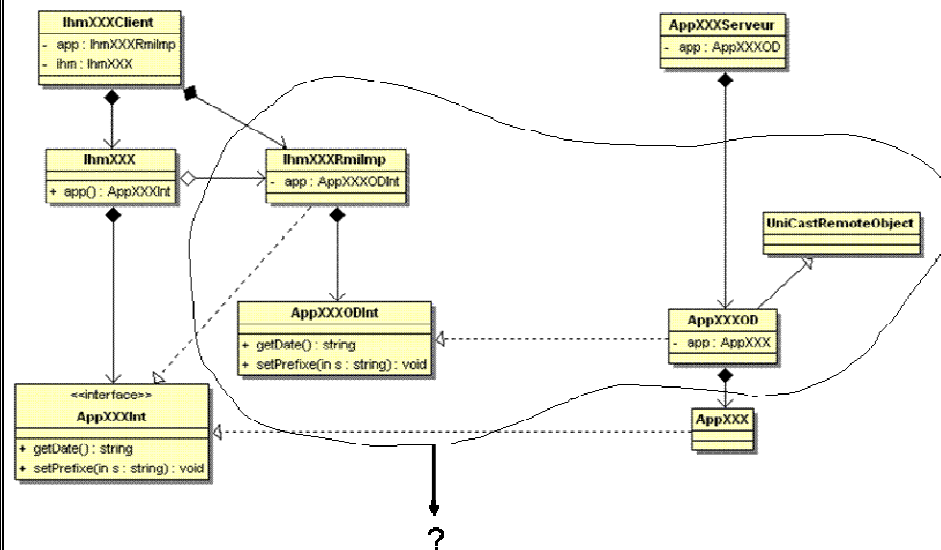
	Soit le code suivant : <pre>public class OD extends UnicastRemoteObject implements Inter {public int x; }</pre> que manque-t-il dans l'écriture de cette classe qui soit indispensable :	Q 9
1	Le constructeur de la classe	
2	L'enregistrement de l'OD dans le service de nommage	
3	L'implémentation des méthodes citées dans l'interface <i>Inter</i> .	X

	Soit le même code suivant : <pre>public class OD extends UnicastRemoteObject implements Inter {public int x; }</pre> que manque-t-il dans l'écriture de cette classe qui soit indispensable et/ou qui permette d'imposer le numéro de port de l'objet distribué	Q 10
1	Le constructeur de la classe	X
2	L'enregistrement de l'OD dans le service de nommage	
3	L'implémentation des méthodes citées dans l'interface <i>Inter</i> .	X

	Que fait ce code : <pre>public class QueFaitCeCode{ public static void main(String args[]){ String t[] = Naming.list("rmi://" + args[0] + ":" + args[1]); for(String e:t){System.out.println(e);} } }</pre>	Q 11
1	Affiche les noms de tous les objets distribués de la JVM (qui héritent de <code>UnicastRemoteObject</code>)	
2	Affiche les noms de tous les objets distribués créés par le factory, situé sur la machine de nom args[0] et de nom args[1],	
3	Affiche tous les noms de tous les objets distribués enregistrés dans un adaptateur RMI situé sur la machine args[0] et dont le numéro de port est args[1]	X

Ceci est le schéma d'architecture de l'atelier 16 dans sa version RMI qui représente une application composé d'un client IHM et de son serveur.

Q 12



Que représente la partie entourée ?

1	Les classes générées par l'IDL du standard RMI-CORBA qui permettent à l'IHM (IhmXXX) de communiquer avec l'Applicatif (AppXXX)	
2	Les classes Java de gestion de l'IHM	
3	Un middleware logiciel qui rend transparent pour l'IHM (IhmXXX) l'utilisation distante des méthodes de l'applicatif (AppXXX)	X

Les avantages d'un système réparti :

Q 13

1	un système réparti est plus à même d'être sécurisé qu'un système non réparti	
2	un système réparti est adapté pour donner une qualité de service identique tant local que distant	X
3	dans un système réparti, il est envisageable de continuer un service même dégradé	X

L'indépendance de la situation géographique d'une ressource informatique est une propriété fondamentale des systèmes répartis

Q 14

1	OUI	X
2	NON	

L'appelle d'une méthode distante peut se dérouler de la manière suivante :

Q 15

1	le client se connecte sur le port du service de nommage qui traduit les requêtes reçues en autant d'appels aux méthodes distantes de l'objet distribué situé sur un serveur métier	
2	le client se connecte sur le port d'un serveur de socket qui traduit les requêtes reçues en autant d'appels de méthodes locales au serveur	X
3	le client se connecte sur le port 8080 d'un serveur HTTP qui traduit les requêtes reçues en autant d'appel aux méthodes distantes de l'objet distribué situé sur le serveur métier	X

SOAP (Simple Object Access Protocol) est un standard de communication permettant de réaliser l'appel de méthode distante (RPC) sur le web

Q 16

1	OUI	X
2	NON	

Dans un système réparti, la transparence d'accès concerne :

Q 17

1	l'accès de l'utilisateur et son authentification aux ressources réparties sur le réseau	
2	l'accès, tant local que distant, des ressources réparties sur le réseau	X

Un Factory est		Q 18
1	une usine de fabrication de liens de persistance entre les objets et leurs représentations en base de données	
2	une usine de fabrication d'interfaces de communication entre les clients et le serveur	
3	un objet distribué qui crée, à la demande, d'autres objets distribués	X

CORBA (Common Object Request Broker Architecture) est		Q 19
1	une API Java permettant la communication de données à travers un réseau	
2	une norme de Middleware dédié à la communication de données à travers un réseau	X
3	un Framework Java et C++ de développement de solution informatique orientée Object Request	

Pour répartir la charge CPU des traitements de requête des objets distribués, sur l'ensemble des machines d'un réseau donné, il faut :		Q 20
1	que chaque machine possède un service de nommage afin que les objets distribués se désenregistrent et s'enregistrent ailleurs quand il change de place	
2	que les objets distribués soient dans un factory afin qu'ils puissent se déplacer de factory en factory	X

L'indépendance de la situation géographique d'un objet distribué passe par la mise en place d'un service de nommage qui mémorise les paramètres de connexion de cet objet distribué.		Q 21
1	OUI	X
2	NON	

<p>Cette représentation UML est celle de l'atelier 16 dans le cas d'une solution RMI. Elle décrit la dépendance des classes et des interfaces entre un client IHM et un serveur RMI.</p> <p>Le rôle de l'interface AppXXXInt est essentiel car :</p>		Q 22
1	Elle décrit les méthodes que la classe IhmXXXRmiImp et AppXXX doivent implémenter	X
2	Elle rend transparent, pour l'IHM, l'implémentation de l'applicatif	X
3	Elle décrit les méthodes distantes de l'objet distribué AppXXXOD	

En JAVA, avec RMI, les paramètres d'une méthode distante d'un objet distribué		Q 23
1	peuvent être de type primitif (int, double, char, ...)	X
2	leurs classes d'appartenance doit implémenter l'interface Serializable	X
3	leurs classes d'appartenance peuvent être une classe dérivée de InputStream et OutputStream	

En JAVA, avec RMI, plusieurs clients d'un objet distribué peuvent utiliser en parallèle une même méthode distante		Q 24
1	OUI	X
2	NON	

En CORBA, appeler les méthodes distantes d'un objet distribué, consiste, pour le client, à appeler les méthodes du proxy récupéré à partir de la référence d'objet CORBA (org.omg.CORBA.Object) de l'objet distribué		Q 25
1	OUI	X
2	NON	

Toutes les méthodes distantes d'un objet distribué		Q 26
1	doivent appartenir à la même interface	
2	peuvent appartenir à plusieurs interfaces	X

Pour utiliser les méthodes distantes d'un objet distribué, il doit être enregistré dans un adaptateur		Q 27
1	OUI	
2	NON	X

En Java, la classe ServerSocket :		Q 28
1	est instanciée à chaque traitement, par le serveur, lors de l'appel d'une méthode distante	
2	permet de traiter les requêtes, envoyées par un client RMI, CORBA, ou HTTP	X
3	est créé par la classe UnicastRemoteObject afin de traiter les requêtes, envoyé par un client RMI	X

On a une base de donnée installée sur une machine A qui persiste les attributs d'un objet distribué qui est créé sur la machine A. On a un servlet d'un serveur HTTP qui s'exécute sur la machine B On a un applet qui s'exécute dans un navigateur sur une machine C. Les machines A et B sont sur un réseau local sécurisé où seul le port 8080 est autorisé. La machine C est un poste quelconque qui est connecté sur le réseau Internet.		Q 29
1	L'applet se connecte à l'objet distribué	
2	Le servlet se connecte à l'objet distribué	X
3	L'applet se connecte au serveur HTTP	X

Dans une architecture distribuée, les données échangées du serveur vers le client sont des objets qui sont passés par copie		Q 30
1	OUI	X
2	NON	

Dans une architecture distribuée, les données échangées du serveur vers le client peuvent être des amorces (ou proxy) d'objets distribués		Q 31
1	OUI	X
2	NON	

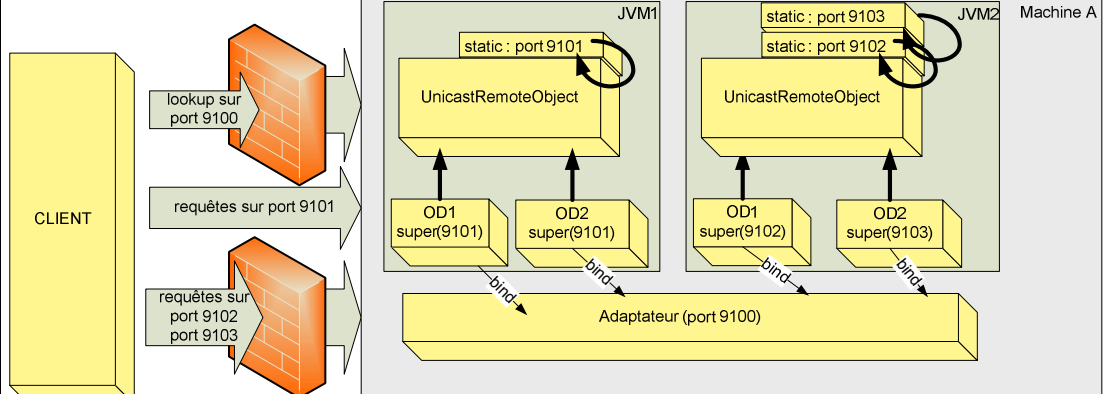
En CORBA, il est possible de faire une programmation basée sur la propagation d'évènement. La création de plusieurs canaux d'évènement permet d'optimiser les performances.		Q 32
1	OUI	X
2	NON	

Soit, un IDL qui contient une interface de nom <i>InterfaceDeviseOD</i> . Cet IDL génère, en Java, une interface utilisée pour implémenter les méthodes distantes. Le nom de cette interface est :		Q 33
1	InterfaceDeviseODPOA	
2	InterfaceDeviseODOperations	X
3	InterfaceDeviseODHolder	

Pour utiliser les méthodes distantes d'un objet distribué :		Q 34
1	on peut demander à un adaptateur le proxy de connexion de l'objet distribué	X
2	on peut demander à un factory le proxy de connexion de l'objet distribué	X
3	on peut demander à un IDL le proxy de connexion de l'objet distribué	

En Java dans CORBA, l'IOR (Interface Object Request) est l'interface java de description des méthodes distantes de l'objet distribué		Q 35
1	OUI	
2	NON	X

Dans le cadre de la communication entre un composant Java et un composant C++ sur un bus CORBA		Q 36
1	on doit créer un socket de communication dans chacun des composants pour les faire communiquer	
2	on peut exécuter les deux composants sur la même machine	X
3	on définit un IDL qui réalise une projection Java et une projection C++ des composants logiciels utilisés pour faire communiquer les deux composants	X

 <p>Ce schéma représente le fonctionnement des serveurs de socket des classes UnicastRemoteObject utilisées dans des programmes Java RMI.</p>		Q 37
1	On peut créer un nouvel OD dans la JVM1 qui s'exécute sur le port 9101	X
2	On peut créer une nouvelle JVM3 dans laquelle, on crée un nouvel OD qui s'exécute sur le port 9103	
3	Dans la JVM2, on peut créer un nouvel objet distribué RMI sur le port 9104	X

En CORBA, un IOR est créé par le serveur pour chaque objet distribué créé		Q 38
1	OUI	X
2	NON	

En CORBA, si on déplace géographique un objet distribué, il est inutile de recréer l'IOR afin que les clients puissent se reconnecter		Q 39
1	OUI	
2	NON	X

Le Domain Name System (DNS) n'est pas un système réparti mais un système à plat de l'ensemble des adresses des domaines et centralisé sur un serveur unique appelée DNSS (Domain Name System Server)		Q 40
1	OUI	
2	NON	X

En CORBA, la redondance, en cas de panne, d'un objet distribué se fait, notamment, en réenregistrant une copie de cet objet distribué dans le service de nommage		Q 41
1	OUI	X
2	NON	

En RMI, le code suivant est un exemple de création d'un objet distribué (<i>HelloOD</i> hérite de <i>UnicastRemoteObject</i>) : <pre> HelloOD od = new HelloOD("Pierre","DUPONT"); Naming.rebind("rmi://localhost:9999/HELLO",od); </pre>		Q 42
1	OUI	X
2	NON	

En CORBA, en cas de redondance d'un objet distribué, les clients doivent rafraichir leurs connexions en demandant la nouvel valeur de l'IOR de l'objet distribué redondé		Q 43
1	OUI	X
2	NON	

Un client qui appelle une méthode distante synchrone , interdit à tout autre client d'utiliser les autres méthodes de l'objet distribué tant qu'il n'a pas fini d'exécuter la méthode		Q 44
1	OUI	
2	NON	X

Pour pouvoir accéder de manière distante aux attributs d'un objet :		Q 45
1	les attributs doivent être publiques	
2	chaque attribut doit être associé à une méthode distante d'accès et/ou de mise à jour	X
3	les méthodes d'accès et/ou de mise à jour de ces attributs doivent être publiques	X

Dans le protocole RMI la communication entre deux objets distribués se fait en utilisant la technologie des sockets		Q 46
1	OUI	X
2	NON	

En JAVA, la persistance des objets en base de données se fait en utilisant le protocole RMI		Q 47
1	OUI	
2	NON	X

En RMI de Java, l'instruction lookup :		Q 48
1	permet de connecter le client RMI à un serveur de socket géré par <i>UnicastRemoteObject</i> qui transforme les requêtes reçues en l'appel de méthodes distantes	X
2	retourne un objet qui est une instance d'une classe qui hérite de <i>UnicastRemoteObject</i>	
3	interroge le service de nommage de l'existence d'un objet distribué et récupère un stub de connexion	X

En RMI de Java, l'instruction lookup peut s'utiliser par l'objet distribué lui-même pour se connecter à lui-même ou à un autre objet distribué.		Q 49
1	OUI	X
2	NON	

Pour mettre en place une redondance chaude d'un nœud d'un système réparti, il est :		Q 50
1	préférable que le nœud maitre et le nœud esclave soient enregistrés dans le service de nommage sous le même nom afin de faciliter le basculement des clients	X
2	préférable que le nœud maitre et le nœud esclave soient sur la même machine	
3	indispensable que le nœud maitre et le nœud esclave soient créés par le même factory	

En CORBA, un IOR (Interoperable Object Référence) contient les informations suivantes :		Q 51
1	le nom de l'objet distribué; l'adresse IP de la machine hôte du service de nommage; le port du service de nommage; une clé désignant l'objet	
2	le nom complet de l'interface IDL de l'objet ; l'adresse IP de la machine hôte de l'objet ; le port du serveur de l'objet ; une clé désignant l'objet	X
3	le nom du fichier .idl de description de l'interface de l'objet; l'adresse IP locale; le port du POA; la description des méthodes distantes	

En RMI de Java, il est possible de transformer une classe quelconque en une classe d'objets distribués même si celle-ci hérite déjà d'une autre classe autre que UnicastRemoteObject		Q 52
1	OUI	X
2	NON	

La sérialisation est un principe informatique qui permet de plier et déplier les attributs de l'objet distribué afin de les rendre accessibles de manière distante		Q 53
1	OUI	
2	NON	X

En CORBA, pousser un évènement par le producteur revient à appeler la méthode <i>push</i> du consommateur		Q 54
1	OUI	X
2	NON	

L'IDL (Interface Définition Language) est un langage de programmation informatique qui permet de créer les traitements des applications distribuées		Q 55
1	OUI	
2	NON	X

La référence d'un objet CORBA (org.omg.CORBA.Object) :		Q 56
1	est créé par le service de nommage	
2	est créé à partir de l'IOR	
3	identifie de manière unique un servent du bus CORBA	X

Dans une architecture répartie, le service de nommage est un "chef d'orchestre" qui centralise la localisation des objets distribués répartis sur le réseau		Q 57
1	OUI	X
2	NON	

Dans une architecture répartie, tous les objets distribués répartis sur le réseau doivent être enregistrés dans le service de nommage		Q 58
1	OUI	
2	NON	X

La tolérance aux pannes d'un servent dans une architecture répartie passe, notamment, par		Q 59
1	la mise en place du doublement du réseau	
2	la mise en place de l'indépendance géographique du servent pour les utilisateurs	X
3	la mise en place de liens de réplication des ressources critiques du servent sur un autre servent	X

Un objet passé en paramètre d'une méthode distante est reçu par l'appelant :		Q 60
1	sous la forme d'une référence d'un objet distant	
2	sous la forme d'un nouvel objet qui est une copie du paramètre	X

2. Questions libres (20 points)

Chaque question est notée sur 4 points.

Vous répondez à ces questions sur une copie vierge en mettant bien le numéro de la question, sans oublier votre nom et prénom.

Q 1

Nous voulons mettre en place une répartition de la charge CPU des objets distribués entre différentes machines du réseau.

Expliquez comment vous vous y prenez.

Correction :

L'objectif est de répartir la création des objets distribués sur les différentes machines du réseau. Pour cela, nous créons et installons un factory sur chaque machine. Le rôle de ces factories est de créer à la demande les objets distribués.

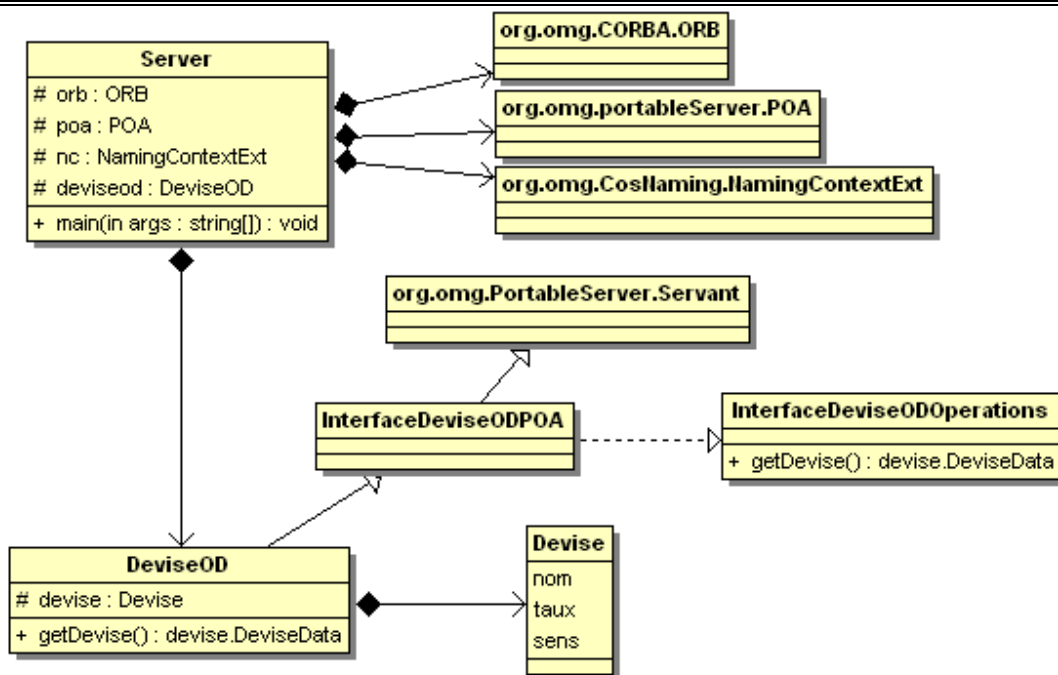
Lors de la demande de création, le factory interroge les autres factories afin de connaître leur nombre d'objets distribués. Si un de ces factory a moins d'objets distribués alors il lui demande de créer l'objet distribué sinon il le crée lui-même.

Chaque factory connaît les autres factory grâce au service de nommage dans lequel ils s'enregistrent tous.

Chaque factory implémente deux méthodes distantes :

- la méthode qui retourne le nombre d'objets distribués créés dans le factory
- la méthode qui crée un objet distribué.

Q 2



Ce schéma décrit les dépendances des composants du serveur CORBA développé dans l'atelier 20.

Quel est le rôle de chacun de ces composants ?

Commentez chacun des liens de dépendance.

Correction :

Server est le programme principal serveur dans lequel sont créés les servants via les éléments prédéfinis de l'ORB (POA, Service de Nommage, ...)

org.omg.CORBA.ORB est la classe de l'ORB dont les méthodes vont permettre de gérer le bus CORBA.

org.omg.portableServer.POA est le Portable Object Adapter, le service CORBA de création et gestion des servants

org.omg.CosNaming.NamingContextExt est l'interface de service du Service de Nommage CORBA. Ce composant permet de créer les noms logiques des servants et de les enregistrer dans l'annuaire.

DeviseOD est notre objet distribué (service de Devise). Il encapsule l'objet **Devise**.

Devise est l'objet informatique encapsulé par **DeviseOD** afin d'utiliser de manière distante ses méthodes.

InterfaceDeviseODPOA est la classe de POA générée par l'IDL. Cette classe contient les mécanismes d'appel distant des méthodes de l'objet distribué qui en hérite.

InterfaceDeviseODOperations est l'interface générée par l'IDL qui contient la déclaration des méthodes distantes de l'interface IDL

org.omg.PortableServer.Servant est l'interface prédéfinie de l'ORB dont toutes les interfaces d'IDL doivent hériter.

Les relations sont, de gauche à droite et de haut en bas :

- Server crée et/ou utilise l'ORB, le POA et le Service de nommage
- Server crée et utilise l'objet distribué (le servant de devise)
- InterfaceDeviseODPOA hérite de Servant (ce code est généré par l'IDL)
- InterfaceDeviseODPOA doit implémenter les méthodes distantes de InterfaceDeviseODOperations qui, ne les implémentant pas, doivent être implémentées par la classe DeviseOD
- DeviseOD hérite de InterfaceDeviseODPOA et doit donc implémenter les méthodes de InterfaceODOperations
- DeviseOD crée et utilise Devise. Elle permet d'utiliser les méthodes de Devise de manière distante.

Q 3

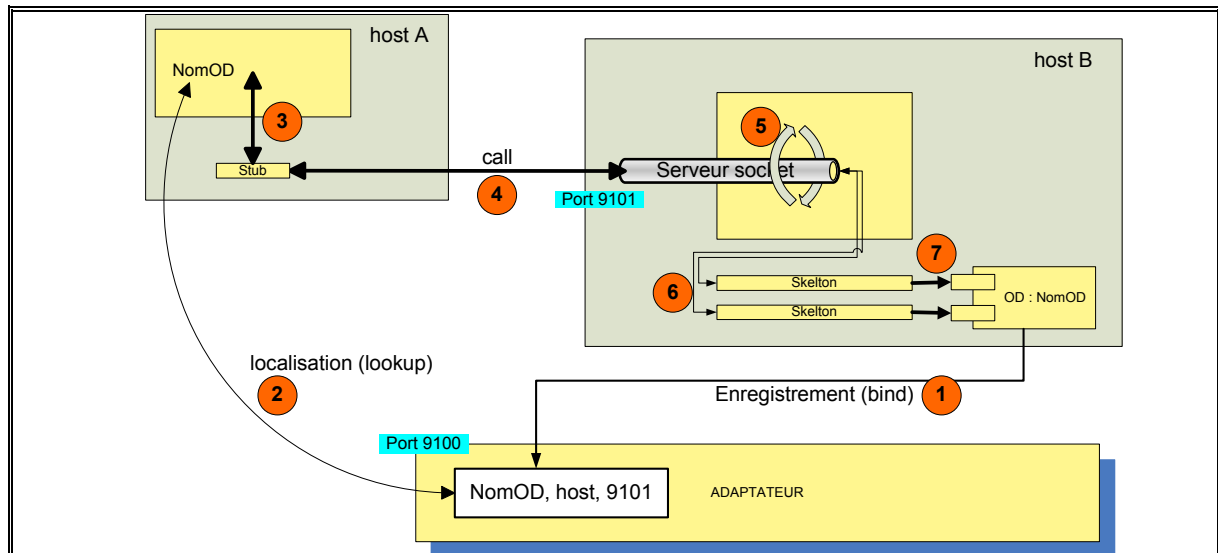
Quels sont les avantages d'un système réparti ?

Correction :

Les avantages d'un système réparti sont :

- partager et répartir des ressources et des services (nœuds) sur un réseau informatique
- faire du load-balancing
- paralléliser les algorithmes de calcul avec des environnements d'exécution spécifique
- continuer un service globalement même dégradé suite à la perte de nœuds
- indépendance de la localisation géographique des ressources
- intégré des composants informatiques hétérogènes en OS et en langages de programmation
- continuité de service pendant la maintenance (remplacement d'un nœud)
- scalabilité : adaptabilité à une forte croissance des besoins informatiques d'une entreprise
- contrôle des accès concurrents aux ressources

(Tourner la page)



Commentez chacun des points numérotés.

Correction :

Point 1: Le serveur enregistre l'OD dans l'annuaire

Point 2 : le client récupère le stub de l'OD via la connexion à l'annuaire sur le port 9100. Le stub contient la référence du port 9101 pour se connecter à l'OD

Point 3: les appels aux méthodes distantes sont réalisés par le stub

Point 4: les appels à ces méthodes sont autant de requêtes qui transitent vers l'OD via le port 9101

Point 5: le serveur de socket sur le port 9101 traite les requêtes reçues

Point 6 : Chacune de ces requêtes sont prises en charge par les skeltons afin d'appeler les méthodes de l'OD

Point 7: Les skeltons appellent les méthodes de l'OD.

(Fin du sujet)