Conception d'application répartie

© Olivier Caron

Polytech'Lille

Conception d'application répartie

√ Problème : définir une application complexe mettant en jeu plusieurs sites, plusieurs acteurs, plusieurs applications

Conception d'application répartie

- √ Problème : définir une application complexe mettant en jeu plusieurs sites, plusieurs acteurs, plusieurs applications
- √ Solution :
  - ▶ Spécifier l'architecture logicielle globale avant toute implémentation

Conception d'application répartie

- √ Problème : définir une application complexe mettant en jeu plusieurs sites, plusieurs acteurs, plusieurs applications
- √ Solution :
  - Spécifier l'architecture logicielle globale avant toute implémentation
  - Limiter les dimensions technologiques

Spécification de l'architecture globale

√ Fonctions de l'application :

- Identification des acteurs
- Identification des fonctions (cas d'utilisation UML)

Spécification de l'architecture globale

- √ Fonctions de l'application :
  - Identification des acteurs
  - Identification des fonctions (cas d'utilisation UML)
- √ Répartition :
  - Identification des sites et/ou types de site
  - Rôle de chaque site
  - Communication inter-sites

Dimension technologique

- √ Quelle technologie ? (EJB, .Net, etc)
- √ Critères (interopérabilité, performances, ...)

Spécification au niveau d'un site

√ Découpage composant :

- Types de composants (session, web, entité, ...)
- Quels composants visibles ?
- Quels composants internes ?
- Comment accèder aux composants ?

Spécification inter-site

- √ Communication entre composants inter-sites
- √ Comment accèder aux composants :
  - Spécifier et introduire un ou plusieurs serveurs de noms
  - Décrire une procédure d'enregistrement des composants à ce(s) serveur(s)

Un exemple

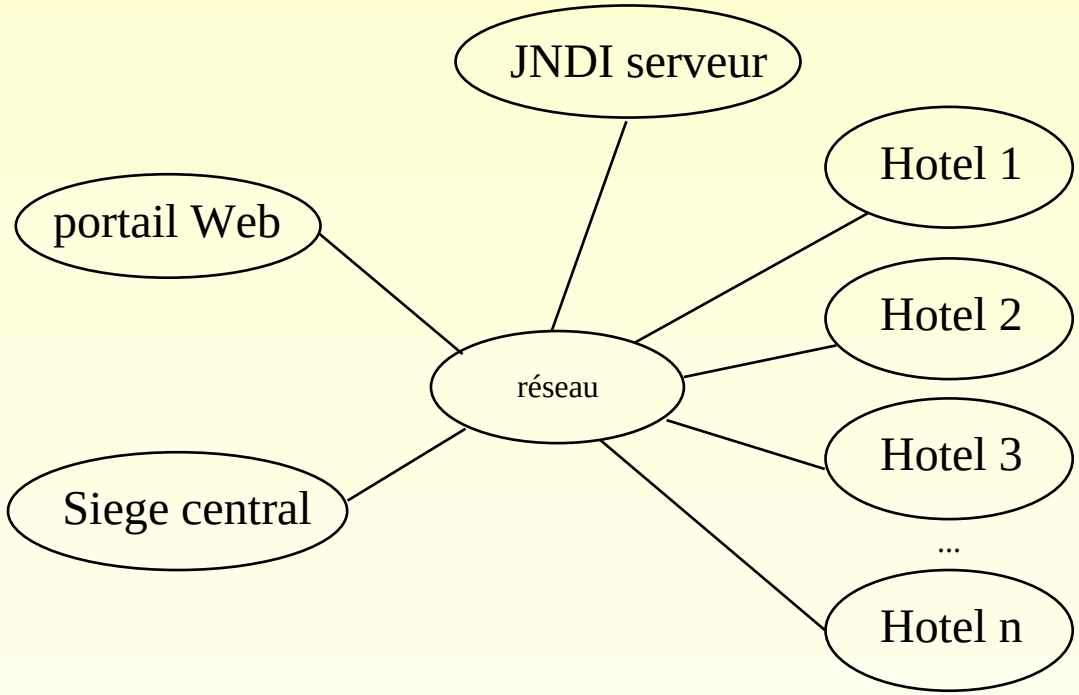
On désire concevoir une architecture client/serveur multi-sites qui permet de gérer une chaîne d'hôtels. C'est à la charge de chaque hôtel de gérer localement la réservation de ses chambres. Chaque hôtel dispose d'un nom unique et doit permettre la réservation d'une chambre à une date donnée, indiquer le nombre de chambres de l'hôtel, le nombre de chambres occupées à une date donnée, le prix d'une chambre (pour simplifier, on considère qu'il n'y a qu'un seul type de chambre).

Le siège central de la chaîne d'hôtels connaît bien sûr l'ensemble de ses hôtels et effectue des statistiques globales sur son réseau notamment le taux d'occupation global pour une date donnée (rapport nombre de chambres occupées/nombre de chambres pour l'ensemble des hôtels).

Un client peut se déplacer dans un hôtel pour demander une réservation dans cet hôtel.

Un serveur web unique existe aussi sur le réseau pour la chaîne d'hôtels et permet la réservation et la consultation d'une chambre d'un des hôtels de la chaîne.

Une solution possible



Au niveau de chaque hôtel

√ Développement d'une application J2EE contenant :
Un composant entité (accès local) pour la gestion des chambres
Un composant session (accessible à distance)

Au niveau de chaque hôtel

- √ Développement d'une application J2EE contenant :
  Un composant entité (accès local) pour la gestion des chambres
  Un composant session (accessible à distance)
- √ Le composant session utilise le serveur de noms JNDI partagé par tout le réseau et s'inscrit à l'adresse symbolique :
  "nomChaineHotel/nomHotel"

Le composant session Hotel

```
-----------------------------------------------------
interface HotelRemoteHome extends EJBHome{
  HotelRemote create() throws RemoteException, CreateException ;
}
-----------------------------------------------------
interface HotelRemote extends EJBObject {
  double getPrixChambre() throws RemoteException  ;
  int  getNbreChambres() throws RemoteException ;
  /** @return le numéro de la chambre réservée **/
  int  reserver(Date d) throws RemoteException,
                               ReservationImpossibleException
  int getNbreChambresOccupees(Date d) throws RemoteException ;
}
```

Le portail Web et le siège central

√ Le portail Web :

- Déploiement d'une application J2EE dotée de composants JSP qui accèdent aux différents composants sessions des différents hôtels
- Les composants webs exploitent le serveur partagé JNDI pour accéder à tel ou tel hôtel.
- Le siège central
- Réalisation d'une application cliente qui consulte le serveur JNDI pour connaître l'ensemble des hôtels et questionner chaque hôtel

Exemple d'utilisation générale de JNDI (1/2)

```
import javax.naming.*;
public class JNDIlist {
 public static void main(String[] args) {
  try {
   Context ctx = new InitialContext();
   if (args.length == 0) listContext (ctx,"");
   else
    for (int i=0; i<args.length; i++) listContext (ctx,args[i]);
  } catch (NamingException ex) {
    ex.printStackTrace(); System.exit(1);
  }
 }
```

Exemple d'utilisation générale de JNDI (2/2)

```
public static void listContext (Context ctx, String subctx)
 throws NamingException {
 System.out.println("Listing Context: "+subctx);
 NamingEnumeration list = ctx.list(subctx);
 while (list.hasMore()) {
  NameClassPair item = (NameClassPair)list.next();
  String cl = item.getClassName();
  String name = item.getName();
  System.out.println(cl+" - "+name);
 }
}
}
```

Variante possible :

- Au niveau de chaque hôtel :
  - un composant session dédié aux fonctions de réservations (portail web)
  - un composant session dédié aux fonctions du siège central