Systèmes d'Information Coopératifs et Répartis

Université Lyon 1
M2 CCI Réseaux

L. Médini, février 2006

Plan des cours

- Protocole HTTP et programmation serveur
- Architectures multi-composants réparties
  - Principes
  - Exemples (CORBA, RMI, RMI/IIOP, J2EE)
- Objets distribués (Javabeans, EJB)
- Web services (SOA, WSDL, SOAP, UDDI)
- Projet

Les architectures réparties

- objets distribués
  Approche orientée objet
  Architectures distribuées
  } objets distribués
  - Exemples d'application : agence de voyage en ligne
  - Exemples d'objets distribués
    - Logique applicative client (connexion, recherche commandes)
    - Gestion sécurisée des paiements
    - Gestion des réservations
    - Interrogation des fournisseurs de voyages
    - ...

Les architectures réparties

- Infrastructures *middleware*
  - But : gestion des communications entre les objets hétérogènes *via* le réseau dans les architectures distribuées
  - Exemples
    - CORBA (OMG)
    - RMI (Java)
    - RMI/IIOP (Java)
    - J2EE (Sun)
    - .Net (MicroSoft)

Les architectures réparties

- CORBA
  - But
    - Communication entre objets hétérogènes et distants
    - Invocation de « services » entre objets d'applications distribuées
  - Principes généraux
    - Séparation stricte Implémentation/Interface
    - Localisation transparente des objets
    - Accès transparent aux objets
  - Caractéristiques techniques
    - Architecture client/serveur
    - Objets distribués ou non
    - Multi-plateforme

Les architectures réparties

- CORBA

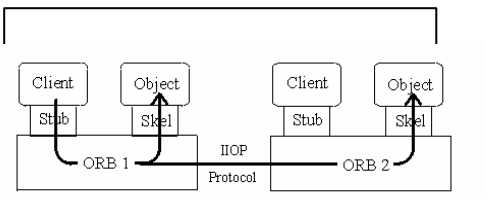
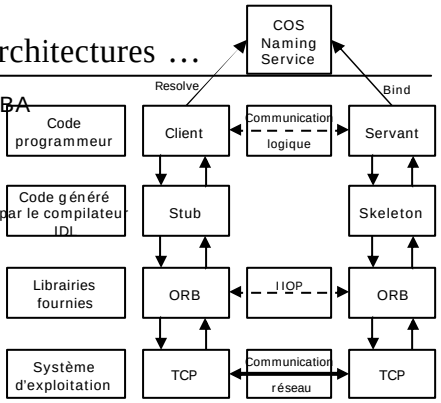


Figure 2: Interoperability uses ORB-to-ORB communication

Les architectures …

- CORBA



Les architectures réparties

- CORBA
  - Langages/protocoles
    - IDL : langage neutre de spécification d'interfaces
      - Représentation des interfaces (informations échangées) dans le même langage
      - Traduction (projection) des interfaces dans différents langages (C, C++, SmallTalk, Ada95 et Cobol OO, Java, Eiffel et Common Lisp)
      - Traduction effectuée à la compilation (statique)
      - Utilise un compilateur IDL spécifique au langage utilisé pour produire
        - Un stub/skeleton lié à un ORB
        - Une description des interfaces des services

```
Interface Chat {
  void SetMessage (in string auteur, in string texte);
}
```

Les architectures réparties

- CORBA
  - Langages/protocoles
    - IIOP : protocole de communication entre ORB
      - Transmission des messages
      - Implémentation de GIOP sur TCP
      - Échange de message entre « ORB »
    - DII : accès dynamique aux serveurs
      - Permet de « court-circuiter » un ORB
      - Découverte dynamique de nouveaux objets
      - Construction et distribution d'invocation
      - Réception de réponses

Les architectures réparties

- CORBA
  - Objets/librairies
    - ORB : Couche « communication » intégrée aux objets
      - Responsable des mécanismes nécessaires pour
        - Trouver l'implémentation de l'objet pour la requête
        - Préparer cette implémentation à recevoir la requête
        - Communiquer les données constituant la requête
      - L'interface que voit le client est indépendante
        - De l'endroit où l'objet est situé
        - Du langage dans lequel l'objet est implémenté
      - Un ORB contient
        - Une interface IDL
        - Un support au service de nommage COS
        - Un support IIOP

Les architectures réparties

- CORBA
  - Objets/librairies
    - Stub : proxy client
    - Skeleton : proxy serveur

    } s'interfacent avec un ORB
  - CORBA COS : services objets communs
    - Naming Service : service de nommage permettant de retrouver les objets servants pour les clients
    - Life Cycle Service
    - Object Transaction Service
    - Security Service
    - …

Les architectures réparties

- CORBA
  - Exemple d'utilisation
    - Fichier Chat.idl : `interface Chat { void setMessage (in string auteur, in string texte);}`
    - Compilation : `idlj –fallTIE Chat.idl`
      - ⇒Interface que doit implémenter le servant :

```
public interface ChatOperations {
  void setMessage (String auteur, String texte); }
```

      - ⇒Interface distante que doit implémenter le servant :

```
public interface Chat extends ChatOperations,
org.omg.CORBA.Object, org.omg.CORBA.portable.IDLEntity {
```

      - ⇒Classe skeleton : `Chat_Tie`
      - ⇒Classe stub : `ChatStub`
      - ⇒Classe contenant des méthodes auxiliaires : `ChatHelper`

Les architectures réparties

- CORBA
  - Exemple d'utilisation
    - À Programmer

```
class ChatServant implements ChatOperations {
  void setMessages (String auteur, String texte){…} }
class ChatServer {
  public static void main (String args[]){…} }
class ChatClient {
  public static void main (String args[]){…} }
```

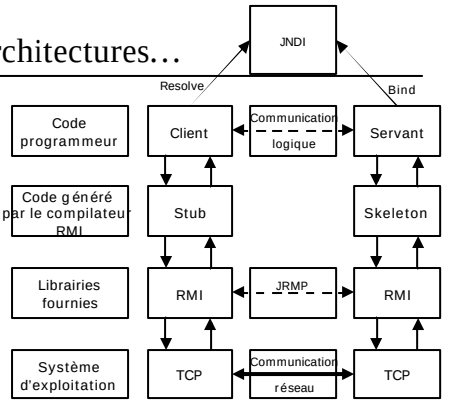
    - Lancement
      - Serveur de noms (machine m1) : `tnameserv` (depuis JDK 1.3)
      - Serveur (machine m2) : `java ChatServer –ORBInitialHost m1`
      - Client(machine m3) : `java ChatClient –ORBInitialHost m1`

Les architectures réparties

- RMI
  - Mêmes principes de base que CORBA
  - Limité à Java (objets non hétérogènes)
  - Plus de langage de spécification d'interfaces
  - Protocole de communication : JRMP
- RMI/IIOP
  - Protocole de communication : IIOP
  - Permet l'interopérabilité entre RMI et CORBA

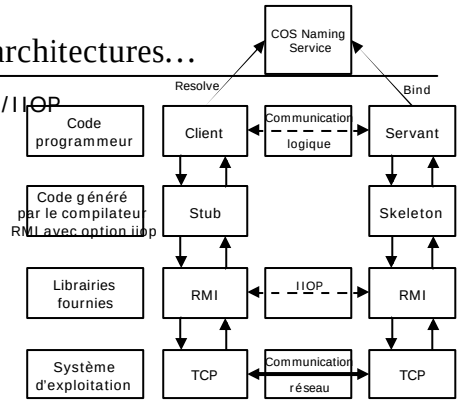
Les architectures…

- RMI



Les architectures…

- RMI/IIOP



Les architectures réparties

- RMI
  - Exemple d'utilisation
    - Interface Java :

```
package monchat;
import java.rmi.*;
public interface Chat extends Remote {
public void setMessage (String auteur, String texte)
throws RemoteException;}
```

    - Serveurs de noms :
      - `rmiregistry` (méthodes `lookup()` et `bind()`)
    - Compilation : `rmic monchat.Chat`
      - ⇒Classe skeleton : `ChatServant_Skel`
      - ⇒Classe stub : `ChatServant_Stub`

Les architectures réparties

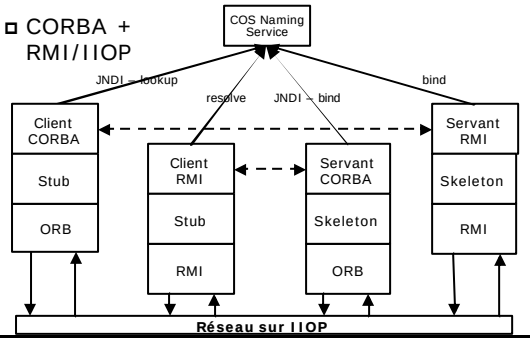
- RMI
  - Exemple d'utilisation
    - À Programmer

```
class ChatServant extends UnicastRemoteObject
  implements Chat {
  void setMessages (String auteur, String texte){…} }
class ChatServer {
  public static void main (String args[]){…} }
class ChatClient {
  public static void main (String args[]){…} }
```

    - Lancement
      - Serveur de noms (machine m1) : `rmiregistry`
      - Serveur (machine m2) : `java -Djava.rmi.server.codebase=http://m1/ ChatServer`
      - Client(machine m3) : `java -Djava.rmi.server.codebase=http://m1/ ChatClient`

Les architectures réparties

- CORBA + RMI/IIOP



Les architectures réparties

- RMI/IIOP
  - Exemple d'utilisation (différences avec RMI)
    - Implémentation du servant

```
class ChatServant extends PortableRemoteObject implements
  Chat {
  void setMessages (String auteur, String texte){…} }
```

    - Transtypage complexe
    - Compilation : `rmic –iiop monchat.Chat`
    - Packages à importer
      - Javax.rmi (servant, serveur, client)
      - Javax.naming (serveur, client)
    - Lancement
      - Serveur de noms (machine m1) : `tnameserv`
      - Serveur (machine m2) :
        `java -Djava.rmi.server.codebase=http://m1/ ChatServer`
      - Client(machine m3) :
        `java -Djava.rmi.server.codebase=http://m1/ ChatClient`

Les architectures réparties

- Infrastructures *middleware* : conclusion
  - Géré : accès aux services transversaux
    - Nommage (serveur de noms)
    - Transactions
    - Persistance
    - Sécurité…
  - Non géré : optimisation des accès aux ressources
    - Pools de connexion ou de threads
    - Activation et désactivation des objets
    - Répartition de la charge
    - Tolérance aux pannes

Les architectures réparties

- Infrastructures *middleware* : conclusion
  - Principes fondamenaux
    - Objets client, servant et serveur
    - Invocation d'objets distants transparente pour le client
    - Échange de messages conformes à des descriptions d'interfaces
    - Service de nommage
  - Limites
    - Requiert la programmation du serveur
    - Requiert l'inscription dans le serveur de noms
    - Interfaces Générées à la compilation (CORBA non-DII)

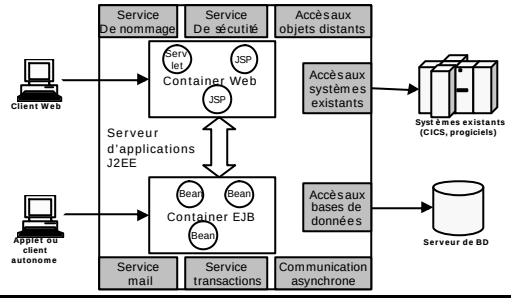
⇒ Mise en place « lourde » pour le développeur

Les architectures réparties

- L'architecture J2EE
  - But : développement, déploiement et exécution des applications distribuées
  - Mêmes principes de base que les infrastructures middleware …
    - Communication synchrone (RMI/IIOP) et asynchrone (JMS) entre objets distribués, serveurs de nom (JNDI), intégration d'objets CORBA (JavaIDL)…
  - …plus de nombreux services techniques supplémentaires
    - Génération d'objets transactionnels distribués (EJB), gestion du cycle de vie des objets, gestion des transactions (JTA et JTS), sécurité, accès aux BD (JDBC), interfaces graphiques dynamiques (Servlets et JSP)…

Les architectures réparties

- L'architecture J2EE



Les architectures réparties

- Références
  - CORBA
    - http://www.omg.org
    - http://corba.developpez.com/cours/
    - http://corba.developpez.com/presentation.htm
  - RMI et RMI/IIOP
    - http://java.sun.com/products/jdk/rmi/
    - http://java.sun.com/j2se/1.4.2/docs/api/
  - Exemples de code RMI et RMI/IIOP
    - http://java.sun.com/developer/codesamples/index.html
    - http://thomasfly.com/RMI/rmi_tutorial.html
    - http://www-128.ibm.com/developerworks/java/rmi-iiop/space.html