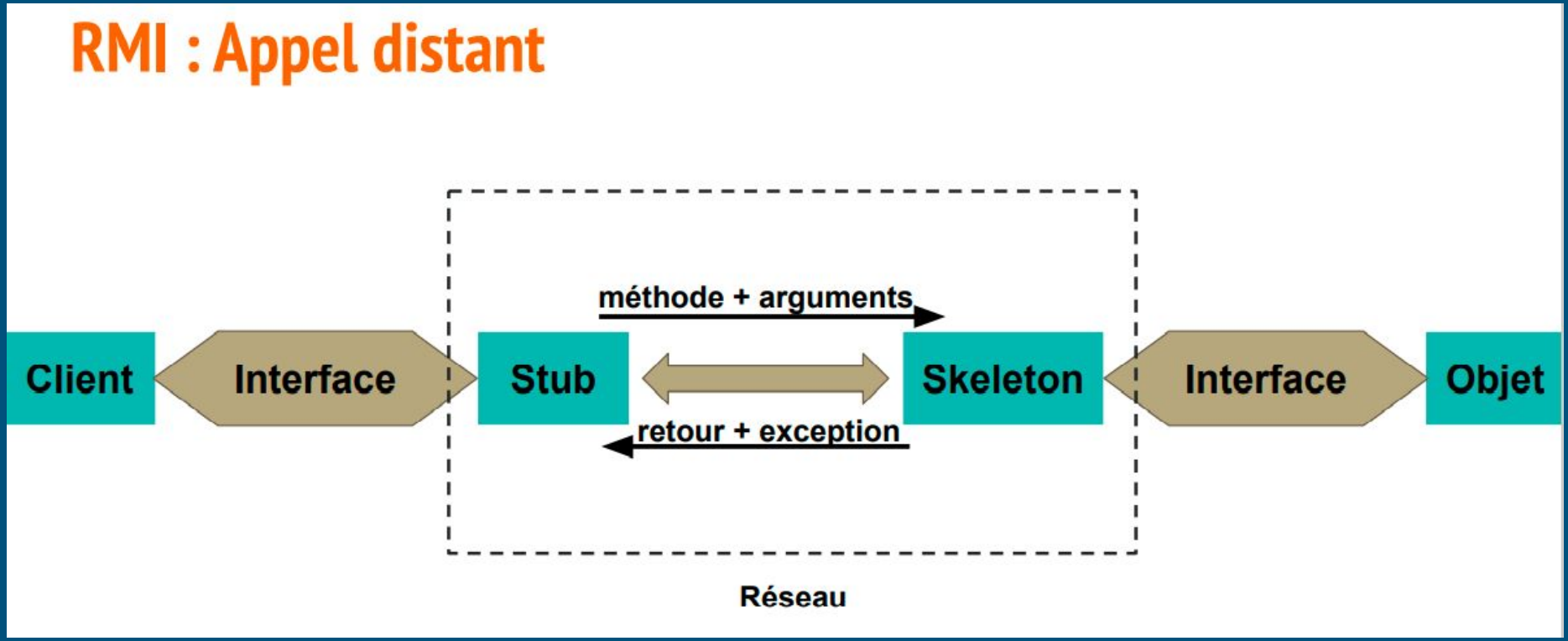
III. Système RMI

- RMI (Remote Method Invocation) est une API Java permettant de manipuler des objets distants (c'est-à-dire un objet instancié sur une autre machine virtuelle, éventuellement sur une autre machine du réseau) de manière transparente pour l'utilisateur, c'est-à-dire de la même façon que si l'objet était sur la machine virtuelle (JVM) de la machine locale.

- Grâce à RMI, un objet exécuté dans une JVM présente sur un ordinateur (côté client) peut invoquer des méthodes sur un objet présent dans une autre JVM (côté serveur). RMI crée un objet serveur distant public qui permet les communications côté client et côté serveur via des appels de méthode simples sur l'objet serveur.

III. Système RMI



III. Système RMI

Objet stub & skeletoon

- Le stub (souche) et le skeleton (squelette), respectivement sur le client et le serveur, assurent la conversion des communications avec l'objet distant.
- Objet stub : L'objet stub sur la machine cliente crée un bloc d'informations et envoie ces informations au serveur. Le bloc se compose de :

  ➔ Un identifiant de l'objet distant à utiliser

  ➔ Nom de la méthode à appeler

  ➔ Paramètres de la JVM distante

III. Système RMI

Création d'applications distribuées à l'aide de RMI, les étapes (côté serveur) :

- **Étape 1 : Définir l'interface distante** : La première chose à faire est de créer une interface qui fournira la description des méthodes pouvant être invoquées par les clients distants. Cette interface doit extends l'interface Remote,le prototype de la méthode dans l'interface doit déclencher throws l'exception RemoteException.

III. Système RMI

Création d'applications distribuées à l'aide de RMI, les étapes (côté serveur) :

- **Étape 2 : Implémentation de l'interface distante** : la deuxième étape consiste à implémenter l'interface distante. Pour implémenter l'interface distante, la classe doit s'étendre à la classe UnicastRemoteObject du package java.rmi. En outre, un constructeur par défaut doit être créé pour lancer l'exception java.rmi.RemoteException à partir de son constructeur parent dans la classe

III. Système RMI

Création d'applications distribuées à l'aide de RMI, les étapes (côté serveur) :

- **Étape 3** : L'écriture d'une classe pour instancier l'objet et l'enregistrer dans le registre Cette étape peut être effectuée dans la méthode main d'une classe dédiée ou dans la méthode main de la classe de l'objet distant. L'intérêt d'une classe dédiée et qu'elle permet de regrouper toutes ces opérations pour un ensemble d'objets distants..

III. Système RMI

Création d'applications distribuées à l'aide de RMI, les étapes (côté Client) :

- **Étape 1** : La mise en place d'un security manager. Comme pour le côté serveur, cette opération est facultative.

III. Système RMI

Création d'applications distribuées à l'aide de RMI, les étapes (côté Client) :

- **Étape 2:** L'obtention d'une référence sur l'objet distant Pour obtenir une référence sur l'objet distant à partir de son nom, il faut utiliser la méthode statique lookup() de la classe Naming.
- Cette méthode attend en paramètre une URL indiquant le nom qui référence l'objet distant. Cette URL est composée de plusieurs éléments : le préfix rmi://, le nom du serveur (hostname) et le nom de l'objet tel qu'il a été enregistré dans le registre précédé d'un slash.

III. Système RMI

Création d'applications distribuées à l'aide de RMI, les étapes (côté Client) :

- **Étape 3:** Appel de(s) méthode()s distante(s),le client détient une référence à un objet distant qui est une instance de la classe de stub. L'étape suivante consiste à appeler la méthode sur la référence. Le stub implémente l'interface qui contient donc la méthode que le client a appelée.

  ➔ il suffit simplement d'appeler la méthode comme suit : nom_interface.nom_methode(paramètres)