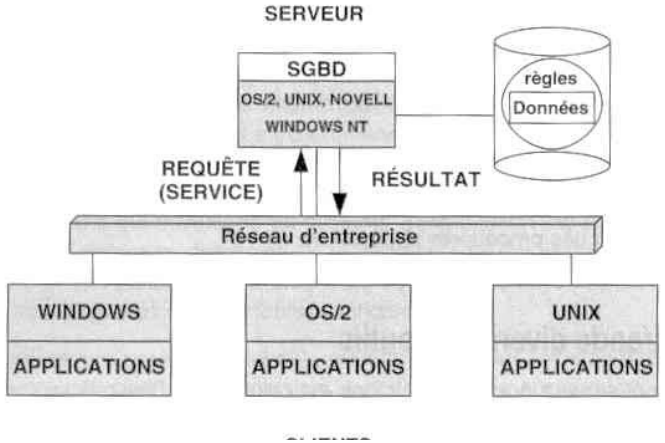
Historique

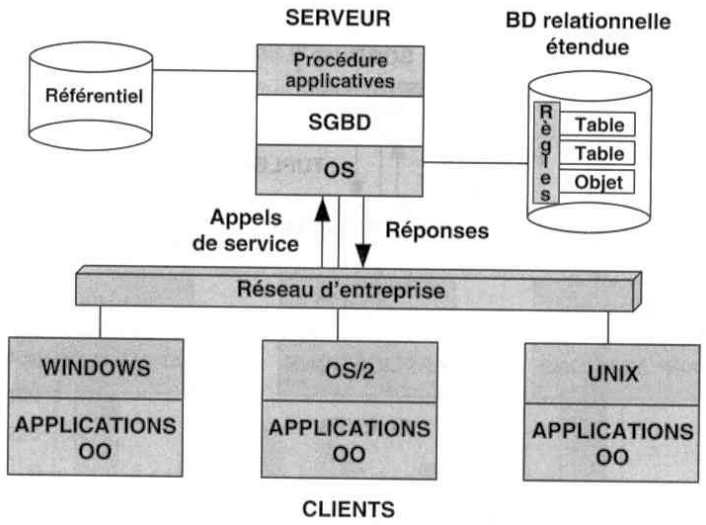
- Avant 1980
  - Les architectures étaient centralisées autour de calculateurs centraux *(mainframe)* de type IBM ou BULL par exemple
- Les années 80
  - Les années 80 ont connu le développement du transactionnel.
  - Les réseaux, notamment les réseaux locaux se sont développés
  - les micro-ordinateurs
- Les années 90 (client serveur)

L'APPROCHE C/S

- **Définition: Architecture client-serveur**
  - Modèle d'architecture applicative où les programmes sont répartis entre processus clients et serveurs communiquant par des requêtes avec réponses.
- **première génération** (**on** née à la fin des années 80)
  - est basée sur des outils clients autour des SGBD relationnels.
  - Le développement s'effectue sur le serveur pour la base de données et sur le client pour l'application.



- 2me génération

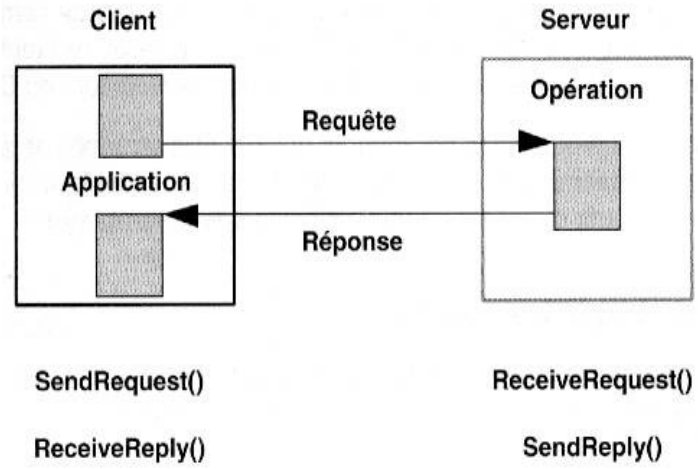


Techniques de communication Client -Serveur

- Le client-serveur est avant tout une technique de dialogue entre deux processus, l'un -le client- sous-traitant à l'autre -le serveur -des fonctions à réaliser. Nous allons dans cette section étudier plus en détail ce mode de dialogue.
- **Notion 1 : Client *(Client)***
- Processus demandant l'exécution d'une opération à un autre processus par envoi d'un message contenant le descriptif de l'opération à exécuter et attendant la réponse à cette opération par un message en retour.
- **Notion 2 : Serveur *(Server)***
- Processus accomplissant une opération sur demande d'un client et transmettant la réponse à ce client.
- **Notion 3 :Requête (Request)**
- Message transmis par un client à un serveur décrivant l'opération à exécuter pour le compte du client.
- **Notion 4 :Réponse(Reply)**
- Message transmis par un serveur à un client suite à l'exécution d'une opération contenant les paramètres de retour de l'opération.

Modèles de communication entre applications

- Protocoles de type question-réponse



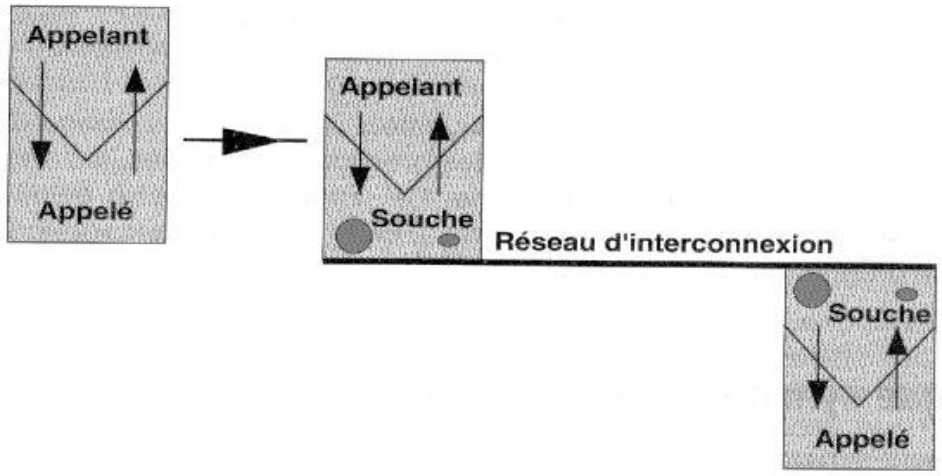
- Des opérations de transport permettant d'envoyer (Send) et de recevoir *(Receive)* des messages sont typiquement utilisées à cette fin. Les protocoles de question-réponse peuvent s'implémenter au-dessus d'une couche session. Celle-ci doit alors être établie entre le client et le serveur par des primitives de connexion *(Connect),* puis de déconnexion *(Disconnect)* en fin de session. Ils peuvent aussi être implémentés directement au-dessus d'un service de datagrammes, qui permet d'envoyer des messages à des ports associées aux clients et au serveur.

2 Assemblage-désassemblage des paramètres

- Lors de l'émission d'une requête, les paramètres doivent être arrangés et codés sous forme de message: c'est **l'assemblage.** A l'arrivée, ils doivent être remis en format interne de manière symétrique à partir du message reçu: c'est le **désassemblage.**
- **Assemblage *(Marshalling)*** : Procédé consistant à prendre une collection de paramètres et à les arranger et coder en format externe pour constituer un message à émettre.
- **Désassemblage (Unmarshallîng):** Procédé consistant à prendre un message en format externe et à reconstituer la collection de paramètres qu'il représente en format interne.

Appel de procédure à distance

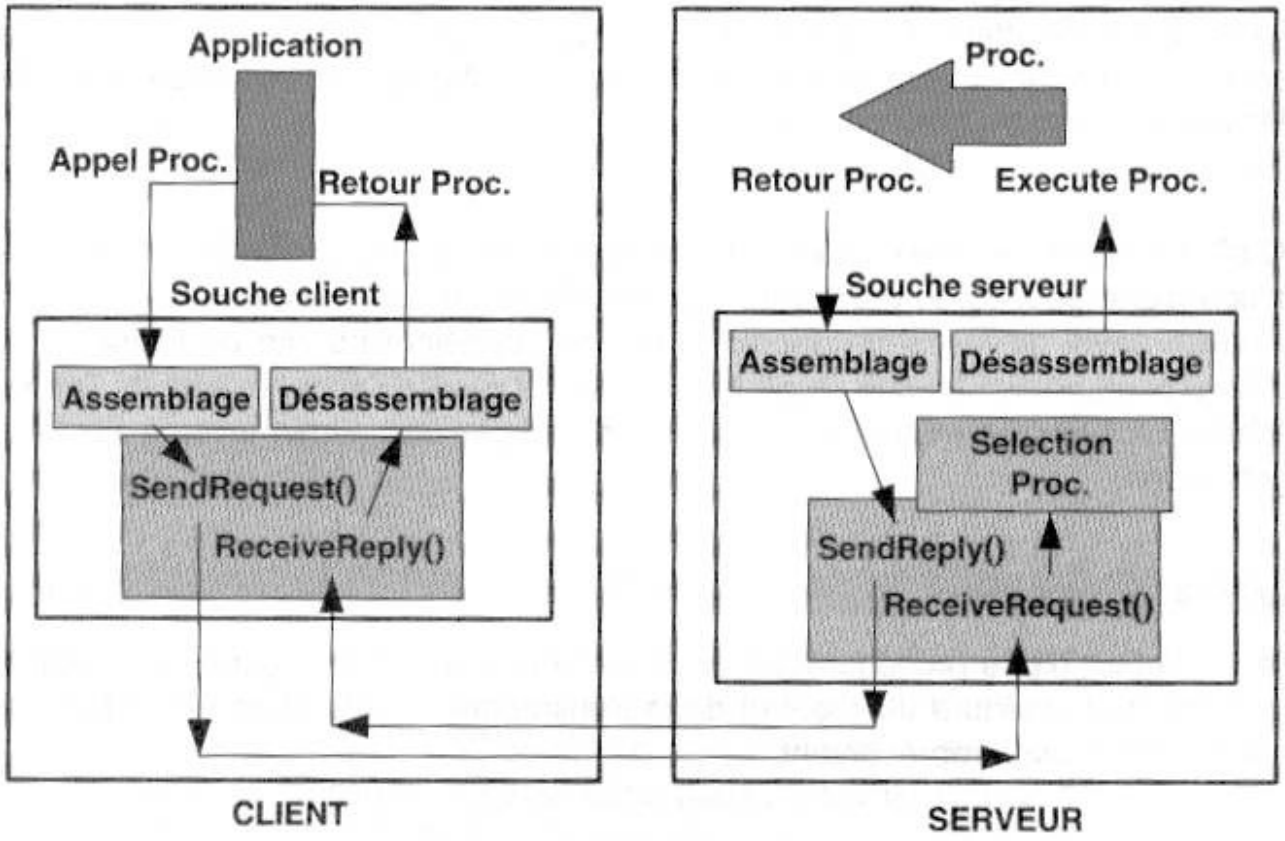
- Dans un système centralisé, l'appel d'opération (fonction ou procédure) s'effectue directement par un débranchement depuis l'appelant vers l'appelé. Ceci n'est plus possible dans un environnement distribué. Afin de rendre transparent le dialogue client-serveur, la technique d'appel de procédure à distance (RPC) a été introduite.



Appel de procédure à distance

- **Appel de procédure à distance (Remote Procedure Call RPC)**
- Technique permettant d'appeler une procédure distante comme une procédure locale, en rendant transparents les messages échangés et les assemblages/ désassemblages de paramètres.
- Le RPC offre les fonctions de l'appelé à l'appelant sur le site de ce dernier. Il est réalisé par introduction d'une souche de procédure (en anglais, stub) pour transformer l'appel de procédure en un envoi de message depuis le site de l'appelant au site de l'appelé. Là, une souche de procédure symétrique reçoit le message et réalise l'appel effectif de l'appelé par débranchement. Les paramètres de retour transitent par un chemin inverse via le réseau d'interconnexion .
- **Souche (Stub):** Représentant d'une procédure sur un site client ou serveur capable de recevoir un appel de procédure du client et de le transmettre en format adapté à l'implémentation ou à son représentant.

Appel de procédure à distance



Appel de procédure à distance

- Les souches peuvent être générées automatiquement à partir d'un langage de description d'interface, permettant de spécifier les noms des procédures appelées, et les types des paramètres d'entrée et de sortie. Ce langage peut être défini comme une extension d'un langage existant (par exemple C), ou comme un langage indépendant de tout langage de programmation (par exemple *IDL Interface Definition Language).*

Dialogue synchrone et asynchrone

- **Dialogue synchrone (Synchronous dialog)** Type de dialogue géré sans file d'attente, où les commandes d'émission et de réception sont bloquantes.
- Typiquement, dans le cas synchrone, le client attend le serveur pendant que celui-ci exécute une opération pour lui.
- **Dialogue asynchrone (Asynchronous dialog)**

Type de dialogue géré avec file d'attente, où l'une au moins des commandes d'émission ou de réception est non bloquante.

Les Middlewares

- Ensemble des services logiciels construits au-dessus d'un protocole de transport afin de permettre l'échange de requêtes et des réponses associées entre client et serveur de manière transparente.
- L'objectif d'un médiateur (Middleware) est donc d'assurer une liaison transparente, c'est-à-dire de cacher l'hétérogénéité des composants mis en jeu. Il s'agit en particulier d'assurer la transparence aux réseaux, aux SGBD, et dans une certaine mesure aux langages d'accès.