Chapitre 2. Architecture des SDs

1. Architectures des SDs
2. Modèle de communication Client/Serveur
3. Modèle de communication peer to peer
4. Modèle à mémoire partagée

78

Chapitre 2. Architecture des SDs

Mots clés

Synchrone, Asynchrone , messages, Middleware, Peer-to peer, client serveur, multi-tiers,

79

79

Chapitre 2. Architecture des SDs

1. Architectures des SDs

- Ensemble d'entités logicielles communiquant entre-elles
- Entités logicielles s'exécutent sur des machines reliées entre elles par un réseau.
- Communication entre entités logicielles :

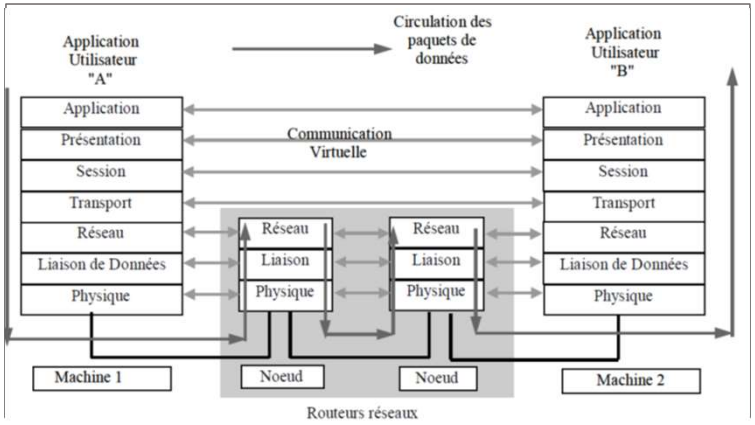
- Le plus basique : en appelant les services des couches TCP ou UDP
- Plus haut niveau : définition de couches offrant des services plus complexes (réalisée en s'appuyant sur les couches TCP/UDP)

Exemple de service : appel d'une procédure chez une entité Distante,

80

Chapitre 2. Architecture des SDs

1. Architectures des SDs



81

Chapitre 2. Architecture des SDs

1. Architectures des SDs

Réseaux TCP/IP exemple : les réseaux locaux, internet ...

- **Couche réseau : IP (Internet Protocol)**

Gestion des communications et connexions entre les machines à travers le réseau avec recherche des routes à travers le réseau pour accéder à une machine.

- **Couche transport**

{ **TCP** : connexion virtuelle directe et fiable entre 2 applications
ou **UDP** : mode datagramme : Envoi de paquets de données sans gestion de l'ordre d'arrivée, ni gestion des paquets perdus.

82

Chapitre 2. Architecture des SDs

2. Modèle de communication Client Serveur

- **Motivation:**

- Accepter des **postes clients de moindre capacité** (bande passante de communication, capacité de mémoire, capacité d'affichage (résolution, couleur), puissance de traitement, consommation).

- Permettre la **mobilité géographique** des postes clients

- Permettre la **modification** possible de l'environnement et de **la qualité de service** de la communication.

- **Réagir** à une augmentation de la charge (par exemple nombre de clients connectés)

83

83

Chapitre 2. Architecture des SDs

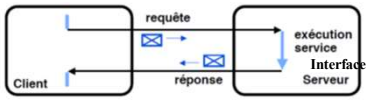
2. Modèle de communication C/S
Exemple : une BD en C/S

Problème : comment maintenir un **niveau acceptable de qualité de service** (par exemple temps de réponse moyen) ?

-Client et serveur (pas nécessairement **localisés** sur deux machines distinctes):

- Le **client demande** l'exécution d'un **service**

- Le **serveur réalise le service**



- **Indépendance interface-réalisation**

84

84

Chapitre 2. Architecture des SDs

2. Modèle de communication C/S
Exemple : une BD en C/S

Communication par messages (plutôt que par partage de données, mémoire ou fichiers)

Requête : paramètres d'appel, spécification du service requis

Réponse : résultats, indicateur éventuel d'exécution ou d'erreur.

85

85

Chapitre 2. Architecture des SDs

2. Modèle de communication C/S
Exemple : une BD en C/S synchrone

On peut implémenter une application C/S de nombreuses façons :

Monolithique : les différents niveaux sont imbriqués dans un programme unique

Deux couches (2 niveaux/ 2 tiers): presque obligatoire quand on travaille avec un SGBD

sur le serveur le SGBD (**D**), avec une partie de la logique applicative Traitement (**T**)(contraintes d'intégrité, procédures stockées, etc.)

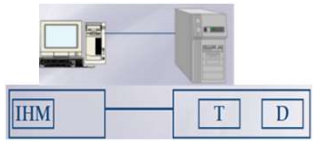
sur le client : présentation (**IHM**) et l'autre partie de l'application (**T**)

86

86

Chapitre 2. Architecture des SDs

2. Modèle de communication C/S
Exemple : une BD en C/S



Client léger Serveur

Client lourd Serveur

87

87

Chapitre 2. Architecture des SDs

2. Modèle de communication C/S
Exemple : une BD en C/S

Trois couches (ou plus) :

–Client : présentation (**IHM**) et partie des traitements et des objets métier (**T**)

–Serveur d'applications le reste de la logique applicative (**T**)

–Serveur de données héberge le SGBD / stockage des objets métier (**D**)

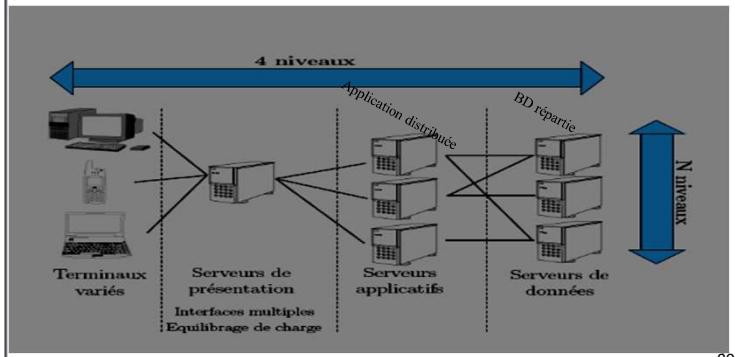
88

88

Chapitre 2. Architecture des SDs

2. Modèle de communication C/S
Exemple : une BD en C/S

Plus de trois couches :



89

89

Chapitre 2. Architecture des SDs

2. Modèle de communication C/S
Exemple : une BD en C/S

- Serveur de fichiers (AFS, NFS)
- Serveur d'impression (lpd)
- Serveur de calcul
- Serveur d'application (spécifique à l'application)
- Serveur de bases de données
- Serveur de temps
- Serveur de noms (annuaire des services)

90

90

Chapitre 2. Architecture des SDs

2. Modèle de communication C/S
Exemple : une BD en C/S

- **Structuration :**

–Découplage du code : séparation des interfaces service et réalisation, maintenance plus facile

–Client et serveur peuvent être modifiés (remplacés) indépendamment.

–Découplage des équipes: chacun son métier (maquette, administration du SGBD, etc.

91

91

Chapitre 2. Architecture des SDs

2. Modèle de communication C/S
Exemple : une BD en C/S

Répartition → équilibrage de la charge :

–serveur web (présentation) | serveur des objets métier | SGBD

- **Protection:** Client et serveur s'exécutent dans des domaines de protection différents
- **Gestion des ressources**

–Le serveur peut être partagé entre de nombreux clients

–En contrepartie, il doit assurer la gestion des ressources partagées

92

92

Chapitre 2. Architecture des SDs

3. Les SDs pair à pair (peer to peer)

paradigme de construction des systèmes distribués et des applications dans lesquelles les données et les ressources informatiques sont apportées par beaucoup de hôtes sur Internet, qui participent tous à la fourniture d'un service uniforme.

C 'est un modèle de réseau proche du modèle client-serveur mais où chaque client est aussi un serveur

Il peut être centralisé (les connexions passant par un serveur central intermédiaire) ou décentralisé (les connexions se faisant directement).

93

Chapitre 2. Architecture des SDs

3. Les SDs pair à pair (peer to peer)

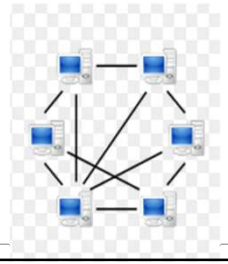
Les SD peer to peer exploitent les techniques de nommage existants , de routage, de réplication de données et de sécurité pour construire **une couche de partage de ressources fiable sur la base** d'une collection d'ordinateurs et de réseaux **non fiable** et non sécurisée.

Les applications peer-to-peer ont été utilisées pour fournir le partage de fichiers, la mise en cache Web, la distribution d'informations et d'autres services

Chapitre 2. Architecture des SDs

3. Les SDs pair à pair (peer to peer)

Les systèmes pair-à-pair permettent à plusieurs ordinateurs de communiquer via un réseau, en y partageant simplement des objets – des fichiers le plus souvent, mais également des flux multimédia continus (streaming), le calcul réparti, un service comme la téléphonie sur IP...



Chapitre 2. Architecture des SDs

3. Les SDs pair à pair (peer to peer)

Des propositions utilisant le modèle pair-à-pair ont été faites pour ne plus utiliser de serveurs, entre autres pour:

-les DNS (Distributed DNS);

-la mise à disposition de logiciels (distributions Linux comme la Mandriva, le système de mises à jour Microsoft Avalanche, World of Warcraft, etc.) ;

-diffuser des contenus multimédias (streaming) ;

-les logiciels de messagerie en ligne ;

-la téléphonie (les premières versions de Skype).

Chapitre 2. Architecture des SDs

3. Les SDs pair à pair (peer to peer)

L'application la plus connue actuellement reste cependant le partage de fichiers par le biais de protocoles comme Bittorent, eDonkey, FastTrack (KaZaA), etc.

Un problème clé pour les systèmes peer-to-peer est le placement d'objets de données à travers de nombreux hôtes et des dispositions subséquentes pour leur accès d'une manière qui équilibre la charge de travail et assure la disponibilité sans ajouter de frais généraux excessifs

4. Les mémoires partagées distribuées

Les systèmes à mémoire partagée distribuée (Distributed Shared Memory DSM) fournissent une abstraction pour le partage de données entre processus ne partageant pas la mémoire physique.

Les programmeurs se voient présenter une abstraction familière de la lecture ou de l'écriture des structures de données (partagées) comme si elles se trouvaient dans leurs propres espaces d'adresses locaux, présentant ainsi un niveau élevé de transparence de la distribution.

L'infrastructure sous-jacente de la DSM doit assurer la synchronisation des accès et des copies et la cohérence des données.

4. Les mémoires partagées distribuées

les processus exécutés sur différents ordinateurs observent toutes les mises à jour effectuées sur les données partagées