Technologie des applications client-serveur

RSX 102
Anne WEI
CNAM Paris

1

Historique de l'UE

- Cette UE a été créé par Professeur Gérard Florin en 2002
- Le but de cette UE a pour connaître le concept *client-serveur* et les applications aux couches hautes Internet

Organisation de l'UE

- 14 cours de 2 heures (le lundi: de 13h à 15h et de 18h30 à 20h30)
  - le concept client/serveur (A. Wei)
  - l'interface Socket TCP/UDP (A. Wei)
  - l'Appel de procédure à distance RPC et le système de fichier NFS (A. Wei)
  - DNS et SMTP (A. Wei)
  - La présentation de données – XML (F. Sailham)
  - Web Service (2 séances) et JSON (F. Sailham)
  - la présentation ASN1 et SNMP (F. Sailham)
  - LDAP (A. Wei)
- 12-14 ED de 2 heures (3 groupes: le lundi après-midi, le vendredi soir et le samedi matin)

3

Plan

1 Introduction
2 Modèle Client-serveur
3 Applications du client-serveur

4

Introduction (1)

- La technologie des applications client-serveur appartient à la technologie de *systèmes répartis* (ou systèmes distribués, *distributed systems* en anglais).
- *« Un système réparti est un système qui s'exécute sur un ensemble des machines sans mémoire partagée, mais que portant l'utilisateur voit comme une seul et unique machine » Tanenbaum, «systèmes d'exploitation» 1994*
- Le *système parallèle* utilise une unité commune pour effectuer le travail en parallèle et en coopération d'un nombre élève de processus

Andrew Start Tanenbaum

5

Introduction (2)

- Les caractéristiques de systèmes répartis :
- La mise en commun de unités permet d'augmenter la capacité de calcules
- L'accès à distance permet de développer des nombreuses applications telles que le guiche de banque et l'agence de voyage par exemple
- Le partage de ressources via un réseau commun (Internet, par exemple) offre les nouveaux services tels que Web (*World Wide Web*) services.
- Le bas coût de matériel et de processus
- …

6

Introduction (3)

- Les systèmes répartis consistent à
- assurer l'interopérabilité des composants (réseaux, logiciels, systèmes, langages)
- assurer la cohérence du résultat (opération idempotent)
- faciliter le développement. C'est-à-dire, le système en question doit être extensible (open source, par exemple)
- sécuriser l'accès à distance (voir l'UE RSX112)
- résister aux pannes matérielles et logicielles
- gérer la mobilité de systèmes (systèmes embarquées, par exemple)
- rassurer la transparents vis-à-vis des utilisateurs et des développeurs
- ….

7

Introduction (4)

- Le client-serveur
- Le système définit **des modèles de répartition**, **des données** et **des traitements** dans une **architecture réseau**
- **L'approche** principale: une approche dissymétrique. L'échange de messages entre client et serveur sont le type de communication « **requête-réponse** »
- Le rôle de client est d'initiateur du dialogue. C'est-à-dire, le client émet des requêtes.
- Le rôle de serveur est d'offrir un (ou plusieurs) service sur un réseau
- Des types de serveurs
- Serveur de fichiers
- Serveur d'impression
- Serveur de calcule
- Serveur de base de données
- Serveur de noms (annuaire des services)

8

Introduction (5)

- Les applications du système client-serveur
- Opération transactionnelle par OLTP (*On Line Transaction Processing*). Elle permet d'effectuer une transaction, une transaction bancaire par exemple. Les principales composants sont des accès à la base de données, des traitements et des modifications et des affichages en mode graphique.
- Aide à la décision par DS (*Decision Support*), la gestion des lignes Air France, par exemple. Les principales composants sont des accès à BD, des traitements de synthèse et de décision et des affichages.
- Les applications Internet : DNS (*Domaine Name System*), SMTP, HTTP…
- World Wide Service
- …

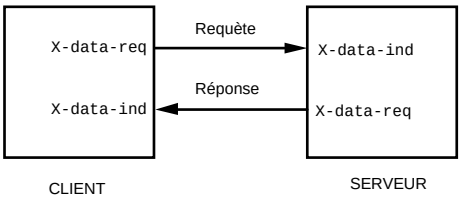
9

Plan

10

Modèle client-serveur (1)

- Le modèle client-serveur se base sur **le mode « message »**
- Le client envoie dans **un premier message une requête** d'exécution d'un traitement à un serveur
- Le serveur effectue le travail et fournit dans **un second message la réponse**
- Ce genre de communication par message de données en mode **asynchrone** est supportée par les protocoles de **transport**



Source : G. Florin

11

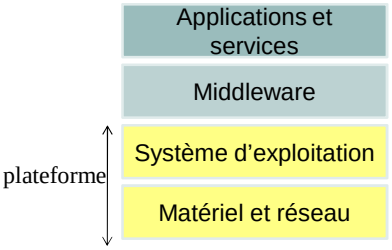
Modèle client-serveur (2)

- Le modèle client-serveur se base sur **l'appel de procédure distance**
- Le client (l'appelant) fait exécuter une procédure à distance par un site (le serveur)
- Le serveur (l'appelé) exécute la procédure
- Le mécanisme de procédure distance se base sur la sémantique et la présentation syntaxique.

12

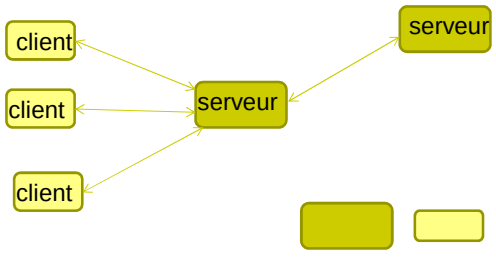
Rappel : Architecture en couche

- La couche Matériel et réseau supporte la communication entre les ordinateurs et le périphérique. Elle offre des services pour les couches hautes
- La couche Système d'exploitation assure le fonctionnement de systèmes
- La couche Middleware gère des objets et leurs fonctionnalités (Java, par exemple)
- La couche Application et services assurent les services d'applications



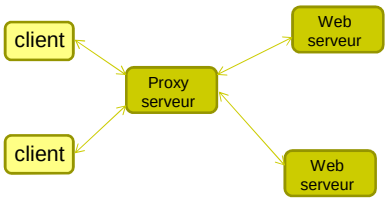
Architecture du modèle client-serveur (1)

- Le modèle client-serveur *mono-serveur*
  - les clients demandent des services en mode « message »
  - le serveur connecté offre le service demandé. Si le service demandé n'est pas local, le serveur demande *à son tour le service* en question en mode « message ». Dans ce cas, le serveur devient un client.
- un exemple, un serveur de fichiers locaux

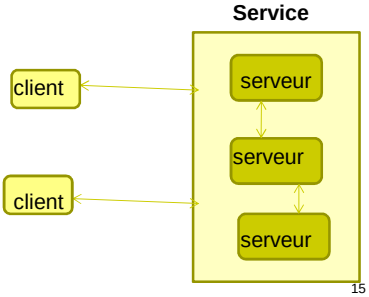


Architecture du modèle client-serveur (2)

- Le modèle client-serveur *multi-serveur*
  - les clients demandent un service en mode « message »
  - un service est assuré par un serveur ou par multi-serveurs
  - la redondance de données est utilisée pour assurer la tolérance du système
  - la cache est souvent utilisée pour garder des données et pour la sécurité
- un exemple, serveur Web



Modèle client-proxy/serveur

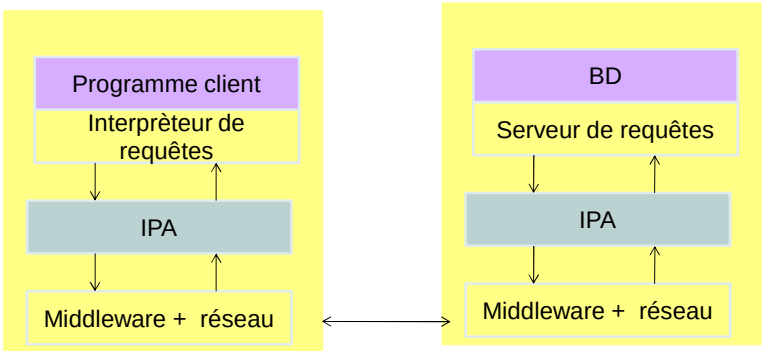


15

Modèle client-multiserveurs

Architecture du modèle client-serveur (3)

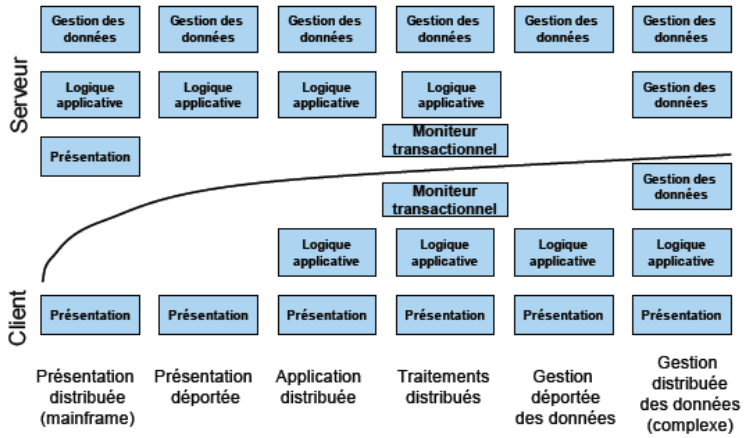
- IPA (*Interface de Programmation d'Application*) est nécessaire pour faciliter de programmer. Web service, par exemple



16

Architecture du modèle client-serveur (4)

- une référence : modèle du Gartner Group (ou Gartner)



17

Principe du modèle client-serveur

- **Du côté client**, il envoie (au moins) une requête et il attend une réponse.
- **Du côté serveur**, il faut gère
  - Gestion de l'accès aux données (priorité, par exemple)
  - Contrôle d'exécution (séquentiel, concurrent)
  - Contrôle des communications
  - Mémorisation ou non l'état de client
  - Sécurité du système

- Un exemple : envoyer des requêtes d'accès à des données (SQL) d'un client vers un serveur et obtenir les résultats.
  - **Établir une connexion** entre le client et le serveur
  - Envoie **des requêtes SQL** vers le serveur
  - **Conversion des formats** de requêtes
  - **Exécution** des requêtes
  - **Conversion des résultats**
  - (gestion des erreurs)
  - **Fermeture de connexion**

18

Outils

- le fonctionnement du modèle client-serveur se base sur les technologies différentes
  - Système d'exploitation local
  - Architecture de communication
  - Base de données
  - Moniteur transactionnel
  - Protocoles additionnels spécifiques

- Des outils
  - Les systèmes de gestion de base de données (SGBD) relationnels centralisés en accès distant (SQL- *Structued Query Language, par exemple*)
  - Les SGBD répartis en accès à distance ou SGBD relationnels objets (ORACLE, par exemple)
  - Les architectures de réseaux (Internet, par exemple)
  - Java (langage, plateforme, interface)
  - Les micro noyaux de systèmes (Windows) répartis (Chorus)
  - Les systèmes d'objets distribués (CORBA - *Common Object Request Broker Architecture, par exemple)*

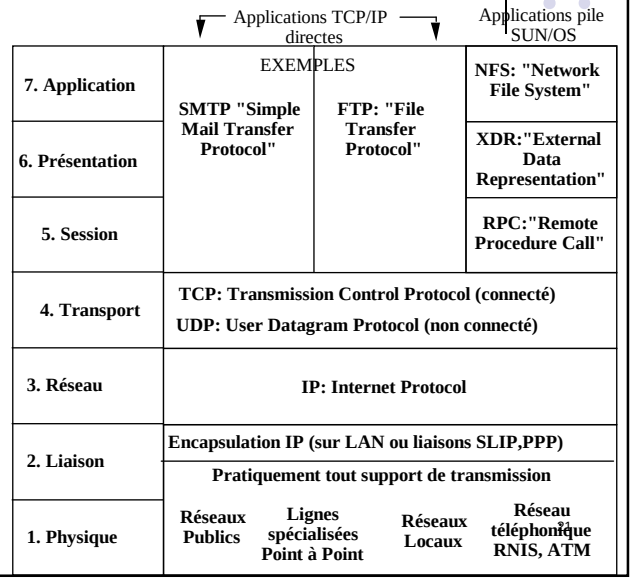
19

Plan

20

Communication via des réseaux (1)

- **Rappel**
- Pile protocolaire



Communication via des réseaux (2)

- **Rappel**
- **Couche Transport** assure un service de transmission fiable de bout en bout entre processus.
- Son principal fonctionnement : gestion des connexions (segmentation, négociation de QoS), contrôle de flux; contrôle d'erreur; contrôle de séquence

- Dans le réseau Internet: les protocoles de la couche Transport sont **TCP** (*Transmission Control Protocol*) et **UDP** (*User Datagram Protocol*)
- Dans le réseau Novell: SPX *(Sequenced Packet eXchange)*

22

Communication via des réseaux (3)

- **Rappel**
- **Couche Session** assure la synchronisation et la gestion des échanges point à point en mode message (définir des « points de synchronisation » pour multi-processus, par exemple)
- elle assure la tolérance aux pannes (re-établissement d'une connexion, par exemple)
- elle gère également l'authentification de connexion

- les protocoles : X225, RPC (*Remote Procedure Call)* , SIP (*Session Initination Protocol*)

- RPC (*Remote Procedure Call)* assure en environnement réparti une sémantique pour l'appel de procédure distante.

- exemples :
- RPC traditionnels : Internet Protocole RPC ; XML-RPC
- Systèmes d'objets répartis: CORBA (*Common Object Request Broker Architecture*); Java RMI (Remote Method Invocation)

23

Communication via des réseaux (4)

- **Rappel**
- **Couche Présentation** traite du codage des données échangées venues des différents sites pour que les différents sites ayant des représentations différentes puissent utiliser les données

- Deux types de syntaxe:
- Syntaxe abstraite : donne la définition d'une grande variété de structures de données
- Syntaxe de transfert : donne une présentation unique dans le réseau des valeurs échangées

- Exemples de standards de conversion :
- syntaxe abstraite ASN1 ((*Abstract Syntax Notation One*)) CCITT X.680 à X.683 (en 2009)
- Syntaxe de transfert : ASN1 X.690 (codages)
- Internet: XDR (*External Data Representation Standard*)
- Systèmes d'objets répartis CORBA : CDR (*Common Data Representation*)
- Web services : XML (*eXtended Markup Language*)

24

Communication via des réseaux (5)

- **Rappel**
- **Couche Application** offre des fonctions à l'utilisateur

- Le développement d'une application informatique, structuration objet par exemple

- les fonctions réseaux qui déchargent l'utilisateur de travaux répétitifs
  - le protocole du transfert de fichiers dans l'Internet – File Transfer Protocol
  - l'accès aux fichiers distants dans l'Internet – Network File System (NFS)

- l'échange du courrier électronique entre usagers
  - Internet SMTP (*Simple Mail Transfer Protocol*)

- l'échange de documents électronique
  - HTML (*Hyper Texe Markup Language*)
  - XML (*eXtended Markup Language*)

25

Applications d'Internet

- L'accès aux informations distantes, WWW, par exemple
  - Systèmes de désignation URL *(Uniform Resource Locator)*
  - Protocole de communication HTTP *(Hyper Text transfer Protocol)*
  - Protocole d'accès SOAP *(Simple Object Access Protocol)*
  - …

26

Conclusion

- Les réseaux et systèmes répartis sont extrêmement complexe au centre de l'informatique actuelle et à venir
- les technologies client-serveur ont bien évoluées (du partage de fichier à web services)
- L'Internet facilite le développement de nouvelles applications (web services sur mobile)
- Le but de cette UE a pour connaître les applications client-serveur, en particulier, les applications d'Internet.

27

Bibliographie

- George Coulouris, Jean Dollimore et Tim Kindberg, « Distributed Systems », ADDISON WESLEY, Third édition, 2001
- Standards d'IETF (*Internet Engineering Task Force*)
- W3C (World Wide Web Consortium)
- Les livres sur SMTP, DNS, HTTP, Web services, XML
- Le(s) plateforme(s) Java
- Gérard Florin, support de cours « Technologie des applications Client-serveur », 2000

28