Architecture client-serveur

Yannick Prié
UFR Informatique
Université Claude Bernard Lyon 1

2005-2006 – Master SIB
M1 – UE 3 / Bloc 4 – Cours 3

CM3 : Architecture client/serveur

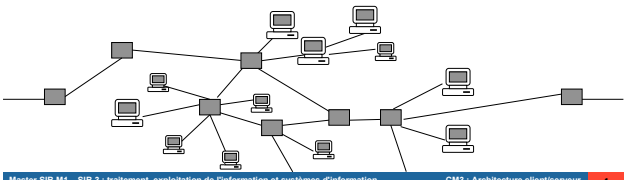
- Objectifs du cours
  - Rappels sur les ordinateurs réseau et Internet. Communication entre programmes et entre machines. Notion de protocole, couches ISO, protocoles de l'Internet. Architecture client / serveur. Considérations de sécurité. Exemples concrets de protocoles : HTTP et Z39.50

Réseau : pour quoi faire ?

- Echanger et partager des informations
  - transferts de fichiers/données
  - accès à des fichiers/données distants...
- Gérer et partager des ressources
  - imprimante partagée
  - puissance de calcul
  - stockage et sauvegarde...
- Communiquer
  - courrier électronique
  - chat
  - publication en ligne...

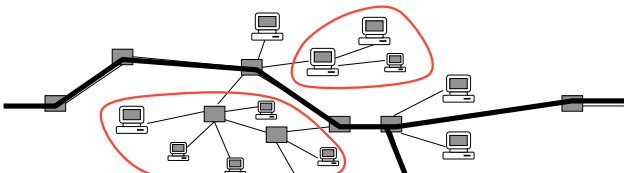
Réseau : c'est quoi ?

- Des machines et leurs programmes (= nœuds du réseau)
  - Ordinateurs réseau (carte réseau + système exploitation réseau)
  - Périphériques réseau (imprimante...)
  - Matériel réseau spécialisé
- Des connexions entre les machines (= branches du réseau)
  - par câble (fibre optique, RJ45, cuivre...)
  - par radio (ondes hertziennes, infrarouge...)

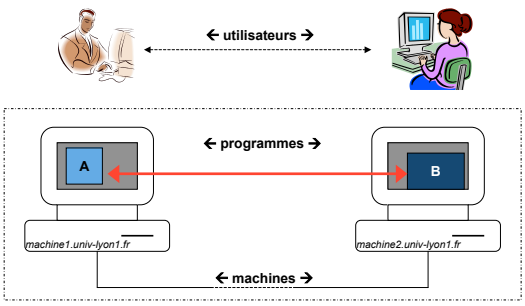


Réseau local / Internet

- Réseau local
  - centré autour d'une organisation (université, entreprise, famille)
  - géré par celle-ci
- Internet = réseau des réseaux mondial
  - ensemble de réseaux locaux
  - reliés par des « backbones » (épine dorsale)



Communiquer entre humains / programmes / machines



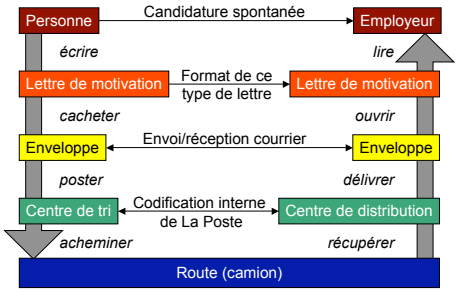
A tous les niveaux, des protocoles

- Définition
  - Ensemble de règles et de procédures à respecter pour pouvoir échanger des données sur un réseau
    - Remarque : exemple de la vie courante
      - Ca va ? (→) / Oui (←) / J'ai pas entendu (→) / Je répète : « Oui » (←)
- Exemples à différents niveaux
  - Niveau programmes (A – B)
    - Tu fonctionnes ? (→) / Oui (←) /Envoie-moi le fichier toto.doc (→) / Ok c'est parti (←) / toto.doc (←) / Bien reçu (→) / Au revoir (←)
  - Niveaux système d'exploitation
    - Toi, système d'exploitation de la machine *machine1.univ-lyon1.fr,* passe le message *« Tu fonctionnes ? »* au programme qui écoute sur le port 3422 (→)
  - Niveau cartes
    - Toi carte réseau, regarde passer des paquets de données sur le câble, attrape ceux qui sont pour toi, et passe-en le contenu au système d'exploitation

Protocoles de communication

- But
  - compréhension entre machines / logiciels
  - communications indépendantes du système d'exploitation ou de la plate-forme
  - limitation des erreurs/risques durant la transmission
- Protocole pour l'échange de messages
  - un langage *et* un ensemble de règles
  - que deux systèmes doivent connaître (parler le même langage)
  - les fabricants doivent se conformer aux normes ISO (International Standardization Organization) pour les protocoles utilisés sur leurs machines/logiciels
- Modèle OSI (Open System Interconnection)
  - découpe le processus de transmission en 7 «couches»
  - chaque couche est responsable de l'un des aspects de la communication en réseau

Modèle en couches



Le modèle OSI les 7 couches

Application : gestion des échanges de données entre programmes et services du réseau

Présentation : mise en forme des informations pour les rendre lisibles par les applications

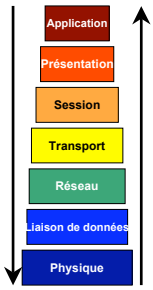
Session : détection du mode de communication à utiliser entre machines et périphériques / Surveillance des connexions

Transport : correction des erreurs de transmission; vérification de l'acheminement

Réseau : identification des machines connectées au réseau

Liaison de données : subdivision des informations en «paquets» pour livraison sur le réseau

Physique : contrôle du support de transmission; circulation de l'information électrique



Protocoles de l'Internet

- Niveau OSI réseau
  - **IP (Internet Protocol)**
    - adressage (routage) des informations
    - identification des machines
- Niveau OSI transport / session
  - **TCP (Transmission Control Protocol)**
    - transfert d'information, contrôle des transmissions
- Niveau OSI application
  - FTP (File Transfer Protocol)
    - transfert de fichiers
  - HTTP (HyperText Transfer Protocol)
    - transfert d'informations sur le web
  - DNS (Domain Name Server protocol)
    - conversion du nom des ordinateurs connectés au réseau en adresses IP
  - etc.

TCP/IP : base de l'Internet

Identification de machines sur Internet

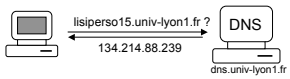
- Une machine = un numéro
- Adresse IP universelle unique
  - 4 nombres de 8 bits (4 octets)
  - séparés par des points
  - ex : 134.214.128.17
- Classes d'adresses / organisations
  - Classe A : 112.x.y.z (ex. NASA)
  - Classe B : 134.214.x.y (ex. Université Lyon 1)
  - Classe C : 56.243.12.x (ex. Cyber-café)
- Remarque : IPV6
  - 16 octets
  - commence à se mettre en place

Noms de ressources sur Internet

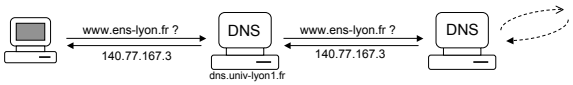
- Idée : associer à l'adresse IP un nom de machine
  - ex. lisiperso15.univ-lyon1.fr # 134.214.88.239
- Nom de machine
  - décomposé hiérarchiquement
    - domaine (critère géographique, institutionnel, organisationnel…)
    - sous-domaine (éventuellement)
    - nom local de la machine
  - exemples
    - ligimpc13.univ-lyon1.fr
    - www.berkeley.edu
    - ftp.berkeley.edu
    - www.education.gouv.fr
- Attribution
  - ICANN : Internet Corporation for Assigned Names and Numbers
    - .gov, .edu, etc.
  - AFNIC : Association Française pour le Nommage Internet en Coopération
    - .fr, .gouv.fr, .asso

Traduction adresse IP / nom de machine

- Service de traduction fourni par des programmes appelés DNS (*Domain Name Server)*
- DNS est aussi le nom du protocole utilisé pour communiquer entre un programme qui désire une traduction, et un serveur
- Un DNS gère un domaine...



- ... et transmet la question à un autre DNS s'il ne sait pas répondre.

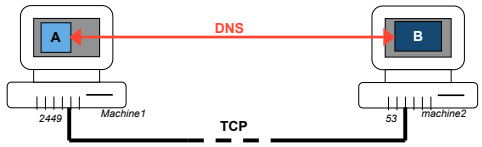


Client / serveur

- Service
  - comportement d'un programme qui peut rendre service à d'autres programmes
    - exemple : service de traduction noms/adresses IP = service DNS
  - un service est appelé par une requête suivant un certain protocole
    - exemple : requête « donne-moi la traduction de lisiperso15.univ-lyon1.fr » envoyée suivant le protocole DNS
- Client
  - *programme* demandant un service à un autre programme ET
  - *machine* sur laquelle tourne ce programme client
- Serveur
  - *programme* fournissant des services à d'autres programme ET
  - *machine* sur laquelle tourne ce programme serveur

Notion de socket

- Port
  - Entrée réseau de la machine
    - sur laquelle un serveur « écoute » en attendant des connexions / requêtes
    - à laquelle un client va se connecter
- Socket
  - « Tuyau » entre deux programmes
  - Quintuplet : (machine1, port1, protocole transmission, port2, machine2)
- Exemple
  - Client sur machine 1 appelle serveur sur machine 2 / port 53, suivant le protocole TCP
  - La connexion s'établit, le canal de communication est ouvert (port de sortie client : 2449)
  - Il devient possible de communiquer suivant un protocole d'application (par exemple DNS)



Client / serveur : exemples (1/2)

- Traduction noms de machines / adresses IP
  - protocole : DNS
  - clients : tout programme réseau utilisant des noms de machine,
  - serveurs : DNS (port = 53)
  - remarque : un DNS peut jouer le rôle de client pour un autre DNS
- Transfert de fichiers
  - protocole : FTP
  - clients : outils de gestion de transferts FTP (Ws_FTP, FileZilla, *etc.*)
  - serveurs : serveurs FTP (port = 21-22)
- Web
  - protocole : HTTP
  - clients : navigateurs web (Mozilla, IE, Firefox, Opera...)
  - serveurs : serveur web (IIS, Apache, ...) (port = 80)

Client / serveur : exemples (2/2)

- Machine connectée au réseau
  - protocole : ping
  - clients : ping
  - serveurs : serveurs ping
- Peer to peer
  - protocoles : envoi de fichier, échange d'informations, ...
  - client : client P2P
  - serveurs : client P2P, serveurs de métadonnées

Client / serveur : remarques

- Un programme serveur
  - tourne en permanence, attendant des requêtes
  - peut répondre à plusieurs clients en même temps
- Nécessité
  - machine robuste et rapide, qui fonctionne 24h/24
    - grande mémoire,
    - disques suffisants
    - sécurité des disques
    - *etc.*
  - présence d'administrateurs réseau pour gérer les serveurs

Architecture client / serveur

- Des échanges entre programmes sur réseau de machines suivant les principes client / serveur
  - des machines serveur peu nombreuses
  - des postes clients pour les différents utilisateurs
- Fiabilité et avantages (*cf.* « Comment ça marche ? »)
  - Ressources centralisées : les serveurs sont au centre du réseau, gèrent les ressources communes à tous les utilisateurs, et permettent d'éviter les problèmes de redondance et de contradiction
  - Meilleure sécurité : faible nombre de points d'entrée pour l'accès aux données
  - Administration centralisée au niveau des serveurs : les clients ne sont pas des ressources critiques
  - Réseau évolutif : ajouter/enlever des clients sans perturber le réseau
- Inconvénients
  - Coût élevé des machines serveurs, car fiabilité vitale

Sécurité des échanges réseau

- Protection des données qui circulent
  - Cryptage des données
- Protection des données sur les machines
  - Identification
- Protection des attaques
  - Firewall
  - Antivirus

Notion de session

- Session
  - Connexion maintenue entre un logiciel client et un serveur
  - Par exemple
    - identification sur un intranet,
    - navigation sans donner à nouveau mon login/mdp
    - le lien entre le client et le serveur est maintenu même quand il ne se passe rien
  - Une session est en général coupée si elle dure trop longtemps

Echanges sur le web : HTTP

- HyperText Transfert Protocol
  - Défini par le W3C (World Wide Web consortium)
- La plupart des URL
- Envoi de documents web
  - d'un serveur web (serveur HTTP)
  - vers un client web (navigateur)
- Principe
  - Requête du client au serveur
    - demander une ressource web (page, image, service)
  - Réponse serveur au client
    - envoyer une ressource (page web, image, réponse)
- HTTP 1.0
- HTTP 1.1
  - gère les sessions (permet de garder une connexion)

HTTP : requêtes / réponses

- Référence
  - RFC 2616 (Request For Comments =/= standard de fait)
- Requête client
  - Contenu de la requête (type + URI + version protocole)
  - En-têtes
  - Ligne vide (indique fin de requête)
- Réponse serveur
  - Code réponse (version HTTP + code + chaine)
  - En-têtes
  - Ligne vide (indique fin en-tête)
  - Contenu de la réponse (souvent le document qu'on veut)

HTTP requêtes / réponses

- Requêtes
  - GET : demande de document ou de service
  - POST : demande de service avec envoi de paramètres
  - HEAD : demande d'information concernant un document
  - ...
- Réponses
  - Codées
  - 2xx : succès
    - 200 : ok
  - 3xx : redirection
    - 304 : document inchangé
  - 4xx : erreur client
    - 401 : non autorisé
    - 404 : inexistant
  - 5xx : erreur serveur
    - 500 : erreur dans l'exécution d'un service
    - 505 : version HTTP non supportée

En-têtes de requêtes

- From : adresse email
  - Non envoyée par la majorité des clients pour des raisons de confidentialité
- Accept : liste de types MIME
  - Exemples
    - audio/mid, image/jpeg
    - application/pdf
- Accept-Encoding : liste de méthodes de codage MIME
  - Exemples : compress, x-gzip, x-zip
- Accept-Langage : liste des langues acceptées
  - En réalité, n'est pratiquement jamais utilisé
- User-Agent : l'identification du logiciel client
  - Permet de répondre différemment suivant le client
  - Ne devrait pas être le cas (car il y a des normes qui devraient être respectées par tous les clients)
- Referer : page d'où l'on vient
  - Peut être utile pour faire des statistiques de parcours dans le site
- Autorization : login password
  - Niveau faible de sécurité (tout passe en clair)
- If-Modified-Since : date
  - Ne transmet la page que si elle a été modifiée depuis la date spécifiée
  - Utile pour les caches
- etc.

En-têtes de réponses

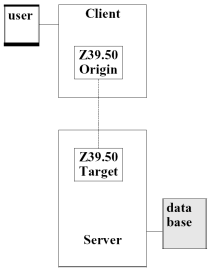
- Server : type du serveur
- Date : date du traitement de la requête
- Last-Modified : date
  - Utile pour le cache
- Content-Type : type MIME du document renvoyé
  - Doit faire partie en théorie de la liste des types acceptés dans la requête (Accept)
- Content-length : longueur des données (octets)
  - On peut savoir quand (et si) le transfert est fini
  - Permet au navigateurs d'indiquer des barres de progression
  - Non obligatoire
- Content-Encoding : encodage MIME
  - Doit faire partie en théorie des méthodes spécifiées dans la requête (Accept-Encoding)
- Content-Langage : langue
- etc.

Exemple

```
eguerin >telnet bat710 80
Trying 134.214.88.10...
Connected to bat710.univ-lyon1.fr.
Escape character is '^]'.
HEAD / HTTP/1.1
Host: www710.univ-lyon1.fr
Connection: close
HTTP/1.1 200 OK
Date: Mon, 09 Sep 2002 14:50:22 GMT
Server: Apache/1.3.9 (Unix) Debian/GNU
Last-Modified: Thu, 11 Jul 2002 09:36:01 GMT
ETag: "27ec6-1811-3d2d5181"
Accept-Ranges: bytes
Content-Length: 6161
Connection: close
Content-Type: text/html; charset=iso-8859-1
Connection closed by foreign host.
eguerin >
```


Interrogation de catalogues Z39.50 : principes

- Base de données bibliographiques hétérogènes
- Serveur Z39.50 = passerelle
  - traduction des requêtes Z39.50 dans le langage de la base
  - traduction des réponses de la base suivant le protocole Z39.50
- Dialogue client/serveur suivant Z39.50
- Le client Z39.50 envoie les requêtes et affiche les réponses



(ZIG : Z39.50 tutorial)

Remerciements

- Certaines diapositives proviennent de cours du Permis de Conduire Informatique (Université Lyon 1 – http://pci.univ-lyon1.fr).
- D'autres sont inspirées du cours « Web avancé », IUT A, UCBL (Eric Guérin)