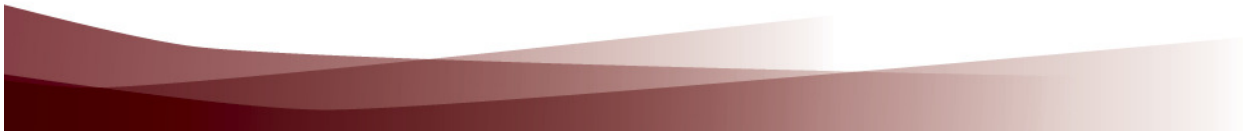
Telecommunication Services Engineering (TSE) Lab

Chapitre 2: Couche Application I

Roch Glitho, PhD

Associate Professor and Canada Research Chair

My URL - http://users.encs.concordia.ca/~glitho/



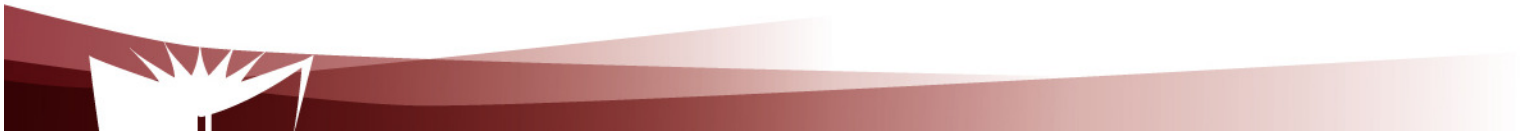
Concordia University
Engineering and Computer Science
Concordia Institute for Information Systems Engineering

Telecommunication Services Engineering (TSE) Lab

Université du Québec
École de technologie supérieure

Couche Application I

(Infrastructures d application)



Université Concordia University

Couche Application

Application
Transport
Réseau
Liaison de données
Support physique

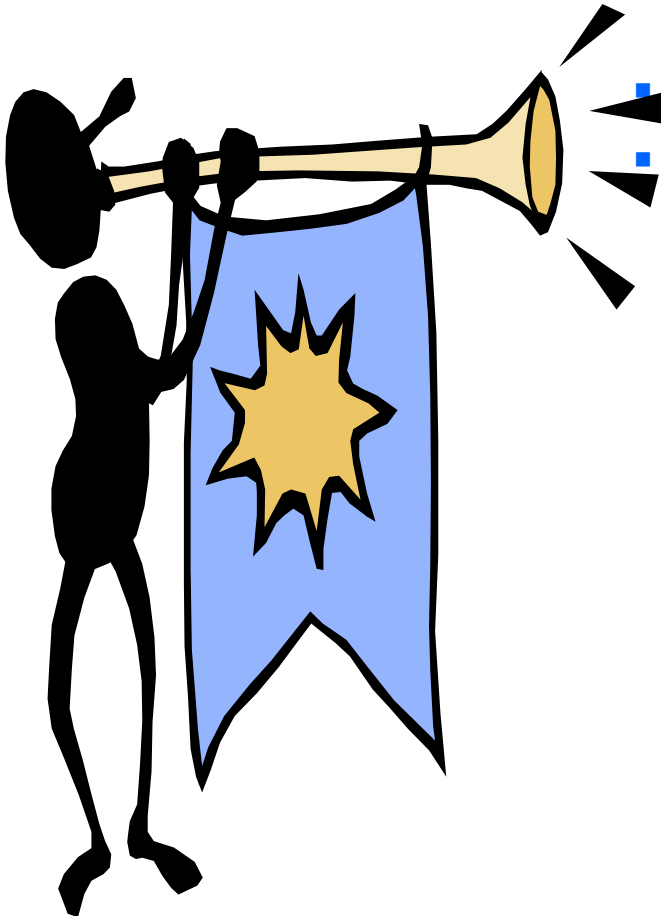
Couche Application

Infrastructure d application

Composants génériques re-utilisés par d'autres protocoles du niveau applicatif

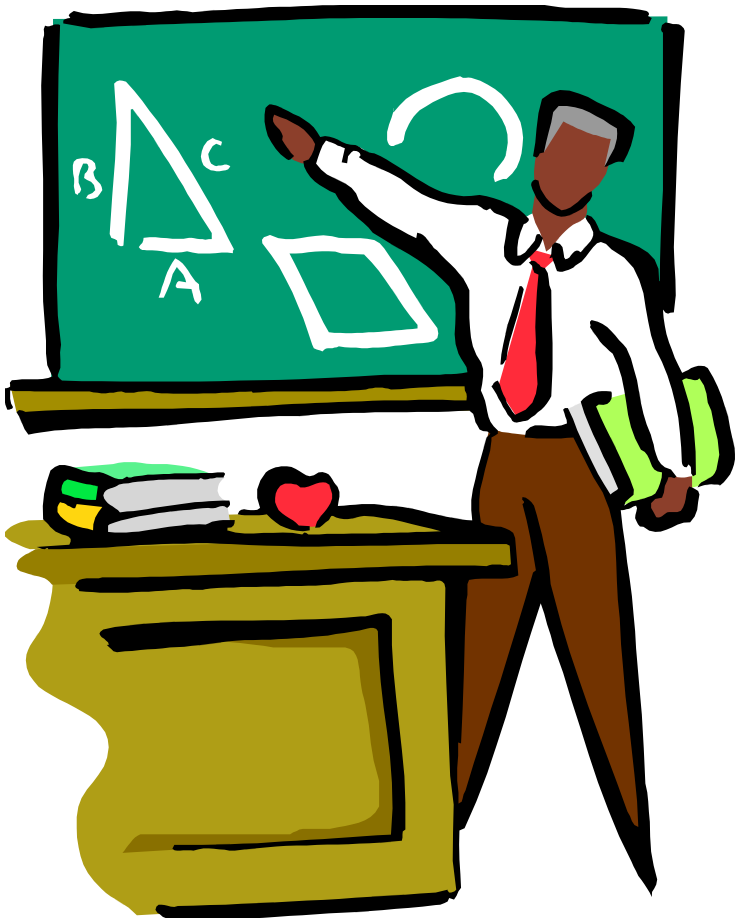
- Espaces de noms de systèmes (Domain Name System – DNS)
- Pair à pair (ou poste à poste) – Peer to Peer (P2P)

Table des matières (Couche Application – Première partie)



- DNS
- P2P

DNS



- 1 – Cadre conceptuel
- 2 - Implémentation

Cadre conceptuel

Genèse

- Début des années 80
- Remplacement du fichier HOSTS.TXT utilisé au début
  - Le fichier HOSTS.TXT
    - Contient la liste de tous les hôtes Internet et le mappage entre les noms symboliques et les adresses IP
    - Géré de manière centralisée
      - Information distribuée périodiquement à tous les hôtes par transfert de fichier

Cadre conceptuel

Genèse

- Début des années 80
- Remplacement du fichier HOSTS.TXT utilisé au début
  - Lea problèmes de l'approche fichier HOSTS.TXT
    - Croissance du nombre d'hôtes sur Internet
    - Adminstration du fichier (qui va le faire?)

Cadre conceptuel

Les requis (Cahier de charges)

- Offrir au minimum toutes les informations disponible avec HOSTS.TXT
- Stockage et maintenance distribués
- Flexibilité dans la syntaxe des noms et la taille des données

Cadre conceptuel

Les requis (Cahier de charges)

- Interopérabilité à travers Internet
- Performance acceptable
- Extensibilité
- Independence de la topologie du réseau et des systèmes d'exploitation

Cadre conceptuel

- Quelques exemples de décisions et leur motivations par les requis
  - **Espace de nom hiérarchisé**
    - Satisfaction des requis suivants:
      - Stockage et maintenance distribués
      - Flexibilité dans la syntaxe des noms et de la taille des champs de données

Cadre conceptuel

- Quelques exemples de décisions et leur motivations
  - **Anté mémoire de données (data caching)**
    - Satisfaction des requis suivants:
      - Performance acceptable

Cadre conceptuel

Concepts clé

- Espace de nom de domaine (Domain name space)
  - Arbre de profondeur variable avec noeuds etiquettés
    - Étiquettes
      - Jeu de caractères de taille variable
      - Minuscules et majuscles sont équivalents

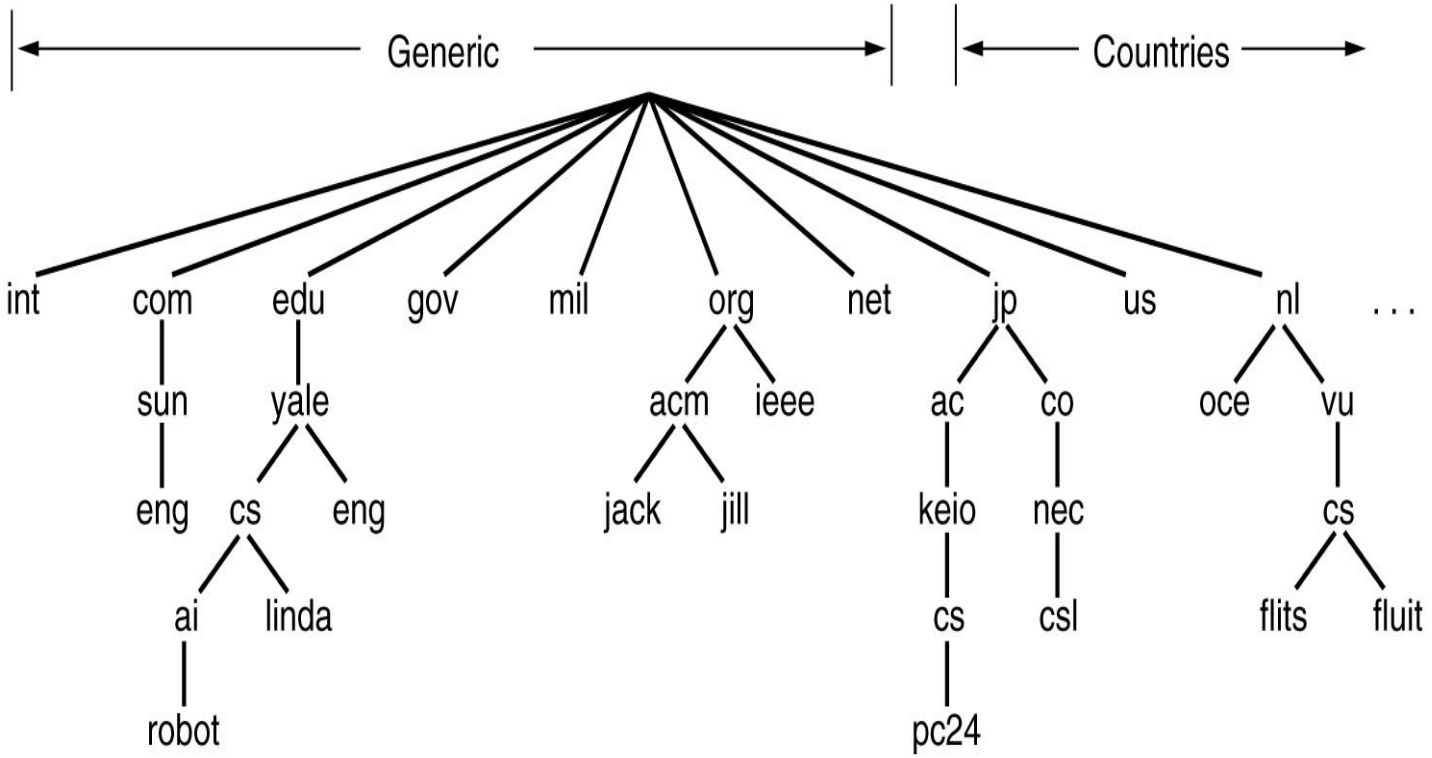
Cadre conceptuel

Concepts clé

- Nom de domaine d un noeud
  - Concaténation de toutes les étiquettes du noeud jusqu a la racine
- Décisions adminstratives
  - Les niveaux les plus élevés correspondent à:
    - Codes de pays (ex: CA)
    - Catégories d'organisation (ex: .com)

Cadre conceptuel

DNS



Cadre conceptuel

Concepts clé

- Enrégistrement de resources (Resource records (RRs))
  - Données attachées à chaque nom
    - Type
      - Resource abstraite (e.g. adresses d hôtes)
    - Champ de la classe (class field)
      - Famille de protocoles (ex: DARPA Internet)
    - Données d application (ex: adresse IP)

Cadre conceptuel

Concepts clé

- Enrégistrement de resources (Resource records (RRs))
  - Note:
    - Enrégistrement dans l anté mémoire peuvent ne plus être à jour

Cadre conceptuel

Concepts clé

- Exemple d enrégistrement

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

Cadre conceptuel

Concepts clés

- Zones
  - Ensembles disjoints de sous arbres sous le controle d une seule organisation
    - Singletons and ensemble formé par tout l arbre sont exclus

Cadre conceptuel

Concepts clés

- Zones
  - Procedure
    - Autorisation initiale obtenue par une organisation parent pour un seul noeud
    - Croissance jusqu à une taille arbitraire sans l implication de l organisation parent

Cadre conceptuel

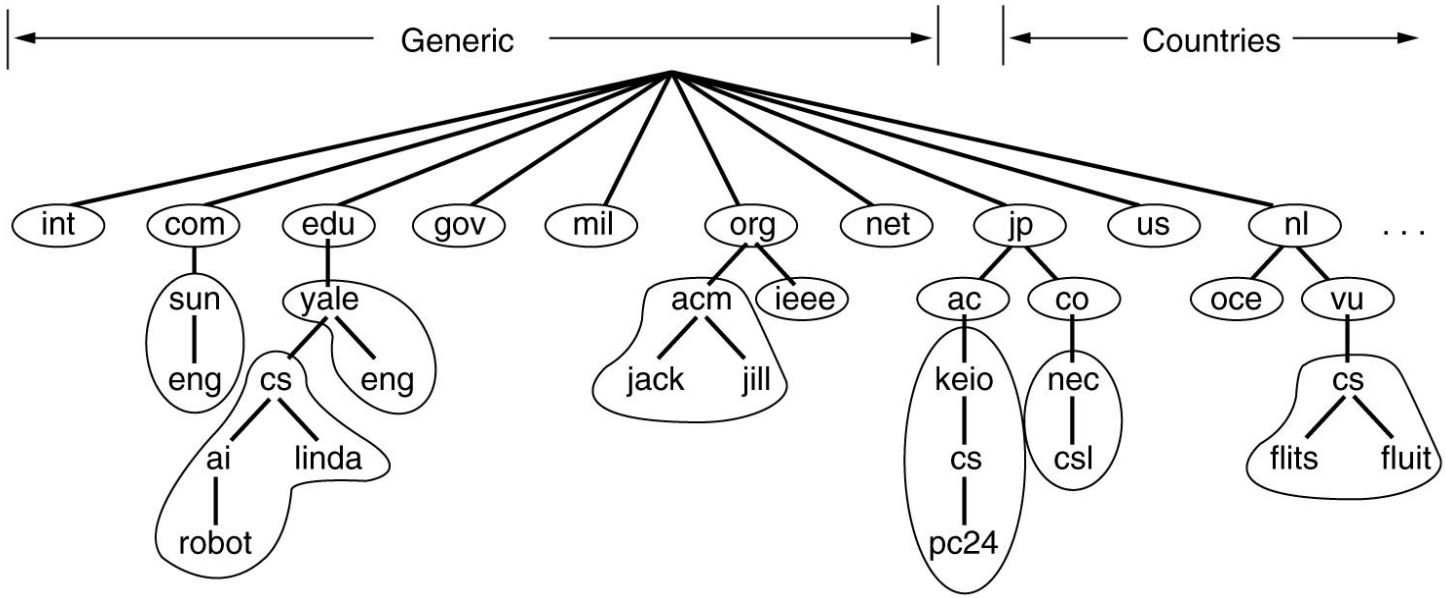
Concepts clés

- Zones
  - Les responsabilités
    - Maintenance des donnés de la zone
      - Redondance peut être utilisée pour obtenir de la fiabilité

Cadre conceptuel

Concepts clé

- Zones



Cadre conceptuel

Operations clés

- Transfer de zones
  - Les organizations font connaitre les informations sur leurs zones aux autres organisations

Cadre conceptuel

Operations clés

- Ante memoire (Caching)
  - Les organisations gardent les informations dans l ante memoire dans la mesure du possible

Implementation

Ce cadre conceptul peut être implémenté d au moins 2 manières

- Client / serveur (implementation actuelle)
- P2P (Encore au niveau de la recherche)

Implementation

Implementation client serveur

Entités fonctionnelles

- Serveur de nom (Name server)
  - Répertoire d information
  - Fait le mappage (i.e name resolution)
  - Peut supporter n importe quel nombre de zones

Implementation

Implementation client serveur

Entités fonctionnelles

- Serveur de nom (Name server)
  - Très flexible
    - Un serveur de nom pour un domaine donné n a pas nécessairement besoin d être dans le domaine
    - La distribution est optimale quand elle suit la même hiérachie que l espace de noms

Implementation

Entités fonctionnelles

- Resolveur (Resolver)
  - Interface offerte aux programmes clients
    - Implémente les algorithms nécessaires pour trouver le serveur de nom qui contient l information de mappage requise.

Implementation

- Serveurs de nom et résolveurs peuvent être dans le même noeud ou dans des noeuds
  - Resolveurs en général centralisés dans des serveurs dédiés au niveau organisationnel
    - Reutilisation de l information de l anté memoire
    - Machines avec moins de puissance n ont pas besoin d implementer leurs propres fonction de résolution

Implementation

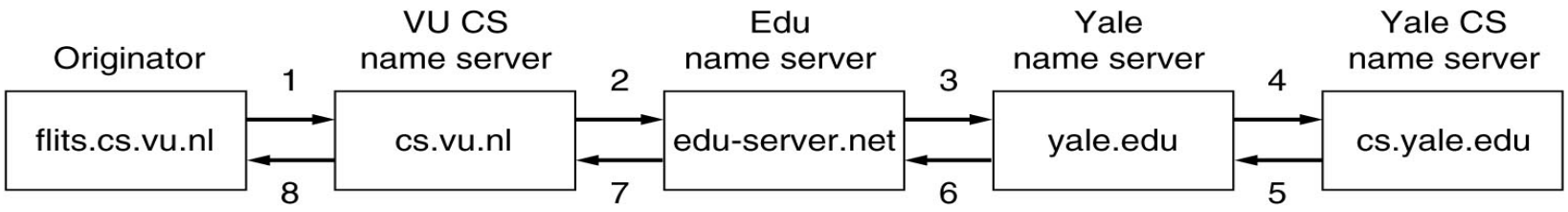
- Requêtes
  - Deux approches
    - Iteration
      - Quand un serveur de nom n a pas l information il retourne au resolveur le nom du prochain serveur sur le chemin

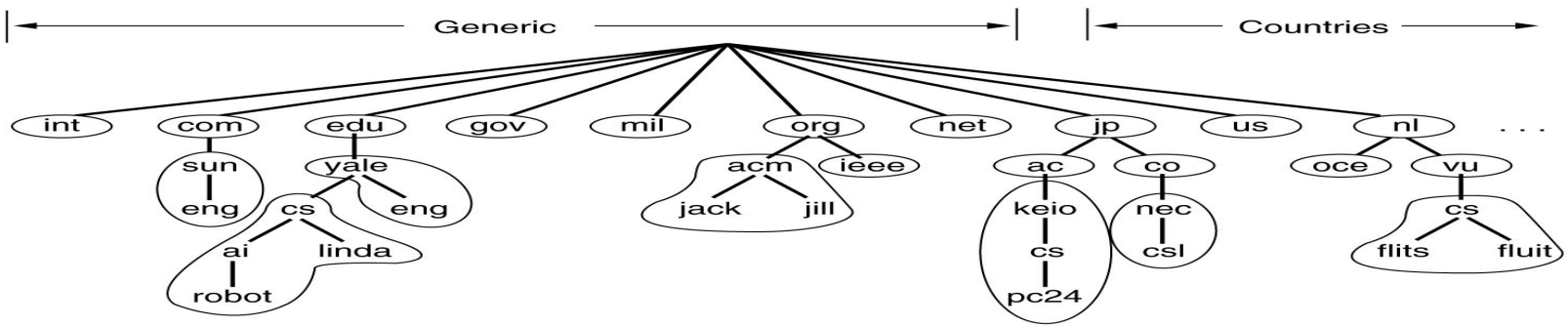
Implementation

- Requêtes
  - Deux approches
    - Recursion (optionel)
      - Quand un serveur de nom n a pas l information requise, il contacte directement un autre serveur de nom qui pourrait l avoir.
      - Le processus est itéré jusqu a ce que l information soit trouvée

Implementation

- Exemple de recherche recursive
- (linda@cs.yale.edu)





Implementation

Messages

- Un seul et même format pour les requêtes et réponses

Header	
Question	the question for the name server
Answer	RRs answering the question
Authority	RRs pointing toward an authority
Additional	RRs holding additional information

Implementation

- Transport (TCP et UDP)
  - UDP
    - Requêtes
      - Re-transmissions éventuelles laissées à la discrétion des applications utilisatrices (resolveur, serveur de nom)

Implementation

- Transport (TCP et UDP)
  - TCP
    - Utilisé pour les informations critiques
      - Mise a jour de l ante memoire

Implementation

Problèmes actuels

- Confirmés par l expérience et l expérimentation
  - Vulnerabilité aux pannes de réseaux et au aux attaques de déni de services (DOS)
    - Raisons essentielles:
      - » Petit nombre de serveurs de noms et peu de redondance
      - » Vulnérabilités des serveurs de nom les plus déployés sont connus

Implementation

Problèmes actuels

- Confirmés par l expérience et l expérimentation
  - Problèmes de performance
    - Latence pour la résolution de nom
      - » Raisons:
        - » Peu de résolution dans l ante mémoire
        - » Erreurs humaines de configuration

Architecture

Un exemple d alternative

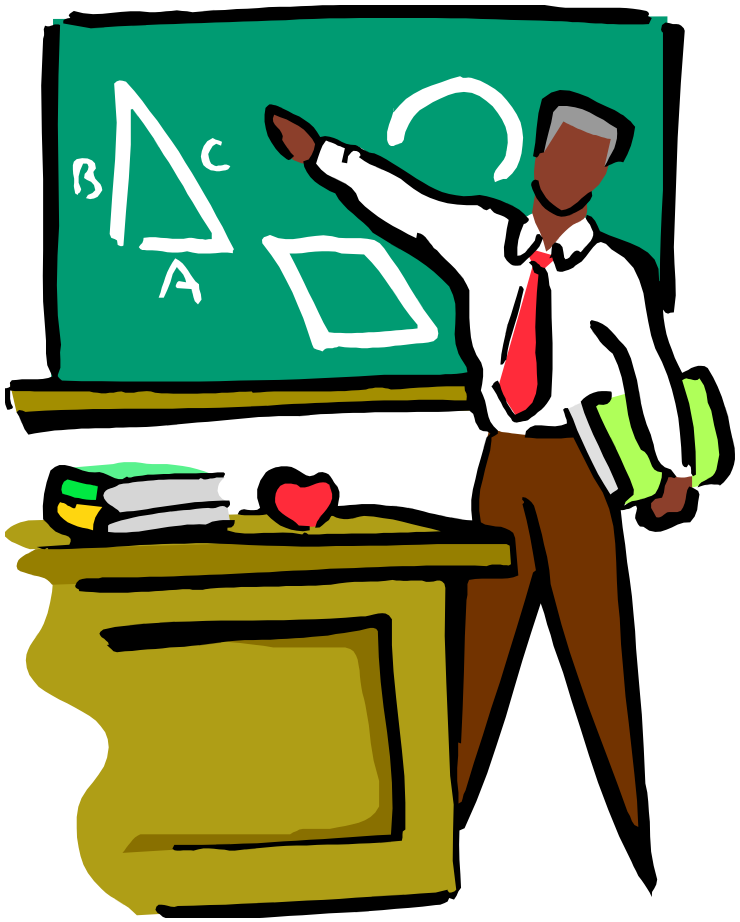
- P2P

References

1, P.V. Mockapetris and K. Dunlap, Development of the Domain Name System, ACM SIGCOMM Computer Review, 1995

2. A. Tanembaum, Computer Networks, Chapter 7

P2P



- 1 – Client/serveur vs. P2P
- 2 - Réseaux de recouvrement P2P
- 3. Chord, Freenet et Skype
- 4. JXTA: Un intergiciel (middleware) pour le développement d applications for P2P

Client / serveur vs. P2P

Client / serveur

- Essence
  - Le serveur offre stockage et capacité de traitement
    - Statique
    - Mise a jour, augmentation de capacité de traitement et de stockage faites uniquement par le fournisseur du serveur
  - Clients accèdent au serveur
    - Role passif
    - Peu ou pas de contribution au stockage et au traitement

Client / serveur vs. P2P

Client / serveur

- Hypothèse de base
  - Clients ont peu de capacité de stockage et de traitement
  - N est plus valide de nos jours

Client / serveur vs. P2P

Client / serveur

- Les problèmes:
  - Racine:
    - Serveur peut devenir un goulot d étranglement d un point de vue stockage et traitement de données

Client / serveur vs. P2P

Client / serveur

- Les problèmes:
  - Exemples
    - » Évolutivité, montée en charge (scalability)
    - » Disponilité (Availability)

Client / serveur vs. P2P

P2P

- Plusieurs definitions existent
  - Dans ce cours:
    - Paradigme de traitement basé sur un réseau de pairs (posts) au lieu d un serveur et qui collaborent pour résoudre les problèmes inhérents au paradigme client / serveur
      - » Évolutivité, montée en charge
      - » Disponibilité

Client / serveur vs. P2P

P2P

- Hypothèse de base
  - Clients ont de nos jours de plus en plus de capacités de stockage et de traitement qui pourraient être utilisées

Client / serveur vs. P2P

P2P

- Clients fédèrent à travers le réseau P2P pour offrir les capacités de stockage et de traitement aux applications
  - Clients sont appelés pairs
    - Chaque pair peut contribuer selon ses capacités
    - Les pairs avec plus de puissance, parfois appelés super pairs peuvent contribuer plus

Client / serveur vs. P2P

P2P

- P2P pur vs. P2P hybride
  - Pure P2P:
    - Architectures totalement distribuée (pas de serveur centralisé)
      - » Plutôt rare (ex: . Freenet)
  - P2P hybride
      - » Un certain niveau de centralisation
        - » Plus déployé (ex: Skype)

Client / serveur vs. P2P

P2P

- Quelques exemples de défis techniques
  - Auto organization
    - Les pairs peuvent rentrer et sortir du système n importe quand
  - Stockage et recherche de données
    - Ou stocker?
      - On peut stocker sur n importe quel sous ensemble de paits
    - Efficicacité de la recherche (performance)

Client / serveur vs. P2P

P2P

- Quelques exemples de défis techniques
  - Tolérance aux fautes
    - Départs volontaires vs. départs involontaires
      - Que faire si un pair quitte?
      - Que faire si un pair tombe en panne?

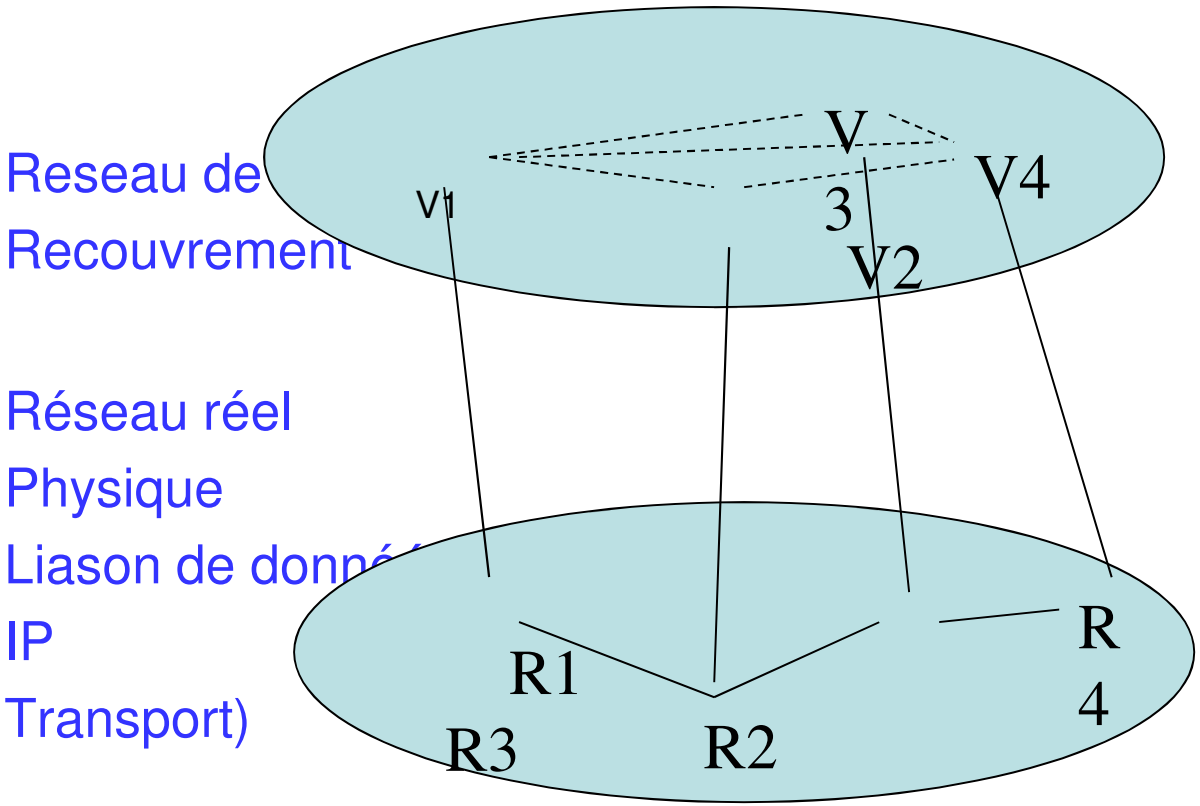
Réseaux de recouvrement (overlays)

- Implémentation actuelle de P2P
  - Réseaux virtuels de niveau applicatif qui fournissent stockage, traitement de données, connectivité et routage

Réseaux de recouvrement (overlays)

- Implémentation actuelle de P2P
  - Construits au dessus de réseaux réels d ou le nom de réseau de recouvrement
    - » Infratructure batie au dessus de la couche transport
    - » Les noeuds réels deviennent des noeuds virtuels dans le réseau de recouvrement

Réseau de recouvrement



Réseaux de recouvrement

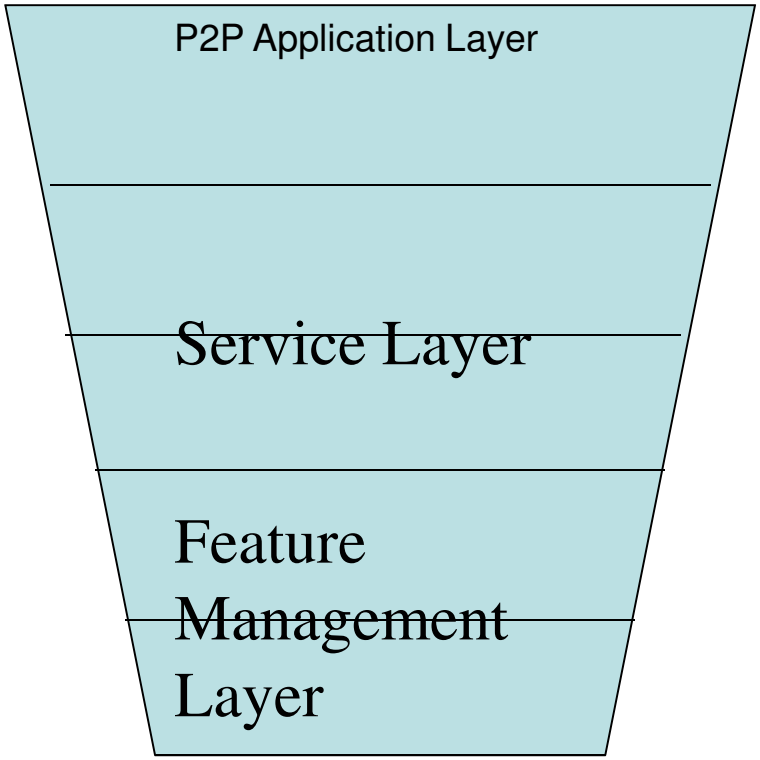
- Characteristiques
  - Topologie peut être différente de la topologie du réseau réel
  - Protocoles peuvent être différents des protocoles du réseau réel

Réseaux de recouvrement

- Characteristiques
  - Deux saveurs
    - Applications incorporées (e.g. Skype)
    - Infrastructure qui peut être utilisée par d autres applications (e.g. CHORD)
      - APIs
      - Outils de développement t

Réseaux de recouvrement

Vue abstraite



Réseaux de recouvrement

Vue abstraite

- Couche application (P2P application layer)
  - L application elle-même (ex. Partage de fichier, Téléphonie IP)
    - Peut être incorporé dans l infrastructure ou développée en utiilisant les APIs s ils existent.

Réseaux de recouvrement

Vue abstraite

- Couche service
  - Services or composants utilisés par le développeur pour batir l application
    - Peuvent être ou ne pas être visible aux développeurs externe

Réseaux de recouvrement

Vue abstraite

- Gestion des features
  - Features utilisées par toutes les applications (e.g. securité, tolérance aux fautes)

Réseaux de recouvrement

Vue abstraite

- Gestion de noeuds virtuels
  - Routage, découverte de resources
- Couche de communication avec le réseau
  - Interface du réseau réel
    - Au-dessus de la couche trnasport

Réseaux de recouvrement structurés

- Topologie très controlée Tightly controlled)
  - Données placées à des endroits spécifiques
    - Recherche de données très efficace
    - Technique: Table de hashage distribuée (DHT)
      - Generation d une clé
        - Put (Key, value)
        - Value = Get (key)

Réseaux de recouvrement structurés

- Quelques examples
  - Content Addressable Network (CAN)
  - Chord
  - Tapestry
  - Pastry
  - Kademlia
  - Viceroy

Réseaux de recouvrement non structurés

- Topologie peu controlée (loosely controlled)
  - Données placées à des endroits aléatoires
    - Techniques de recherche
      - Inondation
        - Très efficace pour les données duplifiées
        - Inefficace pour les autres types de placement

Réseaux de recouvrement non structurés

- Quelques exemples
  - Napster
  - Freenet
  - Gnutella
  - KazaA
  - BitTorrent

Skype

- Skype
  - Plusieurs serveurs
    - Login server
    - Skype-out server (PC to Public Switched Telephony Network (PSTN) calls)
    - Skype in server (PSTN calls to PC)

Skype

Skype

- Architecture du réseau de recouvrement
  - Hierarchique
    - Pairs ordinaires
    - Super pairs
      - Chaque pair ordinaire est connecté à un super pair
      - Super pairs sont interconnectés

Skype

Skype

- Signalisation
  - Au dessus de TCP

References

1, E.K. Lua et al, A Survey and Comparison of Peer-to-Peer Overlay Network Schemes, IEEE Communications Surveys and Tutorials, March 2004

2. I. Stoica et al, Chord: A Scalable Peer to peer Lookup Protocol for Internet Applications, IEEE/ACM Transactions On Networking, 2003
3. I. Clarke et al., Protecting Free Expression On-Line with Freenet, IEEE Internet Computing, January/February 2002
4. S. A. baset and H. Schulzrinne, An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol, IEEE Infocom 2006
5. L. Gong, JXTA: A Network Programming Environment, IEEE Internet Computing 2001
6. The Socket API in JXTA 2.0 http://java.sun.com/developer/technicalArticles/Networking/jxta2.0/index.html

www.ciise.concordia.ca