5.1 – Architecture technique

Introduction

- La notion d'architecture.

- Filer la métaphore : l'architecture renvoie à l'organisation, à la rationalisation.

- L'architecture et son évolution se conçoivent dans un cadre contraint (Budget disponible ? Délais ? Etat de l'existant ?) ...

Introduction

- Notion d'architecture logicielle. Notion complexe à aborder, polymorphisme de la notion

- Conception et organisation des logiciels

- Relation entre les "briques" logicielles. Les briques du logiciel.

- Communication entre les logiciels. Chaque logiciel comme une brique.

Introduction

Architecture technique

Architecture logicielle

Architecture matérielle

Introduction

- Notion d'architecture technique : "l'organisation du mode de fonctionnement des applications et les éléments physiques mis en oeuvre pour supporter leur fonctionnement."

- La notion d'architecture technique recouvre le matériel, le middleware, les logiciels systèmes et les réseaux de télécommunication.

Introduction

- Architecture et vue. Les différents formalismes de modélisation permettent d'avoir des vues sur les organisations. Vue des données, dynamiques des données.

- La notion d'architecture renvoie à une compréhension globale et intégrée, recouvrant notamment la notion de vue
- => Intérêt de la compréhension préalable des formalismes de modélisation (UML, Merise).

Architecture fonctionnelle

Gestion des comptes,
analyse de l'activité,
envoi de mail

Architecture applicative

Excel,
MySQL

Architecture logique

Architecture technique

OS,
SGBD,
Middleware,
type de réseau

Introduction

- Plusieurs niveaux pour comprendre l'architecture

- Le niveau conceptuel => concevoir le système d'information hors des contraintes matérielles.

- Le niveau technique / le déploiement.

- Description des composants + description du fonctionnement de ces composants.

Introduction

- Appréhender l'architecture pour pouvoir comprendre le SI, puis envisager de le faire évoluer

- Risque de destabiliser l'architecture si on fait évoluer un composant sans connaître tout son rôle dans l'organisation -> Risque de dégradation, de mauvais fonctionnement.

Introduction

- Enjeux de l'architecture informatique

- L'organisation des ressources informatiques explique pour partie les processus et les données stockées.

- Comprendre le SI pour le mettre en adéquation avec les besoins et en corriger les manques ou les erreurs
  - → Alignement stratégique
  - → Audit du SI

Introduction

- Faire évoluer l'architecture informatique
  - L'évolution avec la structure (fusion, massification)
  - L'évolution pour répondre à l'évolution de nouvelles formes de travail (travail collaboratif, télé-travail, firme en réseau...)
  - L'évolution pour répondre à de nouvelles opporunités (cloud-computing, externalisation...)
  - L'évolution pour répondre à de nouvelles menaces (sécurités, actualité des données...)

Introduction

- Dans ce qui suit, on considère seulement des architectures multi-postes.

- Dans ce qui suit, on traite principalement des architectures les plus standards qui sont rencontrées (on se contente d'évoquer les mainframes pour se centrer sur le modèle client-serveur)

Plan

1) **Vers l'architecture client-serveur**

2) Les différents types de serveurs

3) Le cas des serveurs de base de données

4) Fonctionnement des serveurs

5) Le client

6) Autres architectures

7) Evolutions récentes

1) Vers l'architecture client-serveur

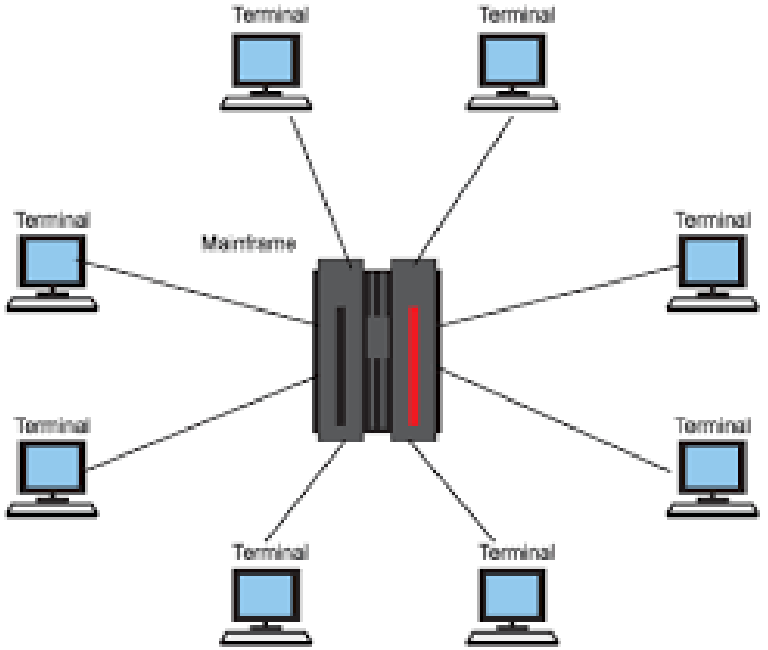
Mainframe / terminaux

Client / serveur

Pair à pair

1) Vers l'architecture client-serveur

- Le cas de l'architecture mainframe



- Cette architecture se retrouve encore dans les centre de recherche ou de traitement massif de données.

1) Vers l'architecture client-serveur

- Les architectures mainframe modernes
  - Architecture propriétaire.
  - Fournisseur unique en matériel et logiciel de base (OS).
  - Efficacité lié à l'homogénéité du système : le concepteur de logiciel n'a pas à se préoccuper du fonctionnement de ses logiciels sur des architectures physiques diverses.
- Avantages et inconvénients d'une solution entièrement propriétaire.
- Fournisseurs de ce marché : BULL et IBM
- Approximation du prix d'une solution mainframe : 15 000 $ à 200 000 $

1) Vers l'architecture client-serveur

- Mainframe => de nouvelles solutions avec l'émergence du micro-ordinateur (fin 70s).

- Mainframe -> micro-ordinateur -> PC => réduction de taille, miniaturisation (microprocesseur d'Intel)

- Mainframe traduit par "gros ordinateur" ou "ordinateur central".

1) Vers l'architecture client-serveur

- Premier microprocesseur commercialisé en 1971. Intel 404. 108kHz
-
- 1972 : Traf-O-Data : premier système commercialisé avec un microprocesseur candencé à 200 kHz et 16 Ko de mémoire vive.

-
- Traf-O-Data -> Microsoft.
- 80s => des ordinateurs de taille raisonnable qui ne sont plus seulement des terminaux.

1) Vers l'architecture client-serveur

- Début 80s : des systèmes fonctionnels produits par différents constructeurs, mais inaptes à communiquer entre eux

- 1983-84 : vers la compatibilité. Une norme s'impose le "compatible PC".

- Dès lors, on converge vers un monde avec deux architectures (outre celles des mainframes) : l'architecture PC et l'architecture d'Apple.

1) Vers l'architecture client-serveur

- 2005 : les machines d'Apple migrent vers des architectures à base de processeurs Intel (auparavant Motorola puis PowerPC)

- Evolution des applications qui ont du être redéfinies pour des processeurs Intel.

1) Vers l'architecture client-serveur

- L'évolution des micro-ordinateurs conduit à ce qu'il est possible de faire un serveur d'un micro-ordinateur.

- Développement conjoint des réseaux et des micro-ordinateurs => déploiement de nouvelles architectures

- Client serveur comme base de ce déploiement, même les mainframes sont utilisés en client-serveur désormais.

1) Vers l'architecture client-serveur

Exemple du Crédit Agricole : d'une architecture mainframe vers du client serveur (2007)

http://bfmbusiness.bfmtv.com/01-business-forum/le-credit-agricole-ose-cobol-sur-windows-351006.html

1) Vers l'architecture client-serveur

- Serveur : machine permettant le partage de ressources, machine rendant un service.
  - Serveur d'application
  - Serveur de données.

- Le serveur est souvent boosté : multiprocesseurs, support de charge de connexion, taille de la mémoire vive etc...

1) Vers l'architecture client-serveur

- Architecture client – serveur :
  - Des traitements sur le client et sur le serveur
  - Le client ne se contente pas de fonctions d'affichages
  - Différents niveaux de la relation sont possibles : cf client lourd VS client léger.

1) Vers l'architecture client-serveur

- Avantage d'un serveur de données.
  - Pas de resaisie
  - Même données partagées par tous
  - Les données évoluent pour tous dans le même temps
  - Sauvegarde du contenu d'une seule machine.
  - Securité des données : elles ne sont pas stockées sur les postes clients mais sur un point unique
- Les questions que cela peut poser :
  - Question de la modification simultanée de données ?

1) Vers l'architecture client-serveur

- Avantage d'un serveur applicatif.
  - Tous les clients bénéficient de la puissance du serveur.
  - Déploiement et mise à jour sur une mise à jour et non pas sur plusieurs.
- Les questions que cela peut poser :
  - Problème de licences.

1) Vers l'architecture client-serveur

- On parle d'architecture client / serveur
- On parle aussi d'architecture deux-tiers.
- Le principe et les avantages de l'architecture client-serveur se retrouve sur des architectures plus complexes : 3 tiers, n-tiers.
- On répartit les tâches entre des serveurs spécialistes efficaces qui mutualisent leur travail.

1) Vers l'architecture client-serveur

- La notion de client désigne aussi bien :
  - La machine qui se connecte à un serveur
  - L'application qui se connecte à un serveur,
  - La personne qui se connecte à un serveur.

Plan

1) Vers l'architecture client-serveur

2) **Les différents types de serveurs**

3) Le cas des serveurs de base de données

4) Fonctionnement des serveurs

5) Le client

6) Autres architectures

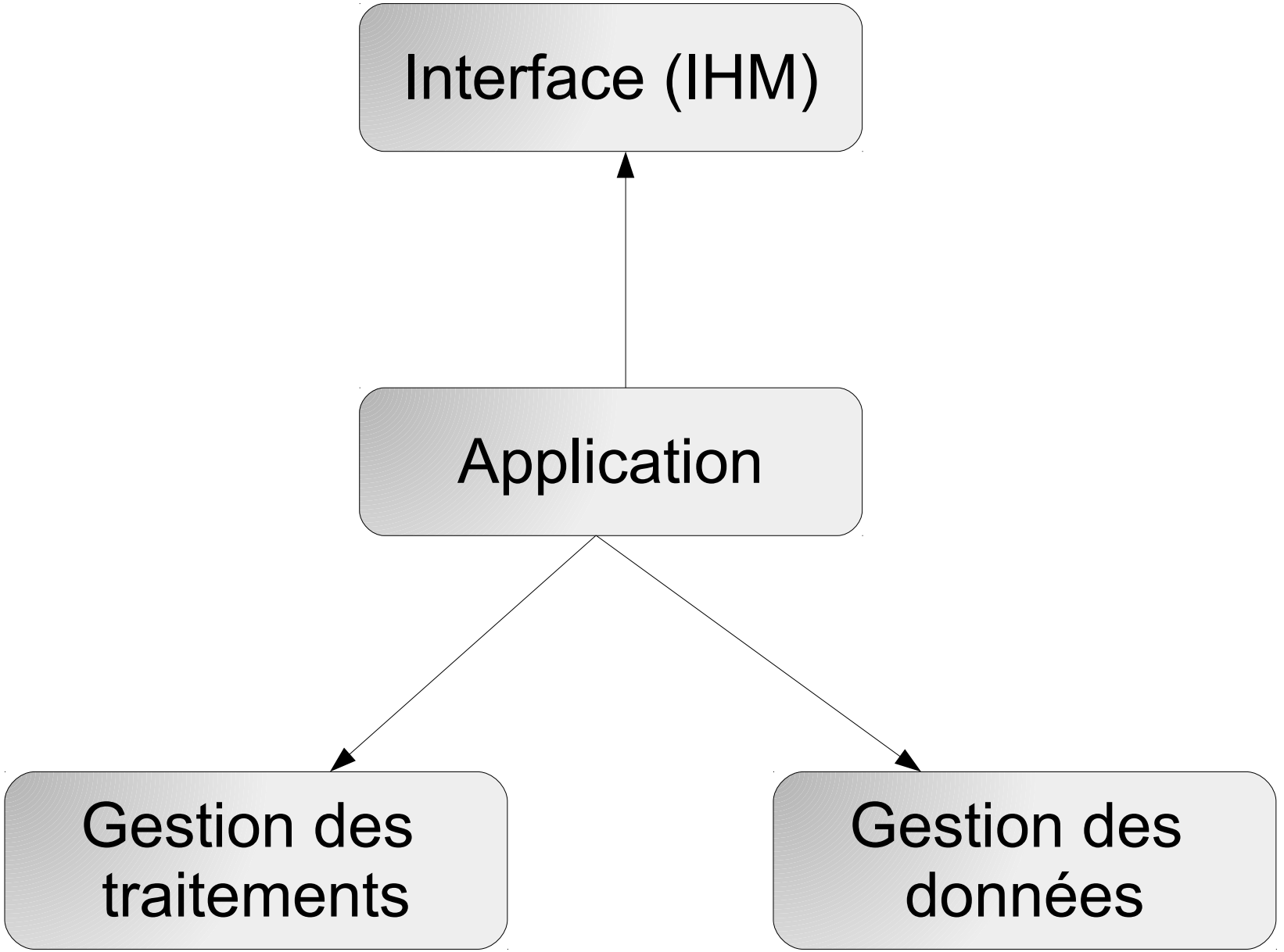
7) Evolutions récentes

2) Les différents types de serveurs

- Le modèle client-serveur définit le mode de fonctionnement d'une application dont les composantes se répartissent entre un logiciel client et un logiciel serveur qui communique en utilisant un protocole de communication.

- Par extension, "client" et "serveur" sont utilisés pour désigner les machines physiques qui supportet ces logiciels.

2) Les différents types de serveurs



2) Les différents types de serveurs

- Distinction par fonction des serveurs.

- Les différents plateformes.

2) Les différents types de serveurs

- **Serveur de fichiers.**
  - Serveur de données.
  - Espace de stockage centralisé et unifié.
  - Espaces publics et privés.
  - Définitions de niveaux de droit sur les fichiers

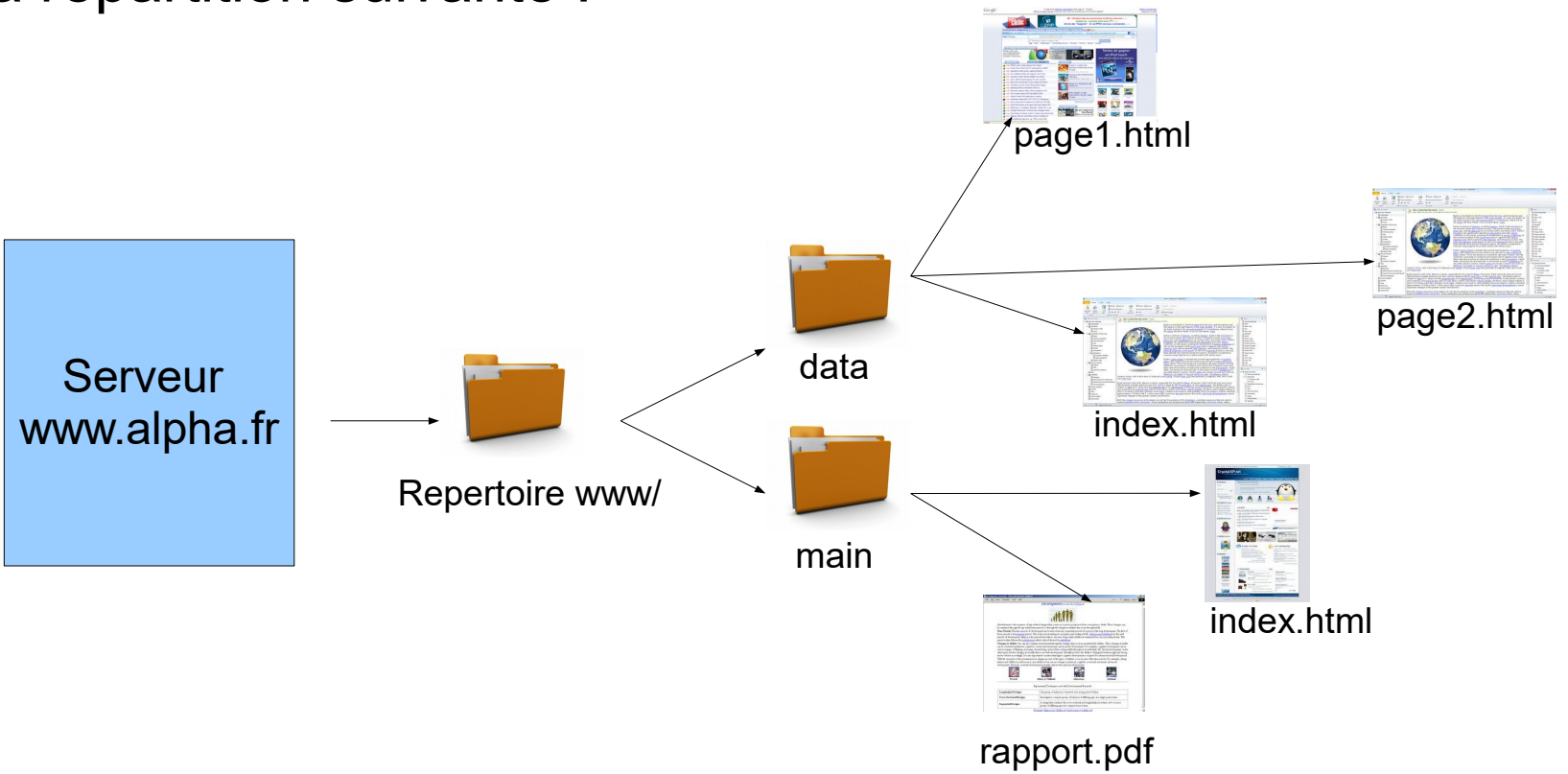
- Exemple de fichiers clients stockés sur un serveur de données commerciales. Le poste client a accès à une interface logicielle lui permettant de demander des fichiers selon certains critères. Il envoie sa demande aus serveur. Le tri des fichiers est exécuté sur le serveur pour ne pas saturer le réseau. => Répartition des traitements.

2) Les différents types de serveurs

- **Serveur Web.**
  - Serveur destiné à répondre à des requêtes HTTP.
  - Ce serveur dispose d'un répertoire www. La hiérarchie de ce répertoire est la hiérarchie du site Web.

2) Les différents types de serveurs

- Exemple du serveur www.alpha.fr. Sur ce serveur, on a la répartition suivante :



- Par suite, les adresses des document seront de la forme :

http://www.alpha.fr/data/page1

http://www.apha.fr/main/rapport.pdf

2) Les différents types de serveurs

- Dans le cas d'un serveur Web, le logiciel client est souvent un navigateur.
- Langage HTML / code source d'une page Web
- Normalisation du langage HTML.
- Autres langages supportés par un navigateur : CSS, XML, javascript
- Un navigateur dispose égalemente de plug-ins.
- Web 2.0 => XHTML (HTML normalisé en XML, CSS, javascript ou AJAX)
- Langage côté serveur et côté client.
- Côté client : javascript, CSS...
- Côté serveur : PHP, ASP, ...

2) Les différents types de serveurs

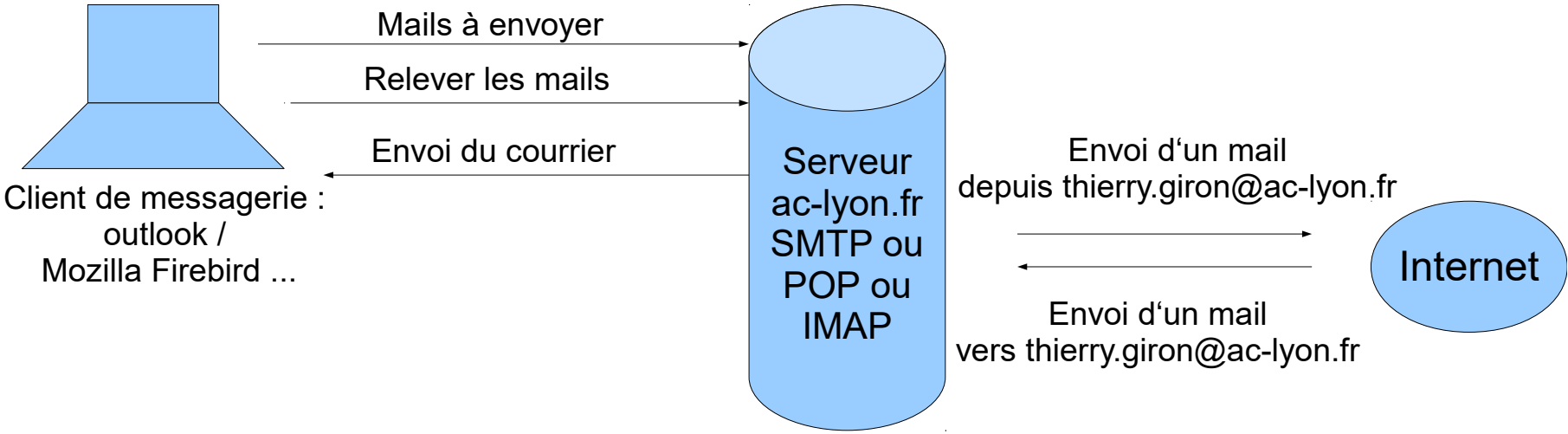
- **Serveur FTP**

2) Les différents types de serveurs

- **Serveur d'application**

2) Les différents types de serveurs

- **Serveur de messagerie**
- IMAP / POP / SMTP.
- Cas de l'académie de Lyon : les adresse sont de la forme thierry.giron@ac-lyon.fr



2) Les différents types de serveurs

- Cas classique en entreprise, cas des PGI

2) Les différents types de serveurs

- Différentes plateformes
- J2EE / .Net ("Dot Net") / LAMP
- http://www.journaldunet.com/solutions/dossiers/pratique/net_j2ee.shtml
- http://www.supinfo.com/articles/single/1337-j2ee-vs-net-choix-depend-vos-besoins

Plan

1) Vers l'architecture client-serveur

2) Les différents types de serveurs

3) **Le cas des serveurs de base de données**

4) Fonctionnement des serveurs

5) Le client

6) Autres architectures

7) Evolutions récentes

3) Les serveurs de base de données

- Le serveur de base de données est une référence pour le stockage des données

- Serveur sur lequel est déployé un SGBD.
- Repose souvent sur des bases de données relationnelles.
- Optimisation / Minimisation de l'espace de stockage.

3) Les serveurs de base de données

- Le logiciel de gestion des BD :
  - Assure l'organisation des données
  - Leur stockage
  - Permet l'accès au données
  - Assure des fonctions de sécurité
  - Permet de faire évoluer les données stockées.

3) Les serveurs de base de données

- Rôle central du SQL, adopté par tous les SGBD. Structured Query Language.
  - Commandes DDL – Data Definition Language. Création des tables, modification des tables, ...
  - Commandes DML – Data Manipulation Language. SELECT, INSERT, UPDATE, DELETE.
  - Commandes DCL – Data Control Language. Sécurité de structures et des données
  - Commandes TCL – Transaction Control Language. Elles permettent de rendre cohénretes, sur la base de données, un ensemble d'opérations liées.
  - Commandes SQL Procédural. Permettent d'automatiser des traitements sur la base de données

3) Les serveurs de base de données

- Exemples de requêtes DDL : CREATE TABLE, ALTER TABLE...
- Exemples de requêtes DML + sémantique enrichie par l'imribication
- Exemples de requêtes DCL :

```
CREATE USER Util_127 IDENTIFIED BY "tkf821"
GRANT CONNECT TO Util_127
GRANT SELECT ON Clients TO Util_127
```

3) Les serveurs de base de données

- Les commandes TCL (Transaction Control Language). Gestion des transactions. Une transaction => "le regroupement logique d'un ensemble d'ordres SQL de modification des données d'une base."
- Une transaction a pour but de garantir la cohérence des données lors d'une mise à jour.
- Cf ACID : Atomicité, cohérence, Isolation, Durabilité.

3) Les serveurs de base de données

- Dans une trasaction, on distingue :
  - Un début de transaction
  - Une série d'ordres DELETE, INSERT, UPDATE
  - Un ordre de validation ou un ordre de retour en arrière, au dernier état stable connu.

3) Les serveurs de base de données

- Exemple du cas d'une gestion commerciale
- Base de données avec des commandes et des lignes de commandes.
- Lors des livraisons, les lignes de commandes passent de "à livré" "en cours")
- Les commandes pour lesquelles toutes les lignes sont "en cours" devienent "livrées".
- => beaucoup de mises à jours conjointes
- Problème pendant le traitement : il faut pouvoir revenir en arrière.

3) Les serveurs de base de données

- Tant qu'elle n'est pas terminée, une transaction est réversible.
- On peut revenir dans l'état stable antérieur.
- Fonctionnement similaire à celui du brouillard d'écriture dans les logiciels comptables.

3) Les serveurs de base de données

- SGBDR qui sont des standards
- Oracle, PostgreSQL, MySQL, ...
- Il existe d'autres formes de données : bases de données hiérarchiques, fichiers séquentiels indexés, base de données objet.

Plan

1) Vers l'architecture client-serveur

2) Les différents types de serveurs

3) Le cas des serveurs de base de données

4) **Fonctionnement des serveurs**

5) Le client

6) Autres architectures

7) Evolutions récentes

4) Fonctionnement des serveurs

- La communication entre le serveur et le client.

- Le client et le serveur échangent des données

- Interface native => ok, mais que se passe t'il dans le cas de logiciels différents, de plate-formes hétérogènes ?

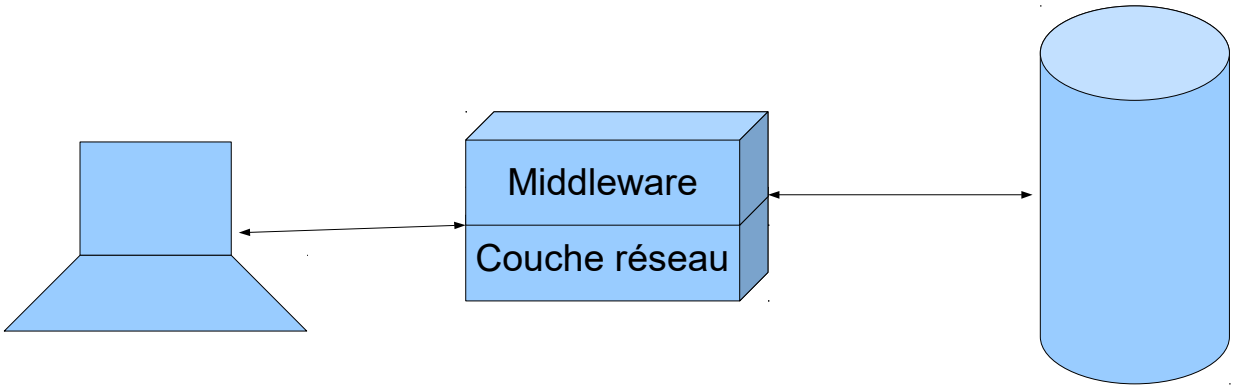
4) Fonctionnement des serveurs

- Notion de middleware.
- Il s'agit d'un logiciel ou ensemble de logiciels permettant :
  - De faire communiquer les programmes situés sur des machines différentes, voire avec des systèmes d'exploitation différents
  - De faciliter le travail des informaticiens pour l'élaboration d'applications nouvelles en fournissant toute une série de services qui évitent du travail de programmation.
  - De masquer à l'utilisateur la réparatition des donnée et des traitements d'une application. L'utilisateur a l'impression de travailler "en local"

4) Fonctionnement des serveurs

- Notion de middleware.
- Il s'agit d'un logiciel ou ensemble de logiciels permettant :
  - De faire communiquer les programmes situés sur des machines différentes, voire avec des systèmes d'exploitation différents
  - De faciliter le travail des informaticiens pour l'élaboration d'applications nouvelles en fournissant toute une série de services qui évitent du travail de programmation.
  - De masquer à l'utilisateur la réparatition des donnée et des traitements d'une application. L'utilisateur a l'impression de travailler "en local"

4) Fonctionnement des serveurs



4) Fonctionnement des serveurs

- Exemple d'un traitement de texte sur une machine dans un réseau avec différentes imprimantes.
- La machine client ne peut pas être capable de gérer l'ensemble des imprimantes possibles
- L'imprimante dispose donc d'un pilote ou driver qui sait communiquer avec l'extérieur de manière standard, et avec les programmes propres à l'imprimante.

4) Fonctionnement des serveurs

- Exemple d'OBDC qui permet à un logiciel de manipuler plusieurs SGBD ne disposant pas d'interfaces natives entre eux.
- ODBC => d'abord développé par Microsoft puis repris sous Unix et intégré à Java.

- Exemple de RPC : Remote Procedure Call.

4) Fonctionnement des serveurs

- Le cas des Webs services : un Middleware universel

- HTTP et Xml.
- WSDL : Web Services Description Language (standard W3C)
- SOAP : RPC entièrement basé sur XML.

4) Fonctionnement des serveurs

- Problème de la sécurisation quand la communication passe par le réseau Internet.

- Cf Intranet et extranet + télétravail
- Solution de canaux spécialisés.
- Solution de l'utilisation du réseau Internet + VPN

4) Fonctionnement des serveurs

- VPN : authentification + cryptage.
- Il exsite deux protocoles principaux : SSL et IPSec.
- SSL est supporté par les navigateurs, IPSec demande l'utilisation d'un client spécifique.

4) Fonctionnement des serveurs

- Architecture classique d'un ERP.

Plan

1) Vers l'architecture client-serveur

2) Les différents types de serveurs

3) Le cas des serveurs de base de données

4) Fonctionnement des serveurs

5) **Le client**

6) Autres architectures

7) Evolutions récentes

5) Le client

- Une distinction principale est à faire entre client lourd et client léger.
- Dans le cas du client lourd, la machine cliente prend en charge une large partie des traitements, alors que dans le cas du client léger, c'est le serveur qui prend en charge la majeure partie des traitements.
- Dans le cas d'un client lourd, le déploiement est plus complexe : le client doit être configuré pour exécuter les traitements nécessaires.

5) Le client

- Exemple de client lourd.
- Dans un cabinet comptable, on installe une nouvelle solution informatique.
- Cette installation se fait autour d'un serveur de données central qui contient toutes les informations / données client.
- Toutes les machines des collaborateurs doivent être configurées : un logiciel doit y être installé, qui permet l'accès et le traitement des données du serveur de données.
- On parle ici de client lourd.

5) Le client

- Un client Web est un client léger : la machine cliente doit disposer d'un navigateur et ne gère que l'affichage (ou quasiment).

- Le navigateur est le client léger de référence.

- Il pose cependant le problème de la pauvreté des interfaces qu'il permet.

5) Le client

- Des solutions sont proposées pour émuler un client riche à partir de XML

  - XAML ("Zammel", standard Microsoft),
  - XUL ("zoul", standard XML de Mozilla)
  - Flex (standard XML de Macromedia)

- Ces technologies permettent de décrire, en langage XML, une application qui pourra ensuite apparaître sur le client.

5) Le client

- Autres exemples
- Un client de messagerie (outlook) est considéré comme un client lourd
- Un jeu en ligne sera considéré comme un client lourd
- Le logiciel installé pour une application comptable et communiquant avec le serveur d'application.
- Etc..

Plan

1) Vers l'architecture client-serveur

2) Les différents types de serveurs

3) Le cas des serveurs de base de données

4) Fonctionnement des serveurs

5) Le client

6) **Autres architectures**

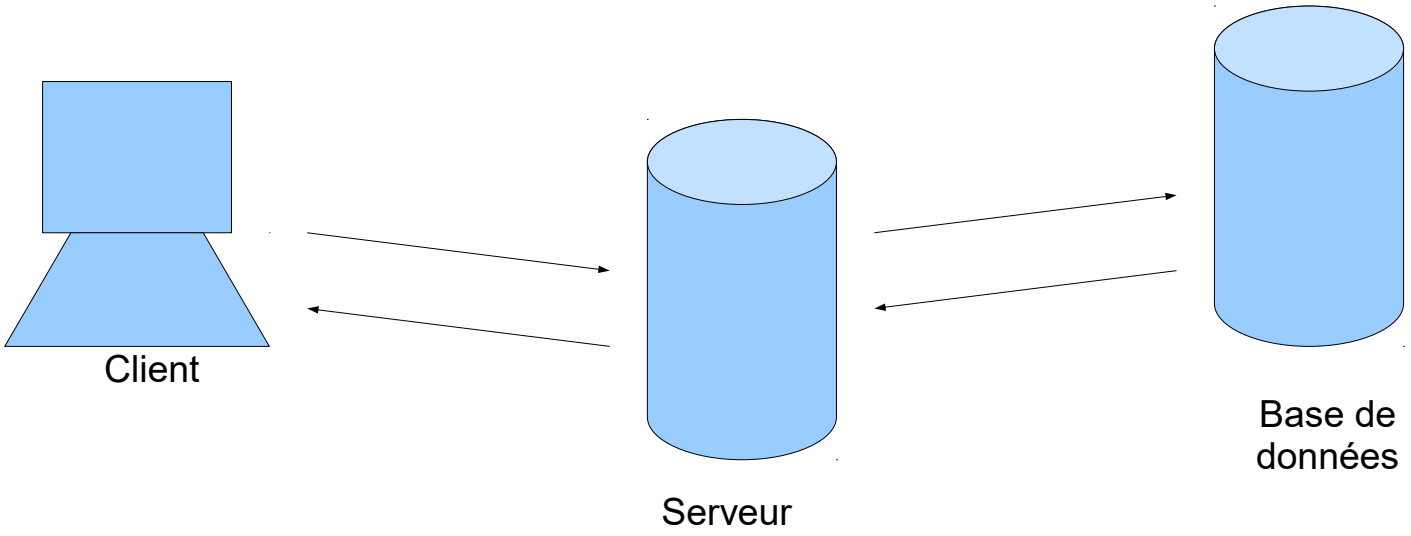
7) Evolutions récentes

6) Autres architectures

- A partir du client serveur.
- Les serveurs comme des clients d'autres serveurs.
- Client-Serveur / 2 tiers -> 3 tiers -> N tiers.
- Application répartie.

6) Autres architectures

- Architecture 3-tiers classique :



- Exemple d'une société de commerce en ligne. Exemple d'interaction

6) Autres architectures

- Les serveurs DNS : des requêtes entre eux.

- Le cas des routeurs et des tables de routage.

- Le cas de base de données sur plusieurs serveurs.

- Architectures pair à pair ?

Plan

1) Vers l'architecture client-serveur

2) Les différents types de serveurs

3) Le cas des serveurs de base de données

4) Fonctionnement des serveurs

5) Le client

6) Autres architectures

7) **Evolutions récentes**

7) Evolutions récentes

- Intranet
- Avantages et inconvénients

- Extranet
- La notion de portail

7) Evolutions récentes

- La Virtualisation. Il s'agit de faire tourner des logiciels, voire différents OS sur un même serveur

- Du point de vue de l'extérieur : plusieurs serveurs distincts.

- Des serveurs dont l'utilisation peut être de 10%, passent à une utilisation de 80% (Vmare).

7) Evolutions récentes

- Cloud computing.

- Définition de Syntec Infrmatique : "L'interconnexion et la coopération de ressources informatiques, situées au sein d'une même entité ou dans diverses structures et dont les modes d'accès sont basés sur des protocoles et standard d'Internet."

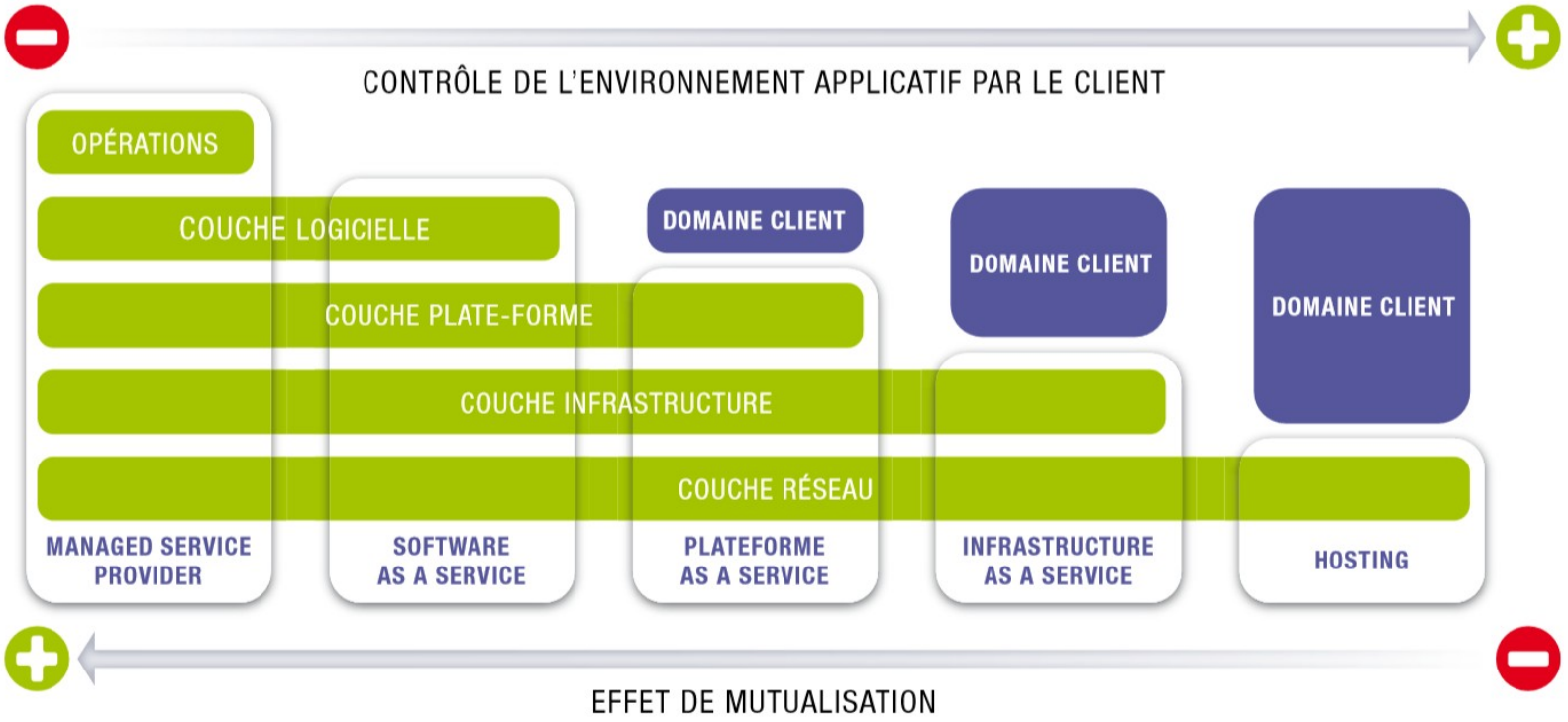
7) Evolutions récentes

- Cloud computing.

- Les caractéristiques :
  - Considéré comme un service, l'entreprise utilisatrice ne dispose plus des produits ni des technologies.
  - Le paiement s'effectue selon la consommation. Une charge fixe devient une charge variable.
  - Mutualisations des ressources physiques => retour d'expérience, économies d'échelles, nouveaux processus et dernières innovations sont disponibles sans veille et sans formation

7) Evolutions récentes

- IAAS
- PAAS
- SAAS
- Depuis le livre blanc du cloud compunting, site de Syntec Informatique :



7) Evolutions récentes

- Avantages du Cloud ?

- Risques et inconvénients ?

7) Evolutions récentes

- Les nouveaux déploiements se coordonnent avec des évolution des modes de travail

- Nommadisme et télétravail <-> Intranet, client serveur, middleware standardisés.

- Extrenalisation <-> Cloud-computing, client serveur, middleware standardisés.

- Des technologies séléctionnées par des modes d'organisation ou des modes d'organisation induits par des technologies ?

Conclusion

- Chercher la meilleure configuration :
  - Part d'externalisation ?
  - Sécurité ?
  - Adaptation à l'organisation
  - L'inertie de l'existant.
  - Gestion du changement