Chapitre 5

Vision Informatique Logique
Vision Technologique

NFE107

Chapitre 5

Vision Informatique Logique
Vision Technologique

5.8 Les différents types de midleware

Les différents types de middleware

- Le middleware orienté transactions (MOT) assure la gestion cohérente des transactions complexes
  - Exemple : CICS, IMS, Tuxedo, TDS (GCOS), …
- Le middleware orienté objets (MOO) permet de relier des composants logiciels réalisés sous forme d'objets et de gérer des transactions
  - CORBA, EJB, DCOM/COM+
- Le middleware orienté messages (MOM) permet aux applications de communiquer et d'échanger des informations de manière asynchrone par la transmission et la mise en file d'attente de messages
  - MQSeries, SonicMQ, JMS, …
- Le bus logiciel ainsi formé par cet ensemble de middleware distribué assure la communication synchrone et asynchrone entre les applications

Mode de communication
Synchrone ou asynchrone ?

- Les architectures distribuées et l'émergence de technologies de communication entre applications imposent des choix de conception architecturale. Une des décisions importantes à prendre reste le mode de communication : synchrone ou asynchrone ?

- **Mode synchrone**
  - Dans un mode synchrone, l'émetteur doit localiser le destinataire sur le réseau, se connecter à l'une de ses instances et attendre la réponse
  - Dans un tel mode, la requête n'est jamais perdue. En cas d'erreur ou de rejet, l'application émettrice en est directement avertie
  - Les technologies RPC, Corba, RMI, EJB Session/Entity, DCOM/COM+, Web Services (SOAP/HTTP) entrent dans cette catégorie

- **Mode asynchrone**
  - A l'inverse, dans un mode asynchrone, la disponibilité du destinataire au moment de l'émission n'est plus indispensable
  - Celui-ci traitera la requête en temps voulu. Charge au middleware utilisé de garantir alors que la requête ne se perde pas
  - Les technologies de Message Oriented Middleware comme IBM MQSeries, Microsoft MSMQ, SonicMQ, les EJB MDB entrent dans cette catégorie

Mode d'échanges
Batch ou TP ?

- **Batch**
  - Conçu bien pour des gros traitements (nécessite beaucoup de temps CPU et de mémoire)
  - Mise à jour fichier quand le batch est traité
  - La performance d'un Batch est mesurée en capacité de traitement. Le temps de réponse mesurés en heures
  - Traitements (jobs) non interactifs
  - Une application Batch (Batch Processing) est généralement monoprogrammé (une transaction à la fois), aussi la sérialisation du traitement est simple
  - Le Batch peut être optimisé en triant les enregistrements par la clé du fichier, on peut aussi fusionner les enregistrements dans un fichier avant de faire appel au traitement batch
  - Le Batch a une charge fixe et prévisible, contrairement au TP

- **Transaction Processing**
  - Conçu pour de petits traitements
  - Mise à jour quand la transaction est traitée
  - La performance TP est mesurée en temps de réponse. Il est de l'ordre de quelques secondes après chaque transaction
  - Le TP doit gérer l'arrivée des transactions de façon aléatoire

Mode de couplage
Fort ou faible ?

- Fort
  - Mode connecté
  - Echange en mode synchrone (concerne principalement le mode TP)
  - Service avec état (stateful)

- Faible
  - Mode déconnecté
  - Echange en mode asynchrone (middleware orienté message)
  - Service sans état (stateless)

- Dans les SI complexes, il faut bien doser l'usage du couplage fort et du couplage faible en évitant les couplages forts dans des périmètres étendus

Intégrité transactionnelle

- La transaction ACID
  - **Atomicité** : une transaction doit être entièrement effectuée ou pas du tout (par exemple, lors de la mise à jour de n éléments d'une table)
  - **Cohérence** : la cohérence entre tables d'une même base doit être respectée, même en cas d'incident
  - **Isolation** : pendant une transaction complexe, les autres transactions voient la totalité des données à modifier dans l'état antérieur au démarrage de la transaction jusqu'à son achèvement
  - **Durabilité** : lorsque la transaction est achevée, les modifications sont définitives, même en cas d'incident

- Dans le cadre d'architecture distribuée, il est primordial de traiter cette problématique
  - Exemple : Utilisation d'une SOA avec des Services Web XML. Comment garantir l'ACIDité d'une transaction répartie ?

Volumétries

- **Volumétrie Statique**
  - Nombre d'enregistrements
  - Taille d'enregistrement (octets)
  - Historisation (rétention, archivage)
  - Estimation du volume total des données fonctionnelles, y compris les données historisées
  - Volume physique total…

- **Volumétrie Dynamique**
  - Nombre d'utilisateurs du système (internes, externes, filiales et partenaires)
  - Nombre de transactions simultanées
  - Estimation du nombre de connexions SGBD simultanées
  - Estimation du nombre de requêtes SQL (profils de requêtage)
  - Ratios écriture/lecture
  - Batchs
    - Nombre
    - Fréquence
    - Durée…

Les tendances du marché

- Les progiciels : le développement des notions de gestion de la relation client (CRM), chaîne logistique (SCM), achats, ou conception de produits ont été intégrées aux offres des mastodontes du secteur (Oracle/PeopleSoft, SAP, …) par acquisition ou développement interne

- La convergence vers des socles d'infrastructure standard : J2EE et .NET pour l'exécution des traitements, bus EAI, ESB et Services Web pour l'interopérabilité entre modules, outils PKI pour la sécurisation des échanges, ...

- L'Open Source : pour contrer cette offensive, une course à l'intégration va s'engager. La contre-offensive consiste à proposer le socle de base - à la limite gratuitement, en y associant une offre de modules, d'outils et de services connexes

Chapitre 5

Vision Informatique Logique
Vision Technologique

5.9 Architecture REST

Soyons utopiques et réalistes

- Cette présentation a pour but de montrer comment accéder facilement aux données :
  - Distribuées
  - De façon performante
  - De générer rapidement des milliers de services
  - Avec de bonnes performances

ou comment tirer parti des moteurs de recherche ...

L’architecture du Web appliqué aux Systèmes d’informations

- Quels sont les grands principes du Web ?

REST : le style d'architecture du Web

Le style d'architecture REST a été défini par les fondateurs de l'Internet.
C'est le style d'architecture promu par Google, Amazon, Microsoft (pour la partie Live)

REST : les ressources

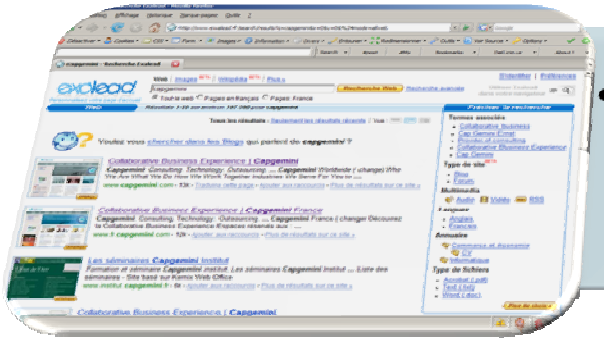
- Tout est “ressource” : documents, PC, Vidéos, Fiche client …
- Toutes les ressources sont accessibles via des URL
  - http://entreprise/Client/33/
  - http://entreprise/Contrat/246532/
  - http://monpc/os/network
  - skype:myaccount?call

- Les ressources doivent être compréhensibles et mémorisables (par un être humain ou par un navigateur web)
- Une ressource peut avoir différents types : html, xml, image … C’est a l’appelant de demander le format souhaité
- Les ressources peuvent être manipulées via une API universelle

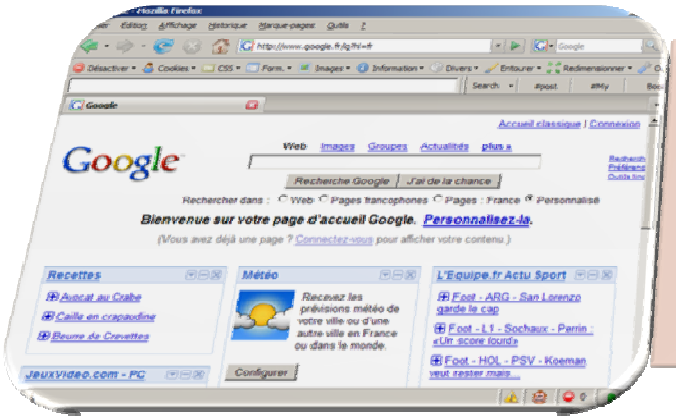
REST : les liens

- Les ressources sont ‘hypertext’
  - La fiche client contient les liens vers les adresses, contrats …
- We don’t want “keys”, we want links!
  - Une ressource a une URI unique, c’est un identifiant universel
  - Coté pratique : il n’y a pas besoin de routage pour retrouver donnée, l’identifiant est l’adresse !
  - Le Web sémantique exploite ce concept pour permettre de ‘requêter’ le Web
- Les liens doivent être durables et logiques (versus physiques)
  - http://entreprise/Client/33 (et non http://EGED234.FTE/DRG_34_V3/servletloader?CID=ERFRKFE)

Les utilisateurs en entreprise connaissent les mêmes problèmes que les internautes …

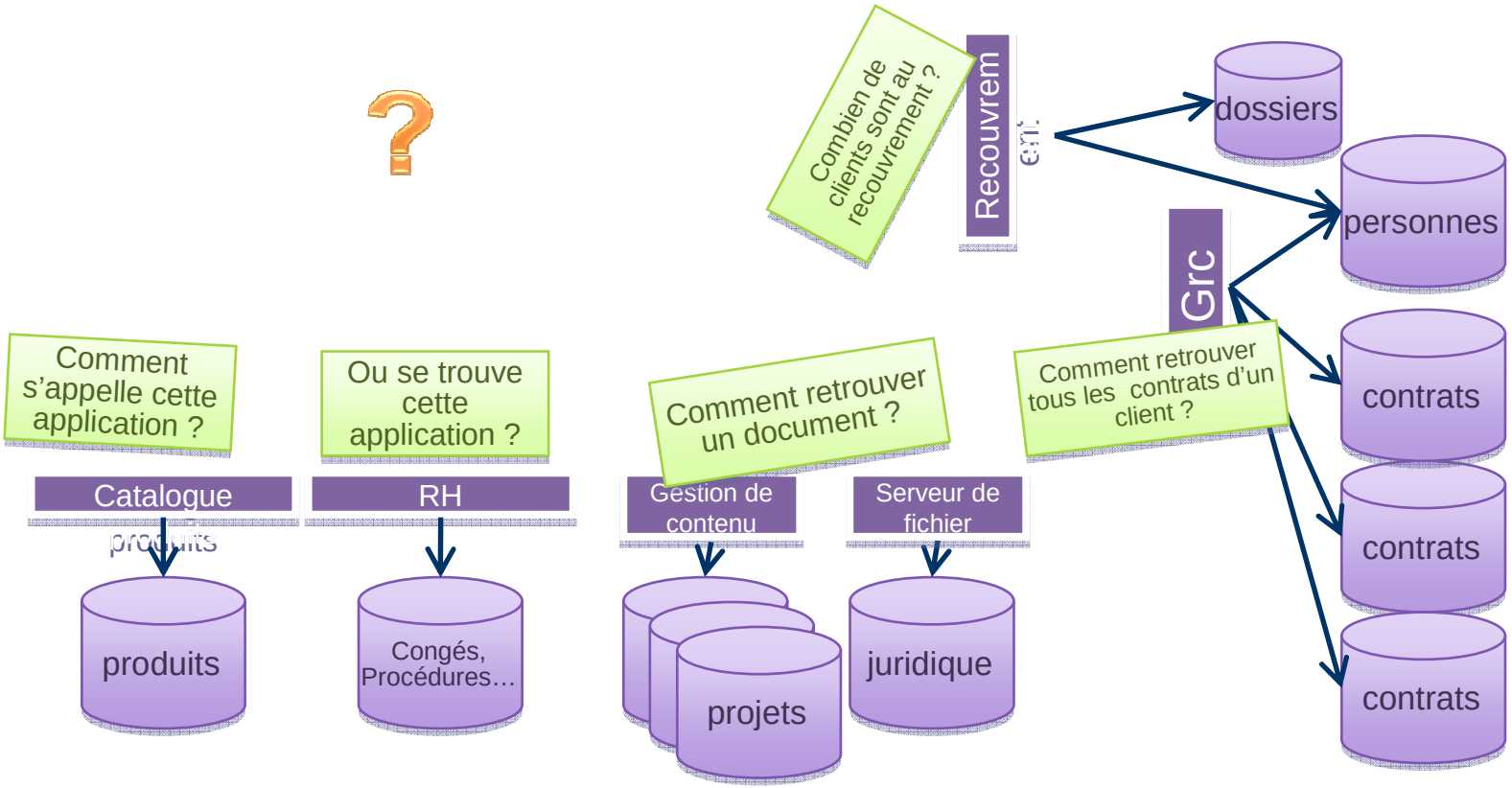


- Comment retrouver de l'information parmi les millions de sites web ?



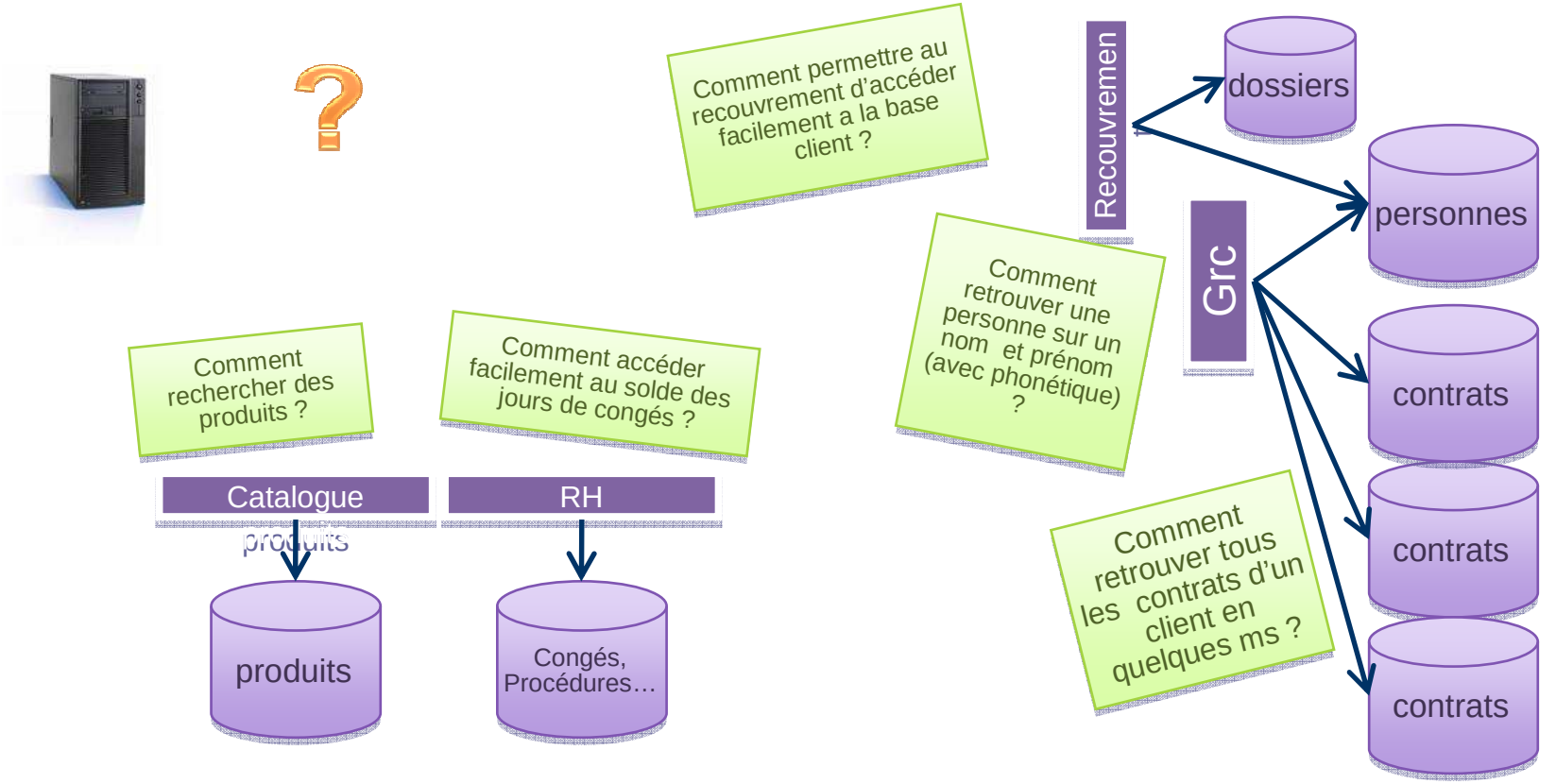
- Comment être alerté des ajouts/modifications d'articles sur les sujets qui m'intéressent ?

Des nombreuses informations sont stockées…



Mais restent peu accessibles par les utilisateurs…

…et peu accessibles par les applications



De nombreuses interfaces a développer, interfaces hétéroclites …

Les moteurs de recherche ont étendu les fonctionnalités

Indexation

- Sources
  - SGBD
  - Annuaires
  - Fichiers
  - …
- Formats
  - Ppt, docs, Excel
  - Images…

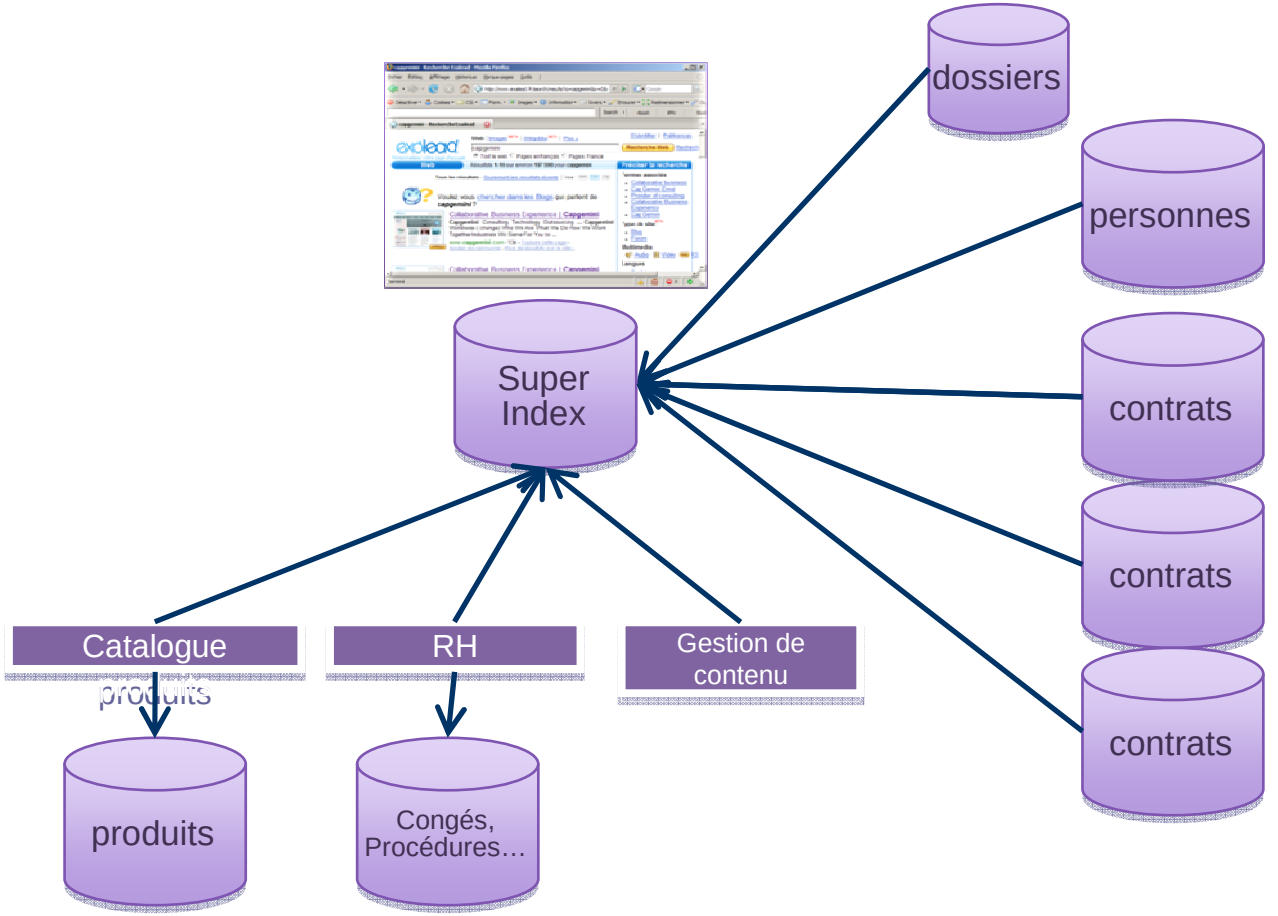
Recherche

- Recherche de **mots** dans les documents
- Recherche **approximative** (phonétique, lemmatique)
- Recherche **formelle** (recherche des documents ou auteur=Dumas).

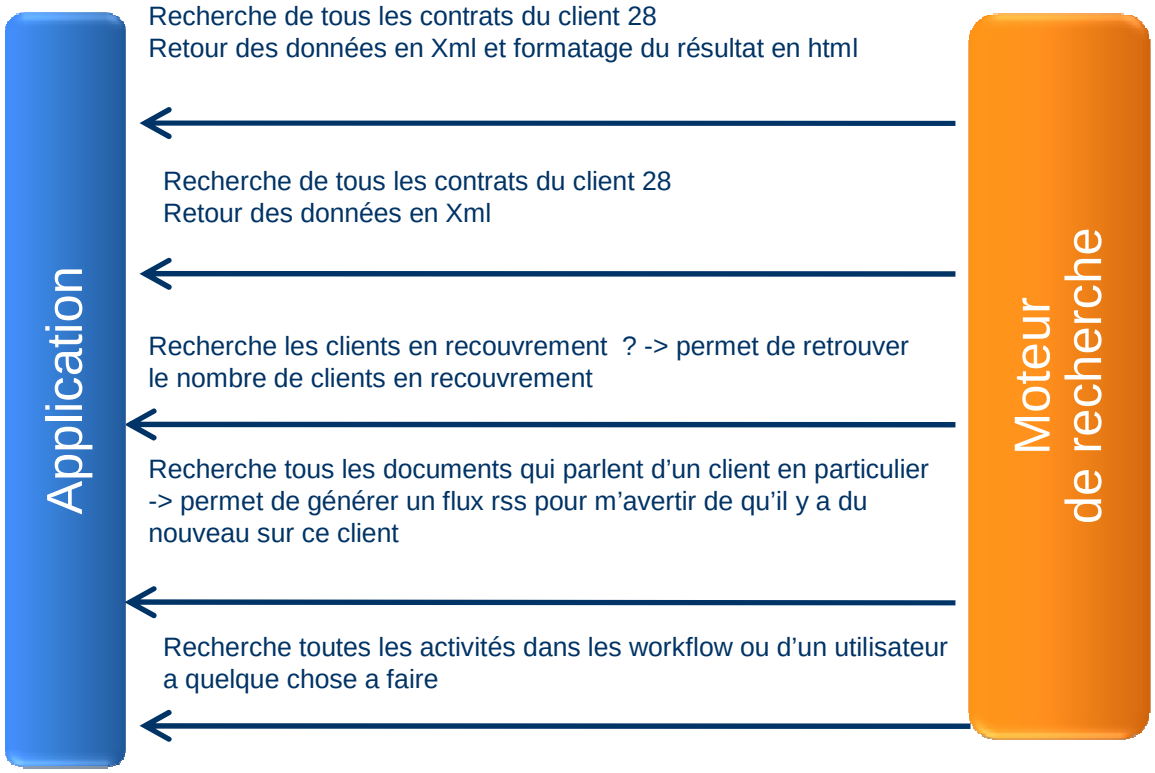
Ce qui permet maintenant d'adresser de nombreux besoins propres aux systèmes d'informations des entreprises

Une seule interface de recherche d'informations

Une recherche sur un client (par exemple) permet de retrouver les mails échangés, sa fiche client, ses contrats … Un click permet d'accéder a la ressource souhaitée (fiche client, mail…)



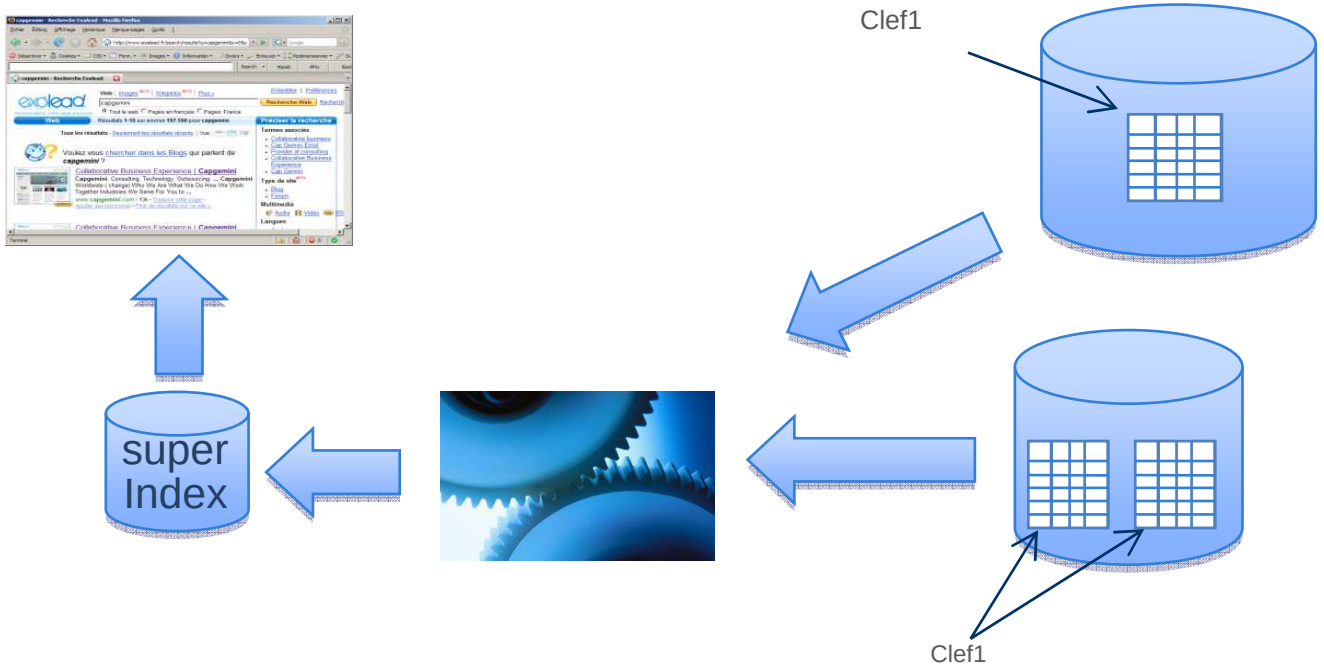
Ce qui permet également d'avoir une "super" API



Le moteur permet la recherche formelle et peut retourner les données indexées en xml ce qui permet aux applications d'exploiter l'information.
Il est possible de filtrer sur les sources (base client, contrat…), sur des valeurs de champs (clientId …), des dates …
Les données sont retournées en quelques centaines de ms, pour les outils de relation client le moteur peut se justifier uniquement par la possibilité d'acceder aux contrats sur n backoffice en quelques ms

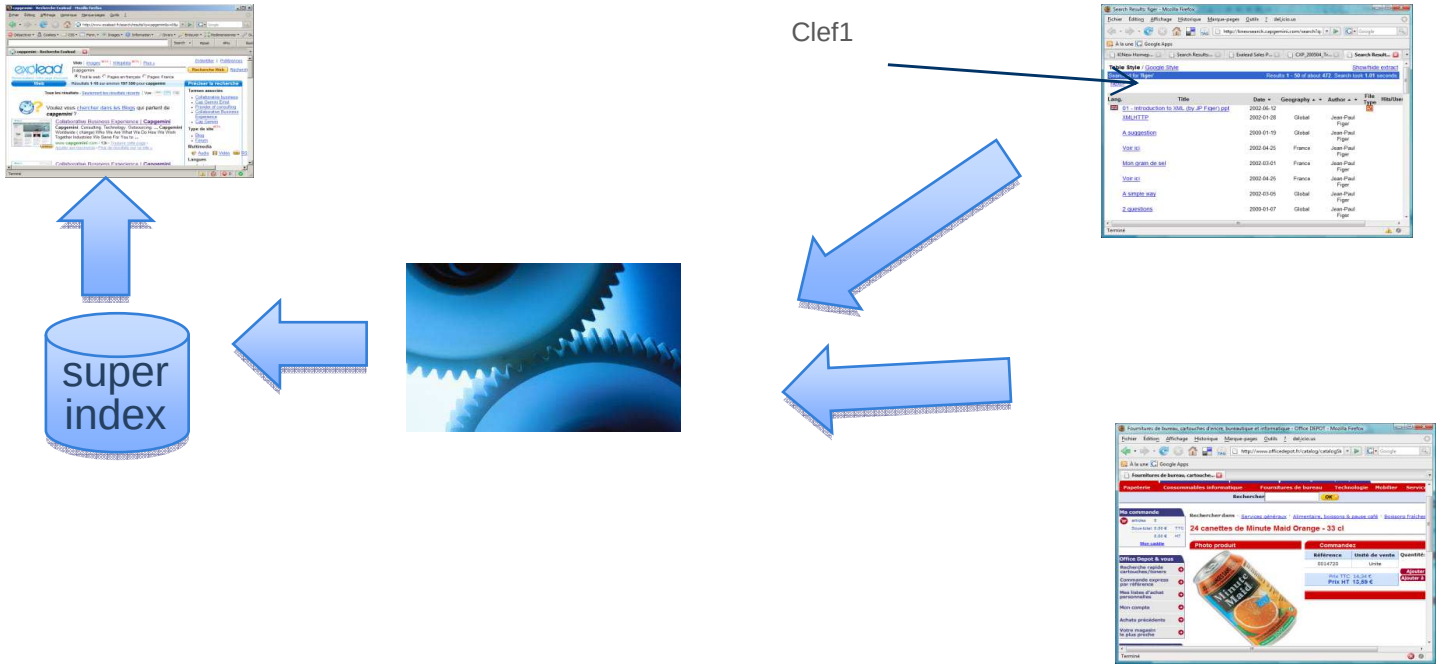
Les applications peuvent consommer des données aussi facilement qu'un utilisateur peut faire des recherches. Les services n'ont plus besoin d'être définis à priori

Le moteur sait indexer des tables ou des vues des bases de données ...



Ce qui permet des interrogations rapides sur les données du système d'information même si l'interrogation porte sur des dizaines des bases de données, les données sont dans l'index !

Il peut aussi indexer des sites Web



L'indexation de sites est possible pour les applications internes …. mais aussi externes !
Les meta données des pages web peuvent être utilisées pour faire de la recherche formelle

ou même des services ...

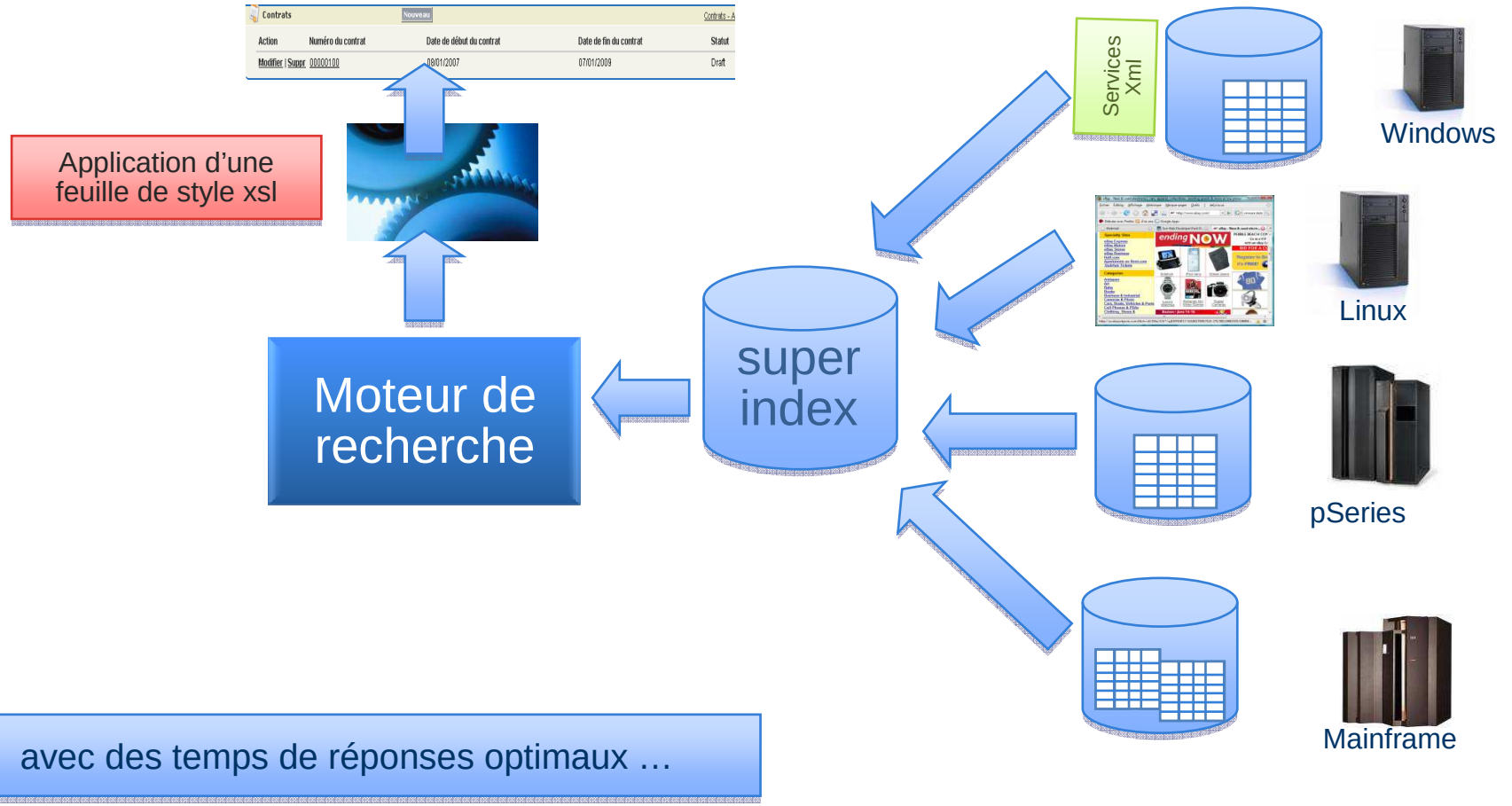
super Index

Services Xml

Fichiers plats ....

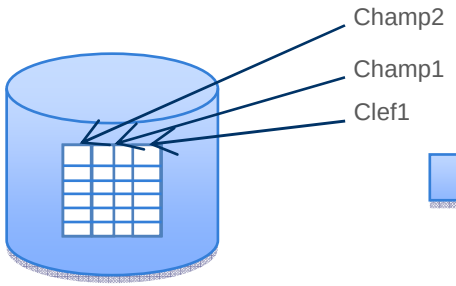
Il est possible d'intégrer par appels de services ou de fichiers

Exemple d'affichage des données consolidées …



avec des temps de réponses optimaux …

Comment indexer et renvoyer des liens ?



Résultats de la recherche

Champ1

Lien : http://entreprise/ressource/Clef1

Quelles tactiques pour disposer des liens vers des ressources ?

Utiliser le mod_rewrite dans apache pour disposer d'url 'propres' en façade des url physiques utilisées par les applications (ressource.asp, client.php…)

Créer les pages Web par l'utilisation du moteur de recherche.
Exemple :
• Recherche sur le champ1 'AAA' ➔ les liens retournent des liens vers le moteur de recherche en filtrant sur la clef de chaque ligne correspondante au critère
• L'affichage du moteur est personnalisé pour afficher la ressource :

Champ1 Champ2

Modifications temporaires (mais rapides) de l'application :
• Création des 'points d'entrée' qui permettent d'afficher directement les données voulues. C'est possible avec les applications existantes en Java, Php … mais aussi avec des outils de revamping mainframe (via des scripts)
• Cela permet de réutiliser de la logique applicative

Points essentiels

- Les moteurs de recherche savent indexer les bases, les applications…
- Ils permettent des recherches approximatives (phonétique…) et formelles
- Ils permettent d'obtenir des vues agrégées de données de nombreux systèmes en gardant de très hautes performances
- Et c'est performant
  - Un seul index, recherche distribuée avec de nombreux threads, traitements en mémoire, pas de transactions

Bibliographie

- Wikipedia: Achitecture REST (http://fr.wikipedia.org/wiki/Representational_state_transfer)
- Présentation de l'architecture REST, le bureau électronique de JP Figer (http://www.figer.com/Publications/REST.htm)
- Communauté des architectes Capgemini (http://architectes.capgemini.com/images/innovmail.pdf)
- Introduction à REST (http://www.xfront.com/REST-Web-Services.html)

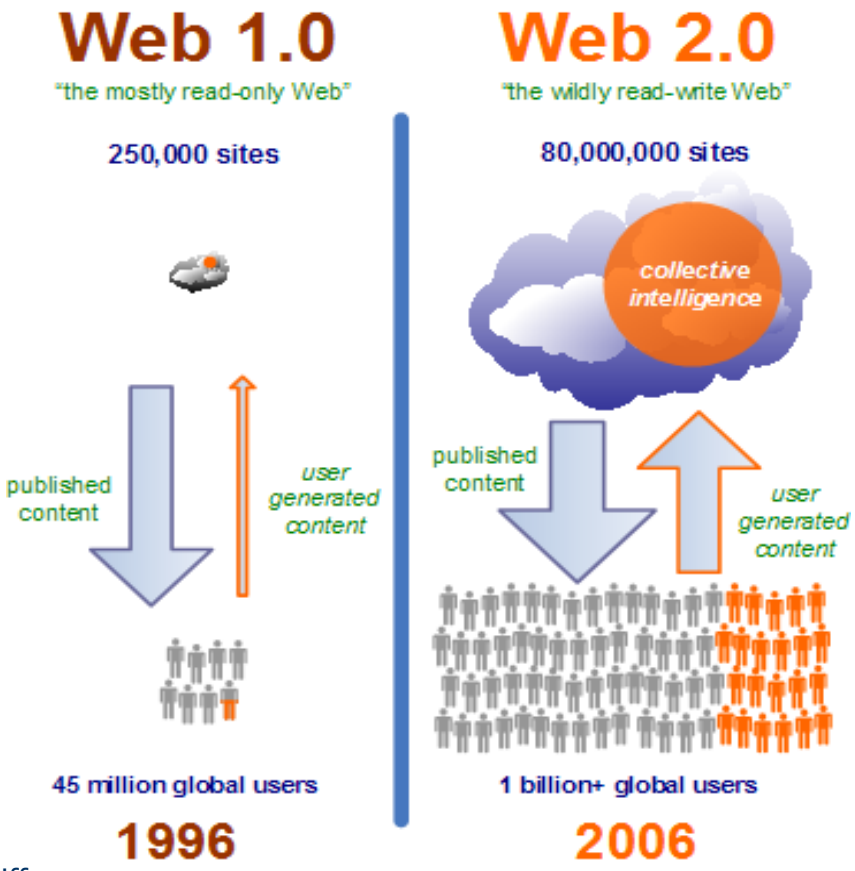
Chapitre 5

Vision Informatique Logique
Vision Technologique

5.10 Web 2.0

Le Web 2.0, un média dans lequel le contenu est produit par les utilisateurs

Aller sur Internet



Etre sur Internet

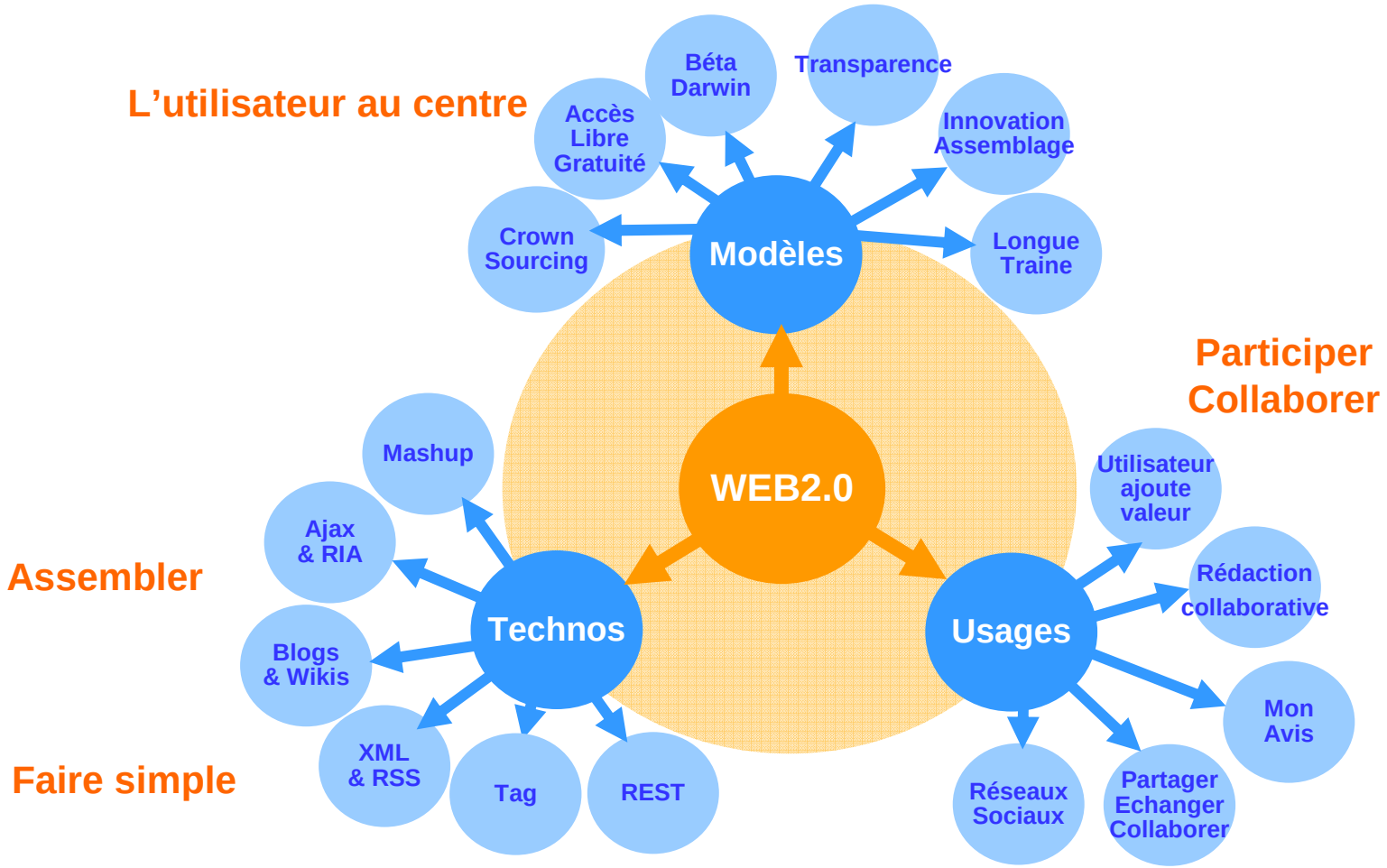
Le Web participatif
Augmentation de l'Intelligence collective

Internautes organisés en réseaux sociaux

SOME RIGHTS RESERVED

Dion Hinchcliffe

Evolution des technologies, Révolution des usages



Concepts du « web 2.0 »

- Concepts Web 2.0 apparu lors d'une conférence de « brainstorming » entre O'Reilly et Medialive International (Article daté du 30/09/2005)

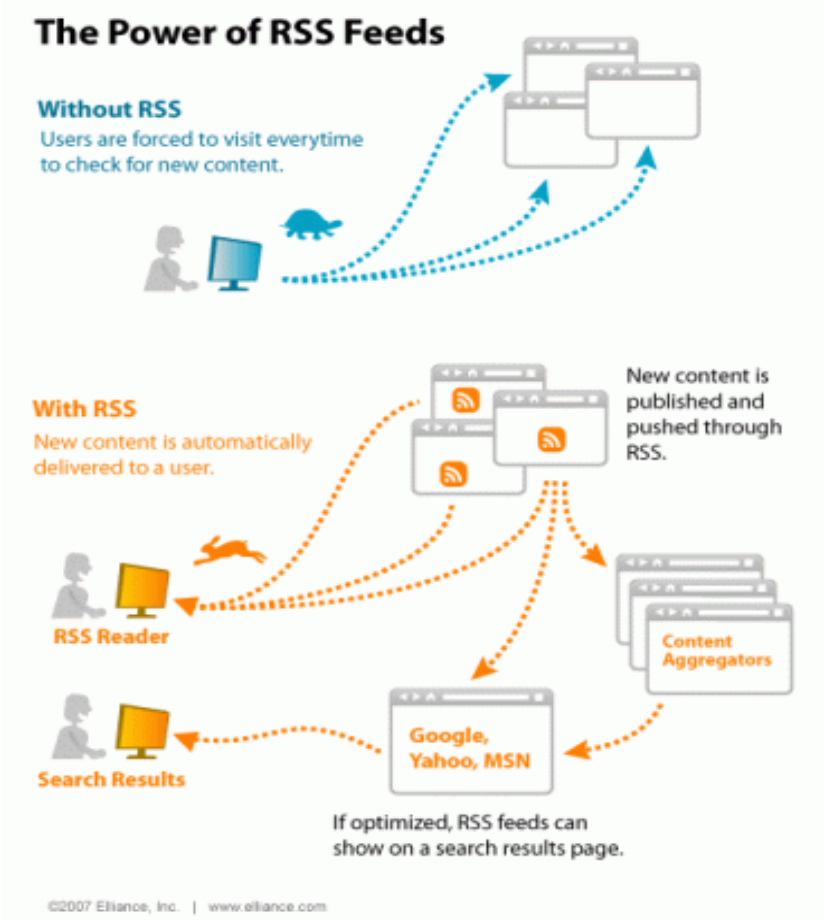
Etre un pur produit web	Plus besoin d'être téléchargé puis installé, le service s'utilise directement en ligne. Amélioration continue et mise à jour permanente
Tirer parti de l'intelligence collective. Favoriser le partage	Les utilisateurs deviennent producteurs de contenu, voire de ressources (bande passante, espace disque). Considérer les utilisateurs comme des co-développeurs
Capitaliser sur la richesse des données	Avoir le contrôle sur des sources de données uniques, difficiles à recréer, et dont la richesse s'accroit à mesure que les gens les utilisent
S'appuyer sur des technologies souples et interopérables (RSS, AJAX, REST)	Proposer des interfaces plus intuitives et plus rapides au service de nouveaux outils interactifs et de nouvelles pratiques collaboratives (blogs, réseaux sociaux, syndication, ...)
Se libérer du PC	Les applications peuvent être portées sur un assistant personnel, un téléphone mobile, la télévision, …
Voir grand	Sur le principe qu'il faut non seulement viser le cœur du marché, mais aussi sa périphérie à travers la mise en place de service « prêt à consommer »

Technologies du « web 2.0 »

- Un site utilisant une approche Web 2.0 met généralement en œuvre les technologies suivantes:
  - Utilisation de feuilles de styles **CSS**, d'un balisage **XHTML** sémantiquement valide
  - Syndication et agrégation de contenu **RSS / ATOM**
  - Services et architecture Web **REST**
  - Techniques d'applications riches **RIA** (Rich Internet Application)
  - Mise en œuvre d'applications composites (**Mashup**)
    - Une application composite est un site Web qui combine du contenu provenant de plusieurs sites Web. Le principe d'un mashup consiste à agréger du contenu provenant d'autres sites, afin de créer un site nouveau
      - utilisation le plus souvent l'objet XMLHttpRequest, AJAX du côté client, et les API ou les Services Web des sites dont on agrège le contenu
      - de plus en plus d'éditeurs de contenu proposent gratuitement des API ou des Services Web afin d'encourager la communauté des développeurs à créer des mashups utilisant leur contenu
    - Exemple: Google, Yahoo!, Amazon, eBay, ou FedEx (permet à un commerçant de présenter sur son propre web l'état d'avancement d'une livraison qu'il a sous-traitée au logisticien), **leur intérêt étant d'inciter les développeurs à répandre et à diffuser leur contenu**

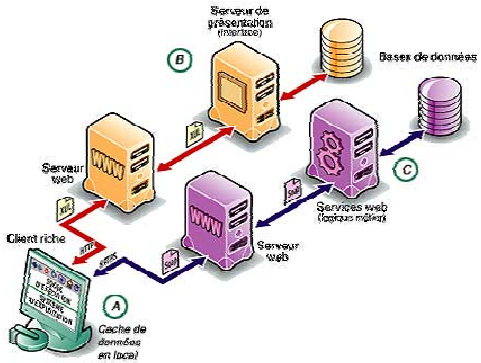
Flux RSS

L'information vient automatiquement dans votre environnement selon vos besoins



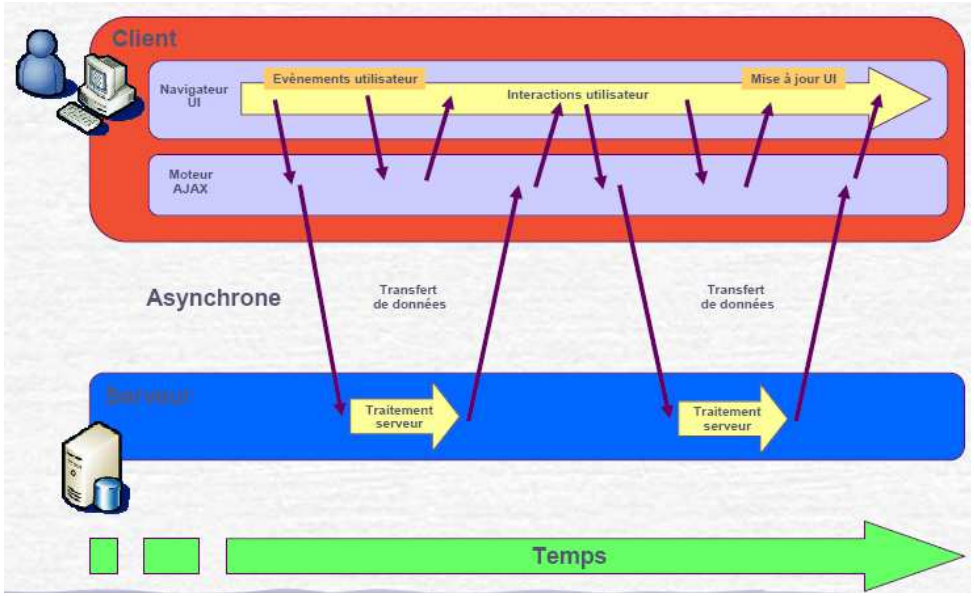
Les solutions technologiques pour la mise en œuvre d'un site « web 2.0 »

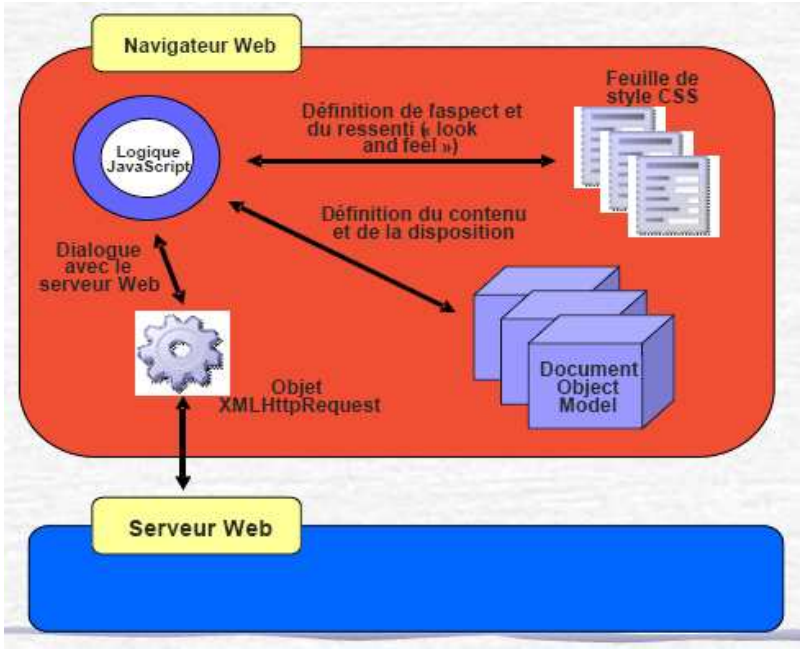
- Un site « web 2.0 » se base sur des approches dites « client riche ». L'architecture du « client riche » concentre les avantages du client lourd et du client léger
  - Interface graphique ergonomique (onglets, glisser-déposer, etc.) des clients lourds
  - Dialogue avec un serveur au travers d'une architecture client léger : les protocoles utilisés sont des standards tels que HTTP, XML-RPC, XUL (XML User interface Langage) pour la présentation et Soap
- Les réponses envisagées
  - Les applications « conventionnelles »: Elles s'appuient sur des frameworks complets de type Ajax, en proposant des composants Javascript avancés. Ces approches font évoluer « sans rupture » les applications web classiques, avec de réels apports du point de vue ergonomie et des avancées majeures d'un point de vue technique
  - Adobe Flex: Le client est un navigateur Web adossé d'un lecteur Flash qui prend en charge le dialogue avec le serveur. L'interface est construite de manière classique avec Macromedia Flash MX, ou à partir d'une description XML des composants de l'interface, traitées côté serveur
  - Microsoft SilverLight: Solution concurrente de Flex, nécessite l'installation du plugin SilverLight Runtime (2 Mo)
  - Google Web Toolkit: C'est un framework permettant de créer des pages web dynamiques en Ajax, en traduisant le code d'une application Java classique proche de Swing



Ajax

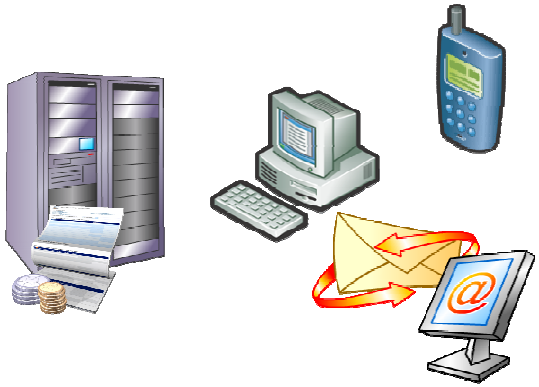
- AJAX, ou Asynchronous JavaScript And XML (« XML et JavaScript asynchrones ») est un acronyme désignant une méthode de développement d'applications Web, basée principalement sur les technologies :
  - (X)HTML / CSS
  - DOM du Navigateur
  - XML / XSLT
  - Objet XMLHttpRequest
  - Javascript





Collaborateur 2.0
High-Tech / Low-Tech

Les outils performants pour traiter l'information apparaissent d'abord dans l'entreprise

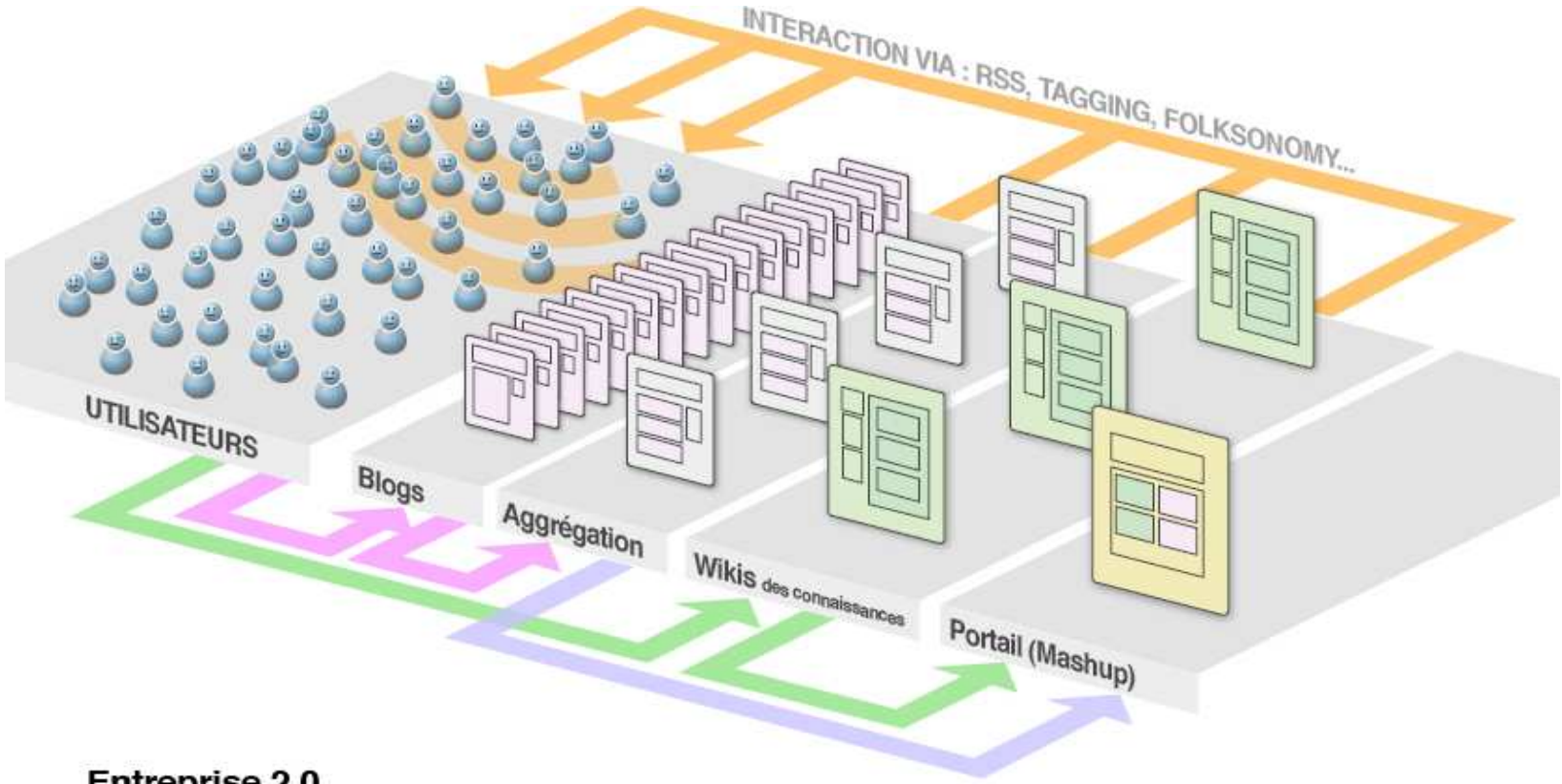


Le web 2.0 a induit des changements très rapides de comportement et d'attitude face à l'information et la connaissance

Le monde du travail a été rattrapé puis dépassé par le grand public

Décalage des pratiques et des usages entre la vie professionnelle et la vie privée

Entreprise 2.0: De la gestion des transactions à l'intelligence collective



Entreprise 2.0
Applications des modèles collaboratifs à la production de contenu et la gestion des connaissances.

Bibliographie

- Wikipedia: Le web 2.0 (http://fr.wikipedia.org/wiki/Web_2.0)
- <XML>fr (http://xmlfr.org/actualites/decid/051201-0001)
- Wikipedia mashup (http://en.wikipedia.org/wiki/Mashup_(web_application_hybrid)

Chapitre 5

Vision Informatique Logique
Vision Technologique

5.11 Sauvegardes

Modes de sauvegarde

- Sauvegardes dites « à froid » :
  - La procédure de sauvegarde « à froid » consiste à :
    - Arrêter la ou les bases du silo de données (et donc suspendre les services associés au silo de données)
    - Sauvegarder l'ensemble des fichiers (données, redo log, contrôles) constitutifs du périmètre prévu de la sauvegarde (totale, incrémentale)
    - Redémarrer la base ou les bases du silo de données (et donc relancer les services associés au silo de données)
- Sauvegardes dites « à chaud » :
  - La procédure de sauvegarde « à chaud » ne présuppose pas la suspension des services du silo de données, ni l'arrêt de la ou des bases du silo de données
  - La sauvegarde s'effectue en parallèle des accès nominaux au silo de données (appels de services)

Sauvegarde « à froid »

- **Avantages**
  - **Simplicité de mise en œuvre**
  - **Capitalisation sur les procédures et solutions existantes**
- **Inconvénients**
  - **Le silo de données et ses services sont indisponibles durant toute la durée de la sauvegarde**
  - **Les durées possibles des plages d'indisponibilité planifiée sont très courtes et peu compatibles avec les volumes de données des référentiels**
  - **Fréquence de sauvegardes trop faible, augmente le volume potentiel de données perdues en cas d'incident physique ou logique**

Sauvegarde « à chaud »

- **Avantages**
  - **Pas d'impact sur la disponibilité du silo de données et de ses services, meilleure continuité de service**
  - **Offre une bien meilleure flexibilité pour la construction du plan de production**
  - **Augmentation possible de la fréquence des sauvegardes pour limiter les volumes de pertes de données potentielles**
- **Inconvénients**
  - **Plus grande complexité d'appropriation et de mise en œuvre**
  - **Nécessité de vider les données en cache (snapshot) selon la solution de sauvegarde retenue**
  - **Peut provoquer des locks selon le SGBD retenu**

Modalités de sauvegarde « à chaud »

- Les modalités d'implantation des modes de sauvegarde « à chaud » restent à définir, plusieurs pistes sont envisageables pour une étude détaillée ultérieure, autour des thèmes suivants :
  - Réplication
    - Au niveau physique (disques, baies, volumes)
    - Au niveau logique (SGBD, File System)
    - Synchrone vs Asynchrone
  - Virtualisation des espaces de stockage
  - Solutions spécifiques de sauvegardes (incrémentales, totales) de fichiers ou de bases de données

Notion de réplication

- La duplication de données est nécessaire lorsqu'il s'agit d'offrir un accès plus large et plus rapide à l'information
- Les données sont copiées d'un serveur principal vers un ou n serveurs secondaires en de multiples lieux pour:
  - Les mettre à disposition des différents services selon leurs besoins spécifiques
  - Augmenter la disponibilité des applications en faisant face à toutes sortes de pannes des équipements
- Les mécanismes généralement retenus permettent de dupliquer l'information via TCP/IP sur des systèmes indépendants sans partage des ressources
- La réplication s'applique à quatre types de ressources:
  - Le contrôleur de disque, le volume logique de disque, le système de fichiers et le SGBD
- La problématique de sauvegarde à chaud revient à définir les mécanismes de réplication de données à la volée sans passage par des procédures de batch

La réplication synchrone des données

- La réplication synchrone repose sur le mécanisme « 2-phase commit » existant sur la plupart des SGBD relationnelles. Le principe étant de s'assurer du bon déroulement des mises à jour effectuées avant de permettre la mise à jour suivante:
  - Le système principal bloque l'accès aux bases de données de tous les systèmes cibles concernés pour assurer la mise à jour simultanée des bases. Si l'un des systèmes n'est pas disponible, l'opération est rejetée
  - La réplication s'effectue en temps réel. Elle offre aussi une garantie d'un niveau de mise à jour identique de l'information sur toutes les bases de données concernées.
- Les données répliquées ne sont jamais utilisées en fonctionnement normal en production mais uniquement lors des procédures de restauration
- Performances:
  - La nécessité de s'assurer de la disponibilité des bases de données secondaires retarde l'enregistrement proprement dit. Ce retard comporte plusieurs facteurs, tels que :
    - le temps de traitement du serveur principal induit par la gestion des serveurs secondaires ;
    - le temps de réaction des serveurs secondaires ;
    - le temps de transit des informations véhiculées sur le réseau.
- Contraintes:
  - La méthode est très sensible à la disponibilité du réseau et des équipements, ce qui induit fatalement un certain coût de mise en œuvre
  - Assurer la disponibilité du réseau ne suffit pas au bon fonctionnement du mécanisme. Il est aussi nécessaire de dimensionner les serveurs secondaires en accord avec les performances du serveur principal pour pouvoir suivre le rythme
- Points faibles:
  - Les principaux points faibles de cette méthode s'expriment en termes de performances et de coûts, du fait même du principe du commit à deux phases et des moyens à mettre en œuvre pour l'assurer
- Du fait des moyens à mettre en œuvre, le mode synchrone est utilisé quand la qualité de la mise à jour de l'information est vitale, là où les informations les plus récentes sont essentielles

La réplication asynchrone des données

- La réplication asynchrone repose sur la capture d'une copie des modifications effectuées sur la base de données principale, pour les diffuser ensuite sur les équipements cibles concernés. Elle met en œuvre un moteur de capture et de diffusion des mises à jour, ainsi qu'un système de stockage temporaire
- **La différence fondamentale avec la réplication synchrone réside dans l'aspect non simultané de la duplication**. Ainsi, la méthode asynchrone n'entrave pas le processus d'enregistrement des transactions dans la mesure où le processus de diffusion est découplé de l'enregistrement
- Concernant la duplication proprement dite, elle peut avoir lieu en temps quasi réel (sur chaque événement de mise à jour), à intervalles prédéfinis, ou à date différée selon les nécessités
- Performances:
  - Ce mode permet de différer la réplication aux heures de faible trafic. Le flux global peut être lissé en fonction du débit disponible
  - Le mode asynchrone ne pénalise pas les performances du serveur principal dans la mesure où le processus d'enregistrement des transactions n'en est pas affecté
- Contraintes:
  - La bande passante doit être suffisante pour pouvoir écouler le trafic, sous peine de charger dangereusement le système de stockage temporaire des copies
- Points faibles:
  - Le mode asynchrone n'offre pas de garantie de mise à jour identique de toutes les bases de données sur défaillance du serveur principal. Cette méthode requiert donc une certaine tolérance à l'égard de la perte éventuelle de transactions
  - Comme en mode synchrone, le bon fonctionnement du mode asynchrone ne peut s'affranchir d'un réseau robuste. La redondance du réseau est donc fortement souhaitable.
- Le mode asynchrone est utilisé pour des applications de type gestion de stocks, messagerie, ou mise à jour de documents par exemple ; là où une mise à jour régulière est nécessaire, mais la disponibilité immédiate de l'information la plus récente non vitale

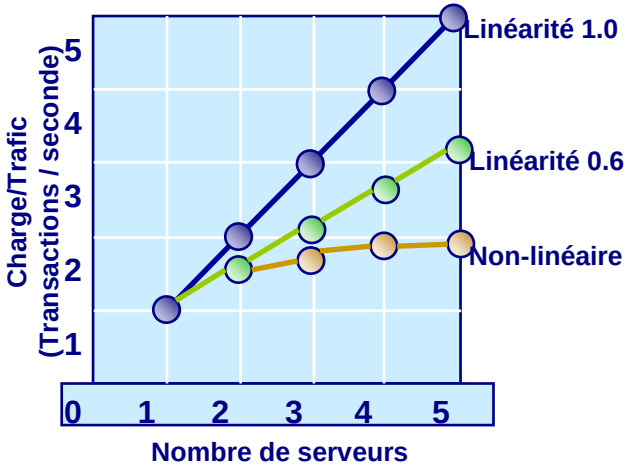
Chapitre 5

Vision Informatique Logique
Vision Technologique

5.12 Capacité de montée en charge

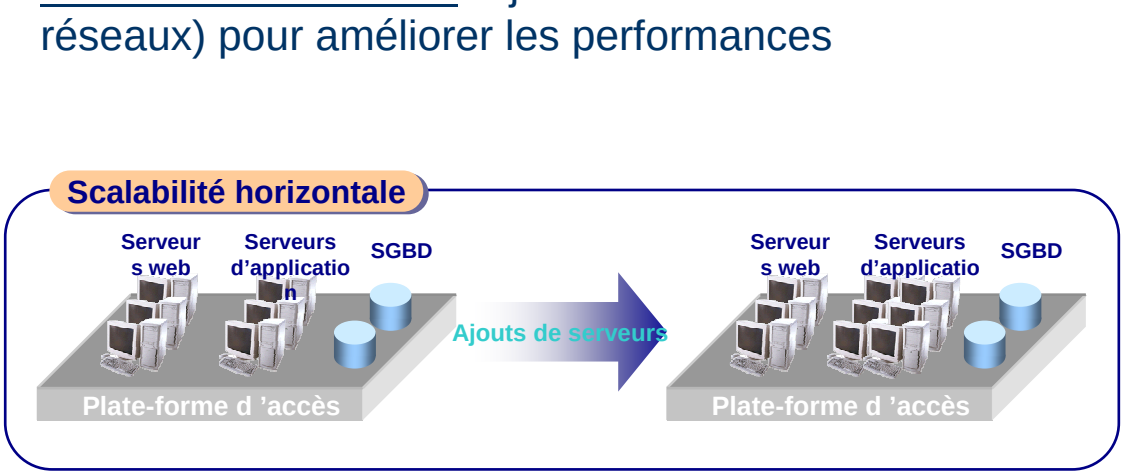
Mise en œuvre de la scalabilité

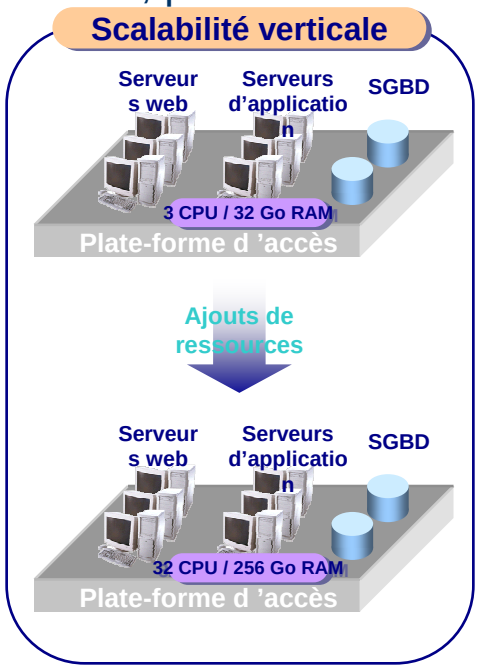
- La scalabilité est la capacité de servir des utilisateurs ou des transactions additionnels sans changer fondamentalement l ’architecture technique ou la conception des applications
  - Une application peut être défini de scalable s ’il est possible de maintenir la performance avec une charge supplémentaire simplement en rajoutant des ressources, comme de la mémoire ou des processeurs
- L’objectif principal est d ’obtenir autant que possible une relation linéaire entre la quantité de ressources ajoutés et les performances accrues résultantes, et cela sans dégradation de la vitesses de traitements des transactions
  - Un ratio de scalabilité 1:1 pour une plate-forme supportant 100 transactions par seconde simultanées à l ’aide d ’un serveur signifie qu ’avec l ’ajout d ’un nouveau serveur, 200 transactions seront servies



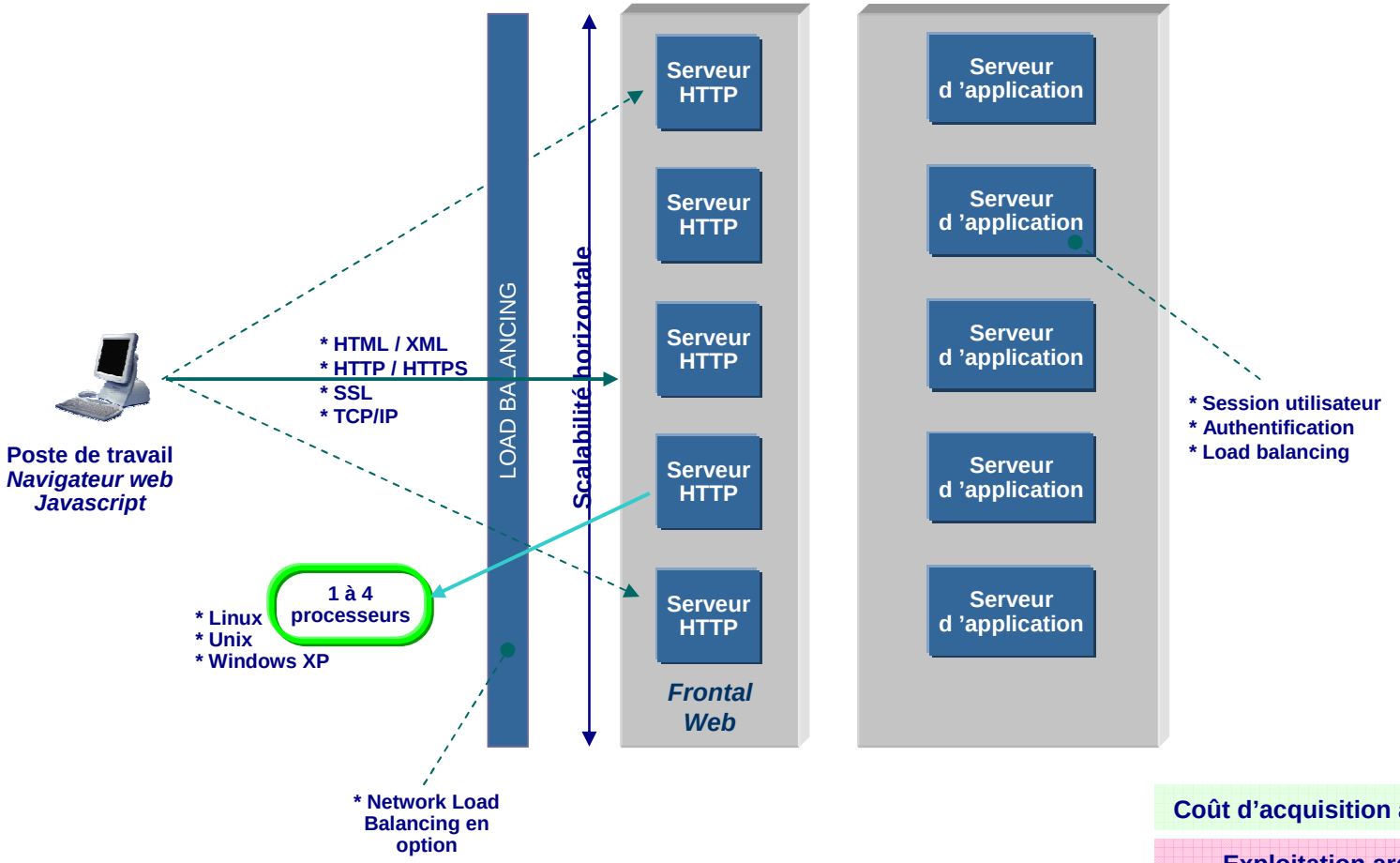
Typologies de scalabilité

- Il existe deux types de scalabilité qui peuvent être appliqués pour affecter les performances globales des plates-formes web:
  - La scalabilité horizontale: Ajouts de serveurs (web, applications, …) pour améliorer les performances
  - La scalabilité verticale: Ajouts de ressources à un serveur (mémoire, processeurs ou cartes réseaux) pour améliorer les performances





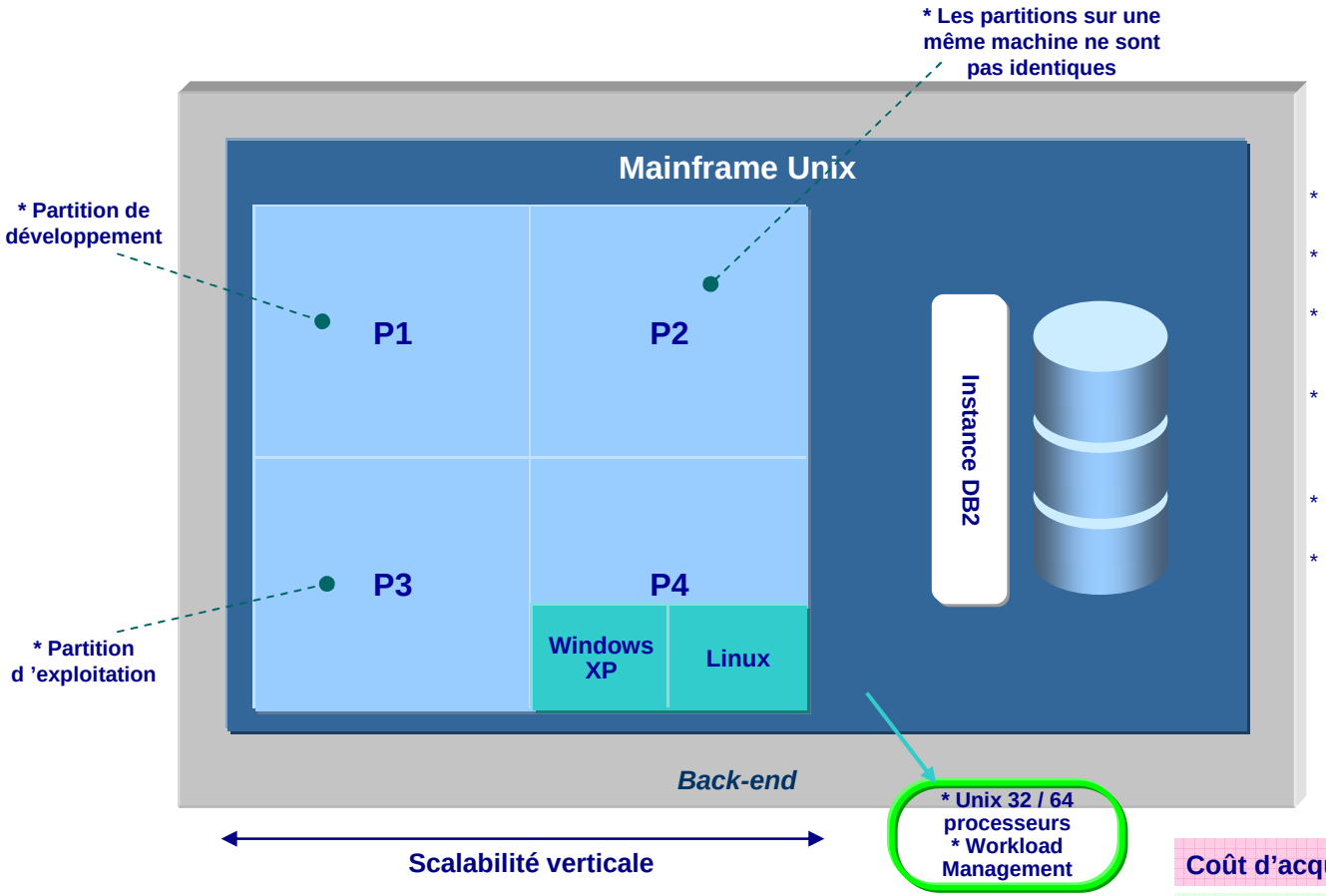
Scénario de regroupement en scalabilité horizontale



Coût d'acquisition attractif

Exploitation ardue

Les opportunités du mainframe Unix



Coût d'acquisition important

Exploitation simplifiée