Java EE

Master 2 MIAHS, Univ. Grenoble Alpes

Jérôme DAVID

Année 2016-2017

Plan du cours

- Introduction à Java Edition Entreprise
  - Les architectures client-serveur, multi-niveaux,
  - Les composants, les conteneurs, les serveurs d'applications
  - Survol des APIs disponibles dans Java EE
- La couche Web de Java EE
  - les servlets,
  - JSP (JSTL, etc).
- La couche métier (EJB) et la persistance
    - persistance : JDBC, Hibernate ?, JPA?

Contexte

- Le développement rapide du Web
  - Les applications doivent être accessibles depuis un navigateur Web
- Des environnements d'entreprise hétérogènes
  - Différents OS (Windows, Unix, Linux, etc.)
  - Différents matériels
- Les solutions proposées
  - PHP : simple
  - ASP de Microsoft : dépendant de Windows
  - JSP - Java Java EE : portabilité, maintenabilité, sécurité

Introduction à Java EE

La but de ce premier cours est de vous donner un aperçu général de Java EE.

Qu'est ce qu'une architecture multi-niveaux (N-Tier) ?
Qu'est ce qu'un serveur d'application ?
Qu'est ce que Java EE ?
Quelles sont les outils disponibles dans Java EE ?

Plan du chapitre

- Contexte et introduction de Java Edition Entreprise
  - Les architectures Client/Serveur
  - Les architectures multi-niveaux ou n-tier
- L'architecture Java EE et ses composants
  - Les différentes couches et leurs composants
  - Les conteneurs et serveurs d'application
- Un survol des APIs disponibles
  - Web, XML, persistance, etc.

Rappels sur JAVA

Principales caractéristiques

- Langage objet, les classes, les interfaces, l'héritage, etc.
- Portable : indépendant de l'architecture
  - utilisation de pseudo-code + machine virtuelle

La version de développement (JDK) JAVA comprend

- Une machine virtuelle JAVA (JVM)
- Une API (Application Programming Interface)
  - un ensemble de librairies utilisées pour développer des composants logiciels

Les versions de la plateforme JAVA

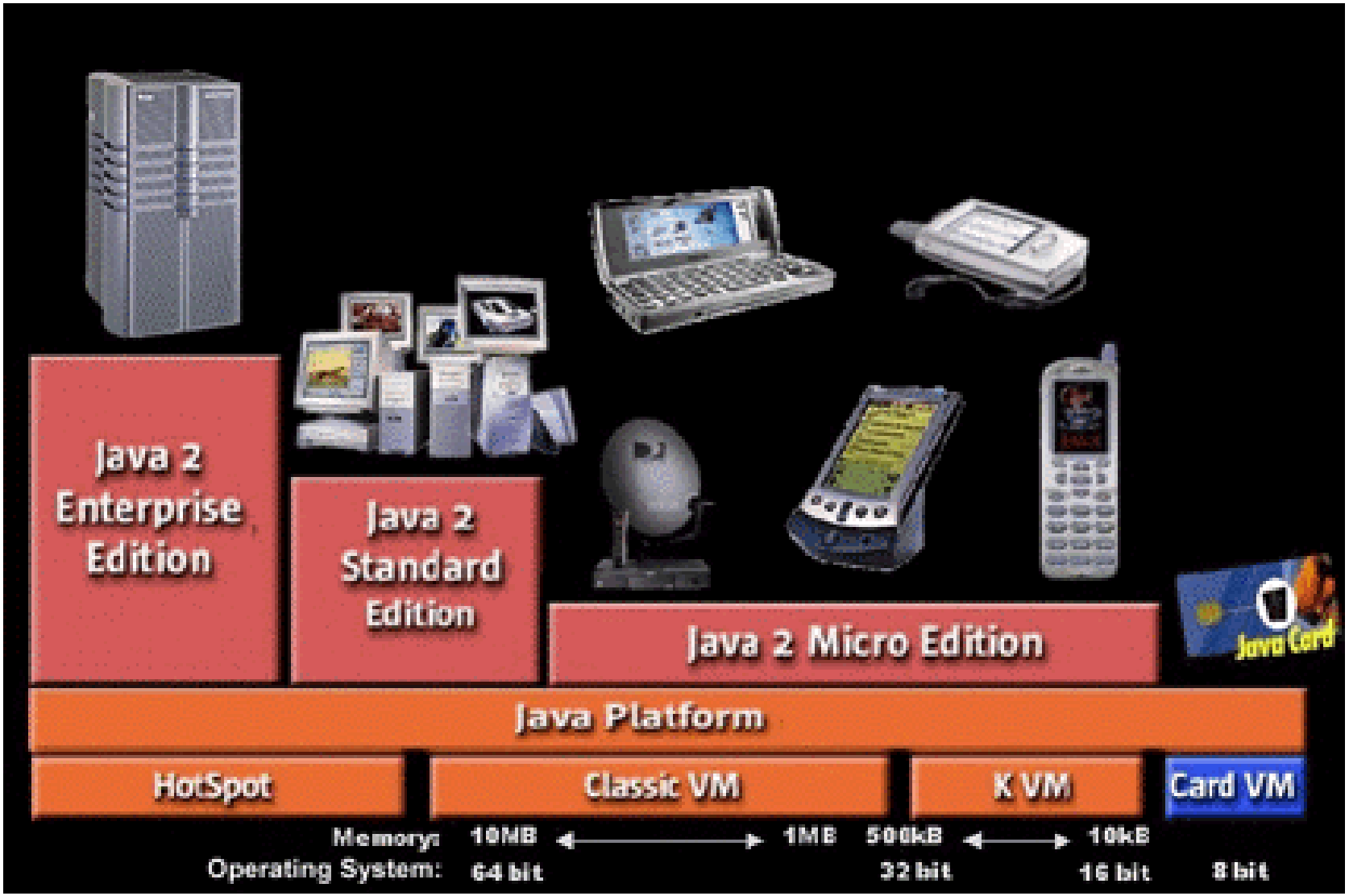
3 versions différentes de la plateforme JAVA :

J2SE (Java 2 Standard Edition) : destiné au développement d'applications pour ordinateurs personnels et aux postes clients

J2ME (Java 2 Micro Edition) : prévu pour le développement d'applications embarquées sur terminaux mobiles (téléphone, PDA, SmartPhones, etc.)

Java EE (Java 2 Enterprise Edition) : destiné à un usage professionnel avec la mise en œuvre de serveurs.

Positionnement des versions de JAVA



La norme Java EE

Introduit (sous le nom de J2EE) par Sun en 1997

- **Norme** supportée par de nombreux acteurs puissants
- Un **standard de développement** d'applications JAVA **réparties**, **multi-niveaux** (n-tiers), basées sur des **composants**

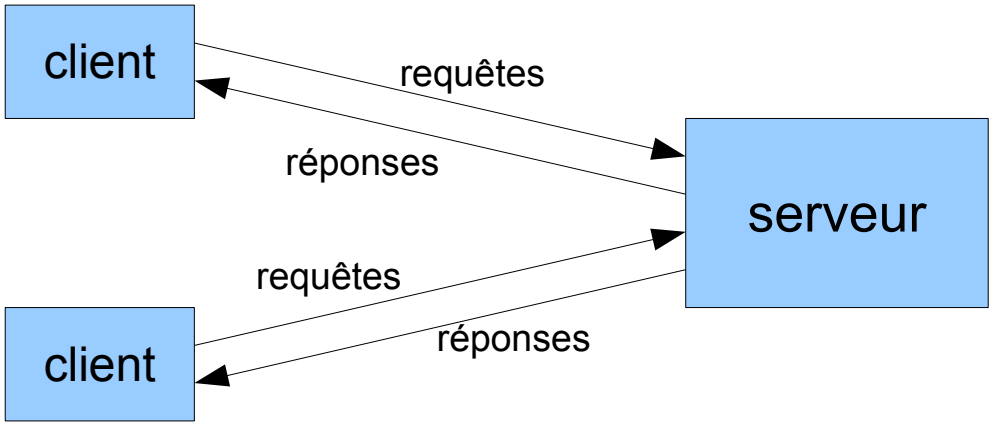
La plate forme Java EE comprend :

- un environnement Java standard (JVM + API)
- les spécifications du **serveur d'application** :c.-à-d. l'environnement d'exécution
- des services sous formes d'API Java

Un peu d'architecture

Les **architectures client/serveur** (modèle HTTP)

- ensemble d'architectures logicielles où des programmes (le client et le serveur) communiquent via un réseau
- Le client envoie des requêtes à un serveur et le serveur fournit des réponses aux clients
  - Exemple : le navigateur Web (client) qui demande une page à un serveur Web



Caractéristiques des clients et serveurs

- Caractéristiques du client
  - Initie le dialogue via une requête
  - Attend et reçoit des réponses
  - Interagit avec l'utilisateur (IHM)
- Caractéristiques du serveur
  - N'initie jamais de dialogue ou d'activité (il écoute)
  - Attend et répond aux requêtes des clients
  - Gère quels sont les clients autorisés à se connecter

Avantages et inconvénients

Avantage : c'est centralisé

meilleure sécurité : points d'accès aux données peu importants

administration simplifiée : elle est faite au niveau du serveur

évolutif : on peut facilement rajouter des clients

Inconvénients

Coût souvent élevé du serveur : c'est une grosse machine

-> ce n'est plus forcément le cas

Le serveur est critique : si il tombe en panne, rien ne fonctionne

-> on duplique les systèmes

Les architectures à 2 et 3 niveaux

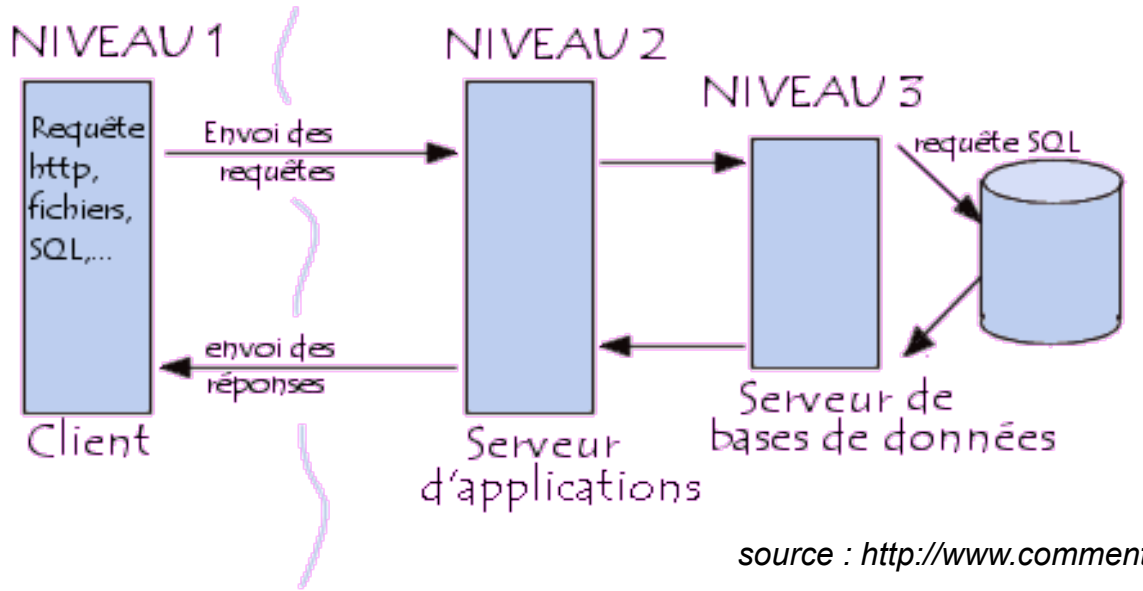
- Architecture **2 niveaux** (2-tiers)
  - client/serveur classique : 1 serveur, des clients
  - un seul serveur pour tous les services
    - Le serveur est polyvalent
    - c'est une architecture très centralisée
- Architecture **3 niveaux** (3-tiers)
  - 1 serveur d'application (couche métier - middleware)
  - 1 serveur de données (couche données - SGBD)
  - et toujours des clients (couche de présentation)

L'architecture à 3 niveaux

Avantages :

permet une plus grand **souplesse** (indépendance SGBD/middleware)

meilleure **performance** : répartition de la charge

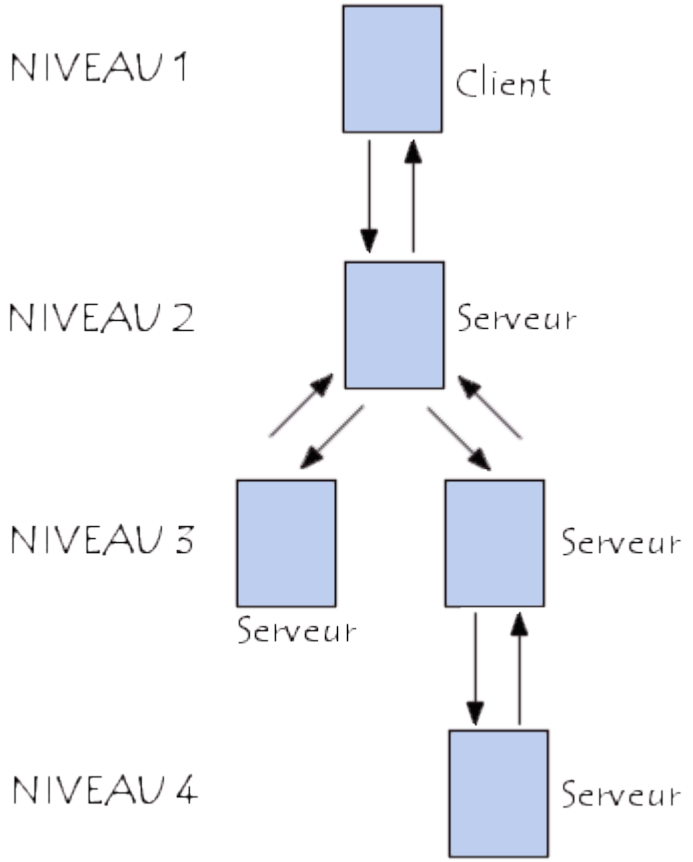


source : http://www.commentcamarche.net

L'architecture multi-niveaux

On spécialise encore plus les serveurs

- toujours les serveurs middleware et SGBD
- serveur d'accès LDAP
- serveur métier (EJB)
- serveur de présentation (JSP)
- etc.



source : http://www.commentcamarche.net

L'architecture Java EE

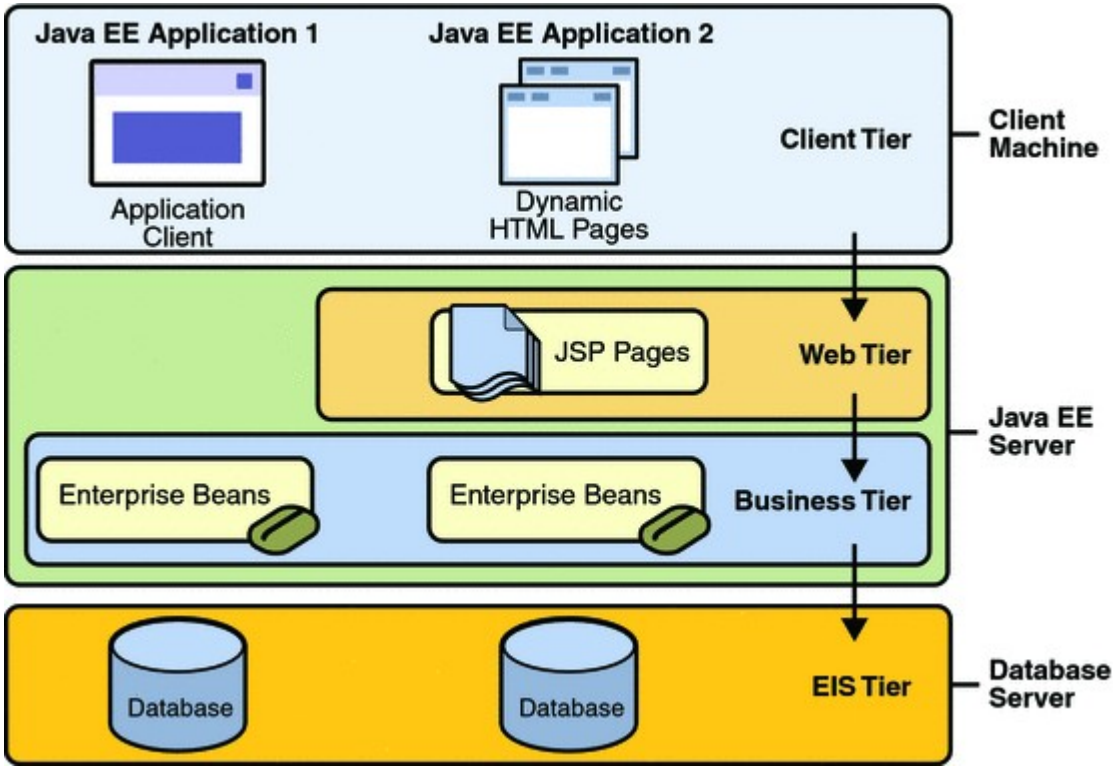
C'est une sorte d'architecture à 3 niveaux où la couche présentation est divisée en 2 :

Le client léger :

un navigteur Web

Un serveur Web

avec JSP, servlets.



source : http://java.sun.com/javaee/5/docs/tutorial/doc/bnaay.html

Les composants Java EE

Les spécifications Java EE distinguent 3 types de composants :

- Composants qui tournent sur le client :
  - les applications clientes et les applets
- Composants Web qui tournent sur le serveur :
  - Servlets, JavaServer Pages (JSP), JavaServer Face (JSF), etc.
- Composants métiers qui tournent sur le serveur :
  - Enterprise JavaBeans (EJB), Java Persitance API, etc.

Tous ces composants sont écrits en JAVA

Les SGBD ne font pas partie des composants

Les composants Java EE clients

2 types de clients : clients Web et clients application

Clients Web (client léger)

- Les pages HTML générées par les composants Web serveur
- Le navigateur qui affiche ces pages

Les applets (client semi-lourd)

- un petite application JAVA qui s'exécute sur la machine virtuelle installée sur le navigateur
- c'est un compromis entre client Web et application

Aujourd'hui, le client Web léger est préféré

Les composants Java EE clients

Le client application (client lourd)

- C'est une application JAVA, avec une interface graphique plus riche écrite en Swing ou AWT
- C'est un client plus lourd que le client Web

Comparaison des 2 approches :

- Il faut installer le client application, pas le client Web
- Le client application peut avoir une interface graphique plus évoluée (avec Web 2.0 ce n'est plus trop le cas...)
- Le client application gère toute la couche présentation, le client Web à besoin d'une sous-couche Web serveur

Composants Web coté serveur

Seulement utiles dans le cas de clients Web

Permettent de générer des pages Web (HTML, XML)

3 types de technologies :

Servlets : classes permettant de répondre à une requête HTTP (classes JAVA qui peuvent écrire du HTML)

JSP : sortes de pages Web exécutables (pages HTML qui contiennent du code JAVA)

JavaServer Faces : technologie au dessus des Servlets et JSP pour fournir des composants de l'interface graphique

Composants métier

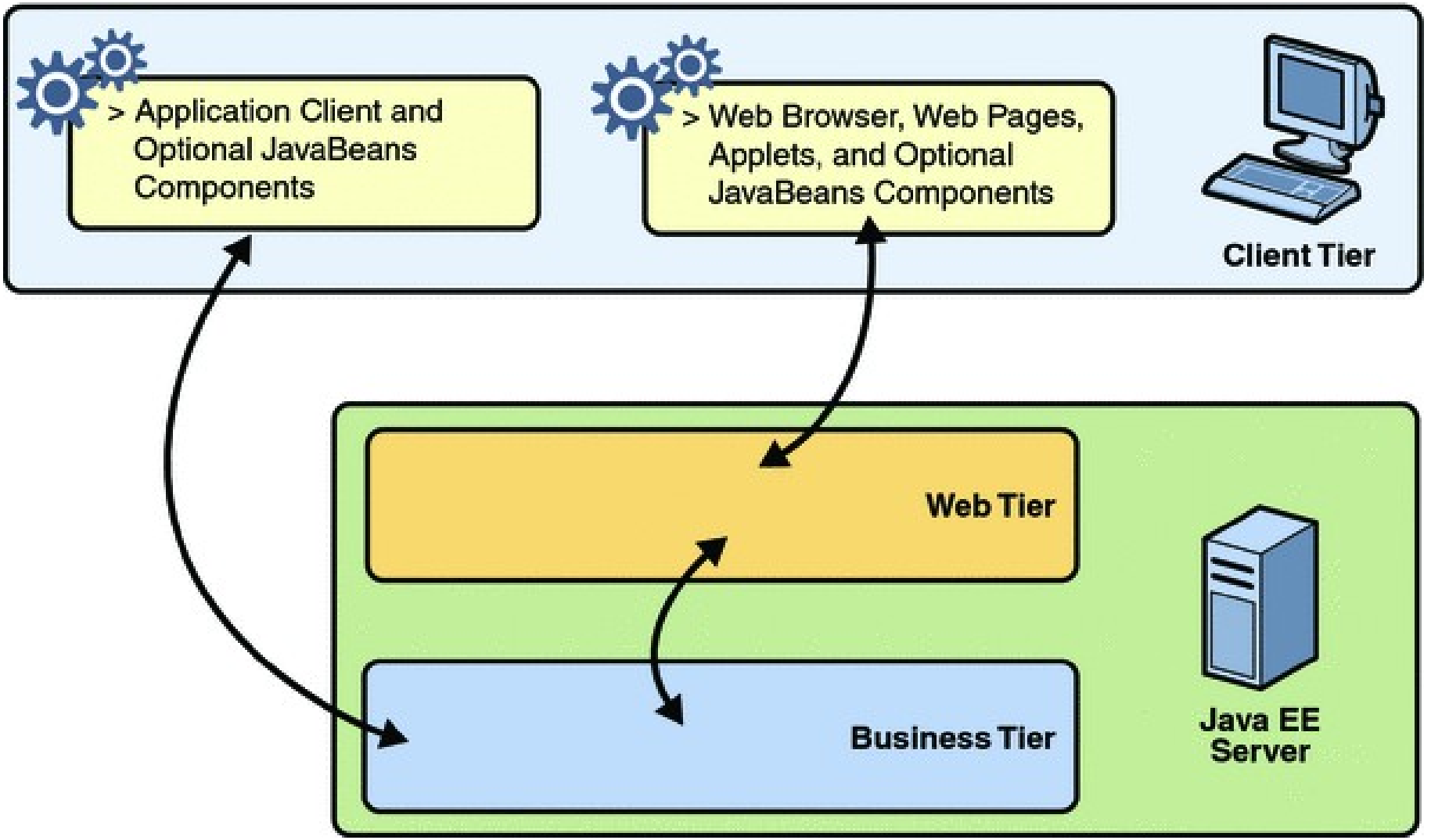
Les composants métiers : classes qui implémentent concrètement les objets du domaine applicatif (c'est le modèle de l'application).

- Exemple : les classes Client, Compte, Prêt d'une banque

Ces composants de la couche métier vont gérer :

- les traitements propres à la logique de l'application
- la persistance des objets dans une BD
- la réponse au programme client (le client application ou les composants Web serveur)

Communication entre client et serveur



source : http://java.sun.com/javaee/5/docs/tutorial/doc/bnaay.html

Les protocoles Internet utilisés

La communication entre les composants des différentes couches se fait par le réseaux (Internet)

Les principaux protocoles utilisés

- HTTP : transfert des pages JSP, Servlet
- RMI : pour invoquer méthodes d'objets distants
- SOAP/WSDL : pour les services Web
- SSL : pour les transferts sécurisés

Les conteneurs Java EE

Un conteneur est l'environnement d'exécution des composants.

Il gère l'interface entre les composants Java EE et les fonctionnalités bas-niveau

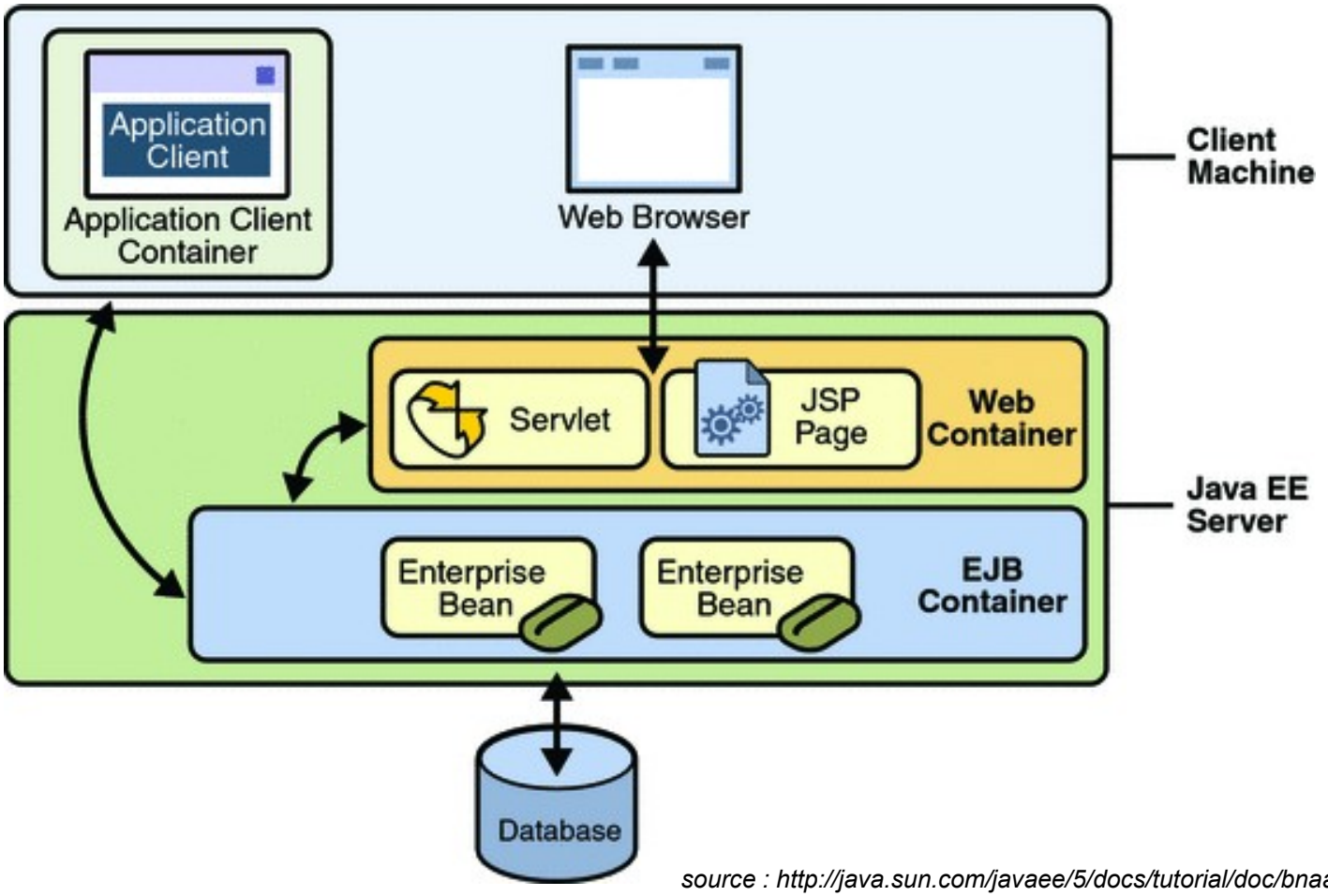
- multi-threading, le cache mémoire, la sécurité, l'accès aux données etc.

Les différents types de conteneurs :

- Conteneur d'EJB (composants métier)
- Conteneur Web (pour l'exécution des servlet, JSP, etc.)
- Conteneur d'application et d'applet (la machine virtuelle)

Les conteneurs Java EE

Le conteneur d'EJB et Web font partie intégrante du serveur Java EE (serveur d'application)



source : http://java.sun.com/javaee/5/docs/tutorial/doc/bnaay.html

Les serveurs d'application

Le serveur d'application va fournir les services systèmes génériques :

- La sécurité
- La reprise sur panne
- Les services transactionnel entre composants
- La gestion des utilisateurs
- L'accès aux sources de données
- etc.

Un serveur d'application Java EE = répond aux spécifications Java EE de Sun

Des serveurs d'application Java EE

Des serveurs commerciaux (payants) :

- BEA WebLogic
- IBM WebSphere
- Sun Java System App. Server
- Oracle Application Server 10g
- SAP Netweaver, ...

Des serveurs Open Source (gratuits) :

- JBoss
- Object Web (OW2) JOnAS
- Apache Geronimo 2.0, GlassFish, etc.

Les serveurs Web Java EE

Les serveurs Web Java EE permettent d'exécuter les servlets, JSP, etc.

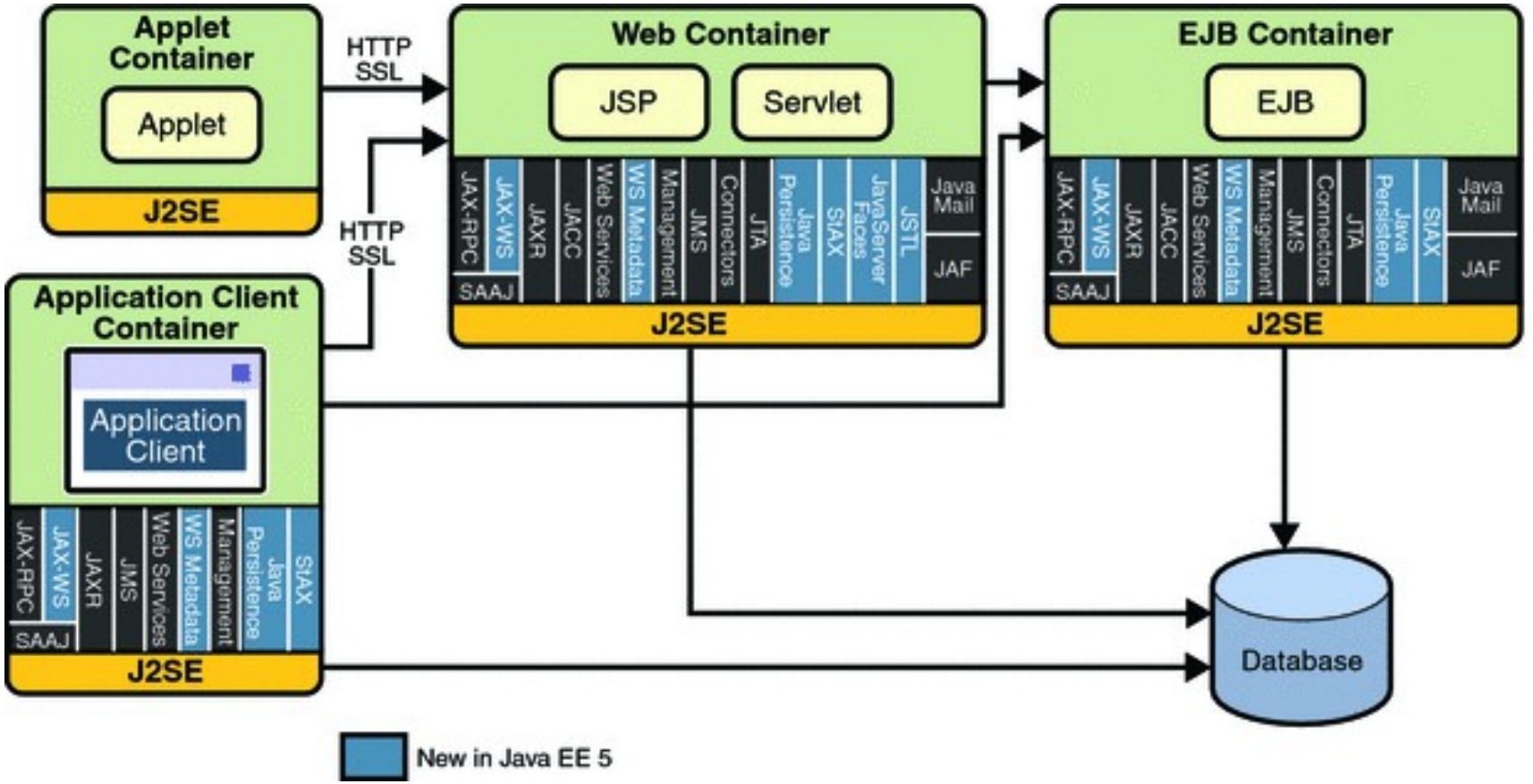
Ils sont souvent inclus dans les serveurs d'application

Quelques serveurs Web Java EE

- **Apache Tomcat**
- Jetty : très léger

Les APIs de Java EE

La plate-forme Java EE fournie un ensemble d'APIs fournissant des services de base au développeur



source : http://java.sun.com/javaee/5/docs/tutorial/doc/bnacj.html

Les Principales APIs

- Au niveau du conteneur Web :
  - Les servlets
  - Les JavaServer Pages - JSP
  - Les JavaServer Pages Standard Tag Library - JSTL
  - JavaServer Faces - JSF
- Au niveau du conteneur d'EJB (conteneur métier)
  - Enterprise Java Beans – EJB
  - Java Persistance API - JPA

Les servlets

C'est la base de la programmation Java EE

- toute la conception d'application Web repose sur cela

Qu'est ce qu'une servlet ?

- Un programme JAVA qui tourne sur un serveur Java EE
- Elle est invoquée lorsqu'un navigateur appelle l'URL qui lui est liée

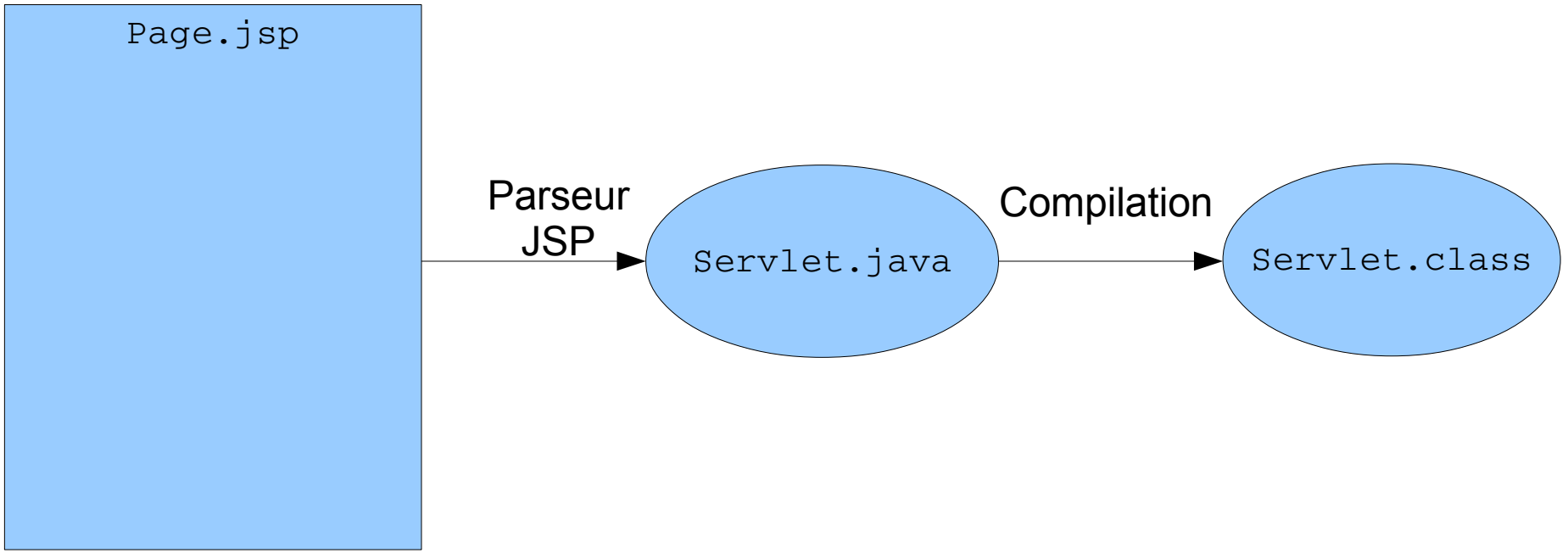
```
public class ServletCoucou extends HttpServlet {

    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
         throws ServletException, IOException {
      PrintWriter out = resp.getWriter();
      out.println("Coucou");
    }

    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
         throws ServletException, IOException {
      doGet(req, resp);
    }
}
```

JavaServer Pages (JSP)

Ce sont des pages Web incluant du code JAVA

plus facile à écrire que des servlets

Les JSP sont transformées en servlet



JavaServer Pages Standard Tag Library

Une API qui recouvre les fonctionnalités souvent utilisées dans les pages JSP (Tag Library)

Quelques fonctionnalités :

- Itérateur et conditionnel
- manipulation de documents XML
- internationalisation
- accès aux bases de données
- etc.

JavaServer Faces

- Plate-forme de création d'interfaces utilisateur Web
- Abstraction de JSP (on pourrait utiliser autre chose)
- Les composants principaux de JSF :
  - Représentation d'interfaces utilisateur (par composants)
  - Gestion des composants (états, événements, validation des entrées, conversion des sorties)
  - Navigation inter-vues

Enterprise JavaBeans (EJB)

EJB = composant logiciel implémentant les méthodes et attributs du métier visée par l'application

Les EJB sont des briques logicielles qui peuvent être utilisées de manière autonome ou avec d'autres EJB

Les EJB peuvent être distribués sur différents serveurs

Le conteneur d'EJB va gérer leur cycle de vie de manière transparente :

- création, passivation, réactivation, destruction

Enterprise JavaBeans (EJB)

- 2 types d'EJB sont utilisés :
  - EJB dit session (session bean) qui propose un service
    - avec conservation d'état entre 2 appels (stateless)
    - avec conservation d'état entre 2 appels (stateful)
  - EJB dit message (message-driven bean) qui réalise des tâches de manière asynchrone
    - le client dépose un message (JMS) dans une file d'attente, un EJB dit message sera instancié pour traiter ce message.
- Un ancien : EJB entité qui représente des données persistantes
  - Remplacé par Java Persistance API depuis la version 3.0.

Les autres APIs

- pour la persistance des données
  - JDBC : pour accéder aux bases de données relationnelles
  - Java Persitence API : mapping relationnel-objet
- pour la gestion distribuée d'objets
  - JNDI : pour donner des noms logiques aux ressources distribuées (exemple : associer une URL aux objets)
  - RMI : pour manipuler des objets distants
- les accessoires
- pour XML et les services Web

JDBC : accès aux bases de données

Java DataBase Connectivity permet d'accéder à des bases de données relationnelles à partir de code JAVA

Principales fonctionnalités

- Abstraction du SGBD utilisé
  - Il existe des drivers pour Oracle, MySQL, Postgres, DB2, etc.
- L'accès aux bases de données (connexion, session, etc.)
- L'envoie de requêtes SQL (Statement, etc.)
- L'exploitation des résultats (Curseur, etc.)

Java Persistence API

- Java Persistence API est une couche au dessus de JDBC (remplace les EJB de données)
- Permet de gérer la translation Objet/Relationnel
  - enregistrer les objets dans tes tables relationnelles
  - lire des objets à partir de tables relationnelles
- Cette API a été introduite dans un but de standardisation (norme EJB 3.0):
  - Nombreux frameworks de persistance : Hibernate, Toplink, JDO
  - Reprend principalement Hibernate

Java EE Connector Architecture (JCA)

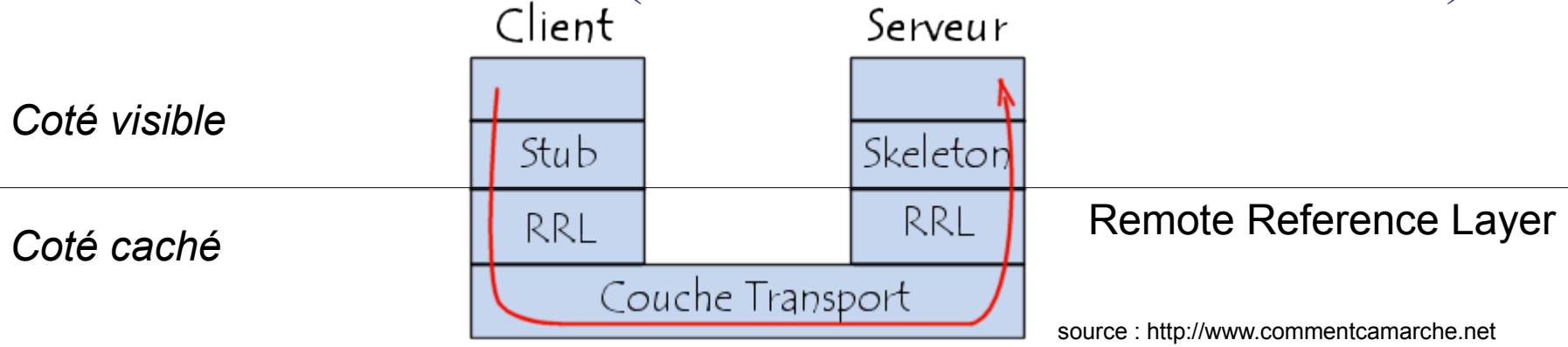
- JCA permet de créer des adaptateurs entre systèmes d'information propriétaires et systèmes Java EE.
  - Cela facilite l'intégration entre systèmes propriétaires et serveur d'application Java EE
  - C'est générique à n'importe quel type de serveur Java EE
  - Mais c'est spécifique au système propriétaire
- Un adapteur peut être exposé comme service Web
  - Les services du système propriétaire deviennent des services Web accessibles par le serveur Java EE

Java Naming and Directory Interface

- JNDI offre les fonctionnalités pour accéder au services de nommage et d'annuaires
  - Un service de nommage permet d'associer une entité technologique à un nom unique (exemple : des données à un nom de fichier, une page web à une URL, etc.)
    - Quelques services de nommage : DNS, NIS
  - Un annuaire est une extension du service de nommage. Il permet d'associer des infos supplémentaires.
    - par exemple une adresse, des numéros de tél. à une personne dans un annuaire LDAP
- JNDI est utilisé par de nombreuses APIs Java EE

Remote Method Invocation (RMI)

- RMI permet de manipuler de objets distants
  - Objets instanciés sur des machines virtuelles différentes, distribuées sur le réseau
- Le serveur d'objets est appelé rmiregistery
  - c'est lui qui possède les objets instanciés
  - Il fait le lien objet/URL (ex : rmi://nomServeur/nomObjet)
- Protocole RMI IIOP (Internet Inter-Orb Protocol)



source : http://www.commentcamarche.net

Java Message Service (JMS)

JMS permet aux applications ou composants Java EE de s'envoyer des messages de manière asynchrone.

- C'est comme le mail mais pour des machines

Cette API permet de créer, d'envoyer, recevoir et de lire des messages.

Avantages :

- Faible couplage (les applis ne se connaissent pas)
- Fiable : garantie de livraison des messages

Java Transaction API (JTA)

JTA fourni une interface de gestion des transactions indépendante du gestionnaire de transaction utilisé

Rappel : une transaction est un ensemble atomique d'opérations élémentaires (tout ou rien)

- Exemple du transfert entre 2 comptes : débit + crédit
- une transaction peut être confirmée (commit) ou annulée (rollback)

L'intérêt de JTA est de gérer les transactions des SGBD, du serveur d'applis, etc via une seule interface commune et partagée

JavaMail

JavaMail est destinée pour gérer le courrier électronique

- Composition de mails
- La lecture des mails
- L'envoi de mails

JavaMail ne contient pas de serveur mail (c'est juste une API cliente)

Elle est indépendante des protocoles utilisés

- POP, IMAP, SMTP, MIME, NNTP, etc.

JavaBeans Activation Framework (JAF)

JAF sert à JavaMail

JAF permet :

- de déterminer les types de donnée (type MIME des PJ)
- de faciliter l'accès à ces données
- de découvrir les opérations possibles sur ces données
- de créer le composant (JavaBean) approprié pour réaliser les opérations

Java Authentication and Authorization Service (JAAS)

- JAAS permet de gérer l'authentification et les droits des utilisateurs d'une application Java EE
  - Gestion individuelle des utilisateurs ou par groupes
- Basée sur PAM (Pluggable Authentication Module)
  - permet de se brancher sur différents systèmes d'authentification :
    - UNIX, Kerberos, annuaires LDAP, NIS

Les services Web

- Les services Web sont des technologies permettant aux applications d'interagir sur Internet
  - indépendamment du langage des applis et de leur plate-forme d'exécution
- Les services Web reposent principalement sur XML
  - XML RPC : Invocation de méthodes à distance
  - SOAP (Simple Object Acess Protocol)
    - protocole d'échange de données entre les applications
  - WSDL (Web Service Description Language) - W3C
    - description des services Web (emplacements, méthodes, paramètres, valeur de retour, etc.

Les services Web

- Suite...
  - UDDI (Universal Description Discovery and Integration)
    - standard de publication et de découverte de service Web bas é notamment sur l'utilisation d'annuaires de services Web
    - publication : mettre la description (WSDL) du service dans un registre
    - découverte : recherche de service dans les annuaires
- Java EE propose un ensemble d'APIs pour les SW
  - Des APIs XML : pour traiter les documents XML
  - Des APIs spécialisées services Web qui s'appuient sur les premières.

Les APIs pour XML

- Java API for XML Processing (JAXP)
  - API de base pour parser et transformer des documents XML
  - Parser SAX, représentation DOM
  - Transformations XSL (XSLT)
- Java Architecture for XML Binding (JAXB)
  - Permet de relier un schéma XML à une représentation Objet JAVA.

Les APIs pour les services Web

- Java API for XML Web Services (JAX-WS)
  - S'appuie sur JAX et fournit des fonctionnalités pour utiliser des services Web XML
  - traitement des requêtes et réponses SOAP, WSDL
- Java API for XML Registries (JAXR)
  - Permet d'accéder à des registres tels que les annuaires UDDI
- SOAP with Attachments API for Java (SAAJ)
  - API SOAP bas-niveau utilisée par JAX-WS et JAXR

Résumé

- Les architectures client/serveur
- L'architecture Java EE
- La notion de composant
  - composants client, Web, métiers
- La notion de conteneur
  - et de serveur d'application
- Les API de Java EE
  - API Web : servlet, JSP etc.
  - API métier : EJB et les autres : JDBC, JavaMail, XML...