

Applications d'entreprise avec JEE

Lom Hillah – Tewfik. Ziadi

Plan du cours Enterprise Applications



Introduction à Java EE



TP JEE



Mini-projet

Technologies



Java SE



Java SE Subscription



Java Embedded



Java EE



Java ME



Java Card



Java TV



Java DB



Developer Tools

Technos Java

- Java SE : applications standards
- Java ME, Embedded : applications embarquées
- Java TV : TV, box, VoD, jeux
- Java Card: applications sur puces intelligentes
- **Java EE : applications d'entreprise**

Contraintes pour les applications d'entreprises

- Fiabilité
- Sécurité
- Extensibilité
- Portabilité
- Intégration
- Adaptabilité
- Disponibilité
- Montée en charge
- Maintenabilité
- **Adéquation aux besoins des utilisateurs !!!**

Types d'architecture

Centralisée

Client / Serveur

Répartie / Microservices

Caractéristiques désirables pour répondre aux contraintes



Normalisation :
simplifier et rationaliser
la production

Intégration,
communication,
adaptabilité,
maintenabilité



Abstraction :
désolidariser un objet
de son contexte

Portabilité,
maintenabilité,
extensibilité, intégration,
adaptabilité



Communication : faible
couplage

Intégration, sécurité,
répartition



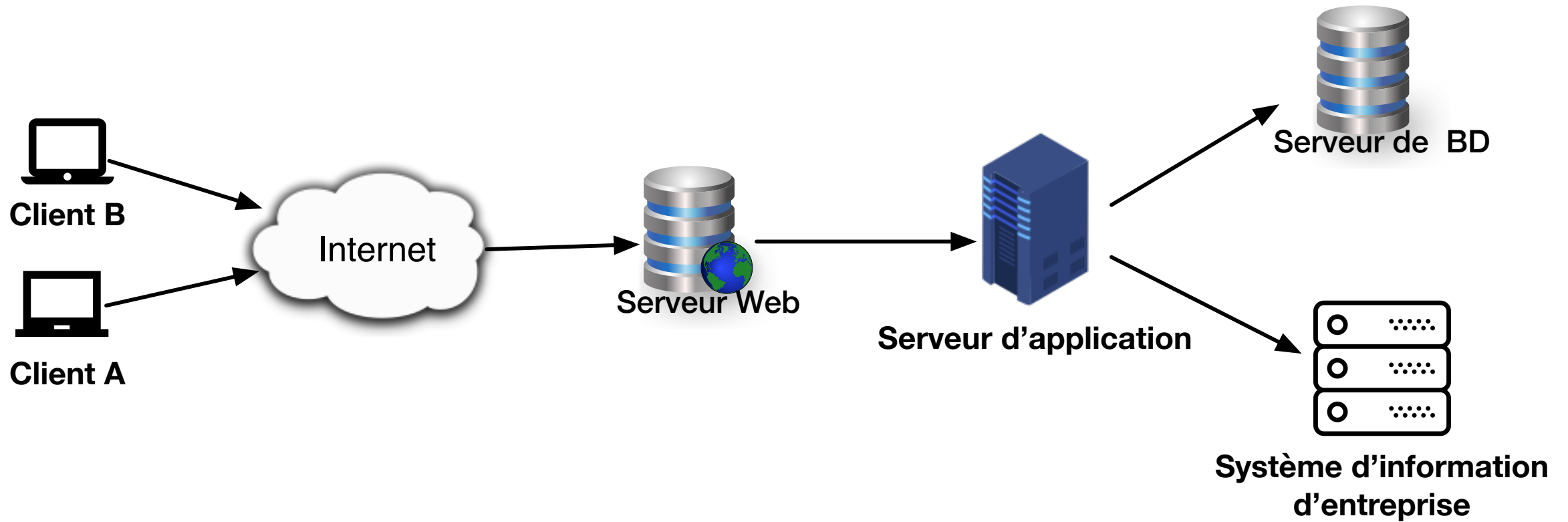
Orienté-Composant,

Maintenabilité, fiabilité,
extensibilité, adaptabilité,
portabilité, disponibilité

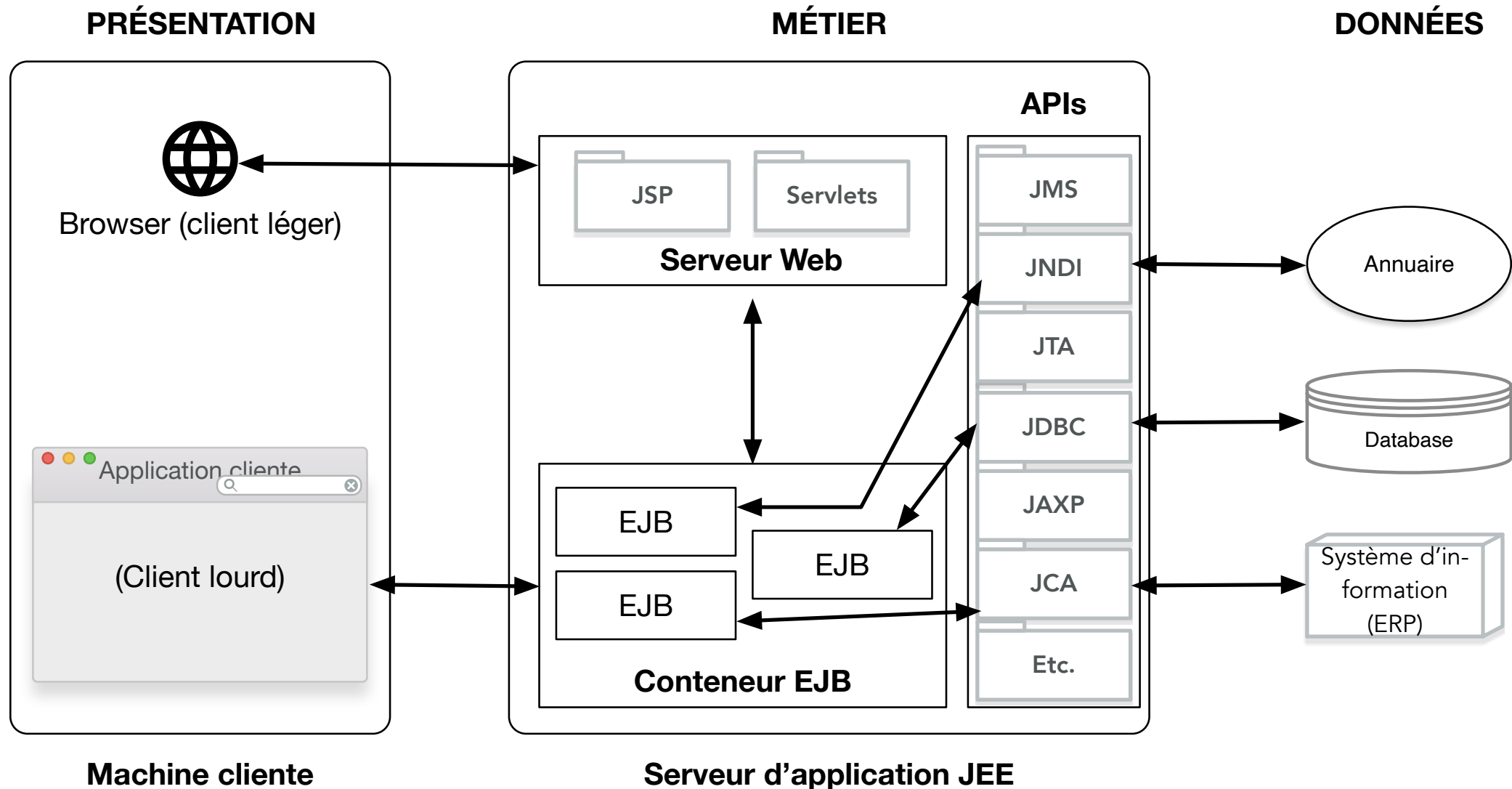
Java Enterprise Edition

- Framework pour le dév., déploiement et exécution d'application d'entreprises
 - Depuis 1998 (à l'époque de Sun)
 - Specs du Java Community Process (www.jcp.org)
- Basées sur des composants, multicouches
 - Séparation des préoccupations : métier vs. non-fonctionnel
 - QoS, persistance, administration, sécurité, transactions
- Développer des applications d'entreprise: **fiables, sûres**, sécurisées, portables, performantes, disponibles, maintenables, extensibles, répondant aux besoins des utilisateurs et ce, à moindres coûts.

Architecture typique d'une application JEE



Architecture générale 3-tiers des applications JEE

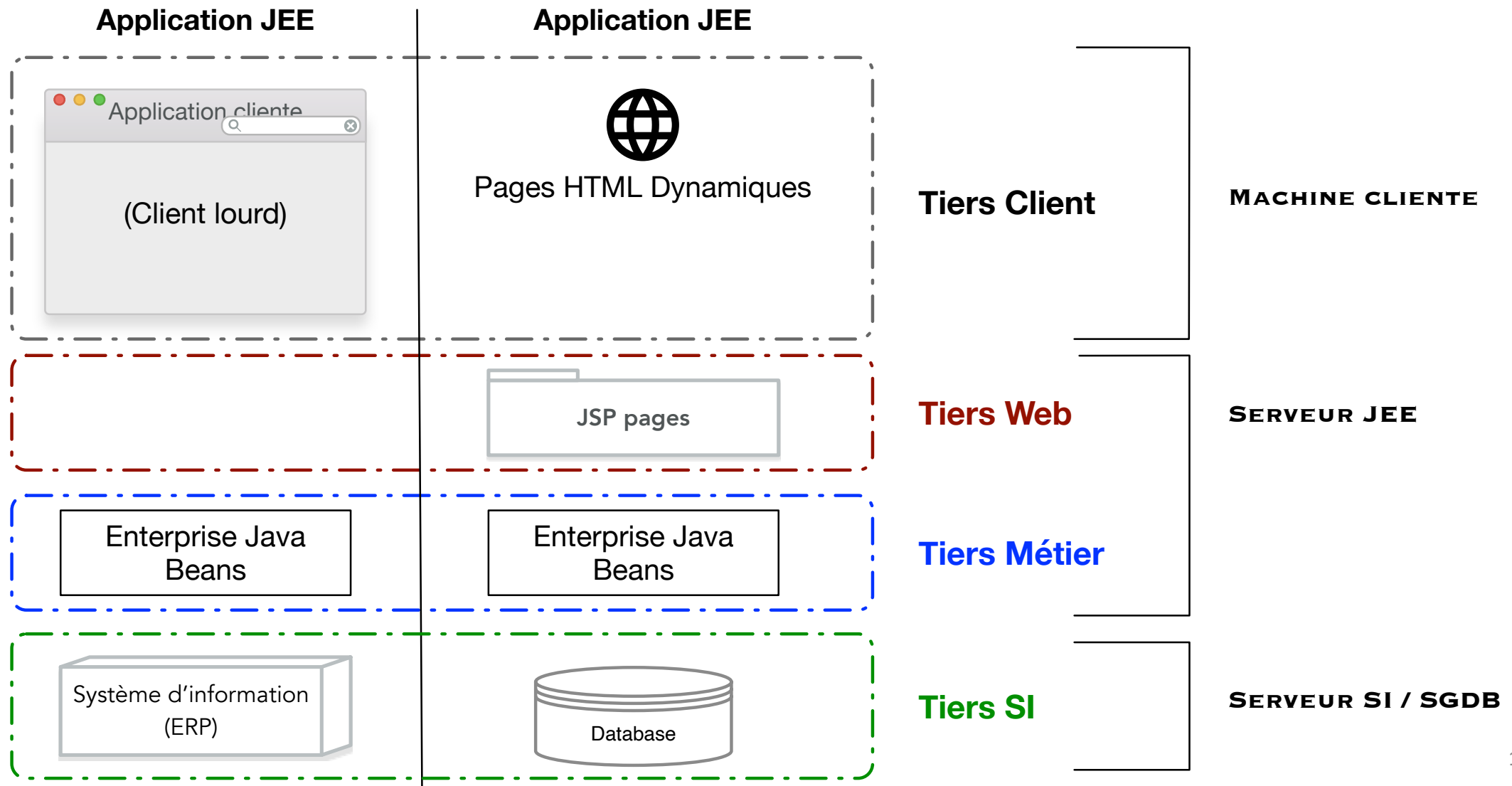


A large, dark blue ink splash graphic on the left side of the slide, with smaller blue droplets scattered around it.

Architecture multicouche (multi-tiers) répartie

- Orientée composants
- Augmentation de la cohésion du code
- Découplage fort entre les couches /
Séparation des préoccupations
- Séparation claire entre IHM, traitements
métiers et données
- Code plus facilement réutilisable
- Composants distincts, interchangeable,
répartis
- Distribution sur plusieurs nœuds (machines)

Exemples d'applications multitiers



Composants et services JEE



Composants clients : application cliente et scripts côté client



Composants Web : Java Servlet et Java Server Pages (JSP)



Composants métiers : Enterprise Java Beans (EJB)

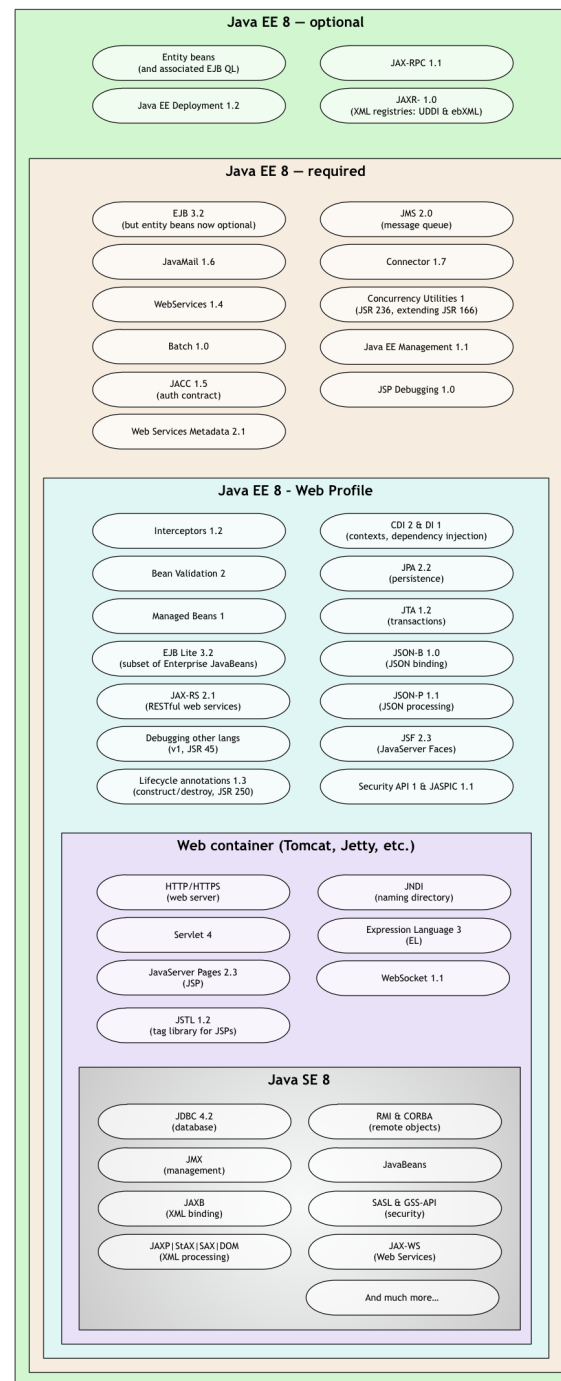


Services d'infrastructure : JDBC, JNDI, JTA, JCA, ...



Services de communication : JAAS, JavaMail, RMI

JEE Stack



Patrons de programmation JEE

- Patrons de conception pour développer des applications JEE
- Organisés selon les couches: présentation, métier, intégration

Core J2EE Patterns Book

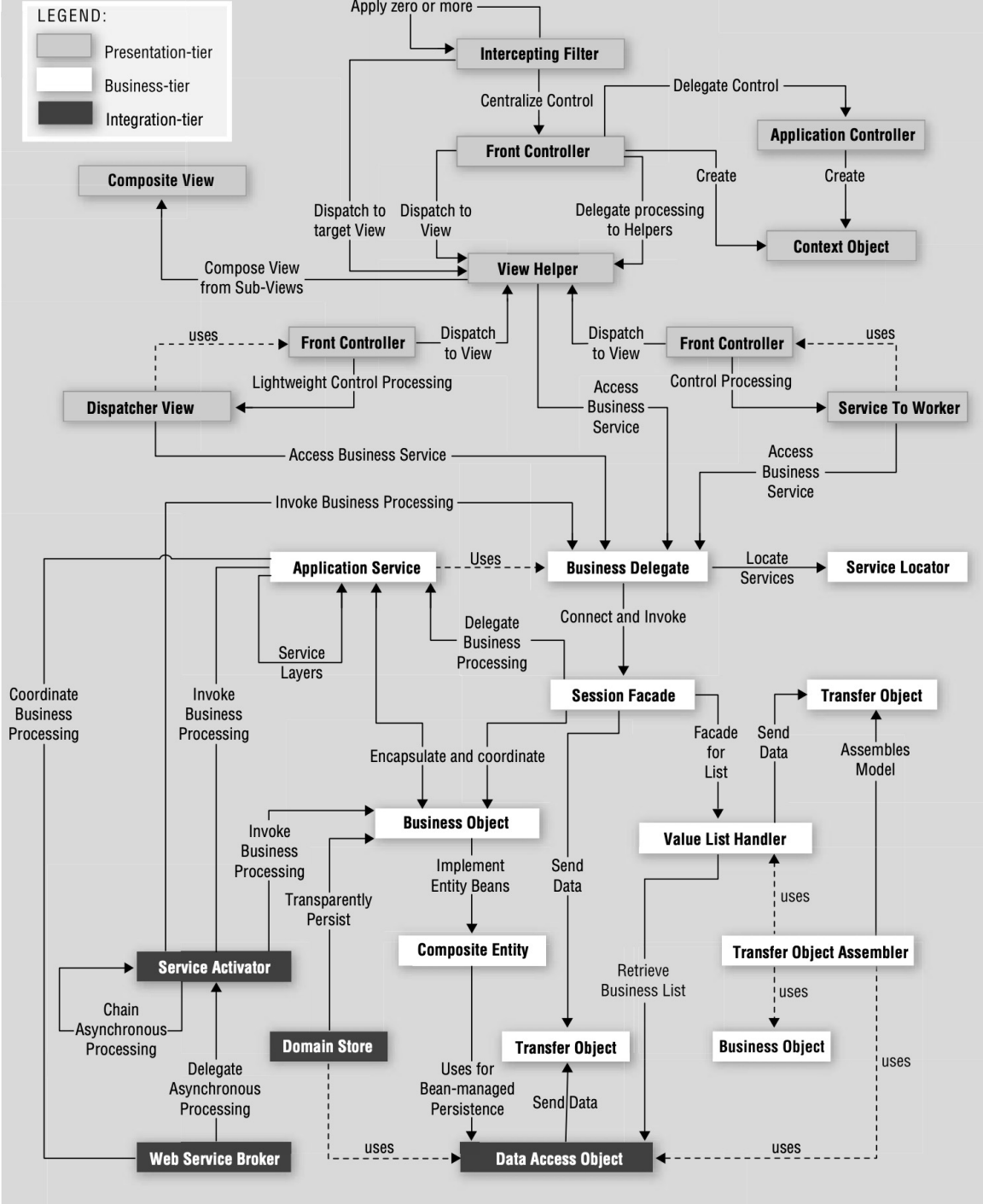


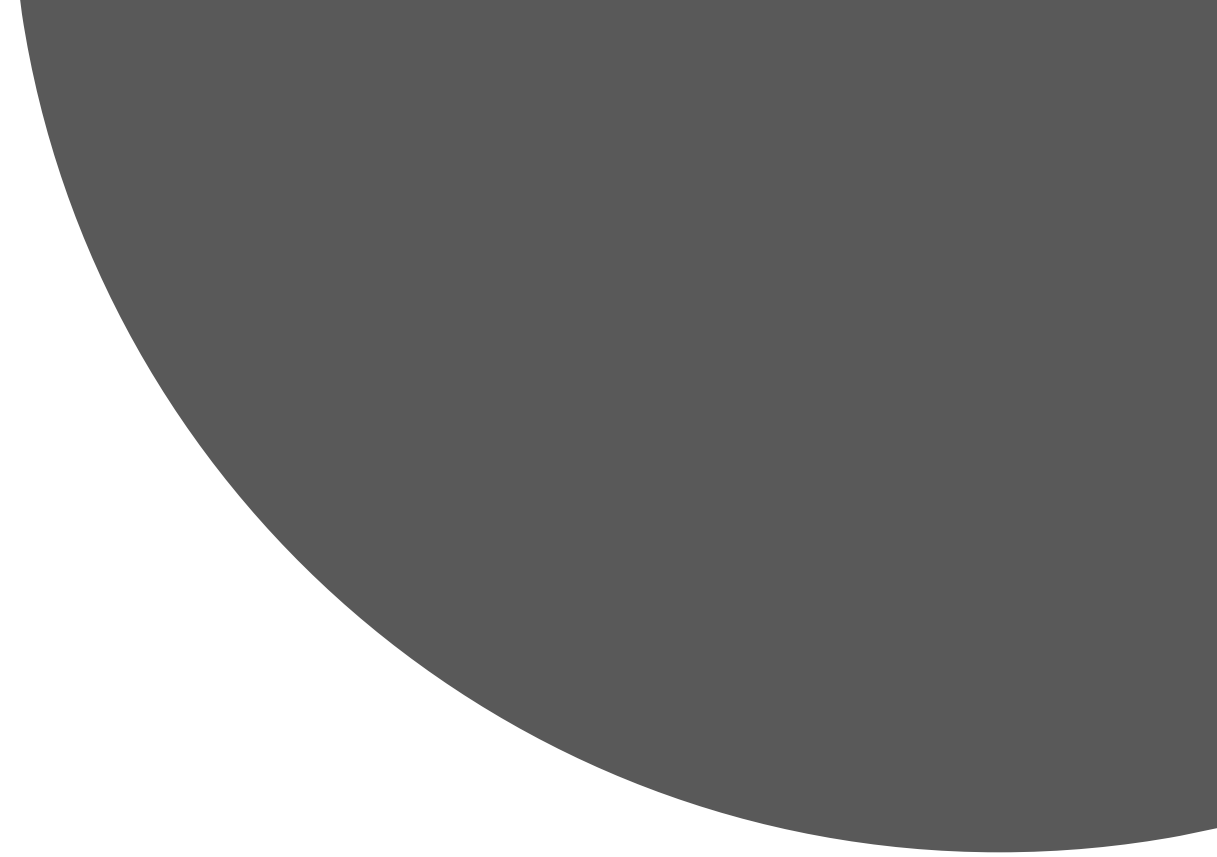
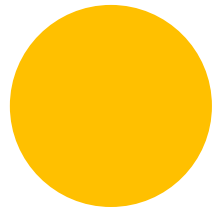
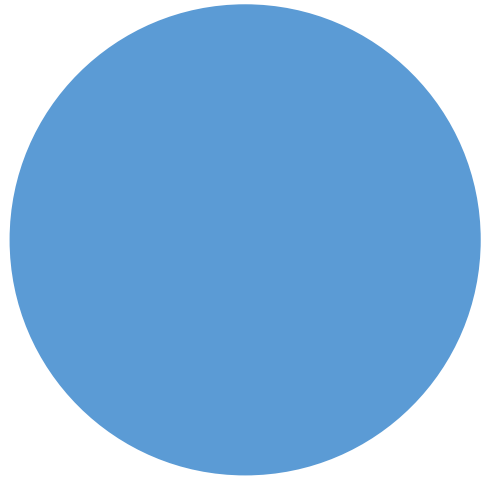
J2EE Pattern Catalog
Addresses 3 Tiers

Quelques Patrons de programmation JEE

- **Data Access Object** : communication avec la couche donnée
- **Data Transfer Object** : transfert de POJO serialisables
- **Session Facade** : encapsule les composants métiers
- **Front Controller** : route les requêtes aux bons contrôleurs
- **Intercepting Filter** : filtre les requêtes selon autorisation, authentication, browser, path constraints, etc.
- Etc.

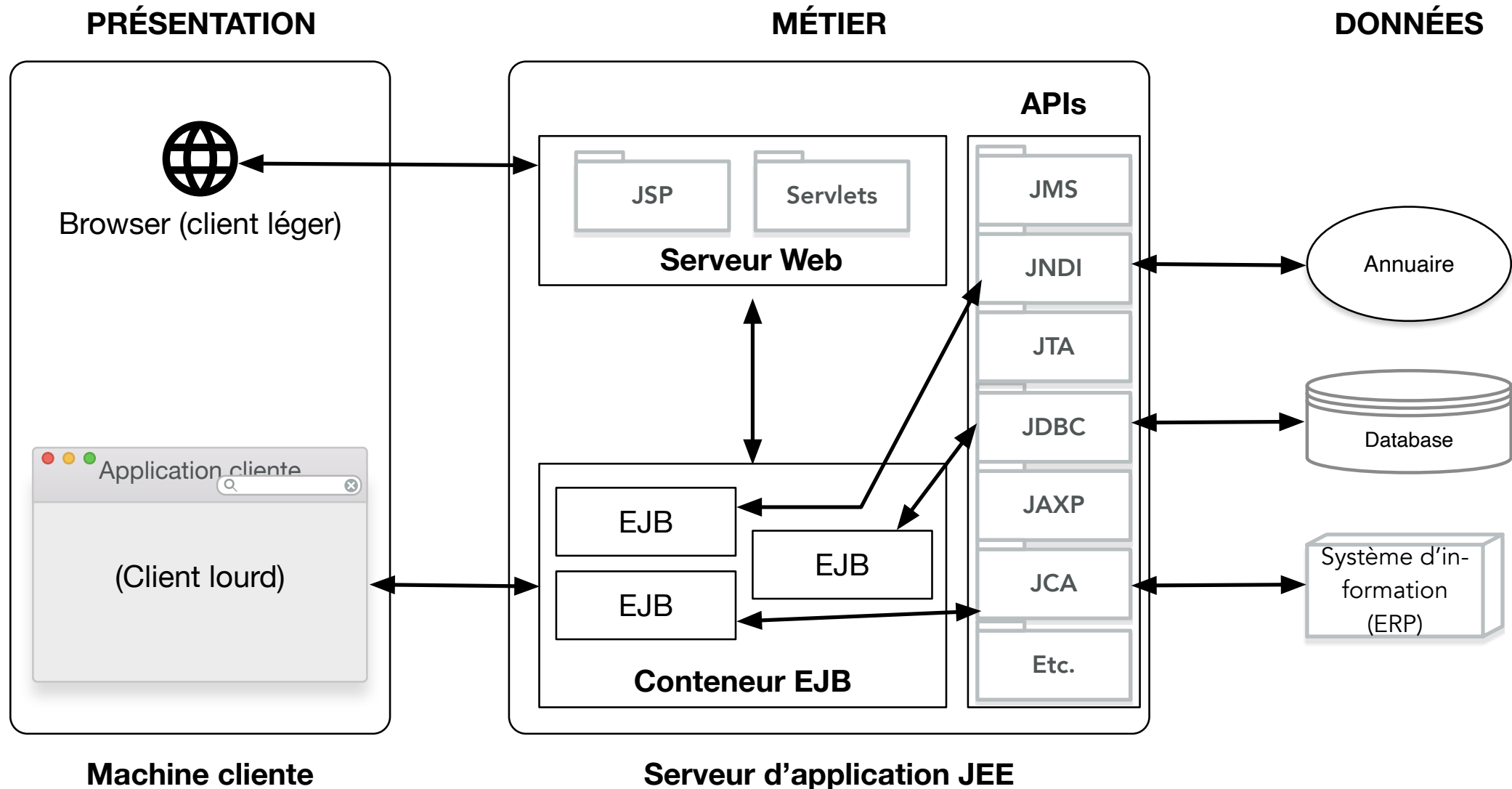
Relations entre les patrons





Composants et services côté serveur

Architecture générale 3-tiers des applications JEE



Les servlets

- Modules **server-side**, utilisés pour traiter les requêtes et envoyer les réponses
 - Classes Java pouvant être exécutées sur un serveur JEE
 - Méthode principale reçoit requête. Renvoie au client page HTML générée.
- Utilisés comme contrôleurs dans les frameworks MVC
- **Générer des réponses ne contenant pas beaucoup de HTML**
 - HTML est un String inclus dans le code Java...
- Peuvent générer des réponses non HTML, comme du PDF

Java Server Pages

- Modules **server-side**, comme les servlets, pour traiter des requêtes
- Code Java et tags JSP combinés avec du HTML, JavaScript et CSS
- **Générer des réponses contenant beaucoup de HTML**
 - Code Java est embarqué dans le HTML
 - Plus facile de générer une réponse HTML d'un JSP que d'une servlet
- JSP et Servlets peuvent être utilisés indépendamment ou de manière conjointe

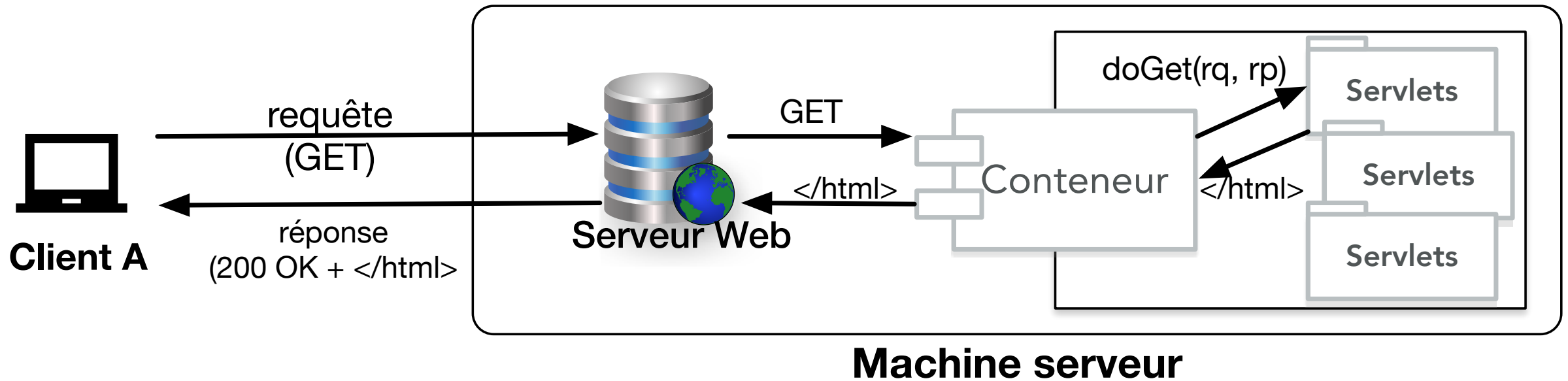
JavaServer Faces

- Création d'interface utilisateur côté serveur
- Patron MVC
 - Gérer côté serveur des événements générés par l'interface utilisateur
 - Spécifier la navigation à travers les pages dans une application
- Tags pour les contrôles dans l'interface
 - Sauvegarder des états à travers des échanges requêtes/réponses multiples
 - Ex: POST (data) -> JSF -> Java Bean (data) -> réponse (avec Java Bean)

Conteneur

- Un conteneur est un **composant** logiciel **système** qui contrôle d'autres composants, dits **métier**
- Un conteneur assure la gestion du cycle de vie des composants qu'il contient
- Fournit des services utilisés par les applications
- Conteneur Web : exécuter les servlets et JSP.
 - Exp: Tomcat dans lequel on peut déployer des servlets.
 - Requêtes adressées au conteneur, pas directement aux servlets
- Conteneur EJB : exécuter les EJB
- Conteneur client : exécuter des application standalone sur des postes utilisant des composants JEE

Application Web avec un conteneur



Services d'un conteneur aux servlets



Gestion du cycle de vie



Support pour la
communication

Élimine le besoin de
programmer soi-même
ServerSocket, Socket, etc.



Support pour le
multithreading

Création automatique de
threads



Support pour la sécurité

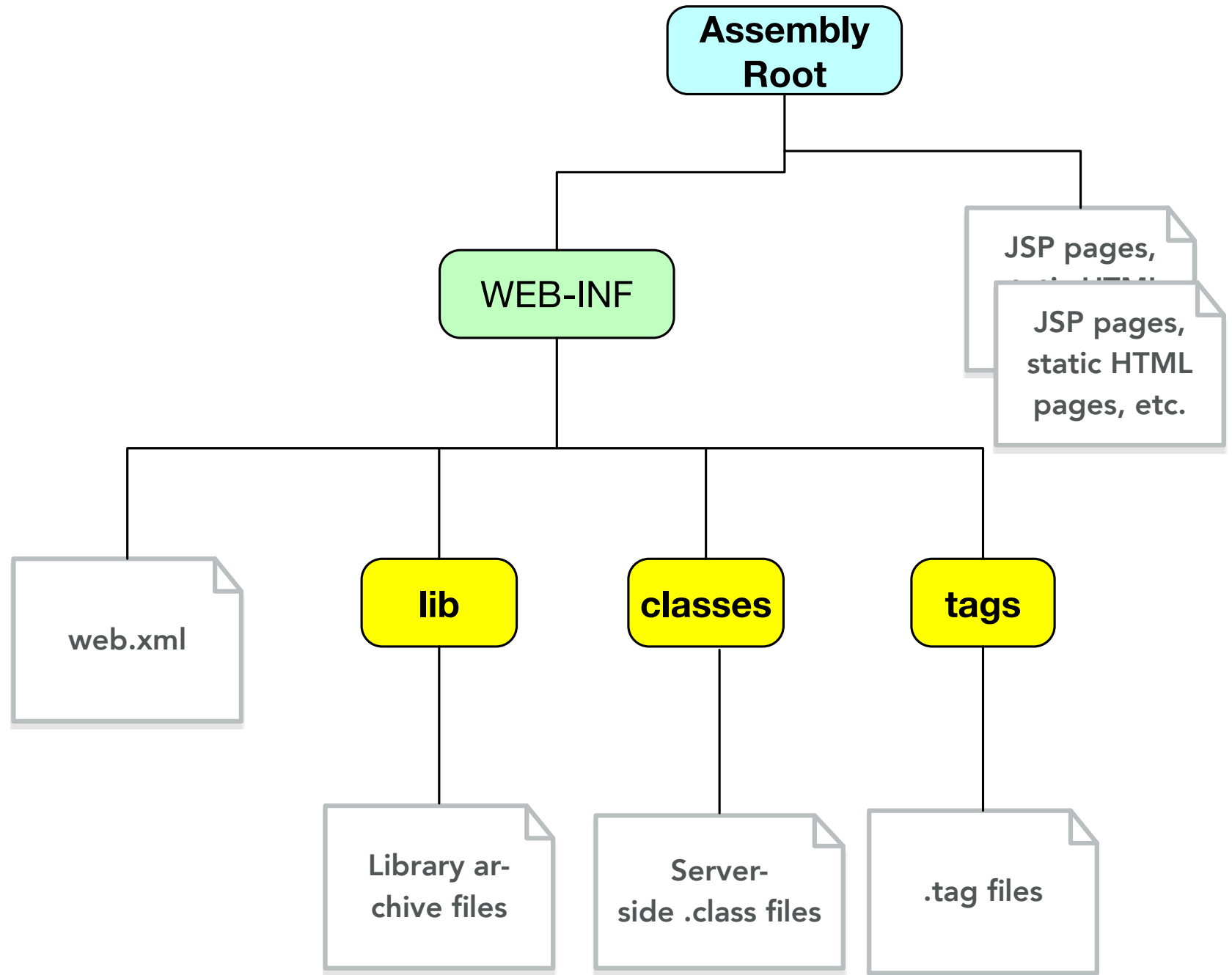


Support pour les JSP

Déployer une servlet dans un module Web

- Un servlet ne peut pas être déployé directement dans un conteneur, il doit être intégré à un **module Web**
- Un module web : ensemble de bibliothèques, de fichiers de config., de code Java (bytecode des servlets), et autres ressources
- Le module Web est **l'unité de déploiement** dans le conteneur

Structure
générale
d'un module
Web



Types d'archives

- **JAR (.jar)** : regroupe des classes
 - Bibliothèque, application client
 - Descripteur de déploiement pour app client -> application-client.jar
- **WAR (.war)** : regroupe des servlets et JSP
 - Toutes les ressources nécessaires à leur exécution (classes, images, balises)
 - Descripteur de déploiement -> web.xml
- **EJB (.jar)** : regroupe les EJB et leurs composants (classes)

Application

- Une application est un **regroupement** d'un ou plusieurs modules dans un fichier EAR (Enterprise Archive)
- Application décrite dans un fichier *application.xml* (dans le fichier EAR)
- Déployer une application dans un conteneur nécessite :
 - Tous les composants (classes, ressources) dans une archive ou module
 - Chaque conteneur a son propre format d'archive
 - Fichier descripteur de déploiement contenu dans le module
 - Précise les options pour exécuter l'application

Serveur d'application



Gère la couche métier de l'application



Fournit un conteneur Web uniquement
(ex.: Tomcat, Jetty)



Fournit un conteneur d'EJB uniquement
(ex.: Jboss, Jonas)



Fournit les deux (ex.: GlassFish,
WebSphere, WildFly)

Enterprise JavaBeans (EJB)

- Classes Java gérant la logique métier
- Peuvent être distribués sur plusieurs serveurs
 - Conteneur EJB se chargera de la recherche (component lookup)
 - Scalabilité
- Deux types :
 - **Session beans** : appelés directement par les clients ou objets middle-tier
 - **Message-driven beans** (MDB) : appelés en réponse à des événements Java Messaging Service (JMS)
- JMS et MDB pour gérer des requêtes asynchrones (message queues)

Java Database Connectivity (JDBC)

- Spécification pour l'accès aux BD relationnelles de manière **uniforme**
- Exécution de requêtes SQL et récupération des résultats avec même API, quelque soit le SGBD
- Un **driver** traduit les appels JDBC vers l'API propriétaire du SGBD
- Appels à BD doivent être séparés de UI et code métier
 - Utilisation des **Data Access Objects (DAO)** pour encapsuler la logique d'accès aux BD

Java Persistence API (JPA)

- Fournit un service **Object Relational Mapping** (ORM)
 - Mapping flexible entre objets Java et données dans la BD relationnelle
 - On peut exécuter les requêtes sans avoir à utiliser directement SQL
 - Java Persistence Query Language
- Résout le problème d'avoir à mapper soi-même la donnée entre les objets Java et la BD
 - Une des limites dans l'utilisation de JDBC
- Hibernate et Spring ont popularisé le concept d'ORM

Java Connector Architecture (JCA)

- API pour communiquer avec le reste du système d'information de l'entreprise
 - Ex.: SAP, Salesforce
- Intégration avec le système d'information
- REST prend de plus en plus le pas sur cette techno

Web Services

- Applications réparties exposant leurs propres API
- Deux standards :
 - Simple Object Access Protocol (SOAP)
 - Representational State Transfer (REST)
- Indépendants des plateformes technologiques
 - Internet of services
- JEE fournit des composants pour simplifier le dév. et l'utilisation
 - JAX-WS (Java API for XML)
 - JAX-RS (Java API for RESTful web services)

Services de la plate-forme JEE

- Service de nommage
 - Service de déploiement
 - Service de gestion des transactions
 - Service de sécurité
-
- Services accessibles via des API
 - Utilisés par les conteneurs et les composants s'exécutant dans ceux-ci



Environnements de développement

Outils pour débuter

- Java EE 8 Platform SDK
 - <https://www.oracle.com/java/technologies/java-ee-glance.html>
 - <https://javaee.github.io/glassfish/documentation>
- JDK 8, 11, or 14
- Eclipse IDE for Enterprise Java Developers
 - <https://www.eclipse.org/downloads/packages/>
- MySQL Community Server
 - <https://dev.mysql.com/downloads/mysql/>
- MySQL Workbench
 - <https://www.mysql.com/products/workbench/>

Outils pour plus tard

- MongoDB Community Server
 - <https://www.mongodb.com/try/download/community>
- Robot 3T
 - <https://robomongo.org/download>
- Spring Tools Suite 4
 - <https://spring.io/tools>
 - <https://www.jetbrains.com/help/idea/spring-support.html>
- Guides pour Spring
 - <https://spring.io/guides#getting-started-guides>

Références

- Java EE 8 Development with Eclipse, 3rd edition
- Core J2EE Patterns 2nd Edition – Best Practices and Design Strategies
- Beginning EJB in Java EE 8
- Java Specification Requests (JSR) for JEE Platform
 - <https://jcp.org/en/jsr/platform?listBy=3&listByType=platform>
- Supports de cours Univ. MLV
- Supports de cours de V. Gardeux