Augmenter la disponibilité des applications JEE grâce au clustering : Le projet open source JShaft

Jérôme Petit, Serge Petit & Pierre-H. Dezanneau

Serli Informatique, ITMatic

ObjectWeb
Open Source Middleware

SERLI & ITMatic

Serli : SSII de 50 personnes

- Systèmes d'informations
- Embarqué & temps réel
- Systèmes & réseaux
- Marquage de documents techniques

ITMatic : cabinet d'architecture JEE

- Architecture et développement Java EE

SERLI & ITMatic

Des équipes impliquées dans l'Open Source

- Utilisation massive de briques Open Source
- Contributions diverses au cours des projets
- Engagement personnel par philosophie

Gestion de projets Open Source Java

- JaasLounge : interopérabilité JAAS pour Java EE
- JShaft : gestion de clusters Java EE
- D'autres projets en gestation

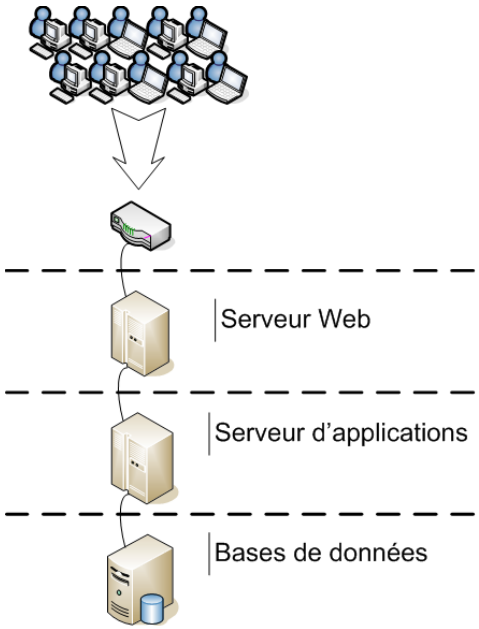
SERLI & ITMatic : l'Open Source, pourquoi ?

Apports de notre engagement Open Source :

- Une école de qualité technique
- Des compétences Up-to-Date
- La crédibilité auprès de nos clients

Plan

Architectures n-tiers



Applications orientés serveurs

Chaque couche logicielle est centralisée :

- Présentation : Pages Web
- Métier : Serveurs d'applications
- Données : Base de données

Architectures n-tiers

Avantages de la gestion centralisée

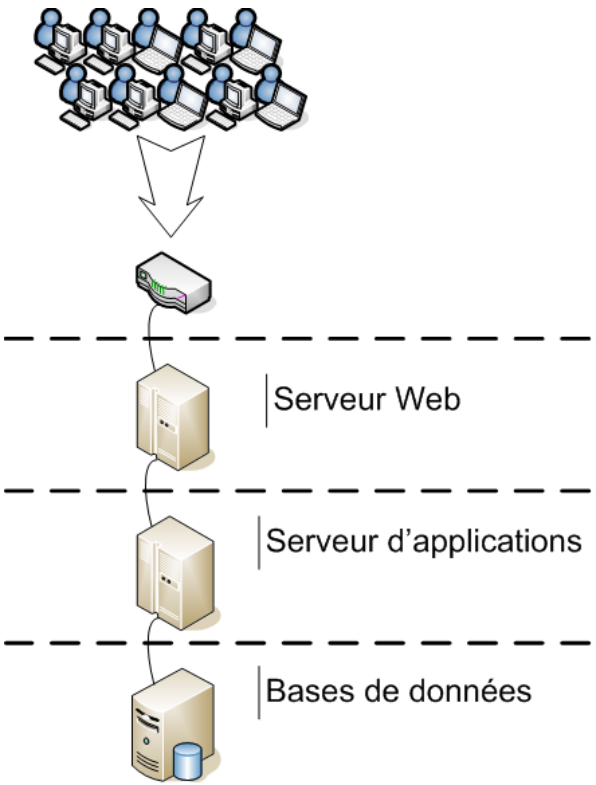
- Architecture transparente pour l'utilisateur (un seul point d'entrée)
- Mises à jour simplifiées
- Administration simplifiée
- Sécurité accrue
- Gestion des transactions, ...

Architectures n-tiers

Inconvénients

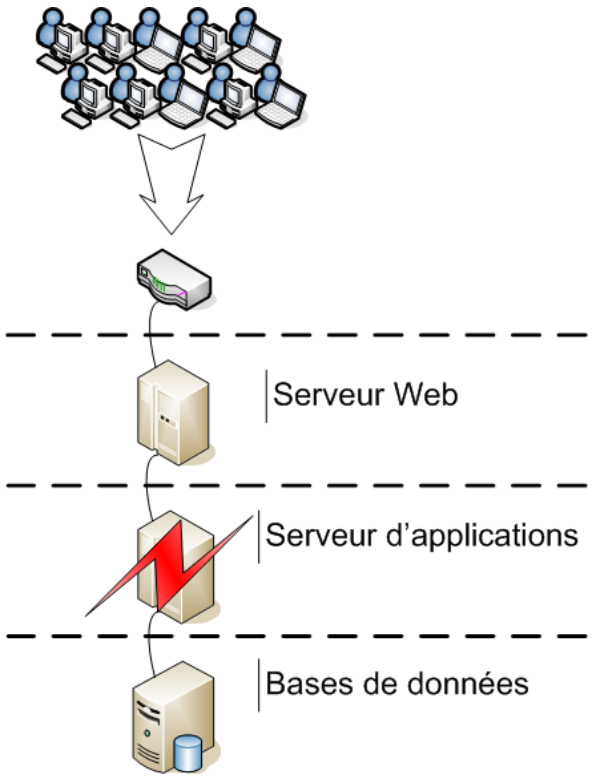
- Hautes performances des serveurs requises
- Apparition des SPOF (Single Point of Failure)

SPOF



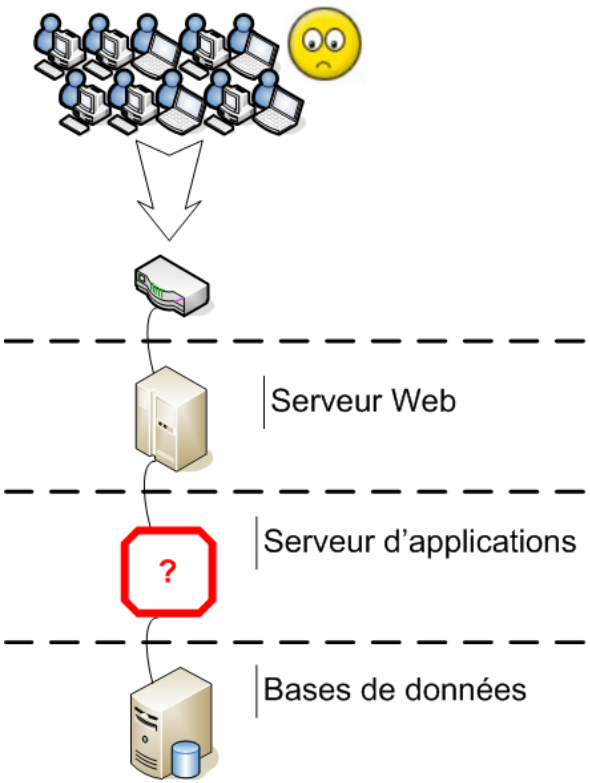
- Chaque tier est représenté par une machine unique

SPOF



- Une machine subit une défaillance

SPOF



- L'architecture *complète* est indisponible

Problèmatique

Problèmatique

- Supprimer les SPOF dans une architecture n-tiers

Moyens

- Augmenter la disponibilité des services
  ➜ Solutions de clustering

Une définition...

✎ Dispositifs visant à garantir la disponibilité d'un service.

Taux de disponibilité	Durée d'indisponibilité
97	11 jours
98	7 jours
99	3 jours et 15 heures
99,9	8 heures et 48 minutes
99,99	53 minutes
99,999	5 minutes
99,9999	32 secondes

Principes

- La redondance augmente la disponibilité
- Mettre en place un système hautement disponible prend du temps

Haute Disponibilité et Clustering

✎ Le clustering est une solution pour rendre une architecture n-tiers hautement disponible.

Le clustering

Définition

- Création d'un ou plusieurs clusters
- Plusieurs ordinateurs vus comme une seule et unique machine
- Noeuds connectés par un réseau haut débit
- Chaque noeud travaille indépendamment des autres

Deux grands types de cluster

- Clusters de calcul (Grid Computing)
- Clusters de haute disponibilité

Le clustering

Définition

- Création d'un ou plusieurs clusters
- Plusieurs ordinateurs vus comme une seule et unique machine
- Noeuds connectés par un réseau haut débit
- Chaque noeud travaille indépendamment des autres

Deux grands types de cluster

- Clusters de calcul (Grid Computing)
- Clusters de haute disponibilité

Avantages

Définition

Le clustering permet :

- Un meilleur support de la montée en charge
  ➜ Scalability
- Augmentation de la disponibilité
  ➜ Fail-Over
- Permet de répartir la charge
  ➜Load-Balancing

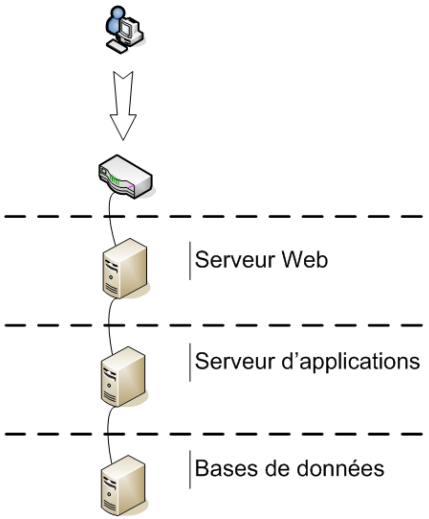
✔ Faible coût.

Scalabilité

Objectifs

- Traitement d'une requête :
  ➙ temps t
- Traitement pour un ensemble de requêtes concurrentes :
  ➙ temps proche de t

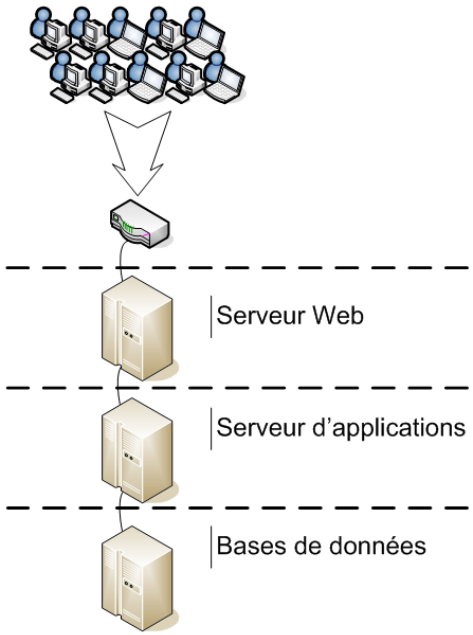
Scalabilité



Problèmatique

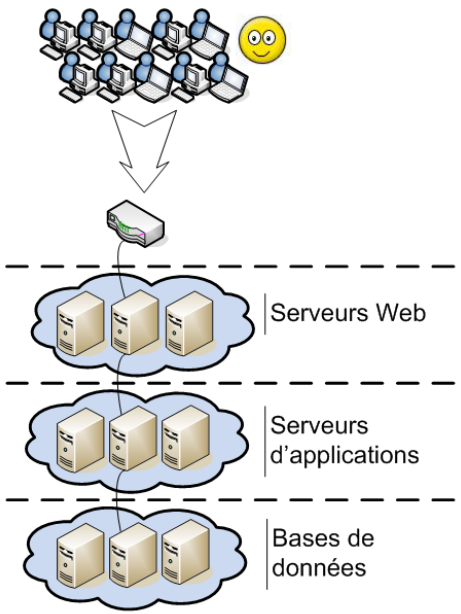
- Comment supporter une charge importante ?

Scalabilité



➙ Scalabilité verticale

Scalabilité



→ Scalabilité horizontale

Scalabilité

Pour assurer la montée en charge d'une architecture n-tiers

1. Augmenter les performances des serveurs
   - RAID
   - Mémoire
   - CPU
   - ...
2. Augmenter le nombre de noeud

Avantage

➜ Deuxième solution moins coûteuse

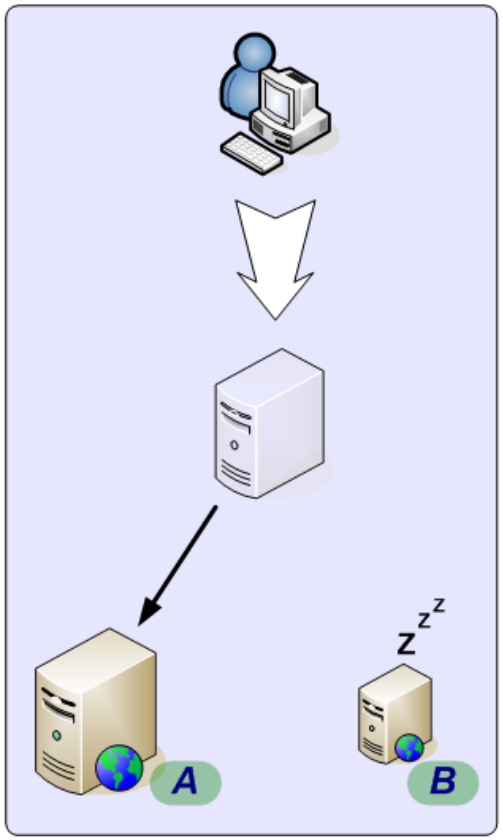
Fail-Over

Objectifs

Garantir une continuité de service en cas de panne d'une machine

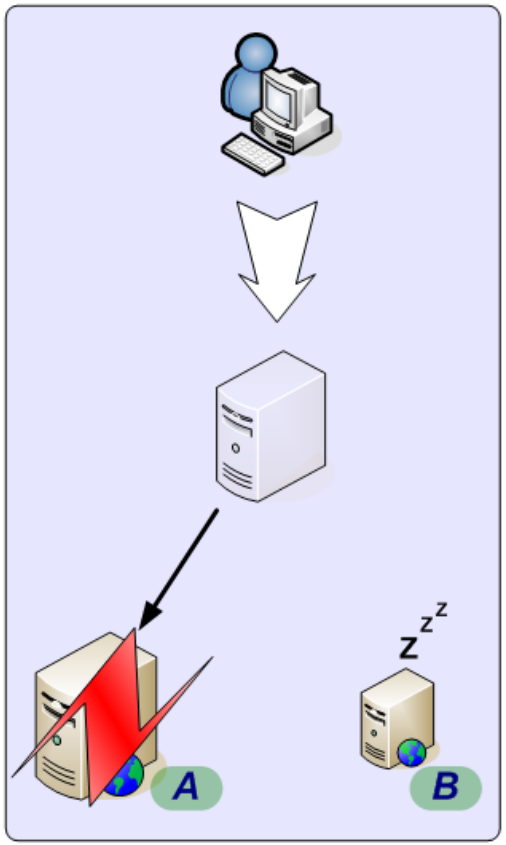
- Sans intervention humaine
- La plus transparente possible pour l'utilisateur

Fail-Over



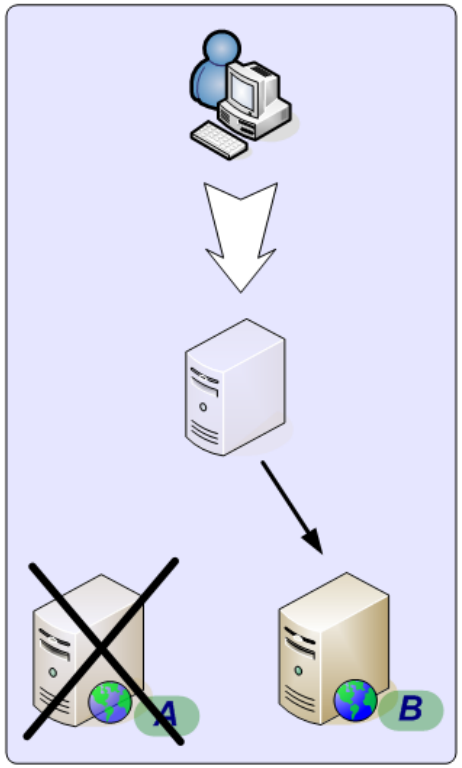
- A est actif
- B surveille l'état de A

Fail-Over



- A subit une défaillance

Fail-Over



- B remplace A
  ➜ Le service est toujours actif

Fail-Over

✎ Une architecture de Fail-Over peut supporter une réplication d'état

Réplication de l'état

- En mémoire centrale
- Base de donnée unique
- Système de fichier partagé

✎ Les mécanismes de réplication permettent la continuité de service

Coût de la continuité

- La réplication de l'état implique des ressources supplémentaires

Fail-Over

Coût de la continuité

Le Fail-Over peut être assuré avec la virtualisation des serveurs :

- Plusieurs instances de serveurs
- Plusieurs instances de systèmes d'exploitation

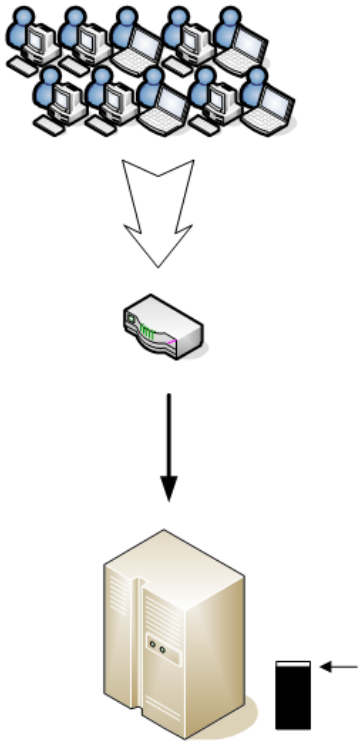
✎ Les besoins guident les choix de fonctionnalités du cluster

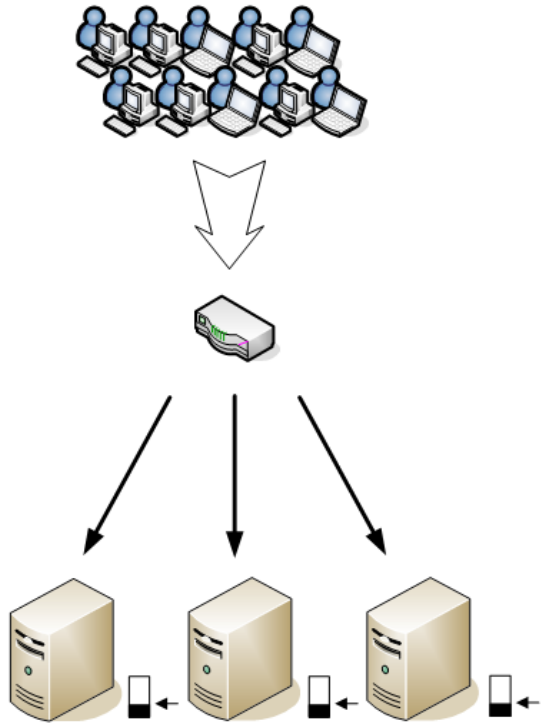
Load Balancing

Objectifs

- Distribution de la charge
- Différents algorithmes
  - Round-Robin
  - Weight-Based
  - Aléatoire
- Différentes solutions
  - Logicielles : machine dédiée
  - Matérielles : DNS Round Robin (Cisco, ...)

Load Balancing





Load Balancing

Avantages

- Perte d'un noeud :
  - Pas la rupture du service
  - Diminution faible de la puissance CPU du cluster

Le clustering et l'architecture JEE

Un niveau d'abstraction supplémentaire

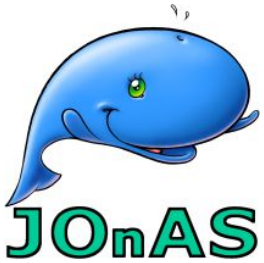
✎ Un cluster est constitué de plusieurs instances du serveur JEE pouvant être hébergées par un ou plusieurs ordinateurs

Inconvénients

- La complexité de mise en oeuvre...
- Les fonctionnalités disponibles...
- Les systèmes de gestion de cluster...

✖ varient énormement d'un produit à un autre.

Présentation



▸ Lien

Version actuelle : JOnAS 4.7.4

- JOnAS est un serveur d'application open-source conforme aux spécifications Sun J2EE 1.4
- Un projet du consortium ObjectWeb
- Communauté très active

Exemple SampleCluster2

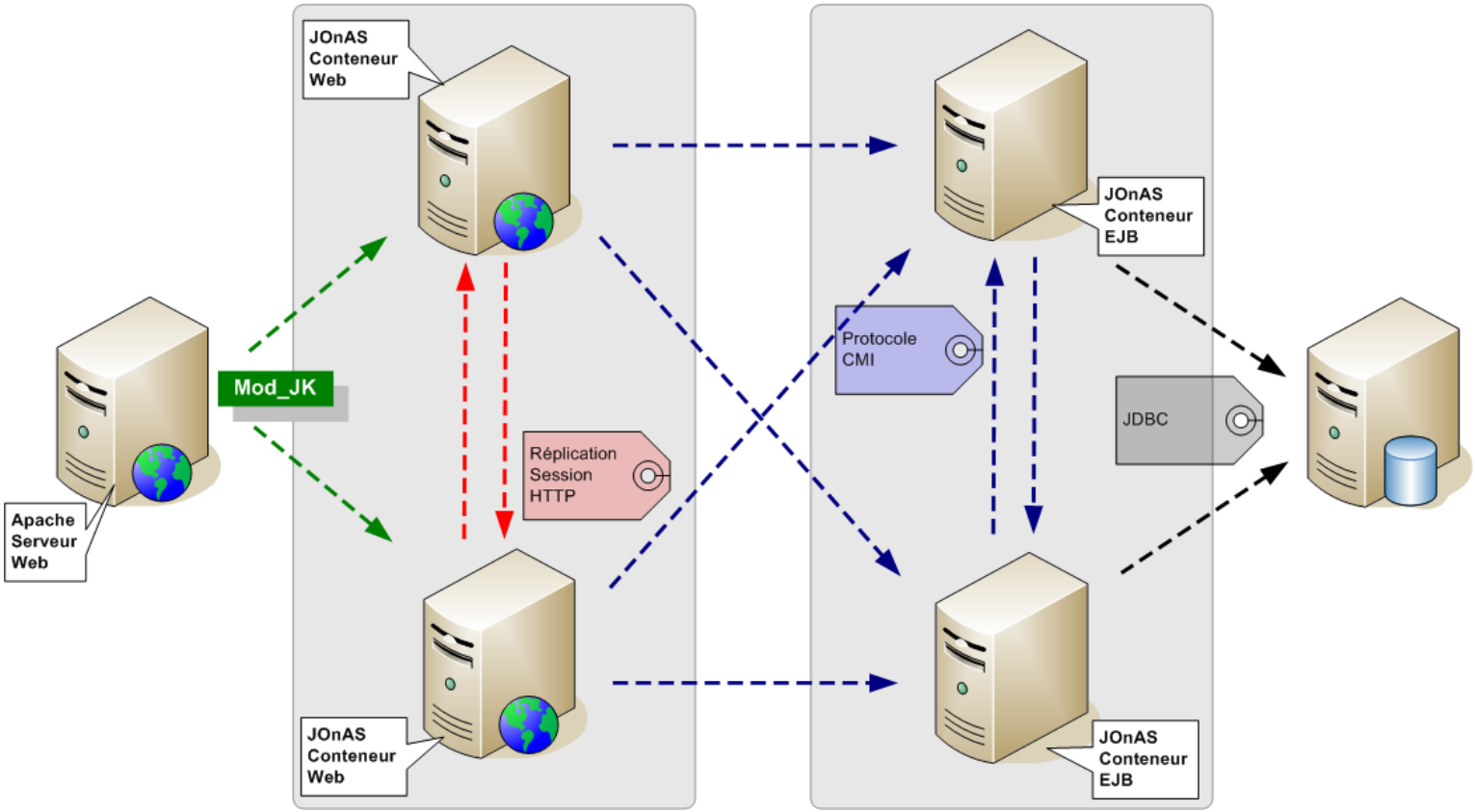
SampleCluster2

- Disponible depuis la version 4.5 de JOnAS ▸ Lien
- Intégration de l'outil *newjc* ▸ Lien

Objectifs de newjc

- Outil de création d'une configuration de cluster JOnAS
- Simplicité de mise en oeuvre
- Déploiement d'une application de test

Architecture de la démonstration



Exemple SampleCluster2

Fonctionnalités mises en oeuvre

- Répartiteur de charge HTTP
- Réplication de sessions HTTP
- Réplication de l'état des EJB

Disponibles

- Fail-Over et Load-Balancing pour le tier Web et pour le tier métier

➜ *Démonstration sur une machine unique*

Fonctionnement

SampleCluster2

- 3 types d'EJB sont utilisés (Stateless et Statefull Session Bean, Entity Bean)
- Au premier appel de la JSP :Un Statefull Session Bean est créé et sa référence est ajoutée à la session HTTP
- Aux appels suivants :
  - Un Stateless Session Bean est créé, les informations qu'il contient sont ajoutées au Statefull Session Bean
  - La page JSP affiche les informations recueillies par le Statefull Session Bean
- Toutes les 10 instanciations du Stateless Session Bean, un Entity Bean est créé

Démonstration

Sequoia

Présentation

Buts

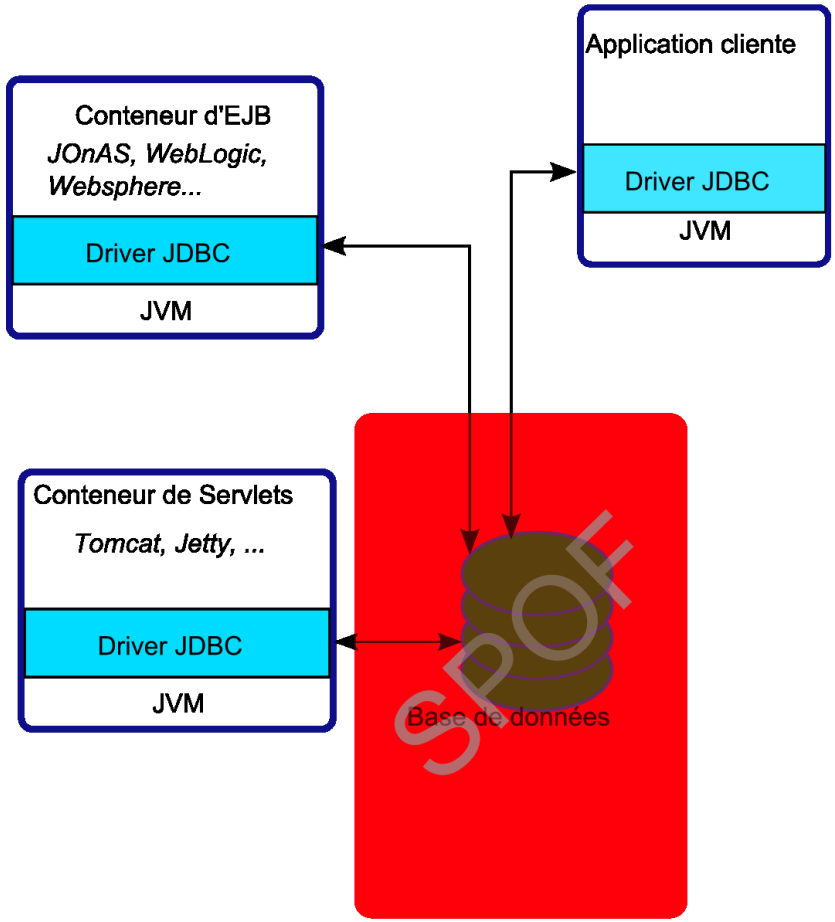
- Que la base de données ne soit plus un SPOF
- Répartition de charge

➜ Sequoia est la suite de C-JDBC.

C-JDBC fut au départ réalisé par l'INRIA, puis d'autres contributeurs ont rejoint le projet. Sequoia est hebergé par Continuent.org, et une partie des premiers contributeurs travaille pour Continuent Inc.

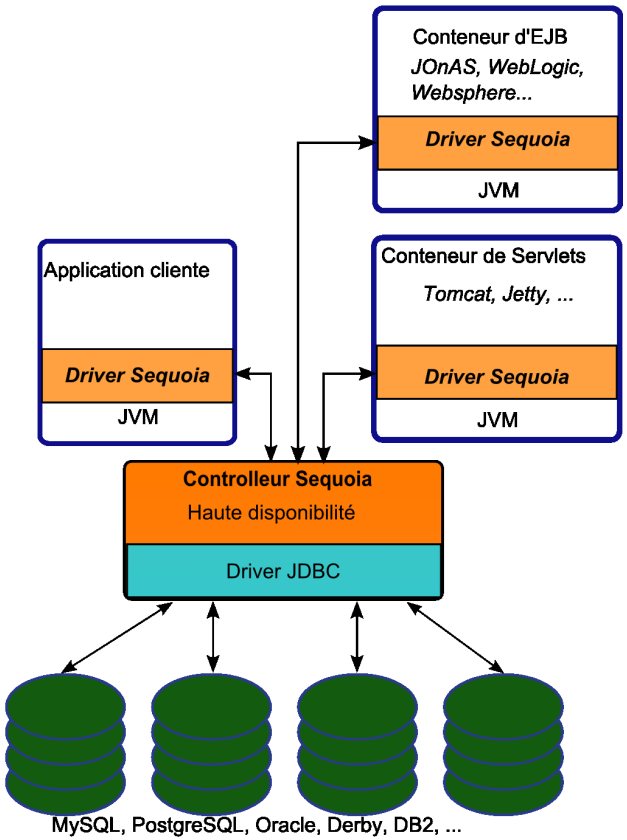
- http ://c-jdbc.objectweb.org LGPL
- http ://sequoia.continuent.org Apache v2

Fonctionnement

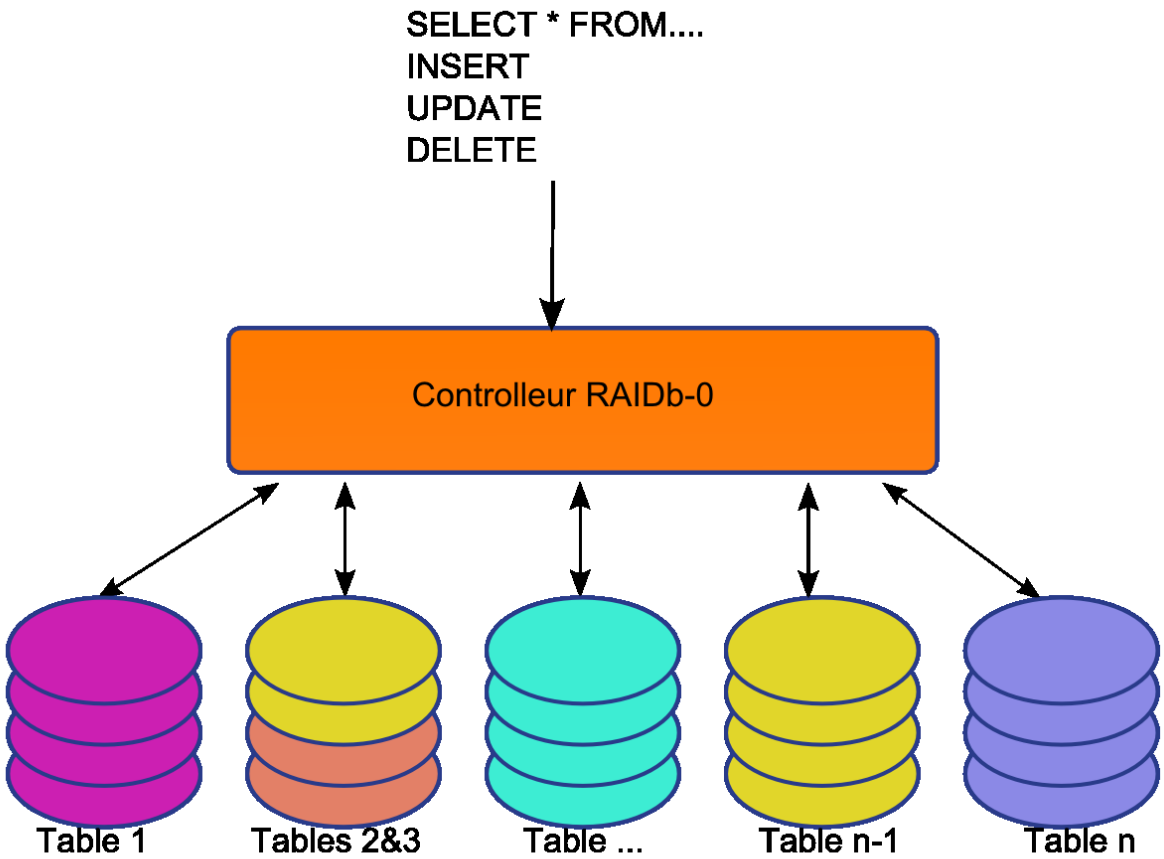


Fonctionnement

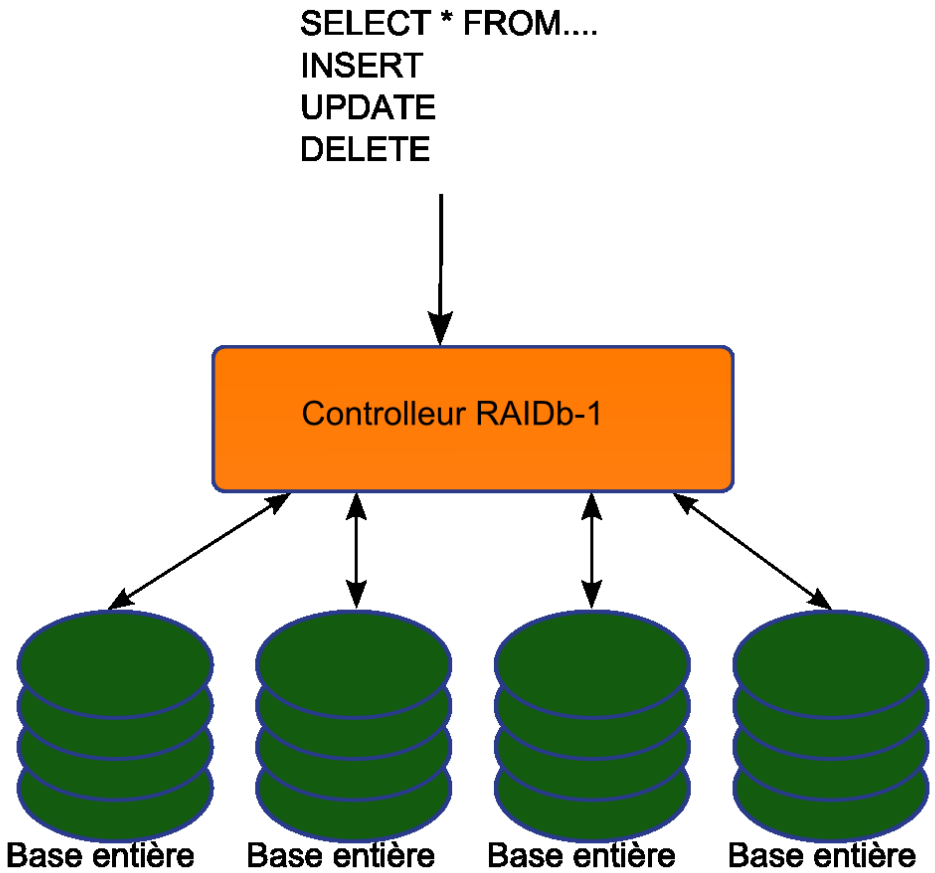
Remplacement du driver JDBC : transparent pour l'application



Raidb 0


SELECT * FROM....
INSERT
UPDATE
DELETE
Controlleur RAIDb-0
Table 1
Tables 2&3
Table ...
Table n-1
Table n

Raidb 1


SELECT * FROM....
INSERT
UPDATE
DELETE
Controlleur RAIDb-1
Base entière
Base entière
Base entière
Base entière

Raidb 2

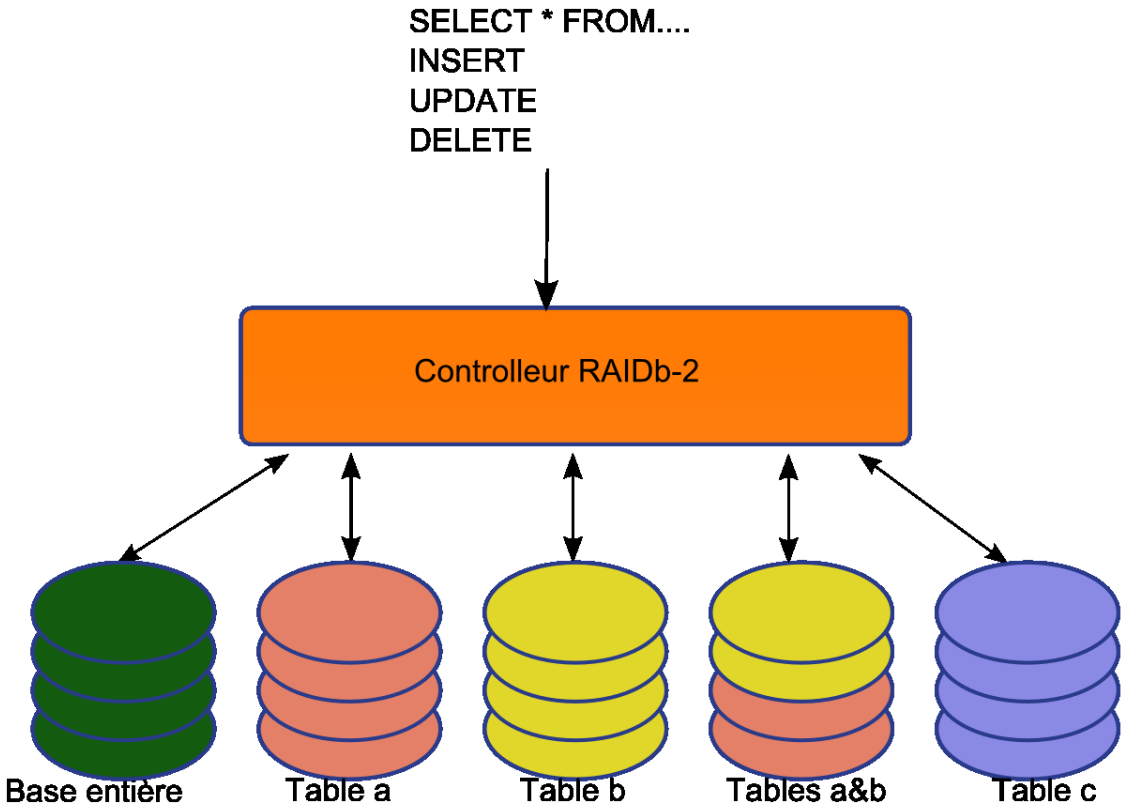
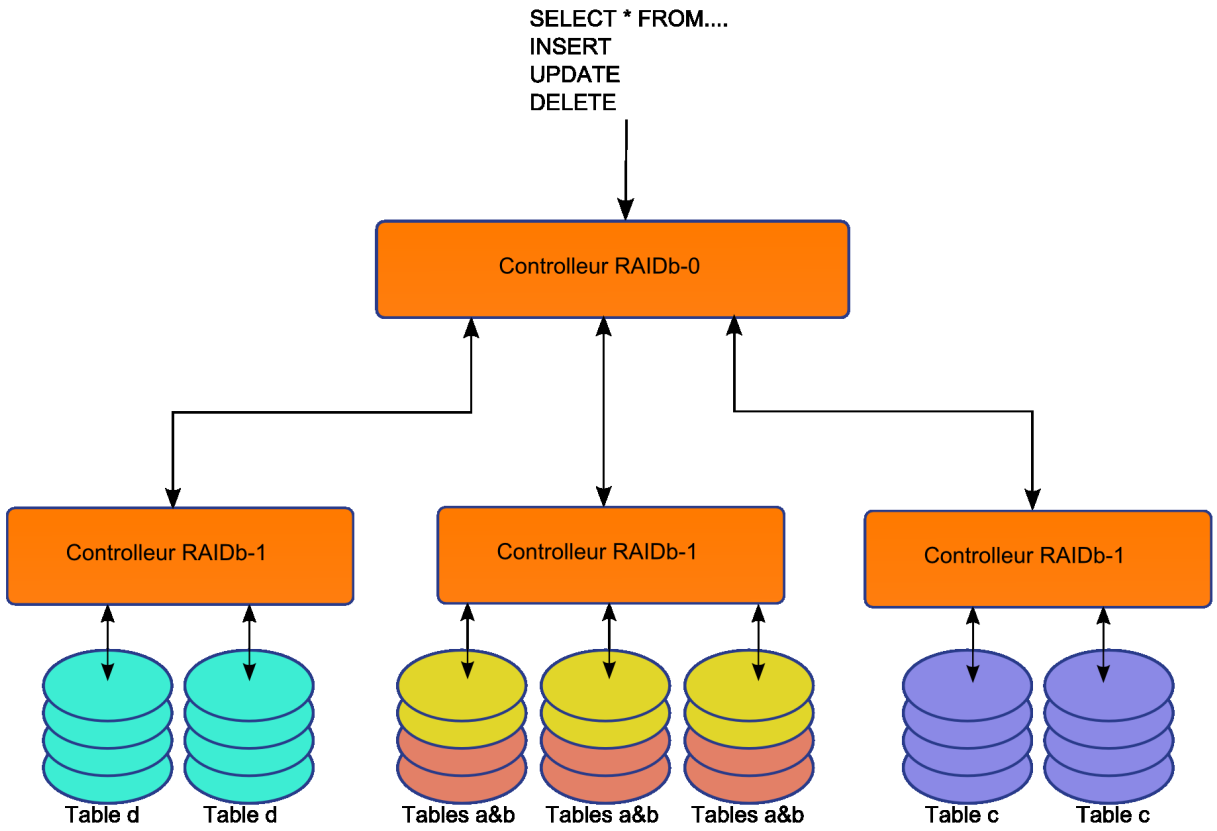
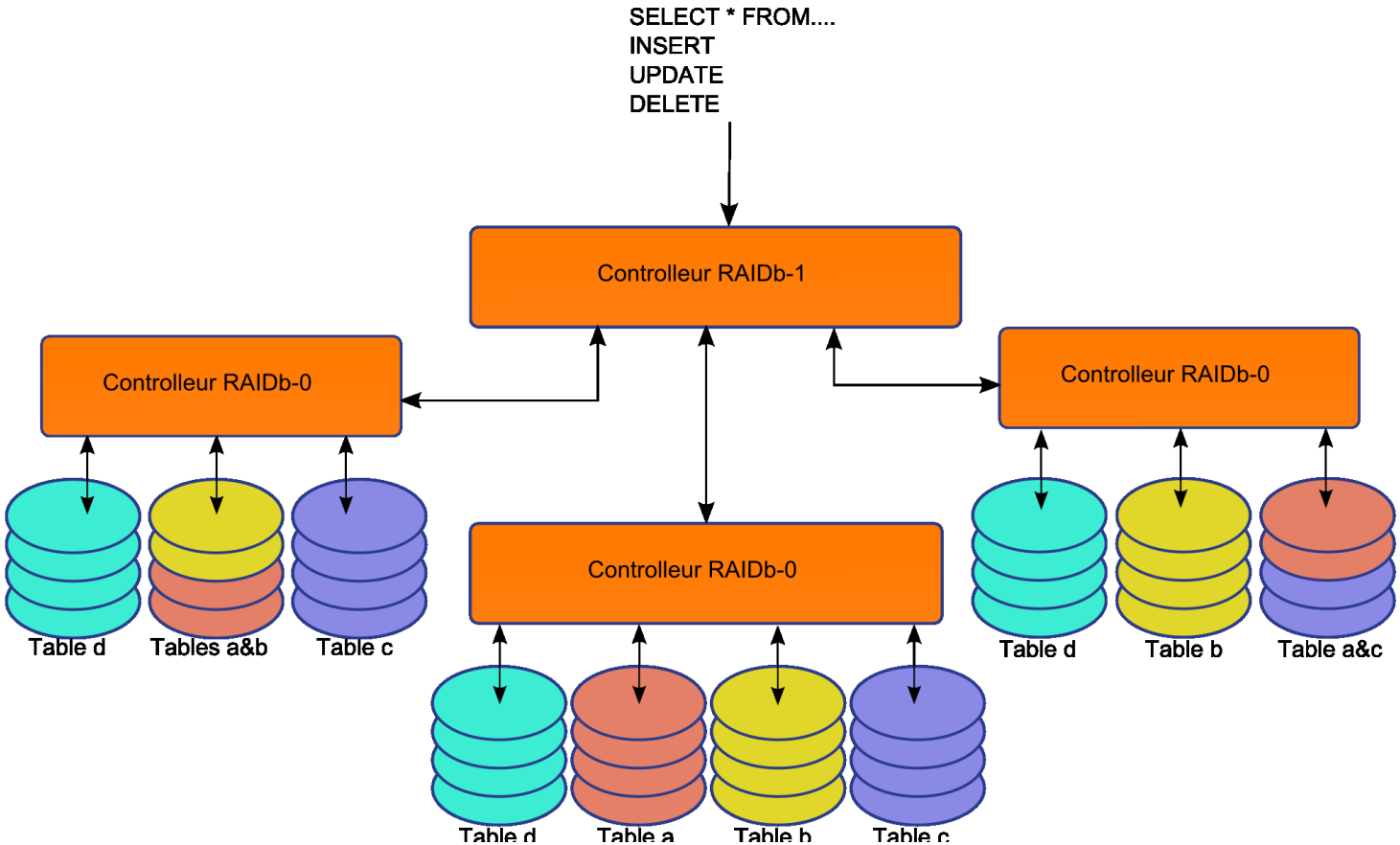


Image extraite de la documentation officielle du projet Sequoia

Raidb 0-1


SELECT * FROM....
INSERT
UPDATE
DELETE
Controlleur RAIDb-0
Controlleur RAIDb-1
Controlleur RAIDb-1
Controlleur RAIDb-1
Table d
Table d
Tables a&b
Tables a&b
Tables a&b
Table c
Table c

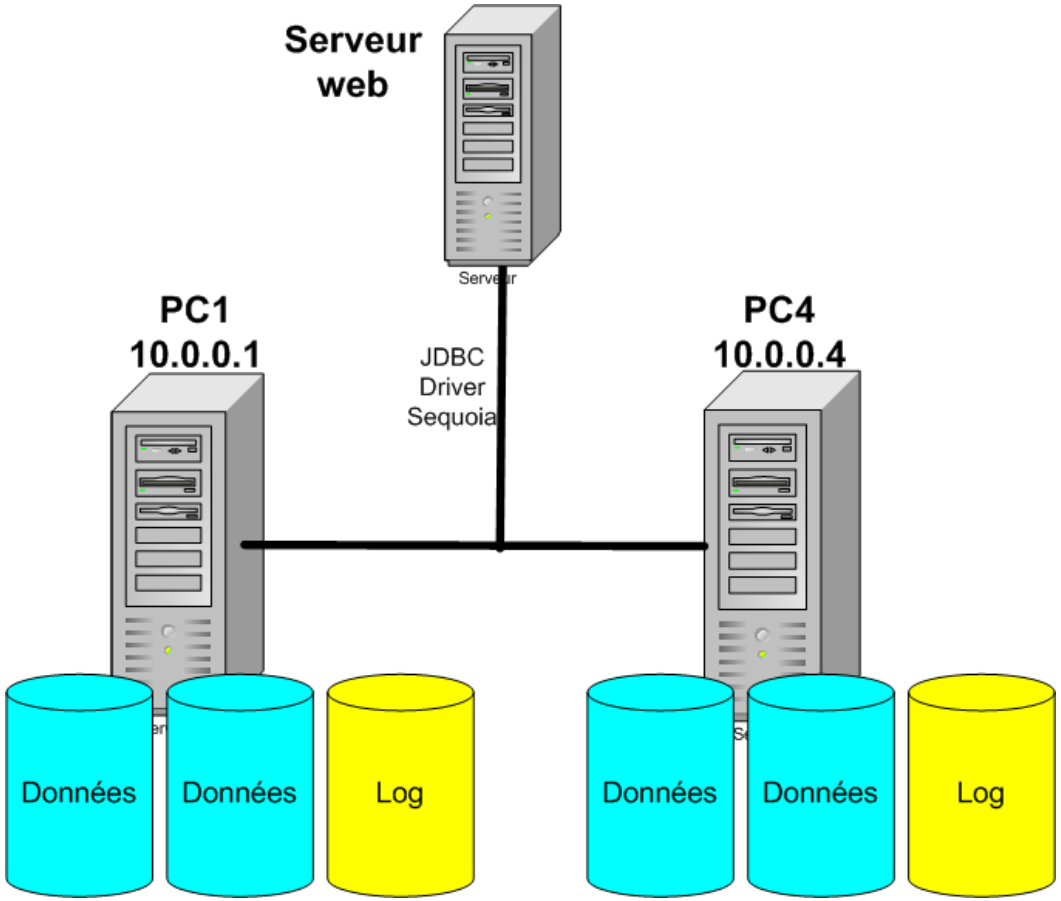
Raidb 1-0


SELECT * FROM....
INSERT
UPDATE
DELETE
Controlleur RAIDb-1
Controlleur RAIDb-0
Controlleur RAIDb-0
Controlleur RAIDb-0
Table d
Tables a&b
Table c
Table d
Table b
Table a&c
Table d
Table a
Table b
Table c

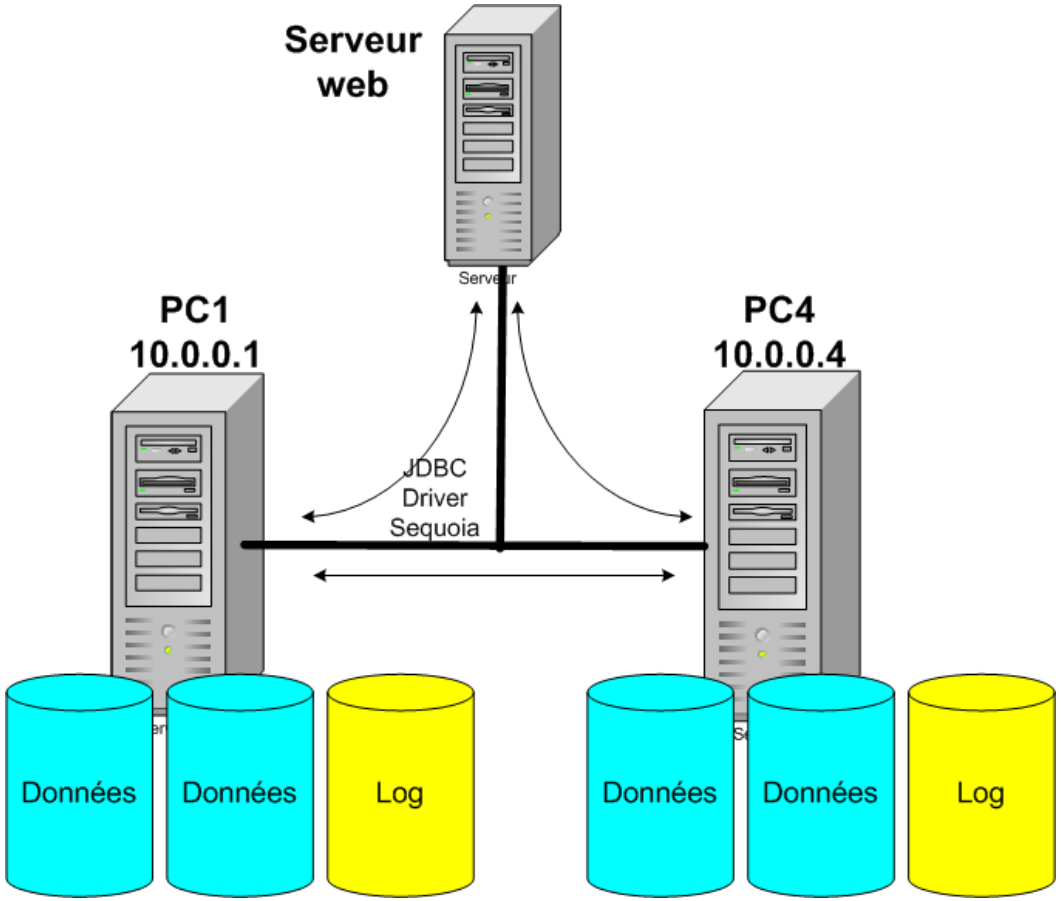
Limitations

- java.sql.Array et java.sql.Ref non supportés
- Connections XA (transactions distribuées) voir le projet XAPool pour le support de XA avec Sequoia

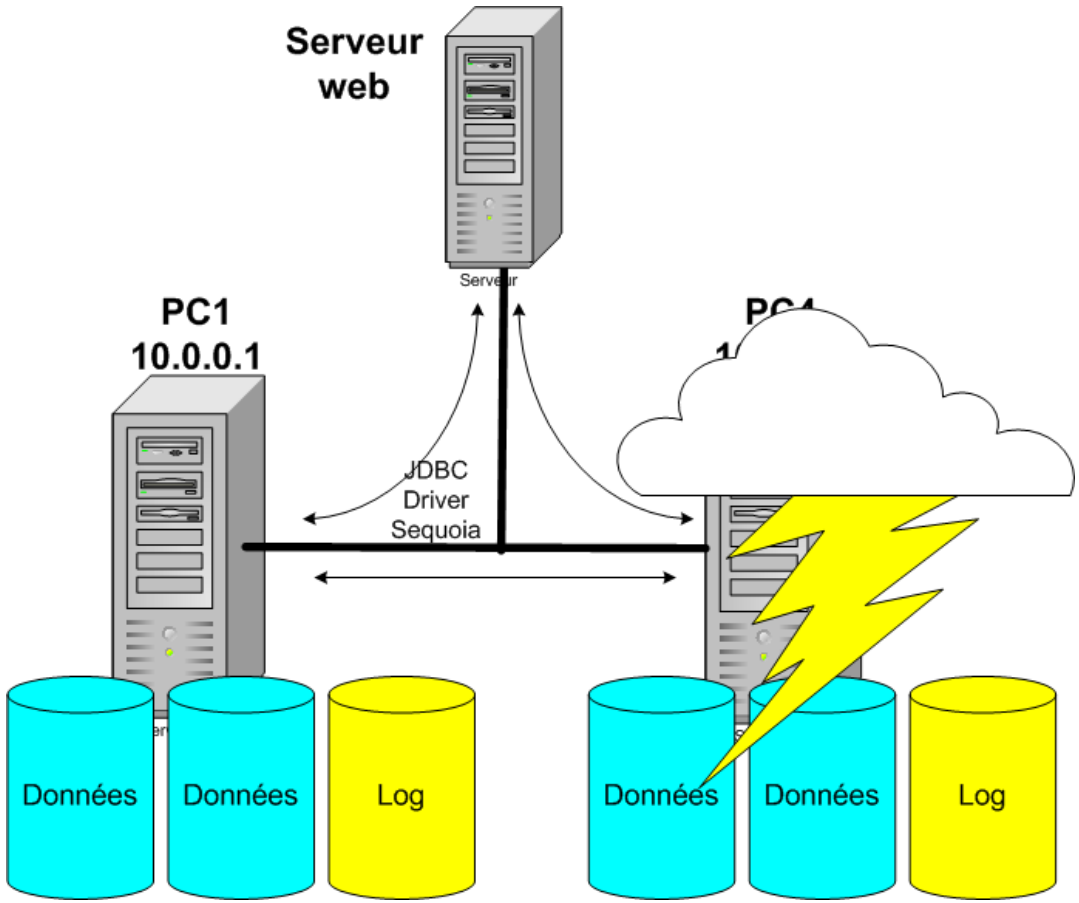
Démonstration



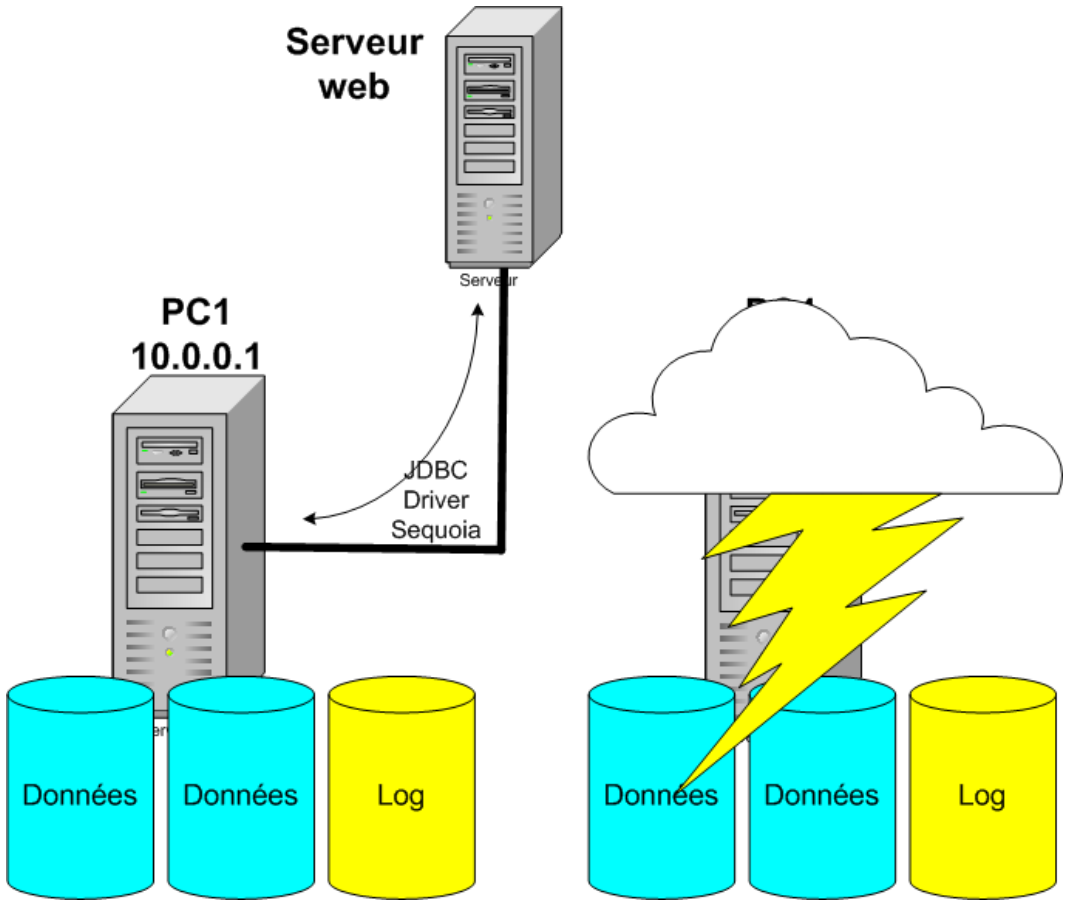
Démonstration



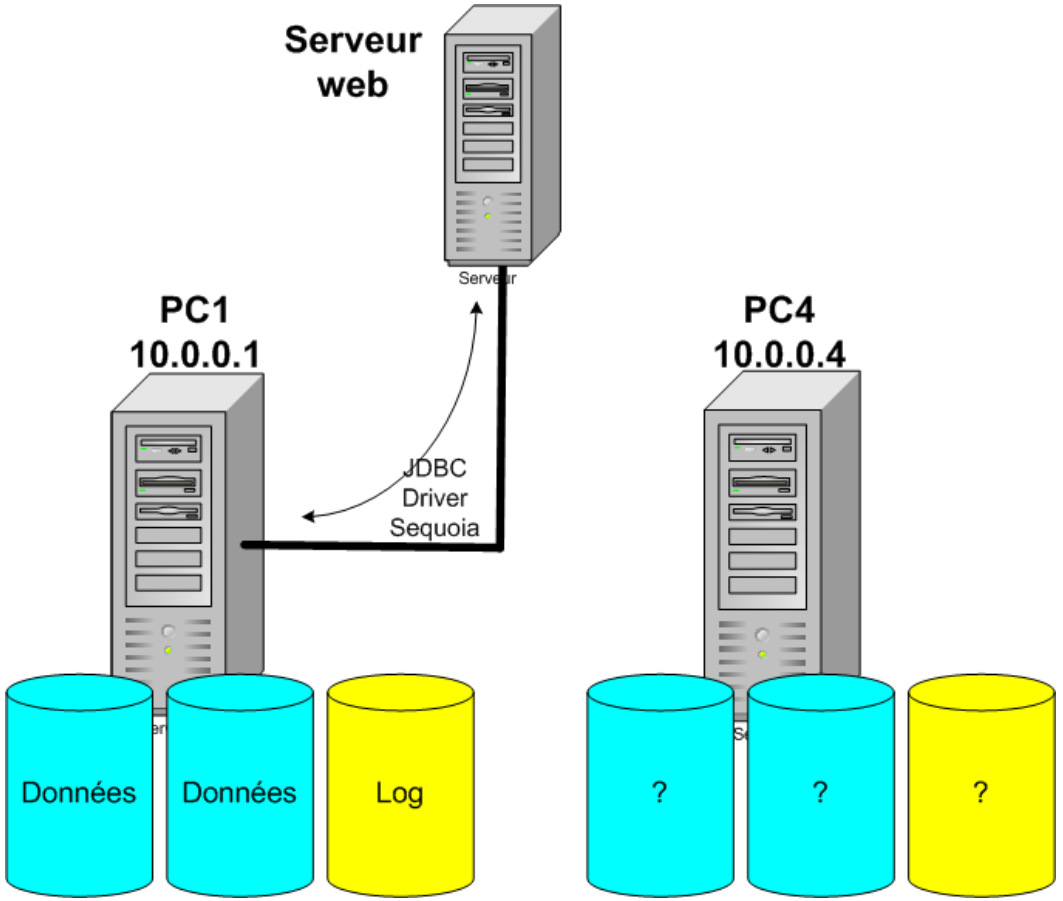
Démonstration



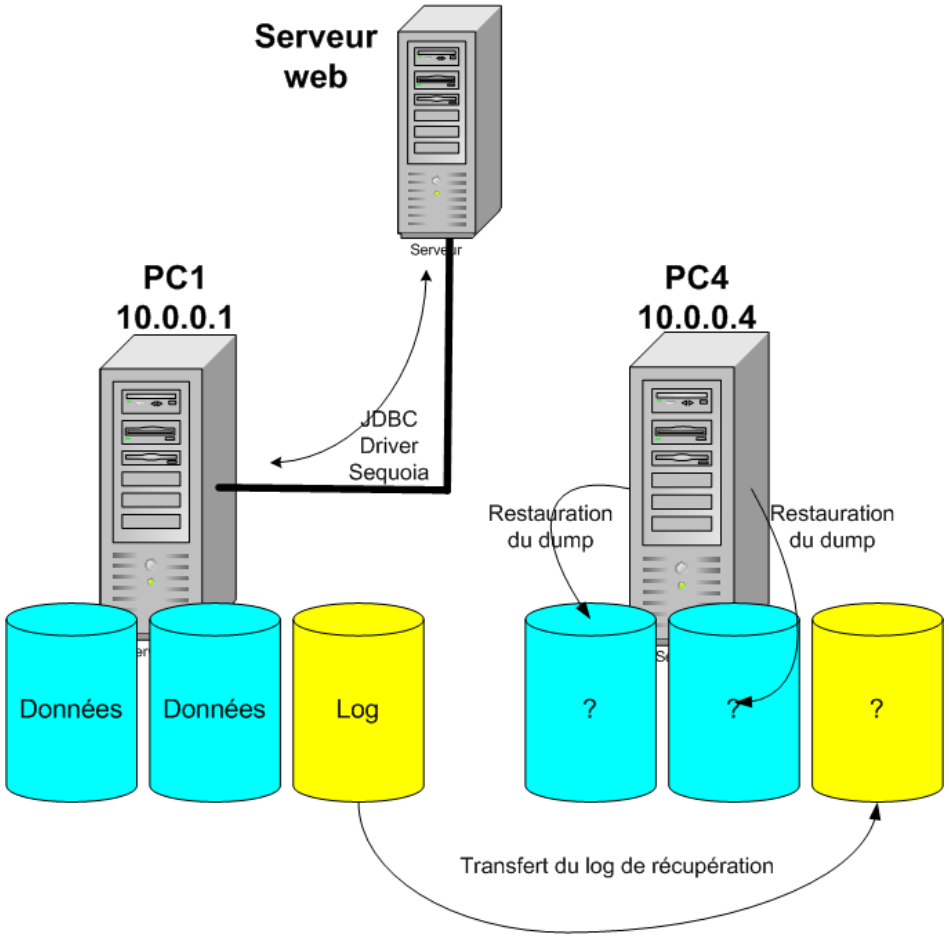
Démonstration



Démonstration



Démonstration



JShaft

Présentation

JShaft

▸ Lien

JShaft est...

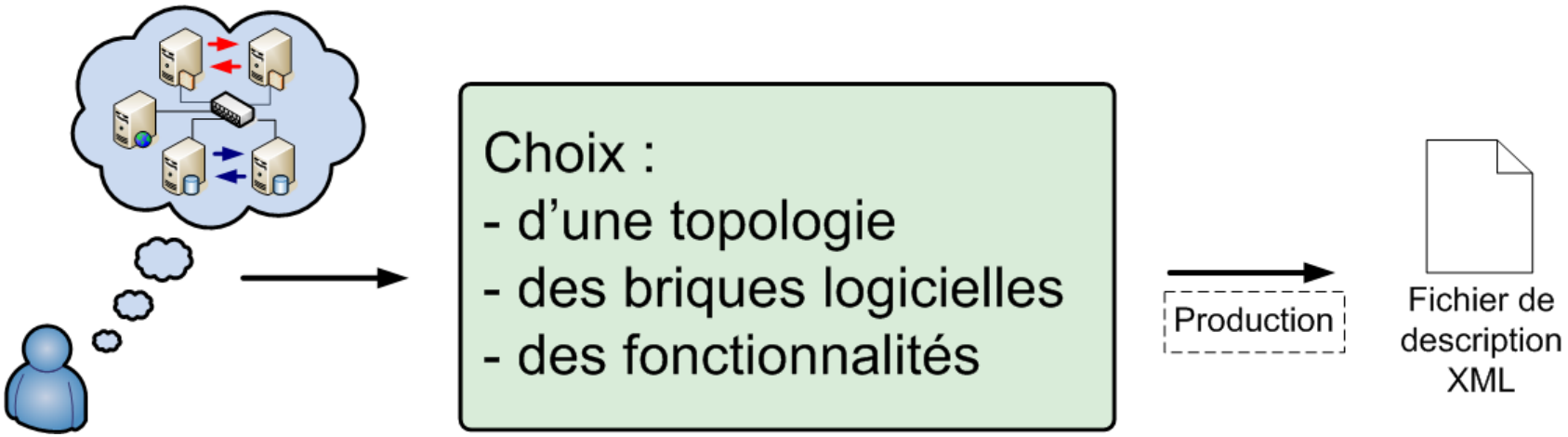
- un projet libre
- dont le but est de configurer et d'administrer simplement "une architecture clusterisée"

Présentation

Rappel des principaux obstacles à la création d'un cluster

- La complexité de mise en oeuvre
  ➜ Solution : Stocker les procédures de configuration et les automatiser
- Des systèmes de gestion de cluster non standardisé
  ➜ Solution : Un système d'administration unique

Production du fichier de configuration

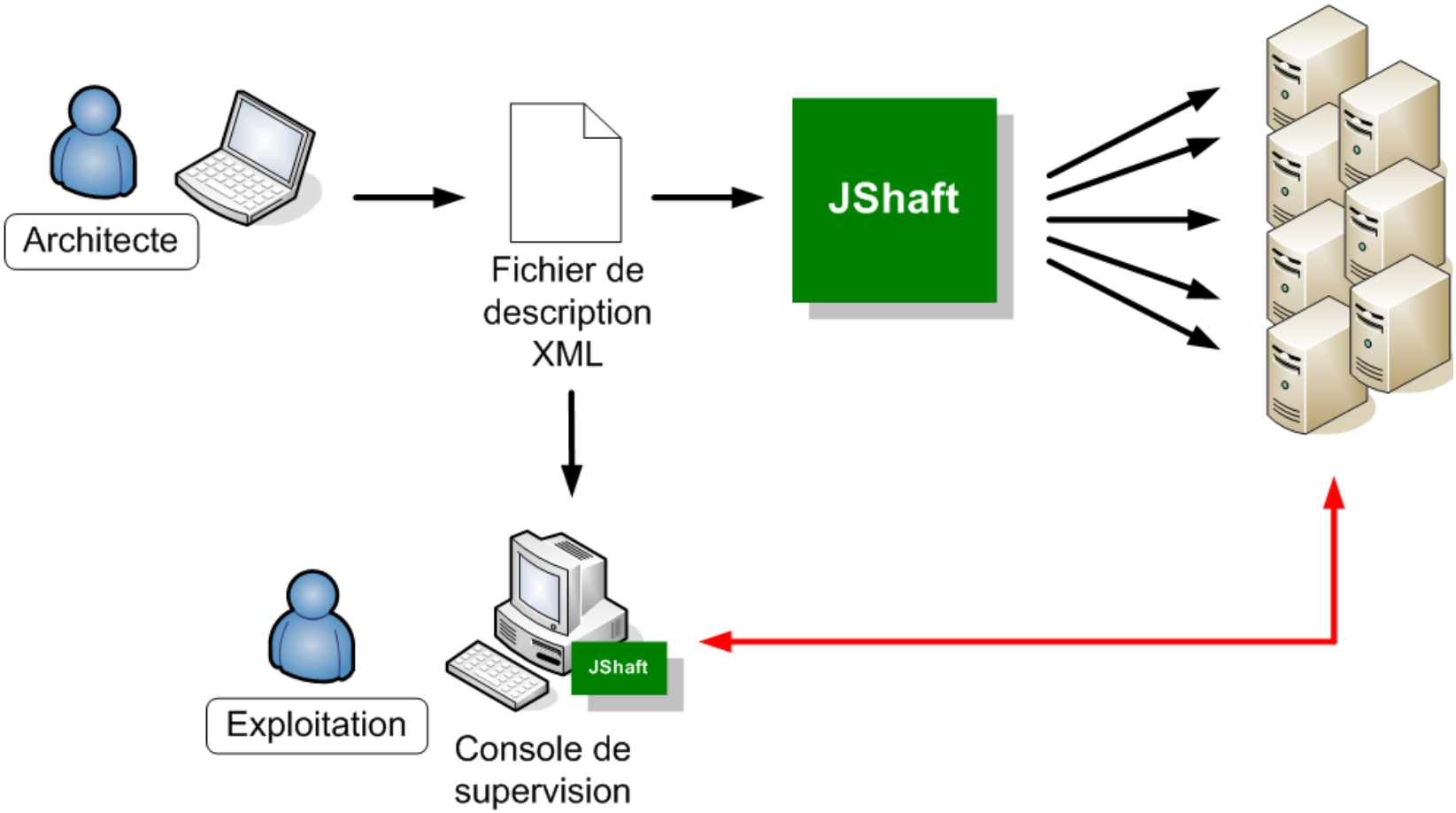


Production du fichier de configuration

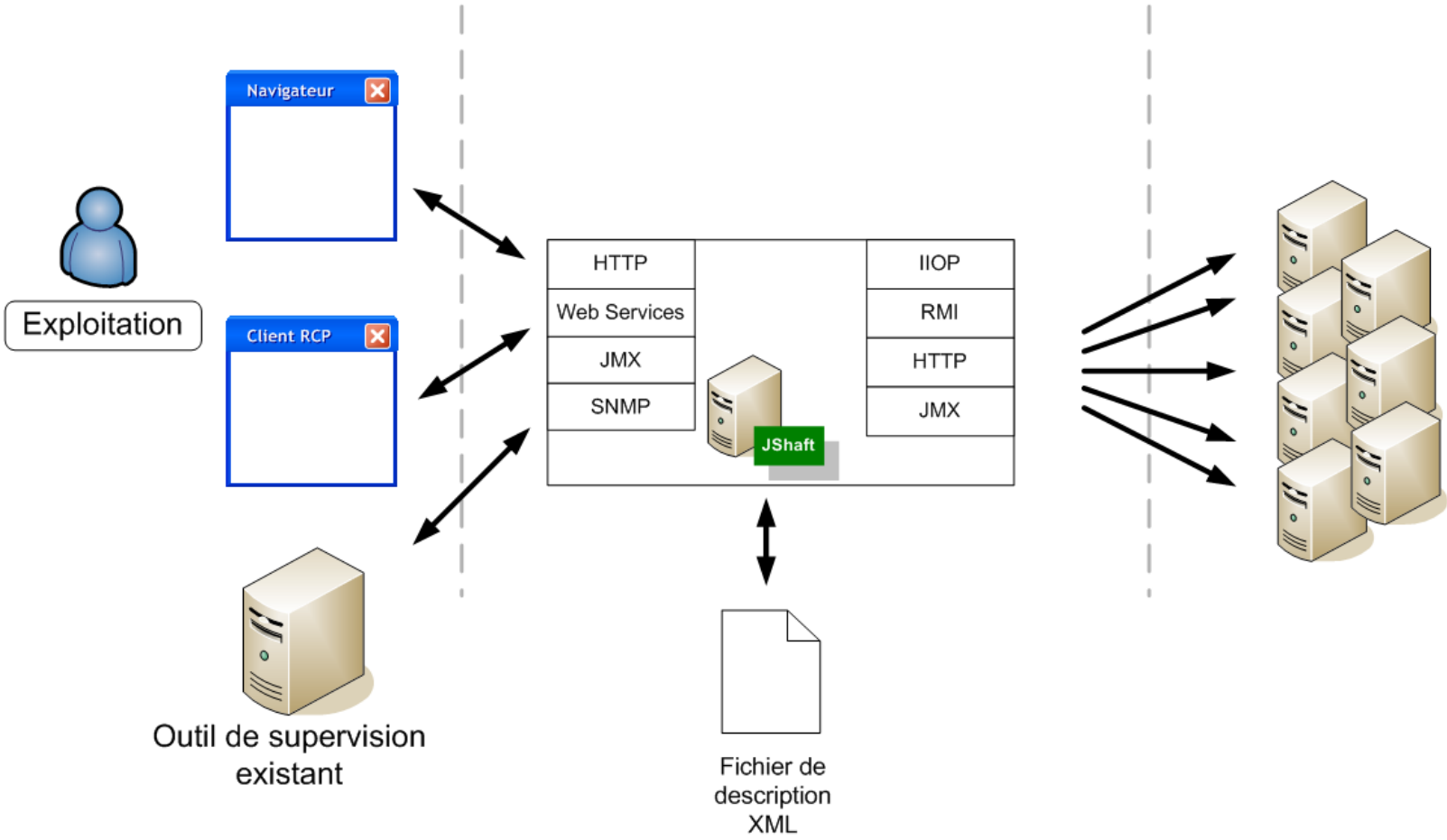
Deux moyens de génération

- En éditant directement le fichier XML (en respectant les contraites du schéma XSD)
- En passant par une interface graphique : Assitant de configuration (Démonstration)

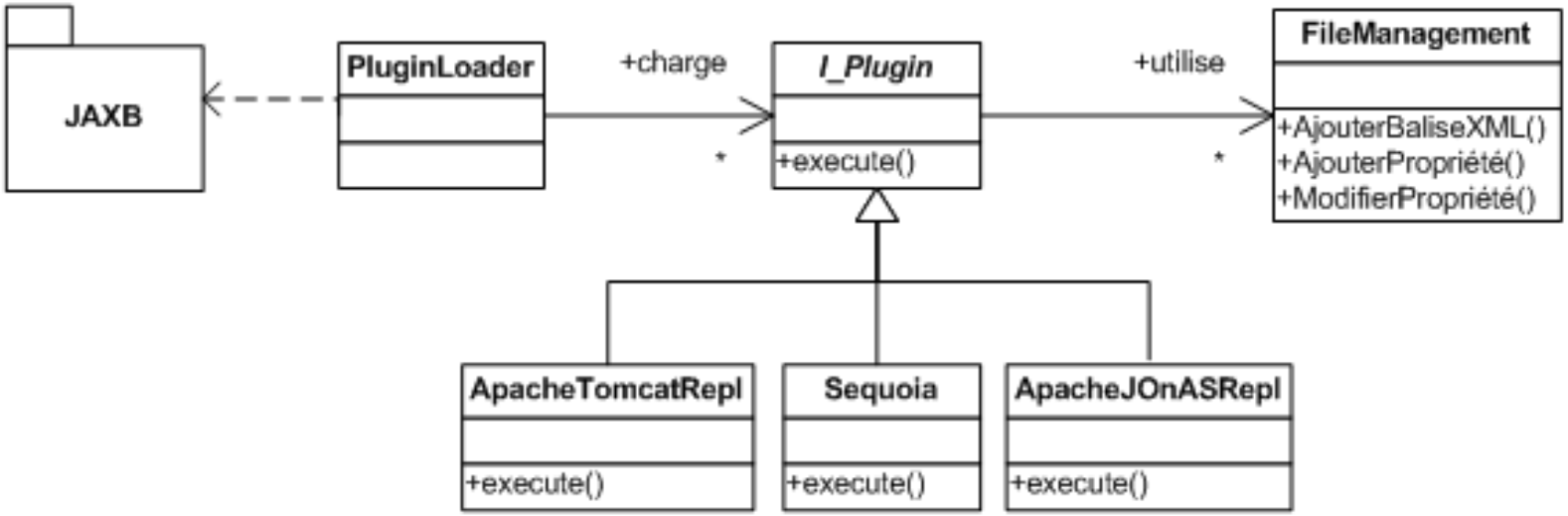
Architecture globale



Architecture serveur d'administration



Conception technique



Conception technique

Briques logicielles

- Parseurs XML (JAXP, JAXB)
  - Modification des fichiers XML existants
  - Gestion du fichier de configuration
- Log4j
- Ant
- JUnit

Version actuelle

Disponible

- Alpha Release ▸ Lien
- Une documentation en ligne ▸ Lien
- 2 plugins

Plugins

- Apache/Mod_JK + JOnAS ou Tomcat
  ➜ Création d'une configuration automatique

Version actuelle

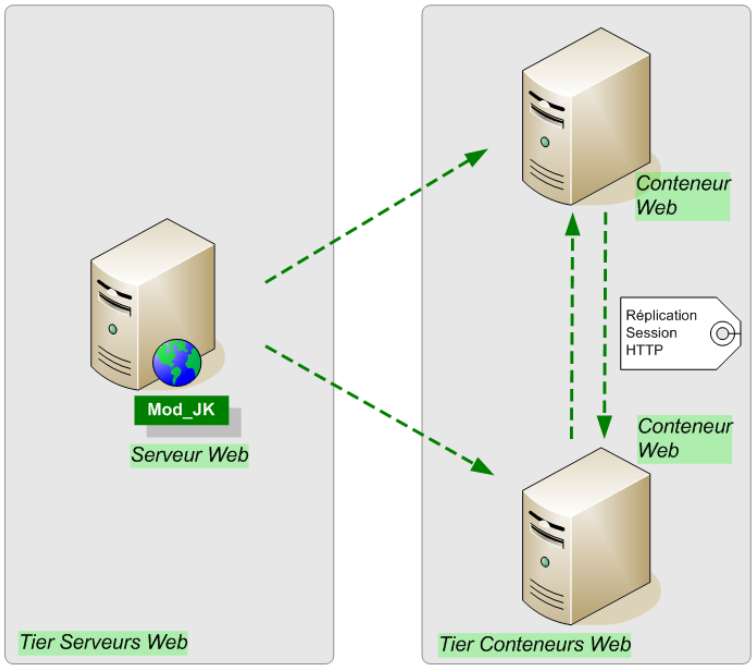
Disponible

- Alpha Release ▸ Lien
- Une documentation en ligne ▸ Lien
- 2 plugins

Plugins

- Apache/Mod_JK + JOnAS ou Tomcat
  ➜ Création d'une configuration automatique

Plugins



Disponible

- Apache/Mod_JK et JOnAS
- Apache/Mod_JK and Tomcat

➜ Création d'une configuration automatique, composée :

- d'un répartiteur de charge
- d'un dispositif de réplication de sessions

RoadMap

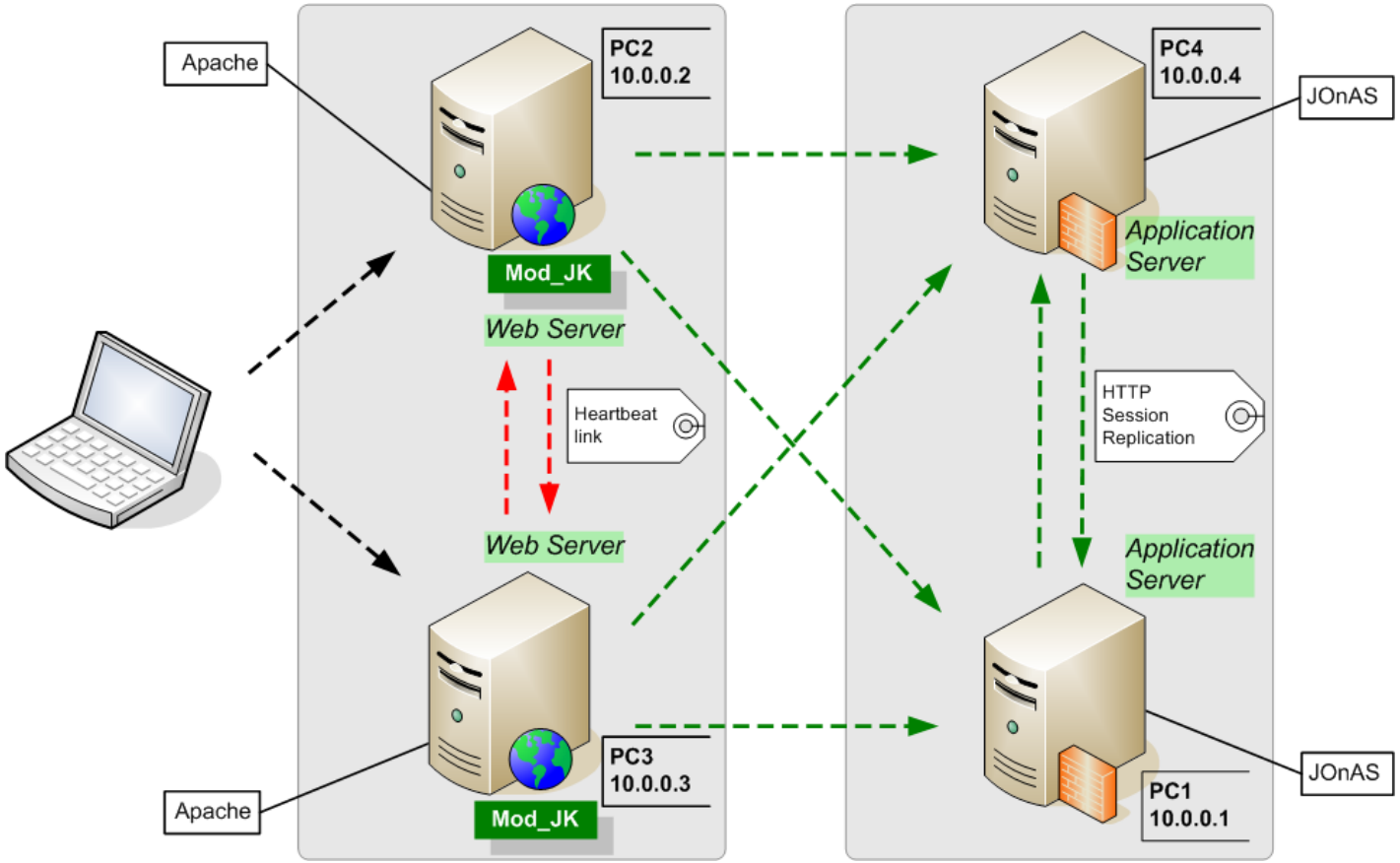
Prochaine Release

1. Création du plugin Sequoia
2. Conversion des sources en tâches Ant
3. Outil de supervision en version alpha

Comment contribuer à JShaft ?

- Production de plugins
- Proposer des modifications du schéma XSD
- Rédaction de documentation

Architecture de la démonstration



Remerciements

Merci :

- Aux organisateurs des RMLL
- Au consortium ObjectWeb
- Stéphane Traumat
- Benoit Pelletier