

### 3.3 Solution technique pour le lot 1

---

La solution décrite dans le paragraphe suivant est le reflet de la réflexion des équipes techniques de Thales à l'instant présent. Elle vise à éclairer le CSN sur ce que Thales pense être la meilleure solution au moment de la rédaction de la réponse. Elle est susceptible d'évoluer au moment de la conception définitive.

Certains chapitres décrivant les détails de mise en œuvre de la solution sont cependant structurants et sont constitutifs de la réponse Thales en ce sens que leur remise en cause invalide le chiffrage détaillé en §9.3.1.

Ce sont les chapitres :

- 3.3.3 - *Sécurisation des fichiers de références immobilières*
- 3.3.7.2 - Eléments de cartographie
- 3.3.11.1 - Principes

La solution technique proposée ci-dessous présente plusieurs atouts limitant au maximum les risques techniques: elle s'appuie sur une architecture et des briques logicielles open source, d'une grande maturité et largement éprouvées, garantissant la fiabilité, la robustesse, la pérennité, la réutilisabilité et l'évolutivité de la solution.

Cette partie décrit dans un premier temps les grands principes d'architecture technique et applicative envisagés. Elle fait ensuite un zoom sur les principales problématiques techniques du projet (administration de données, moteur de recherche, authentification, etc). Elle se termine par une analyse des risques techniques et une matrice de traçabilité des exigences techniques.

### 3.3.1 Architecture technique

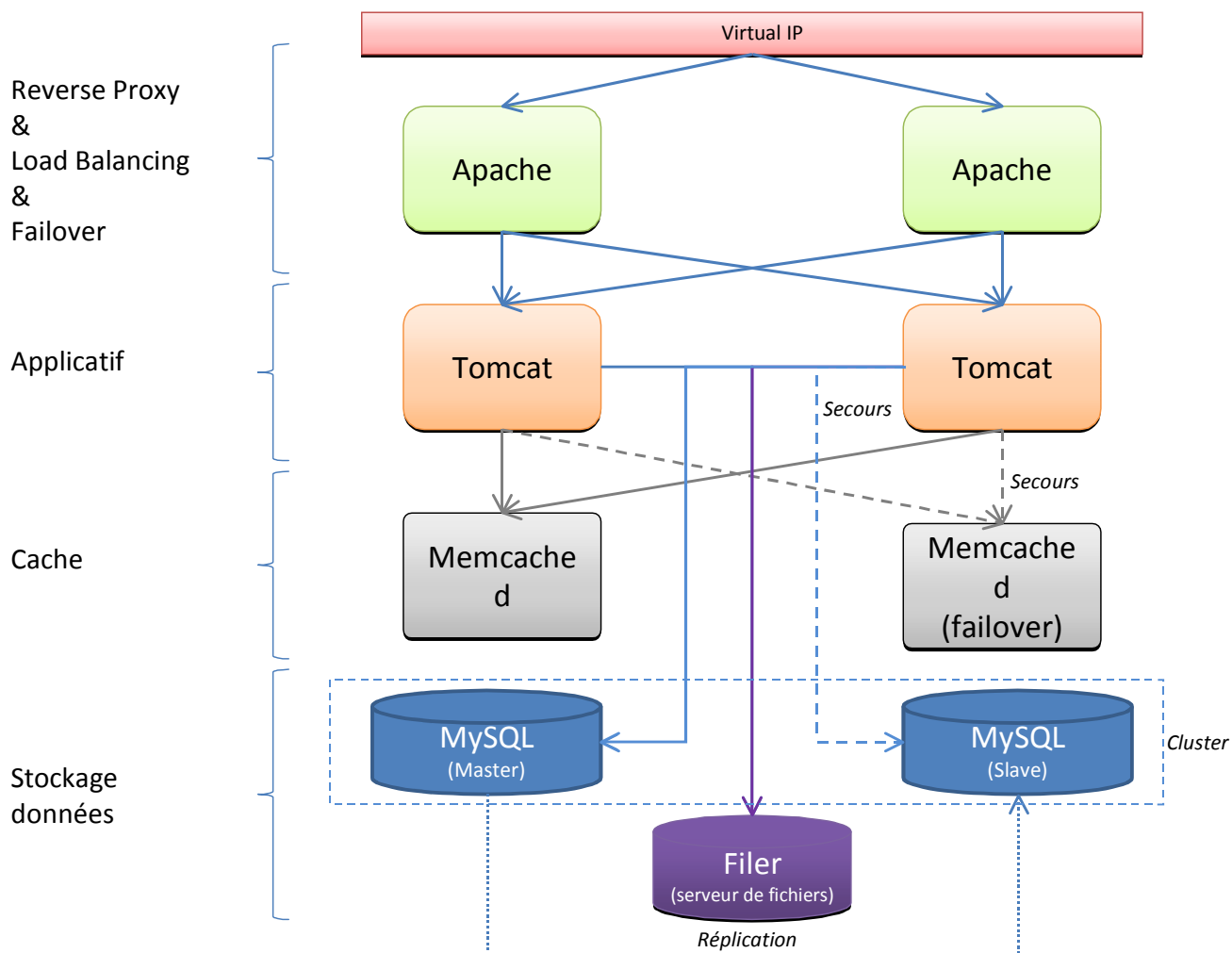
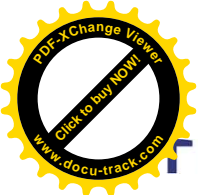


Figure 14 - Architecture technique de la solution



L'architecture technique de la solution proposée par Thales s'appuie sur :

- **un serveur Apache** en frontal qui traite et filtre les différentes requêtes Https qu'il reçoit,
  - ♦ Gestion de la montée en charge : Il permet de répartir les requêtes entre deux serveurs Tomcat (load balancing).
  - ♦ Gestion du failover : il permet de détecter l'inaccessibilité d'un des serveurs applicatifs et redirige automatiquement le trafic vers le serveur restant
  - ♦ Sécurité : il décrypte le flux ssl/tls et décharge ainsi le serveur Tomcat de ces opérations consommatrices de CPU. Il permet également de mettre en place des virtualhosts et par conséquent de masquer l'adresse et le port du serveur Tomcat.
  - ♦ Performance : Il peut être utilisé comme proxy pour améliorer les performances du site au niveau des éléments statiques (images, etc...).
- **un serveur d'application J2EE : Tomcat 7**,
  - ♦ La version 7 de Tomcat est la plus récente tout en étant éprouvée et validée sur de multiples projets.
  - ♦ Elle implémente les spécifications Servlet 3.0 et JSP 2.2
- **un SGBD : MySQL**.
  - ♦ MYSQL est une solution de base de données relationnelle robuste particulièrement adaptée à la volumétrie de données légère du projet
- **un cache mémoire**, utilisé entre autres pour stocker en clair les données de références immobilières en cours d'utilisation.
- **un espace de stockage de fichiers**, utilisé pour stocker les fichiers de données de références immobilières.

### 3.3.2 Architecture logicielle

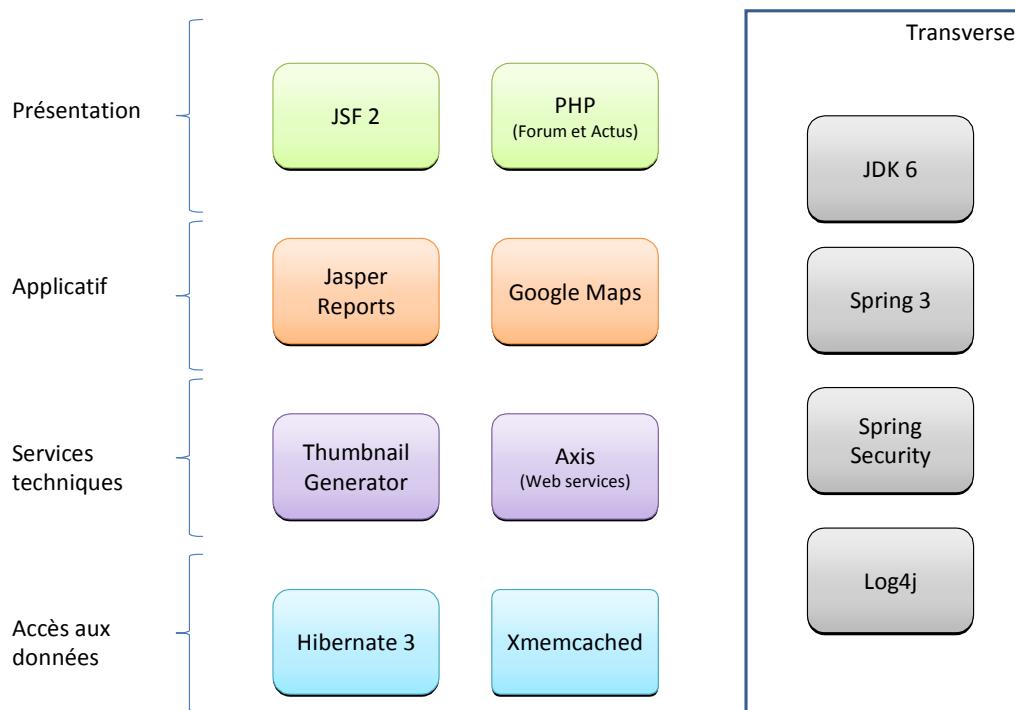


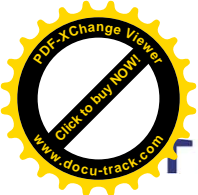
Figure 15 - Architecture logicielle de la solution

D'un point de vue applicatif, les briques logicielles mises en place sont :

- **JDK 6** : Version actuelle du kit de développement Java. Java est un langage de programmation en constante amélioration, portable, objet, supporté par de nombreuses entreprises (Sun, Apache, IBM, Oracle ...). De ce fait, il répond aux problématiques de l'existant: obsolescence et évolutivité.
- **Spring 3** comme Framework pour construire et définir l'infrastructure d'une application Java. Spring est un conteneur léger et est un standard du marché. La version 3 permet d'alléger la configuration XML, et donc d'accélérer le développement.
- **Spring Security** comme framework pour gérer l'accès aux ressources (gestion des autorisations et des rôles utilisateur) d'une application Java et de paramétrer différents mode d'authentification.
- **JSF 2.0 / Primefaces** comme framework de présentation (JSR 314) web 2.0.

JSF implémente le pattern MVC 2. C'est le framework de développement IHM standard de la plate-forme JEE 6:

- ♦ il est indépendant de l'implémentation des servlets du serveur d'application et facilite ainsi d'éventuelles migrations des serveurs,
- ♦ il utilise le langage XHTML et permet ainsi une validation stricte des pages et la possibilité de Templating.



PrimeFaces, une des librairies JSF 2 les plus utilisées :

- ♦ fournit un ensemble riche de plus de 100 composants JSF,
  - ♦ supporte l'habillage des applications avec plusieurs thèmes intégrés et un outil de conception de thème,
  - ♦ fournit un kit d'UI qui supporte la création des applications pour des appareils mobiles (iPhone, Android, Blackberry, etc), potentiellement utile si le CSN souhaite à terme développer une version mobile de son site (voir xxx)
  - ♦ il intègre la librairie Google Maps qui permet d'afficher des cartes et de positionner des étiquettes sur cette carte.
- **JasperReport** : API java de génération de rapports sous différents formats (PDF, HTML, XML, RTF, TXT, CSV, XLS). Cette solution présente l'avantage d'offrir une grande maturité et une forte fiabilité.
  - **Hibernate 3** pour gérer la persistance et l'accès aux données selon la norme (JSR 317) JPA. Hibernate permet de s'abstraire de la source de données.
  - **Log4J** pour gérer les traces applicatives.

L'ensemble des briques logicielles utilisées sont Open Source.