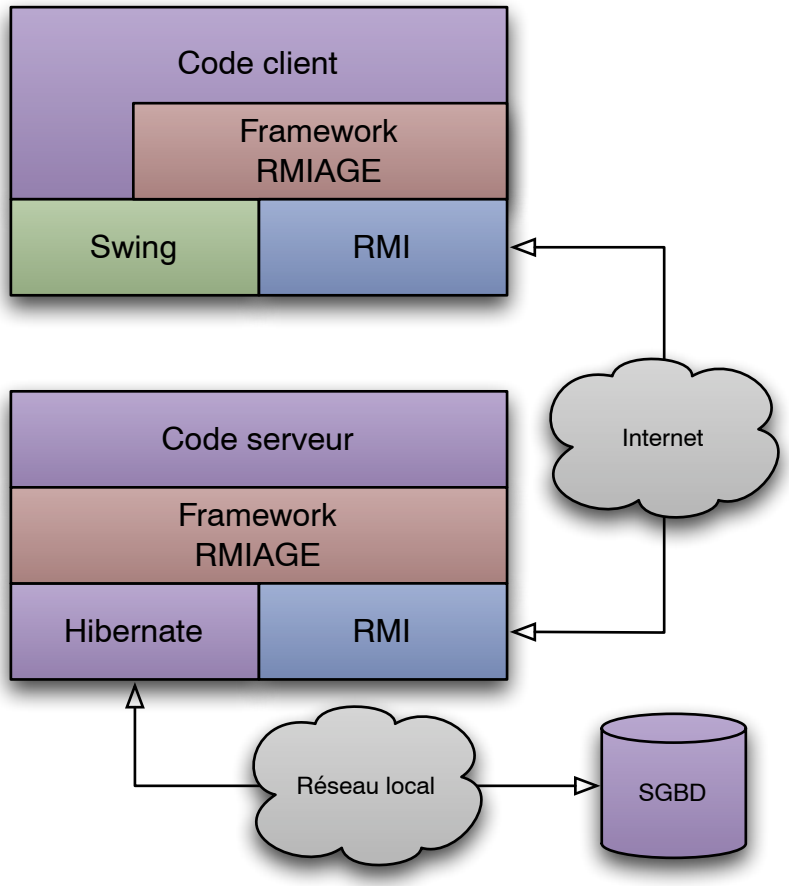
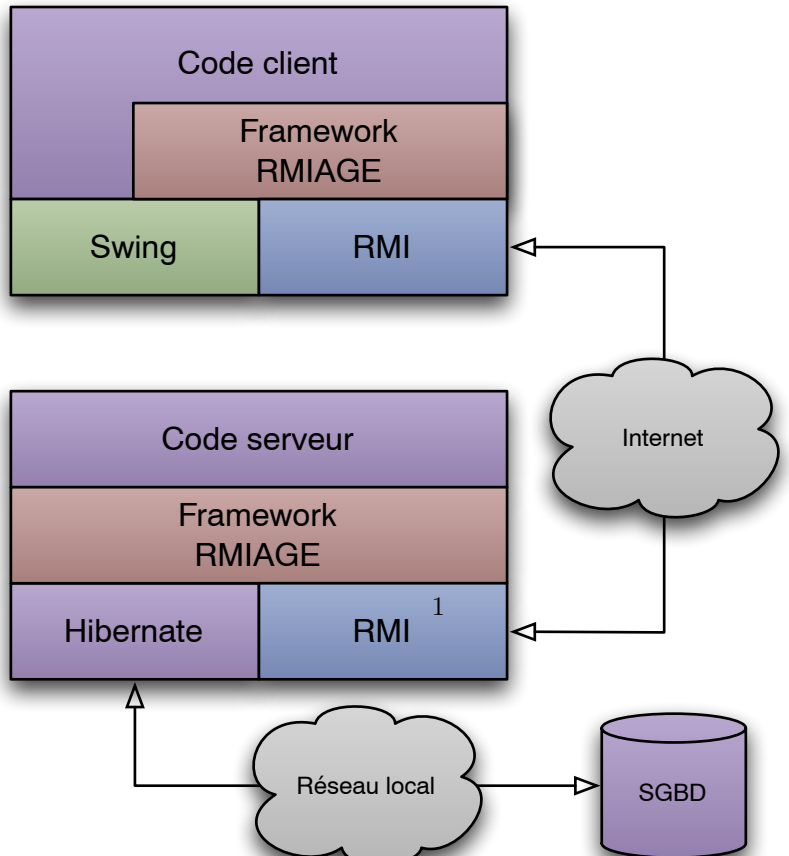


Figure 1: Architecture



Projet RMIAGE : Cahier des Charges

Badre Baba, Geoffroy Carrier, J-Ch Saad-Dupuy, Mickael Scheer

24 avril 2009

1 Présentation générale

1.1 Projet

1.1.1 Finalités

Nous intervenons auprès d'une entreprise de développement souhaitant mettre à la disposition de ses clients des architectures complètes de réseaux sociaux, incluant serveur applicatif et client lourd.

1.1.2 Espérance de retour sur investissement

L'entreprise souhaite prendre place sur ce marché avec pour principal avantage concurrentiel de fournir des produits adaptés aux besoins précis et variés de leurs clients diffuseurs de contenu avec des délais de développement minimaux. Ces derniers attendent des progiciels fournissant des zones d'échange bien spécifiques, et expriment une forte exigence d'ergonomie.

1.2 Contexte

Le développement des réseaux et des moyens de communication offrent désormais beaucoup de facilité pour communiquer et échanger du contenu de nature très diverse et ceci de manière aussi très variée. Quelque soit le contenu et la manière d'organiser ces espaces de communication où l'on partage, échange des informations, leurs objectifs sont très proches. On le voit très nettement dans l'avènement des « réseaux sociaux » qui les déclinent sous des formes très variées.

1.2.1 Situation du projet par rapport aux autres projets de l'entreprise

Notre équipe est compétente et performante en développement, notamment dans les technologies Java, et nous avons mené à bien plusieurs projets dans le cadre universitaire. Il s'agit ici de notre première réalisation commerciale.

1.2.2 Études déjà effectuées

L'équipe a développé au cours des années précédentes une connaissance approfondie du marché des réseaux sociaux à travers une veille technologique. Dans ce cadre, les produits facebook, MySpace, twitter, identi.ca, LinkedIn, viadeo, youtube, flickr, meetic ont été testés en profondeur.

1.2.3 Études menées sur des sujets voisins

Nous avons eu l'occasion de concevoir une architecture bas-niveau client-serveur pour un logiciel de gestion bibliothécaire, et de nous pencher sur le traitement et la diffusion d'informations XML à travers un serveur applicatif.

1.2.4 Nature des prestations demandées

L'entreprise souhaite externaliser le développement de la couche commune aux différents réseaux sociaux.

Afin de répondre au mieux aux attentes clients, l'entreprise souhaite faire développer un *framework* complet et efficace, qui une fois pris en main allégera fortement les délais de développement. Pour couvrir l'ensemble des demandes et s'intégrer dans les infrastructures matérielles et logicielles hétérogènes des clients, la réutilisabilité, l'interopérabilité et la généricité de ce dernier sont fondamentales.

1.2.5 Parties concernées par le déroulement du projet et ses résultats

Durant toute la durée du projet, nous serons en contact avec le maître d'œuvre, M. François Puitg.

Les utilisateurs finaux seront les clients exploitant des produits développés par notre client, diffuseurs de contenu, auprès de qui nous n'interviendrons pas.

1.3 Énoncé du besoin

Les clients, diffuseurs de contenu, souhaitent proposer à leurs utilisateurs des zones d'échange organisées sur le modèle des réseaux sociaux :

- Gestion des relations entre les utilisateurs directement par ces derniers ;
- Partage facile de contenus sans structuration préalable par les équipes déployant l'architecture logicielle.

1.4 Environnement du produit recherché

Le *framework* doit être réalisé en Java, et utiliser la technologie RMI (Remote Method Invocation) pour la communication entre des clients lourds et le serveur applicatif.

1.4.1 Listes exhaustives des éléments

Le produit se compose de trois livrables :

1. Le *framework* ;
2. un programme témoin d'application serveur ;
3. un programme témoin d'application cliente.

1.4.2 Caractéristiques

1. *Framework*

Le *framework* permettra une présentation hiérarchique des données côté client, et reposera sur la technologie de persistence JPA permettant de stocker les données sur un serveur SQL. Cette solution assure leur cohérence, la fiabilité du stockage et la rapidité des recherches.
Il permettra l'ajout, la suppression, la modification de celles-ci à travers une couche de sécurité conçue sur mesure pour répondre aux attentes métier.
Il fournira une application lourde très largement adaptable aux besoins et dont les composants éléments métier pourront être fournis à la volée par le serveur applicatif.
Les arborescences de contenu construites pour chaque utilisateur pourront partager des branches grâce à une gestion en graphe des données.
Des modules de gestion de contacts et de discussion seront fournis, proposant en standard des fonctionnalités fréquemment proposées sur les réseaux sociaux.

2. Application témoin serveur
   L'application témoin serveur sera un simple *package* configuré pour utiliser le serveur SQL embarqué dans la machine virtuelle Java de Sun et ne nécessitant aucune configuration, auquel sera adjoint un système de forums (fils de discussion publics), et un modèle de sécurité minimal.
3. Application témoin cliente
   Il s'agira de l'application déployable directement par l'entreprise, et capable *via* la diffusion des éléments graphiques métier par le serveur de s'adapter aux différents réseaux proposés.
   Elle sera néanmoins adaptable à volonté par l'entreprise au cas par cas (ajouts de fonctionnalités, d'une documentation ad hoc, thèmes visuels, retrait du choix du serveur, etc.).
   Les clients pourront utiliser la technologie Java Web Start pour distribuer l'application à tous leurs utilisateurs sans intervention des administrateurs.
   Un prototype de l'interface permettant d'en percevoir la conception générale est proposé en annexe **??**.

2 Expression fonctionnelle du besoin

2.1 Fonctions de service et contraintes

2.1.1 Fonctions de service principales

Framework accélérant le développement de serveurs applicatifs et clients lourds et proposant :

1. La gestion d'utilisateurs ;
2. Le stockage de contenus ;
3. La création, modification, suppression de ces contenus ;
4. Le partage de ces contenus entre utilisateurs ;
5. Un accès aisé à ces contenus ;
6. Une architecture pour intégrer une politique de permissions ;
7. Des outils de communication entre les utilisateurs.

2.1.2 Fonctions de service complémentaires

Deux programmes de démonstration exploitant les fonctionnalités du *framework* seront fournies :

- Un programme serveur proposant un système de forums ;
- un programme client exposant les fonctionnalités du serveur.

De plus, nous fournirons une batterie de tests unitaires garantissant la qualité de chaque fonctionnalité du *framework*.

2.1.3 Contraintes

1. Développement

   Le framework sera aussi indépendant de choix techniques que possible, notamment de tout éditeur de serveur SQL grâce à la technologie Hibernate.

2. Environnement

   Notre framework se composera d'une librairie, utilisée par les applications témoins livrées.

   L'utilisation de la technologie Java assure une complète indépendance du ou des systèmes d'exploitations hôtes.

   Les applications serveur et cliente pourrons être lancée depuis le même poste, en local, ou sur des machines distantes.

   Les machines hôtes et serveur devrons chacune disposer d'une Java Virtual Machine (JVM) afin de pouvoir executer ces applications.

2.2 Critères d'appréciation

Les principaux critères d'appréciation sont :

- Du coté *framework* :
  - Sa qualité ;
  - Son indépendance ;
  - Sa flexibilité ;
- du coté applications témoin :
  - Leur pertinence vis-à-vis des fonctionnalités principales du *framework* ;
  - La facilité de leur utilisation pour des tests.

3 Cadre de réponse

3.0.1 Outils d'installation, de maintenance

Le produit sera livré sous forme d'archives JAR contenant la librairie et les programmes témoins.

Les instructions concernant la configuration d'Hibernate seront également fournies.