



## ***LasiTalk! Messagerie Instantanée.***

### **1. Introduction générale :**

L'histoire de la communication est aussi ancienne que l'histoire de l'humanité. Au fil du temps l'homme n'a pas cessé d'évoluer, quand il a eu besoin de communiquer, il a mis au point beaucoup de moyens de communication comme le télégramme, le courrier en utilisant les lettres, etc. Mais ça n'a pas duré longtemps car il avait besoin des moyens plus efficaces et plus fiables, c'est pourquoi la communication a subi une grande évolution.

Les deux inventions majeures qui ont révolutionné la communication sont le téléphone et l'internet qui est aujourd'hui le réseau public le plus vaste au monde.

Il existe plusieurs applications qui tournent sur ce réseau que ça soit un réseau vaste comme internet ou bien simple comme un réseau local d'entreprise.

Parmi ces applications, nous nous sommes intéressés à la Messagerie Instantanée qui est un moyen de communication très utilisé dans ces dernières décennies.

### **2. La messagerie instantanée :**

Elle permet l'échange immédiat de messages textuels entre plusieurs ordinateurs connectés au même réseau informatique, et plus communément celui de l'Internet. Contrairement au courrier électronique, ce moyen de communication est caractérisé par l'affichage en temps-réel et la possibilité d'un dialogue interactif. La réalisation de logiciels de messagerie instantanée nécessite des langages de programmation qui offrent des possibilités pour la programmation réseau, la programmation événementielle et celle des interfaces graphiques.

Il existe plusieurs langages informatiques permettant la mise en œuvre de logiciels de t'chat en ligne. Et pour la réalisation de notre projet intitulé «*LasiTalk!*» nous utilisons le langage Java. Le but de notre projet est donc la mise en œuvre d'une architecture client /serveur en java qui permet un dialogue interactif entre les utilisateurs à travers une interface client qui offre des facilités d'utilisation et aussi un certain nombre d'options de configuration.



## ***LasiTalk ! Messagerie Instantanée.***

### **2.1. Notion de la messagerie instantanée :**

C'est un moyen de communiquer en privé avec d'autres personnes de son choix. Le client se connecte à un serveur qui contient les informations sur tous les utilisateurs inscrits (connectés ou non). Chaque personne possède un pseudonyme qui n'est pas unique dans la base de données du serveur. Un mot de passe est aussi indispensable pour se connecter au service. Deux personnes peuvent communiquer en direct si elles sont simultanément connectées au serveur. Sinon, elles ont la possibilité de consulter leurs messages instantanés même en cas où ils ne sont pas connectés.

### **2.2. Avantages de la messagerie instantanée :**

- Des échanges de messages, de documents et de fichier en temps réel.
- D'animation de sessions et de conférences.
- D'une connaissance des personnes en ligne et ainsi pouvoir poser une question qui nécessite une réponse rapide.
- Pas d'encombrement du disque dur : les messages s'effacent une fois la fenêtre du message fermée (si on ne veut pas enregistrer nos historiques).
- De Réaliser un travail collaboratif.
- De discuter à plusieurs personnes en temps réel.
- D'intégrer les fonctionnalités de voix et de vidéo grâce à une webcam.

### **2.3. Inconvénients de la messagerie instantanée:**

- **La sécurité :** Une messagerie instantanée est très facile à pirater et les pièces jointes sont susceptibles de contenir des virus.
- **Le contrôle :** Aucune possibilité de savoir à quelles fins est utilisé l'outil.
- **La compréhension des messages :** Limitation du nombre de mots, utilisation d'un langage abrégé.
- **Les abus :** l'usage personnel, et le risque de ne plus pouvoir s'en passer.



**LasiTalk ! Messagerie Instantanée.**

### **3. LES NOTIONS DE BASE POUR LA REALISATION DE LasiTalk ! :**

Voici Les outils et les notions de base pour la réalisation de LasiTalk ! :

#### **3.1. Le protocole XMPP :**

Ce protocole révolutionne à lui seul tout la messagerie instantanée. C'est un ensemble de protocoles standards ouverts de IETF « l'Internet Engineering Task Force » pour la messagerie instantanée, et plus généralement une architecture décentralisée d'échange de données, et il est basé sur une architecture Client/serveur. XMPP est également un système de collaboration en quasi-temps-réel et d'échange multimédia via le protocole Jingle, dont la Voix sur réseau IP « téléphonie sur Internet », la visioconférence et l'échange de fichiers sont des exemples d'applications.

##### **3.1.1 Ses avantages :**

- Protocole libre et open-source.
- Réseau décentralisé.
- Création d'un compte XMPP extrêmement rapide, en un clic.
- Confidentialité garantie grâce au cryptage.
- Multitude de serveurs XMPP.
- Nombre illimité de contacts XMPP.
- Transmission personnalisée de votre statut à vos contacts.
- Création et configuration simples des Rooms = Salons.
- Sauvegarde de l'historique de la Room sur le serveur XMPP (désactivation possible).

#### **3.2 Le protocole Jabber :**

Jabber est le nom d'un protocole de messagerie instantanée libre et sécurisé. On parle aussi de protocole XMPP.



*LasiTalk! Messagerie Instantanée.*

### 3.3 Raisons d'utilisation du protocole XMPP ou JABBER :

- JABBER est un protocole libre et ouvert. En d'autres termes, il n'y a aucune technologie cachée. Tout le système, c'est à dire tant la partie Client que la partie serveur, Jabber peut être inspectée et corrigée par tout programmeur et donc par nous-même.
- JABBER est un protocole décentralisé. Il n'y a pas un point central pouvant servir à l'espionnage des conversations de messagerie instantanée. Nous pouvons aussi installer et gérer notre propre serveur Jabber, ou utiliser un serveur Jabber publique.
- Le protocole Jabber est sécurisé. Il respecte notre vie privée, les développeurs travaillent dur pour que ces technologies soient le plus sécurisés possible.
- JABBER est réellement gratuit nous n'avons pas à payer pour utiliser le protocole. Des centaines de serveurs Jabber permettent la création de compte gratuit.

### 3.4. Smack API

Smack est une bibliothèque Open Source XMPP client (Jabber) pour la messagerie instantanée et de présence. C'est une bibliothèque Java pure, il peut être intégré dans nos applications pour la création d'un client complet XMPP à de simples intégrations XMPP comme l'envoi de messages de notification et de présence.

### 3.5. OpenFire

Openfire est un logiciel open-source permettant de mettre en place un serveur **de communication** de type **messagerie instantanée** « Jabber » libre et écrit en Java, c'est Totalemment administrable depuis une interface web, et il est simple d'accès. D'autre part, s'il semble possible de compiler Openfire avec l'implémentation libre (openJdk) de Java.



**LasiTalk !** *Messagerie Instantanée.*

#### 4. CONCEPTION ET DEVELOPPEMENT SUIVI DE **LasiTalk !**

- Pour la réalisation de notre application **lasitalk !**, notre démarche a été de diviser le travail en 6 parties :

- ✚ **Faire une étude préliminaire** : On a défini un cahier des charges ouvert, où toutes les propositions sont acceptées.
- ✚ **Classifier le cahier des charges** : On a classifié le cahier des charges en deux parties « besoins nécessaires », c'est-à-dire les besoins basiques de l'application et « besoins optionnels », si le projet a été terminé bien avant son délai, on peut rajouter ces besoins optionnels.
- ✚ **Faire la conception de **lasitalk !**** : On a commencé par le diagramme de cas d'utilisation. On a défini les acteurs principaux, les fonctionnalités de **lasitalk !** On a fait des fiches descriptives pour voir le fonctionnement de chaque cas d'utilisation, des diagrammes de séquences pour les cas d'utilisation les plus pertinents pour voir comment se font les transactions entre l'utilisateur et **lasitalk !** et enfin un diagramme de classes qui nous montre les différentes classes de **lasitalk !**, les différents services de **lasitalk !** ainsi que les classes du package Smack Api utilisé dans **lasitalk !**
- ✚ **Implémenter **lasitalk !**** : On a commencé par créer les classes les plus pertinentes (classe compte, contacts, messages), dans ces classes on a défini les besoins nécessaires comme des méthodes pour ses classes, tout ça grâce au package Smack Api. On a utilisé les classes du package Smack Api pour implémenter nos services, après on a commencé à faire des tests sur chaque classe avec le serveur OpenFire qui est un serveur de communication de type messagerie instantanée « Jabber » libre et écrit en Java. Ce dernier nous a permis de le placer comme serveur de communication pour **lasitalk !**, après la validation de cette étape, on est passé aux interfaces.



## **LasiTalk ! Messagerie Instantanée.**

- ✚ **Conception de l'interface LasiTalk ! :** On a fait une conception de l'interface de LasiTalk ! grâce à un environnement intégré de java qu'on va découvrir dans le chapitre 3, qui concerne l'implémentation.
- ✚ **Intégration des services LasiTalk ! dans l'interface :** Après avoir conçu l'interface LasiTalk !, on a intégré les services LasiTalk ! service par service.

## **5. Les besoins de l'application**

### **Les besoins nécessaires**

#### La classe Compte

- **Inscription :** Elle va permettre à un client d'être enregistré dans le serveur de messagerie donc d'être inscrit.
- **Authentification :** Elle va permettre à un client déjà inscrit de se connecter au service.
- **Changer mot de passe :** Le client peut modifier son mot de passe.
- **Statut de connexion :** Le client peut changer son statut (hors ligne/en ligne).

#### La classe Message

- **Envoi et réception :** Le client peut envoyer et recevoir des messages avec un autre client déjà inscrit et qui est parmi sa liste d'amis.
- **Envoi et réception (hors ligne):** Le client peut également envoyer des messages quand il est hors ligne.

#### La classe Contacts

- **Lister tous les contacts :** Le client peut voir la liste de tous ces contacts en ligne et hors ligne.
- **Lister les contacts connectés :** Le client peut voir la liste de ces contacts connectés.
- **Ajout/suppression contact :** Le client peut ajouter et supprimer des contacts.



***LasiTalk ! Messagerie Instantanée.***

## **Les besoins optionnels**

### La classe Compte

- Inscription sur invitation.

### La classe Message

- Signaler qu'un contact est en train d'écrire.
- Boîte vocale.
- Partage de fichiers.
- Conversation audiovisuel.

### Historique

- Suppression de messages depuis une discussion.
- Exporter une discussion vers un fichier.

### La classe Contacts

- Recherche dans la liste des contacts (Filtrage).
- Accepter/refuser d'être ajouté aux contacts d'un autre utilisateur.

### Conférence

- Administrateur du service.

### Personnalisation de l'interface

- Utiliser un thème pour l'interface.
- Changer la langue de l'interface.
- Changer la taille et la police des caractères.

### Autre

- Rubrique aide, Mot de passe oublié.



**LasiTalk! Messagerie Instantanée.**

## **1. Présentation globale du cahier de charges :**

Notre projet consiste à réaliser une application de messagerie instantanée connectant plusieurs clients localement.

Un Client au niveau de l'application cliente peut :

- S'inscrire.
- Se Connecter à l'aide d'un Pseudo et un Mot de Passe.
- Discuter en privé (Echanger des Messages).
- Discuter avec plusieurs personnes au même temps (conférence).
- Consulter la Liste des Contacts Connectés.
- Consulter la Liste de tous les Contacts.
- Ajouter un Contact dans sa Liste des contact.
- Supprimer un Contact.
- Changer le Mot de passe de son compte.
- Changer de statut de « hors ligne » à « en Ligne » et vice-versa.
- Enregistrer l'historique des conversations.
- Se Déconnecter.

Un administrateur au niveau de l'application serveur peut :

- Voir la liste de tous les Clients existant dans cette application.
- Supprimer un Client définitivement de cette application.
- Modifier les propriétés d'un Client.
- Définir un autre Client comme autre Administrateur de cette application.

## **2. Conception de l'application :**

### **2.1. Diagramme de cas d'utilisation :**

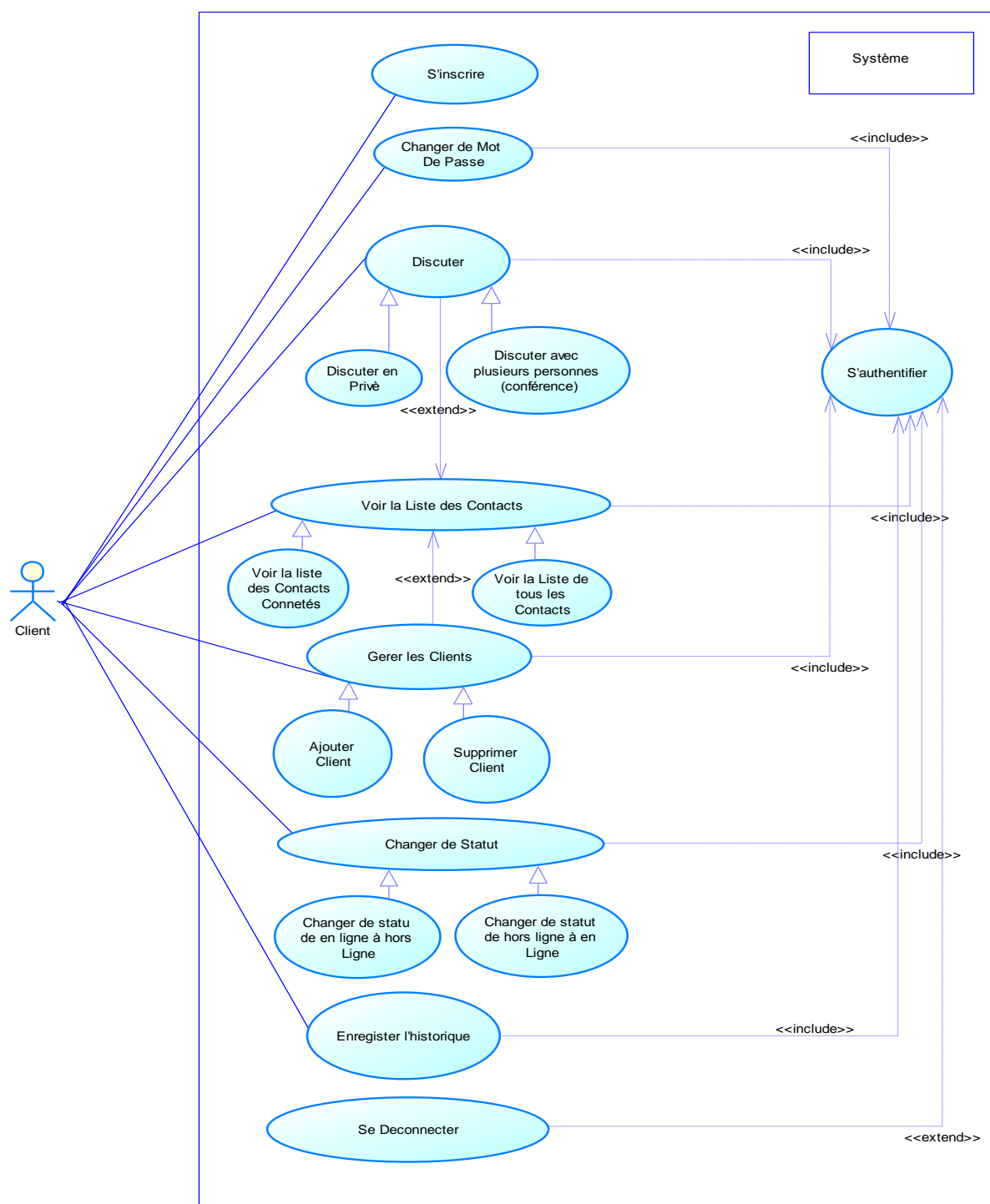
Ce diagramme décrit les relations entre les cas d'utilisations et les acteurs d'un système. Sachant que le processus UP est centré sur les cas d'utilisation et qu'il est





## **LasiTalk! Messagerie Instantanée.**

itératif, il nous a permis d'aboutir après plusieurs étapes de raffinement au diagramme de cas d'utilisation suivant :



**Figure 2.1.** Diagramme cas d'utilisation de l'application **LasiTalk!**



**LasiTalk! Messagerie Instantanée.**

### **2.1.2. Fiches descriptives des cas d'utilisation (Description textuelle)**

Pour les cas d'utilisation les plus pertinents du diagramme précédent, on a établi une fiche descriptive pour chaque cas qui comporte :

- Le nom du cas d'utilisation.
- Le résumé du cas concerné.
- Les acteurs principaux et secondaires interagissant avec le cas.
- Description des scénarios :
  1. Les pré-conditions (pour définir l'état antérieur du système avant le lancement du cas).
  2. Scénario nominal, éventuellement des scénarios alternatifs ou d'erreur.
  3. Les post-conditions (pour définir l'état postérieur après le lancement du cas).

#### **Fiche descriptive (1)**

**Titre :** S'authentifier.

**Objectif :** Ce cas d'utilisation permet à un client de se connecter au service offert par le serveur.

**Acteur Principal :** Client.

**Acteur Secondaire :** /

**Descriptions des scénarios :**

**Pré Conditions :**

- Il faut que le serveur soit bien lancé.
- Il faut que le client soit déjà inscrit.

**Scénario Nominal :**

1. Le client s'identifie à l'aide d'un pseudo et un mot de passe.
2. Le système vérifie si le serveur est bien lancé.
3. Le système vérifie la validité du pseudo et du mot de passe (si le pseudo et le mot de passe appartiennent à la base de données du serveur).



## ***LasiTalk! Messagerie Instantanée.***

4. Le système autorise la connexion.
5. Le client se connecte.

### **Scénario Alternatif :**

- Le serveur n'est pas lancé (hors-ligne)
  - L'application cliente indique au client que le serveur est injoignable.
  - Ce cas d'utilisation se termine par échec.
- Le pseudo ou le mot de passe sont incorrects
  - Le système indique au client que le pseudo ou le mot de passe est invalide.
  - Le système donne au client la main pour refaire son authentification sinon il peut s'inscrire.
- Echec à cause d'une panne dans le système.

### **Enchaînement d'erreurs : /**

#### **Post condition :**

- Le Client est ajouté à la liste des clients connectés dans le serveur.
- La date et l'heure de la connexion sont enregistrées au niveau du serveur.

### **Fiche descriptive (2)**

**Titre :** S'inscrire.

**Objectif :** Ce cas d'utilisation permet au client de s'inscrire pour utiliser et profiter des services offerts par cette application.

**Acteur Principal :** Client.

**Acteur Secondaire :** /

**Description des Scénarios :**

**Pré Condition :** Le serveur doit être bien lancé.

#### **Scenario Nominal :**

1. Le client ouvre la fenêtre d'accueil de l'application et demande de s'inscrire.
2. Le système envoie le formulaire d'inscription au client.
3. Le client remplit le formulaire d'inscription.
4. Le système vérifie la disponibilité des données remplies par le client.
5. Le système affiche au client un message de validation de son inscription.



## ***LasiTalk! Messagerie Instantanée.***

6. Le client se connecte à l'application.

### **Scénario Alternatif :**

- Pseudo est déjà pris
  - Le système affiche au client que le pseudo est pris et lui demande de le changer.
- Echec à cause d'une panne dans le système

### **Enchaînement d'erreurs : /**

**Post Condition :** L'enregistrement du client au niveau de la base de données du serveur.

### **Fiche descriptive (3)**

**Titre :** Discussions

**Objectif :** Ce cas d'utilisation permet à un client de discuter en privé avec un ou plusieurs Clients.

**Acteur Principal :** Client.

**Acteur Secondaire :** Les contacts de la liste des contacts de ce client.

### **Descriptions des scénarios :**

**Pré Condition :** Le client doit être déjà connecté (s'authentifier).

### **Scénario Nominal :**

1. Le client sélectionne dans sa liste des contacts le contact avec qui il veut discuter.
2. Le système démarre une discussion privée avec le contact sélectionné.
3. Le client échange des messages avec ce dernier.

### **Scénario Alternatif:**

- Echec à cause d'une panne dans le système.

### **Fiche descriptive (4)**

**Titre :** Ajouter un client.

**Objectif :** Ce cas d'utilisation permet à un client d'ajouter d'autres contacts à son compte pour pouvoir discuter avec eux.

**Acteur Principal :** client.



## **LasiTalk! Messagerie Instantanée.**

**Acteur Secondaire :** /

**Descriptions des scénarios :**

**Pré Condition :** Le client doit être déjà connecté (s'authentifier).

**Scénario Nominal :**

1. Le client clique sur le bouton ajouter nouveau contact.
2. Le client entre le pseudo du contact qu'il veut ajouter.
3. Le système affiche le pseudo du client dans sa liste de contacts.

**Scénario Alternatif:**

- Echec à cause d'une panne dans le système.

**Enchaînement d'erreurs :**

- Le Client existe déjà.
  - Le système ajoute le contact normalement, mais ne l'ajoute pas une deuxième fois au serveur.

**Poste Condition :** Le pseudo du nouveau contact s'enregistre dans la liste des contacts.

### **Fiche descriptive (5) :**

**Titre :** Supprimer un contact.

**Objectif :** Ce cas d'utilisation permet à un client de supprimer des contacts qui sont dans sa liste des contacts.

**Acteur Principal :** Client.

**Acteur Secondaire :** Le Contact à supprimer.

**Descriptions des scénarios :**

**Pré Condition :**

- Le Client doit être déjà connecté (s'authentifier).
- Le contact doit figurer dans la liste des contacts du Client.

**Scénario Nominal :**

1. Le Client sélectionne le contact à supprimer qui figure dans sa liste, clique sur le bouton supprimer contact.



## ***LasiTalk! Messagerie Instantanée.***

2. Le système réaffiche la liste des contacts sans le contact supprimé (faire une mise à jour de la liste des contacts).

**Scénario Alternatif:** Echec à cause d'une panne dans le système.

**Enchaînement d'erreurs :** /

**Poste Condition :** Le pseudo du contact supprimé ne s'affiche plus dans la liste des contacts.

### **Fiche descriptive (6)**

**Titre :** Se Déconnecter.

**Objectif :** Ce cas d'utilisation permet à un client de se déconnecter et de mettre fin à l'application.

**Acteur Principal :** Client.

**Acteur Secondaire :** /

**Descriptions des scénarios :**

- **Pré Condition :** Le Client doit être déjà connecté (s'authentifier).

**Scénario Nominal :**

1. Le Client demande au système de se déconnecter.
2. Le système affiche une boîte de dialogue pour confirmer la déconnexion.
3. Le Client valide la déconnexion.
4. Le système déconnecte la machine du serveur.
5. Le système affiche chez le client et le serveur un message indiquant que cette machine s'est déconnectée.
6. Le système met à jour la liste des clients disponibles sur le réseau.

**Scénario Alternatif:/**

**Enchaînement d'erreurs :** /

**Poste Condition :** Fin de connexion.

### **Fiche descriptive (7)**

**Titre :** Enregistrer l'historique des conversations.



## ***LasiTalk ! Messagerie Instantanée.***

**Objectif :** Ce cas d'utilisation permet à un client de sauvegarder l'historique de ses conversations.

**Acteur Principal :** client.

**Acteur Secondaire :** /

**Descriptions des scénarios :**

**Pré Condition :**

- Le Client doit être déjà connecté (s'authentifier).
- Le client a déjà échangé des messages avec ses contacts.

**Scénario Nominal :**

1. Le client demande la sauvegarde de l'historique des messages de sa discussion.
2. Le système enregistre l'historique des messages échangés.

**Scénario Alternatif :**

**Enchaînement d'erreurs :** /

## **2.2. Diagrammes de séquence :**

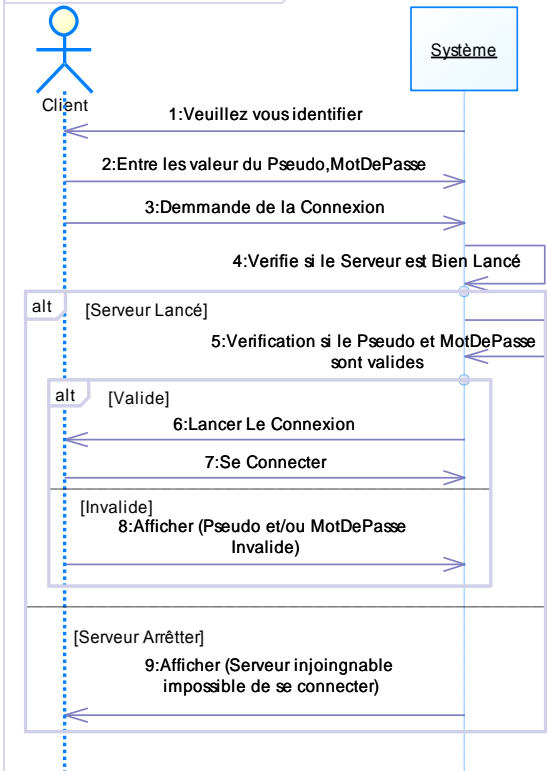
C'est une représentation séquentielle du déroulement des traitements et des interactions entre les éléments du système et/ou des acteurs. Ci-après nous allons présenter les diagrammes de séquence pour les cas d'utilisation les plus pertinents :

1. **s'authentifier.**
2. **S'inscrire.**
3. **Discussion.**
4. **Gérer les Contacts.**

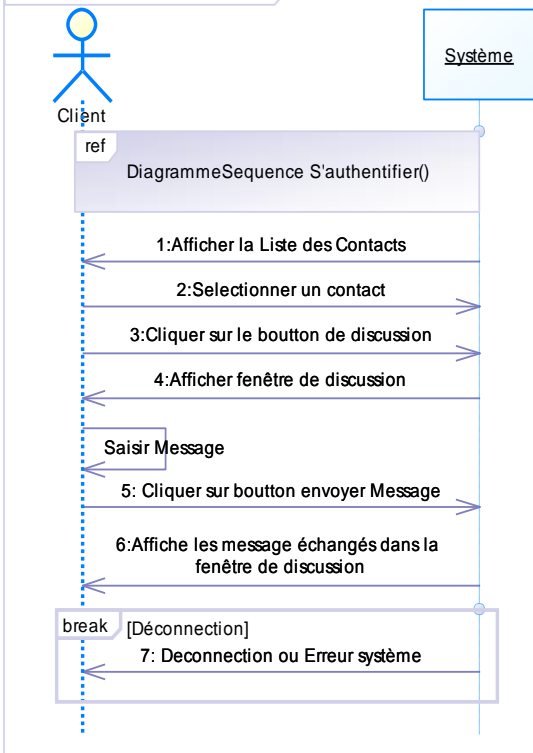


## LasiTalk! Messagerie Instantanée.

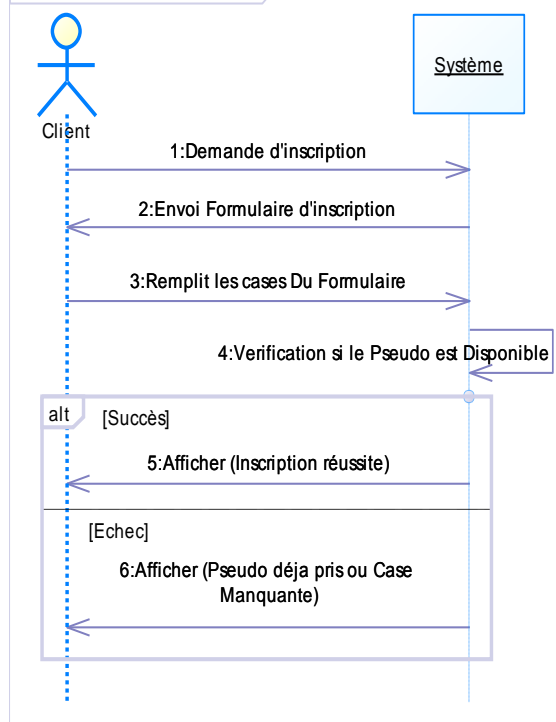
DiagrammeSequence S'authentifier



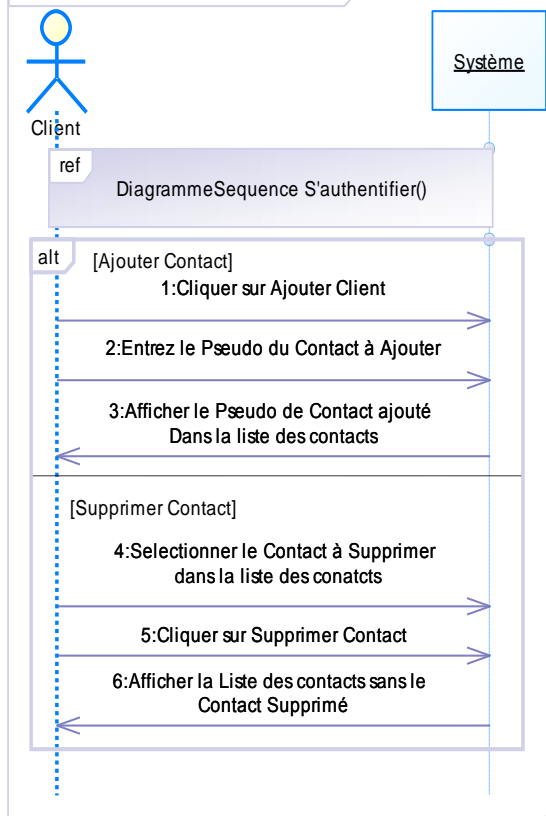
DiagrammeSequence Discussions



DiagrammeSequence S'inscrire



DiagrammeSequence Gérer les Contacts

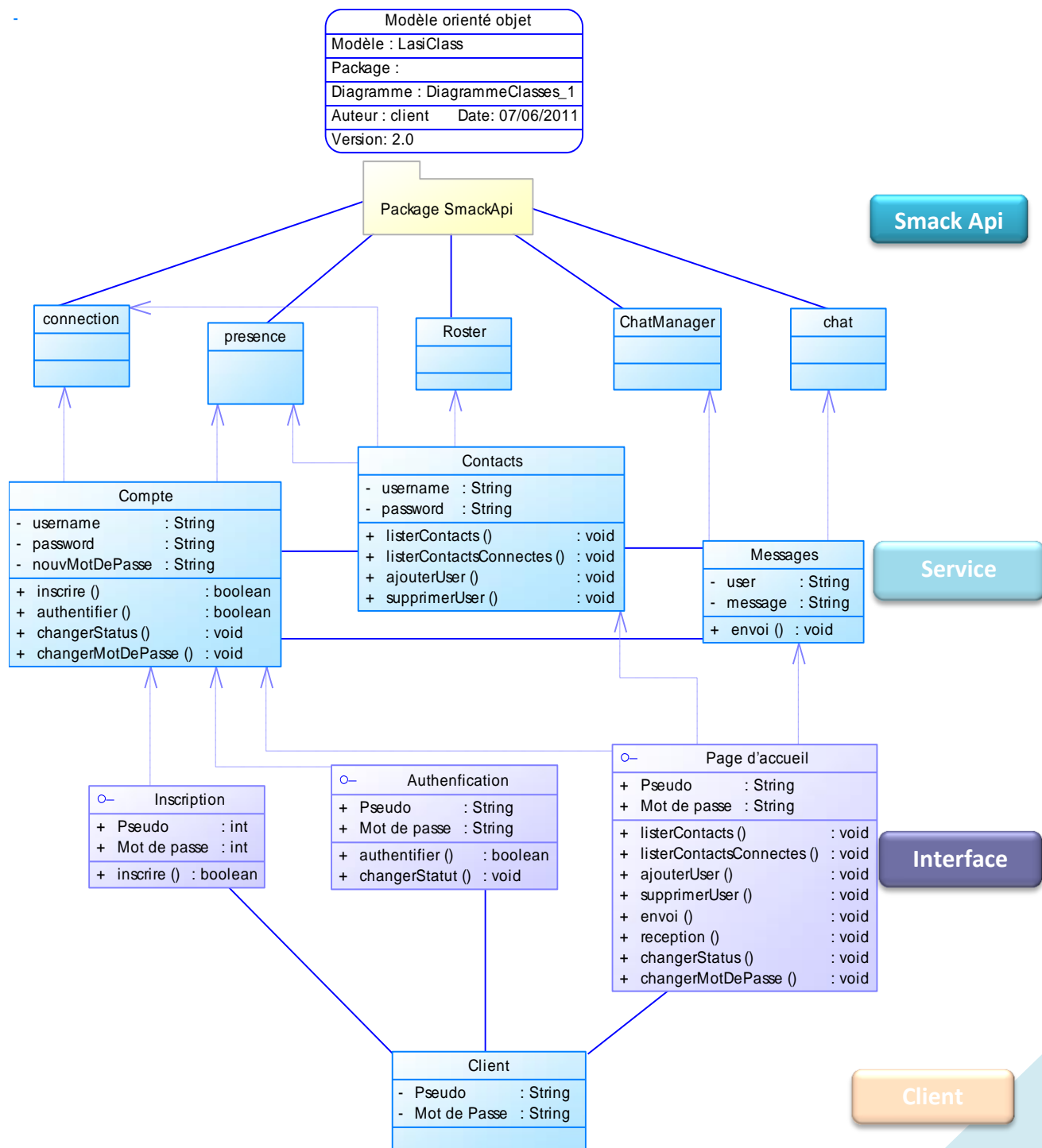






## 2.3. Diagramme de classes

Après avoir raffiné le maximum possible le diagramme de classe, on a pu obtenir un diagramme de classes final contenant cinq classes présentées ci-dessous:





**LasiTalk ! Messagerie Instantanée.**

## 1. Plateforme logicielle

### 1.1. Présentation de l'architecture Client/serveur :

L'architecture **client/serveur** désigne un mode de communication entre plusieurs ordinateurs d'un réseau qui distingue un ou plusieurs clients du serveur : chaque logiciel client peut envoyer des requêtes à un serveur. Un serveur peut être spécialisé en serveur d'applications, de fichiers, de terminaux, ou encore de messagerie électronique.

#### 1.1.1. Caractéristiques d'un serveur :



- Il est initialement passif (ou esclave, en attente d'une requête).
- Il est à l'écoute, prêt à répondre aux requêtes envoyées par des clients.
- Dès qu'une requête lui parvient, il la traite et envoie une réponse.

#### 1.1.2. Caractéristiques d'un client :

- Il est actif le premier (ou maître).
- Il envoie des requêtes au serveur.
- Il attend et reçoit les réponses du serveur.

Le client et le serveur doivent bien sûr utiliser le même protocole de communication. Un serveur est généralement capable de servir plusieurs clients simultanément.

Pour notre application **LasiTalk !** , on a choisi :

- ✓ protocole de communication : Jabber XMPP  Jabber
- ✓ serveur : OpenFire  openfire

**OpenFire :** OpenFire est un serveur Jabber libre (GPL) écrit en java par la Jive Software, l'éditeur du client libre Spark. C'est un protocole, qui est utilisé pour faire de la messagerie instantanée.



## ***LasiTalk ! Messagerie Instantanée.***

Autre fois connu sous le nom de Jive Messenger, Openfire bénéficie du soutien d'une importante communauté. Il est stable et réputé pour sa facilité d'installation et d'administration.

### **Avantages :**

- Support complet de XMPP.
- Facilité d'installation.
- Bénéficie d'une interface d'administration très complète et intuitive.
- Peut s'interfacer avec un grand nombre de composants externes (bases de données...).
- Intègre de nombreux services (serveur de discussion, Proxy de transfert de fichiers, etc.).
- Permet l'utilisation de nombreux plugins.
- Existe sous plusieurs plateformes : Windows, Linux, MacOS.

### **Inconvénients :**

- Pas de possibilité d'avoir des hôtes virtuels (un seul nom de domaine par serveur).
- Openfire n'est pas encore bien connu dans le monde de l'entreprise.

## **1.2 LANGAGE, OUTILS, LOGICIELS ET PACKAGES UTILISES**

### **1.2.1 Le Langage java :**

#### **Caractéristiques Java :**

- Java est interprétée.
- Java est portable : il est indépendant de toute plateforme.
- Java est orienté objet.
- Java est simple.
- Java est fortement typé.
- Java assure la gestion de la mémoire.
- Java est économe.
- Java est multitâche.



## ***LasiTalk! Messagerie Instantanée.***

### **1.2.2 Package Smack Api**

**Smack Api :** Smack Api est une bibliothèque open source conçu pour la messagerie instantanée Jabber XMPP. Cette bibliothèque est écrit en java, elle peut être intégrée dans des applications pour créer des services pour un client XMPP comme l'envoi de messages, ajout, etc.

#### **Avantages du Package Smack Api :**


- ✓ Package gratuit et open source.
- ✓ Simple et lisible.
- ✓ Bien documenté.
- ✓ Ses classes de haut niveau nous permettent de coder des fonctions simples rapidement et efficacement.
- ✓ C'est une couverture complète du protocole Jabber.

#### **Inconvénients du Package Smack Api :**

- ✓ Effectuer plusieurs teste pour comprendre le fonctionnement de chaque classes de ce package.
- ✓ Il existe plusieurs méthodes pour implémenter le même service mais qui ne donne pas le même résultat.
- ✓ Le package peut avoir des bugs de temps en temps.

### **1.3. Environnement de Développement Intégré :**


Pour la réalisation de **LasiTalk!**, on a utilisé deux environnements de développement intégrés qui sont :

**Eclipse**  : est un environnement de développement intégré libre extensible, universel et polyvalent, permettant de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions.




### **LasiTalk ! Messagerie Instantanée.**

**Eclipse et LasiTalk !** : Pour la création des classes de LasiTalk !, qui sont (Compte, Contact, Message) et les tests effectués sur nos services, on a utilisé la plateforme eclipse. Car eclipse est facile à utiliser surtout pour des programmeurs débutants son interface est très intuitive et les erreurs sont faciles à détecter, puisque le programmeur programme tout avec sa main (il n'y a pas de code généré directement par la plateforme).

**Netbeans**  : est un environnement de développement intégré (EDI), en plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++, JavaScript, XML, Ruby, PHP et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages Web).

**Netbeans et LasiTalk !** : Comme vous l'avez remarqué la principale différence entre ces deux plateformes par rapport à notre projet, c'est l'interface graphique. Netbeans a un éditeur graphique d'interface, cela va nous permettre de concevoir notre interface facilement avec l'aide des tutoriaux trouvés sur internet, après la conception de notre interface, on implémente notre l'interface qui est vide avec nos services qui était programmé et testé par eclipse.

**Nb :** Durant nos recherches et nos essais pour la réalisation de LasiTalk !, on a découvert une nouvelle plateforme, qui est peu connue elle s'appelle MyEclipse .

**MyEclipse** : Elle est construite sur l'Eclipse plateforme et elle intègre à la fois des solutions open source et propriétaires dans l'environnement de développement. L'édition standard ajoute des outils de base de données, un web designer visuel, outils de persistance, de printemps des outils.



**LasiTalk ! Messagerie Instantanée.**

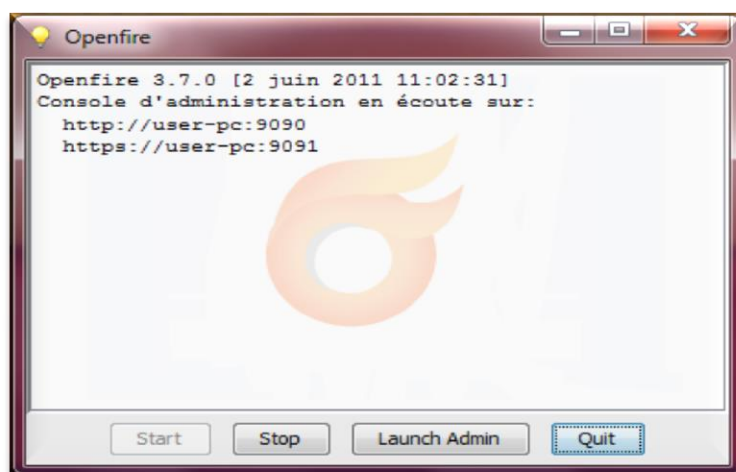
## 2. PRESENTATION DE L'INTERFACE DE **LasiTalk !**

### 2.1. Présentation de l'application cotée serveur : openfire™

#### 2.1.1. Description et fonctionnement d'Openfire

Le lancement d'OpenFire : OpenFire nous affirme qu'il est en écoute sur les ports qui sont : <http://user-pc:9090> : 9090

<https://user-pc:9091>



Pour accéder à la console d'administration OpenFire, on procède de la manière suivante :

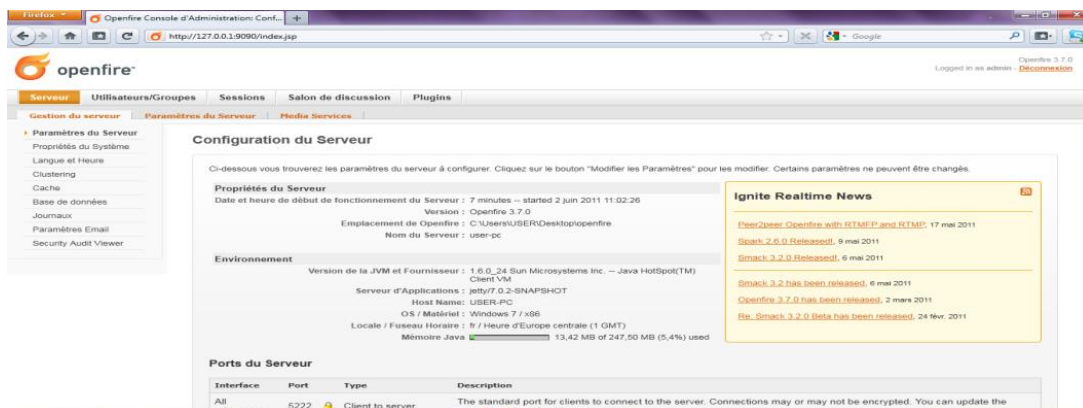
On clique sur **launch admin**, on se connecte en tant qu'administrateur avec le pseudo et mot de passe (admin, admin).



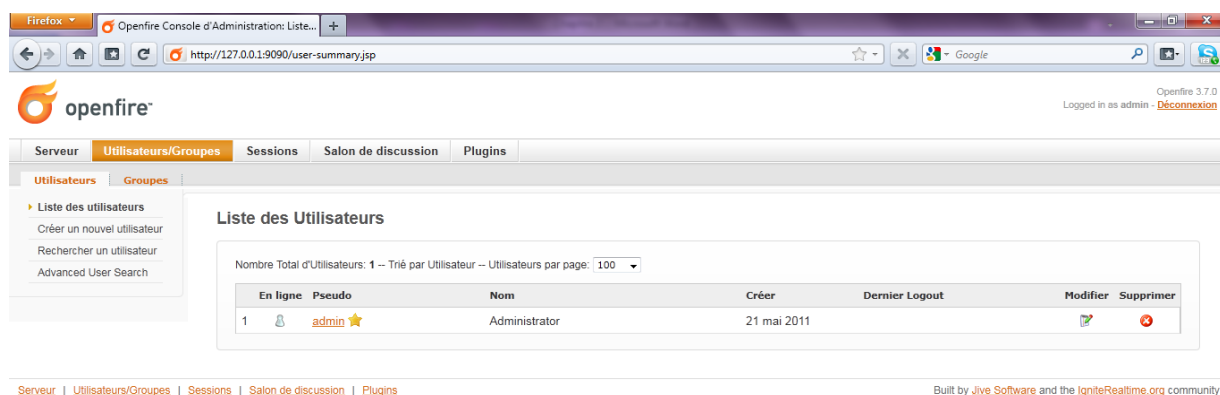


**LasiTalk ! Messagerie Instantanée.**

## Page de Configuration du serveur :



Avec cette page l'administrateur de l'application gère tous les services disponibles comme on le voit ici :



### 1. Créer un nouvel utilisateur :

- L'administrateur peut créer un nouvel utilisateur à travers la console d'administration, cela se fait comme ceci :




## LasiTalk ! Messagerie Instantanée.


### 2. Voir sa liste des contacts :

- L'administrateur peut voir la liste des contacts d'un utilisateur, comme on le voit ici :

### 3. Voir le statut d'un utilisateur :

- L'administrateur peut voir le statut d'un utilisateur avec le chemin **Utilisateurs/GROUPES** puis **liste des utilisateurs**.

**Icône verte**  : signifie qu'un utilisateur est en ligne.

**Icône grise**  : signifie qu'un utilisateur est en mode hors ligne.





**LasiTalk ! Messagerie Instantanée.**



Openfire 3  
Logged in as admin - [Déconnecter](#)

Navigation: [Serveur](#) | [Utilisateurs/GROUPES](#) | [Sessions](#) | [Salon de discussion](#) | [Plugins](#)

Utilisateurs | [Groupes](#)

► Liste des utilisateurs  
[Créer un nouvel utilisateur](#)  
[Rechercher un utilisateur](#)  
[Advanced User Search](#)

### Liste des Utilisateurs

Nombre Total d'Utilisateurs: 3 -- Trié par Utilisateur -- Utilisateurs par page: 100 ▼

	En ligne	Pseudo	Nom	Créer	Dernier Logout	Modifier	Supprimer
1		admin ★	Administrator	21 mai 2011			
2		user1		2 juin 2011			
3		user2		2 juin 2011			

Footer: [Serveur](#) | [Utilisateurs/GROUPES](#) | [Sessions](#) | [Salon de discussion](#) | [Plugins](#) | Built by [Jive Software](#) and the [IgniteRealtime.org](#) commu

L'administrateur peut aussi supprimer un contact et modifier les propriétés d'un contact.

**Découvrir LasiTalk !**

L'utilisateur de n'importe quelle application a besoin de se faire une image mentale et une vision globale pour comprendre la structuration et le fonctionnement de de cette application.

En conséquence, tout auteur d'une messagerie instantanée doit donner une identité visuelle à travers la charte graphique, les polices et les couleurs des caractères utilisés.

Pour **LasiTalk !**, on a :

- La charte graphique de **LasiTalk** : qui est le logo de notre application **LasiTalk** ! les icônes, et les couleurs utilisées, etc.



### **LasiTalk ! Messagerie Instantanée.**

- La charte graphique, simplifiée à son maximum. C'est un peu dans l'esprit de Skype ou MSN, où la couleur dominante est le bleu. On ressent tout de suite un apaisement, rapidement identifié, contrairement aux autres messageries instantanées.
- L'interface de **LasiTalk !** :
  - Le contenu de chaque fenêtre et sa fonctionnalité.
  - L'emplacement des différents éléments au sein de ces fenêtres.
  - Au niveau des options et des possibilités, la simplicité est aussi à l'honneur tout en jouant la carte de l'efficacité.

### **Création du LOGO LasiTalk !**

On a utilisé des couleurs vifs et neutres, une police simple tout en gardant l'esprit de la messagerie instantanée.

Le mot Lasi représente les radicaux des deux prénoms LAMIS et SIRINE par lesquels le projet a été conçu.



Le mot talk qui est un mot anglais signifie parler ou chatter



**LasiTalk !** Messagerie Instantanée.

## Les différentes interfaces de **LasiTalk !**



**Fig 1** Interface de la fenêtre d'authentification de **LasiTalk !**

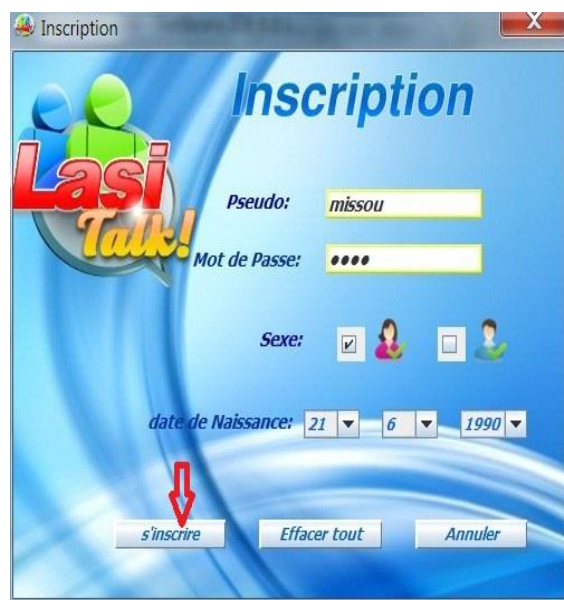
**La figure 1** représente la première interface qu'on obtient au lancement du service.



**LasiTalk !** Messagerie Instantanée.

### S'inscrire sur LasiTalk !

Pour s'inscrire sur **LasiTalk !**, la démarche à suivre est la suivante :



**Fig 2** Interface de la fenêtre d'inscription

La **figure 2** représente l'interface de la fenêtre d'inscription de **LasiTalk !**

Les champs pseudo et mot de passe sont obligatoires par contre les champs sexe et date de naissance sont optionnels.



## **LasiTalk ! Messagerie Instantanée.**



**Fig 2 .1 Interface de la fenêtre d'inscription**

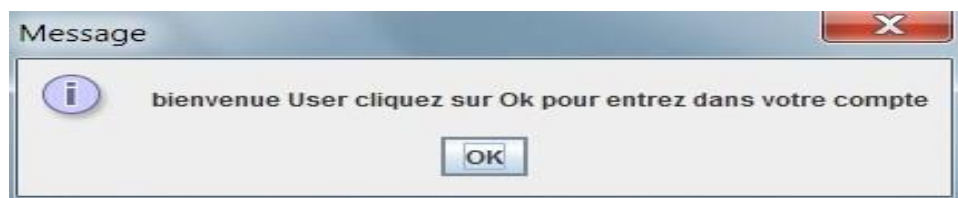


**Fig 2 .2 Interface de la fenêtre d'inscription**

**La figure 2.1 :** Si le pseudo est déjà utilisé, une boîte de dialogue apparaît et prévient l'utilisateur que ce pseudo existe déjà.

**La figure 2.2 :** Si le serveur n'est pas lancé, une autre boîte de dialogue apparaît et demande à l'utilisateur de bien lancer le serveur

Sinon si l'inscription est une réussite, une boîte de dialogue apparaît à l'utilisateur et lui informe qu'il a été inscrit avec succès



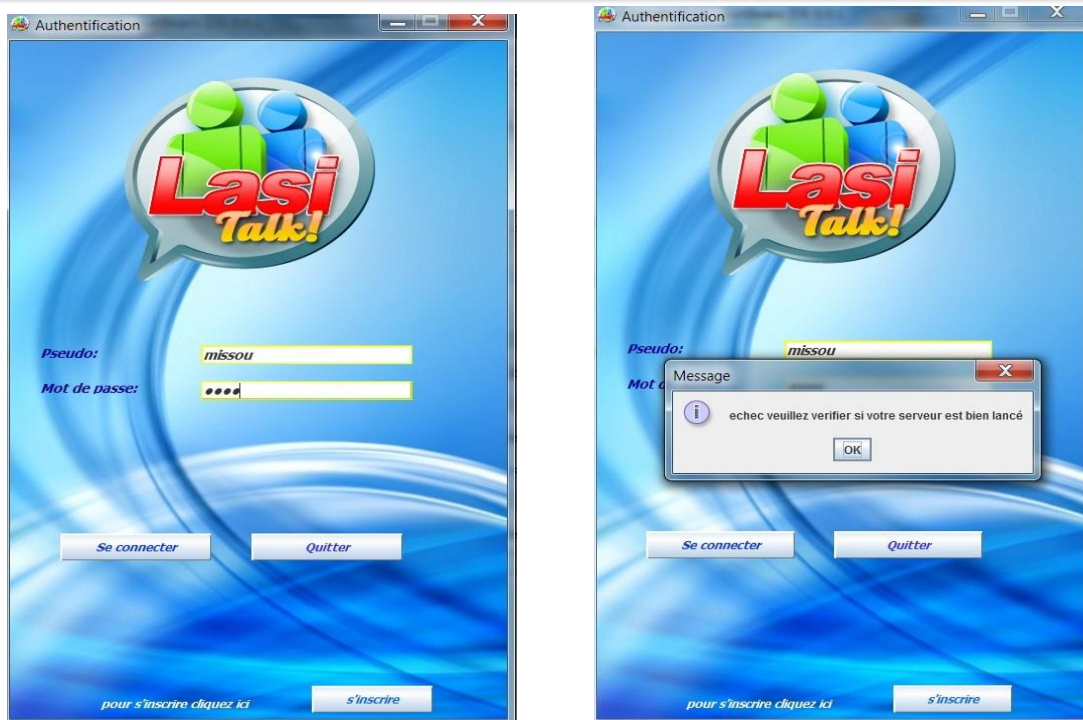
Un clique sur OK permet à l'utilisateur d'accéder à la fenêtre principale de **LasiTalk !**





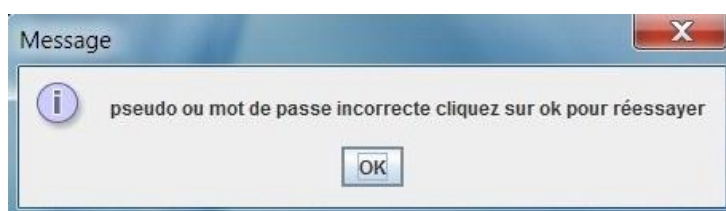
**LasiTalk ! Messagerie Instantanée.**

### Se Connecter à LasiTalk !



**Fig. 3 Interface de la fenêtre d'authentification**      **Fig. 3.1 Interface de la fenêtre d'authentification**

La figure 3 représente un scénario d'enchaînement d'erreurs, quand le serveur n'est pas lancé. L'application informe l'utilisateur par une boîte de dialogue que le serveur n'est pas lancé.



**Fig. 3.2 Interface de la fenêtre d'authentification**

La figure 3.2 représente un autre scénario d'enchaînement d'erreurs, quand le pseudo ou le mot de passe d'un utilisateur sont incorrects, une boîte de dialogue apparaît et informe l'utilisateur que son pseudo ou son mot de passe est invalides. Cliquez sur OK permet à l'utilisateur de ressaisir son pseudo et son mot de passe Et réessayer de se connecter une 2<sup>ème</sup> fois.



**LasiTalk ! Messagerie Instantanée.**

### 3. Les fonctionnalités de LasiTalk !



**Fig. 4 Interface de la fenêtre principale de LasiTalk !**

La Figure 4 représente le cœur de **LasiTalk !**

**LasiTalk ! Contient :**

- **Un menu barre :** qui contient deux items. Un item sert pour la gestion de contacts (ajouter Contacts) et un autre item pour le compte (changer de mot de passe, se déconnecter).
- **Des Boutons :** pour afficher les contacts, supprimer un contact, envoyer un message.
- **Une JListe :** pour afficher les contacts.
- **Un TextArea :** pour afficher les messages entrants et les messages sortants.
- **Un JTextField :** pour écrire le message à envoyer.
- **Une Liste déroulante :** pour choisir le statut de la connexion.



## **LasiTalk ! Messagerie Instantanée.**

### **Fonctionnalités :**

#### **3.1. Ajouter un Contact**



**Fig. 4.1 Ajout**

Pour ajouter un contact, on va dans **menu/ contacts /ajouté Contacts/**. Ensuite, on entre le pseudo du contact qu'on veut ajouter, un message de confirmation apparaît pour nous affirmer que le contact a bien été ajouté.

**Nb :** Pour vérifier qu'un contact est ajouté, on rafraîchit la liste des contacts

#### **3.2 Changer de mot de passe**



**Fig. 4.2 Changer mot de passe**





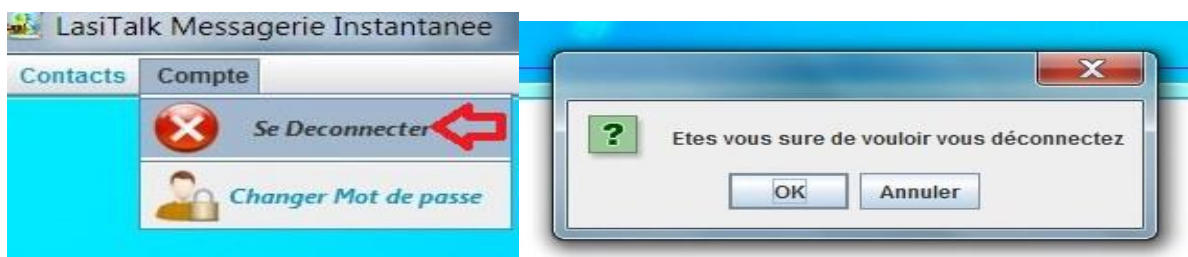
## **LasiTalk ! Messagerie Instantanée.**

Pour changer de mot de passe, on va dans le **menu/ compte/changer de mot de passe**.

On entre le nouveau mot de passe, ensuite, on clique sur changer, un message de confirmation apparaît pour nous indiquer que notre mot de passe a été changé avec succès.

**Nb :** pour vérifier vous pouvez vous déconnecter et vous reconnectez une autre fois.

### **3.3 Se déconnecter**



**Fig. 4.3 Déconnexion**

Pour se déconnecter, on va dans **menu/ compte/se déconnecter**.

Une boîte de dialogue apparaît et nous demande si on veut vraiment se déconnecter et quittez **LasiTalk !**

- ✓ **Clique sur OK** : permet de se déconnecter et de quitter l'application.
- ✓ **Clique sur le bouton Annuler** : nous permettra de rester sur **LasiTalk !**



## LasiTalk ! Messagerie Instantanée.

### 3.4 Afficher la liste des contacts connectés | 3.5 Afficher la liste de tous les contacts



Fig. 4.4.Contacts en Ligne



Fig. 4.5 Tous les Contacts

- ✓ Cliquez sur le bouton vert permet d'afficher la liste des contacts connectés.
- ✓ Cliquez sur le bouton gris permet d'afficher la liste de tous les contacts.

### 3.6 Supprimer un contact

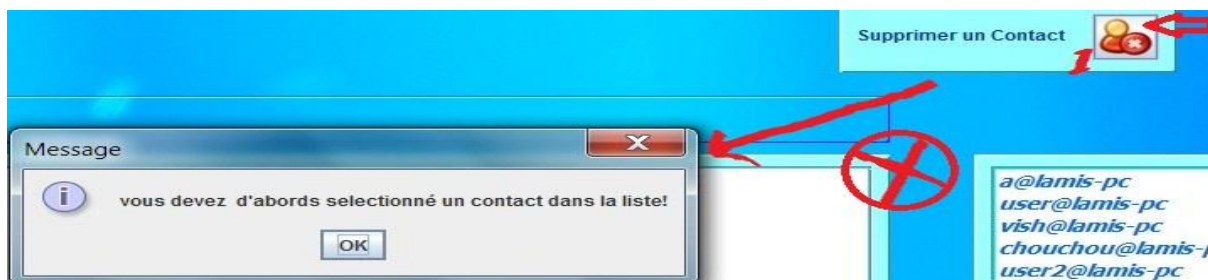


Fig. 4.6.1 Supprimer un contact



Fig. 4.6.2 supprimer un contact



## **LasiTalk ! Messagerie Instantanée.**

Pour supprimer un contact, on doit d'abord le sélectionner dans la liste puis on le supprime avec le bouton supprimer, un message de confirmation apparaît :

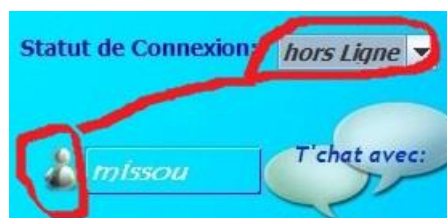
- ✓ **Clique sur OK** : permet supprimer le contact définitivement de la liste.
- ✓ **Clique sur le bouton annuler** : arrête l'opération et le contact ne sera pas supprimé.

**Nb :** Pour vérifier qu'un contact est supprimé, on rafraîchit la liste des contacts.

### **3.7 Statut de la connexion**



**4.7.1 Statut en Ligne**



**4.7.2 Statut hors Ligne**

Avec la liste déroulante, on peut changer le statut de connexion de en ligne à hors ligne et vice versa quand le statut est en mode (en ligne) une icône verte apparaît devant le pseudo du propriétaire de ce compte, si le contact est en mode (hors ligne) une icône grise apparaît.

### **3.8 Echanger des messages**



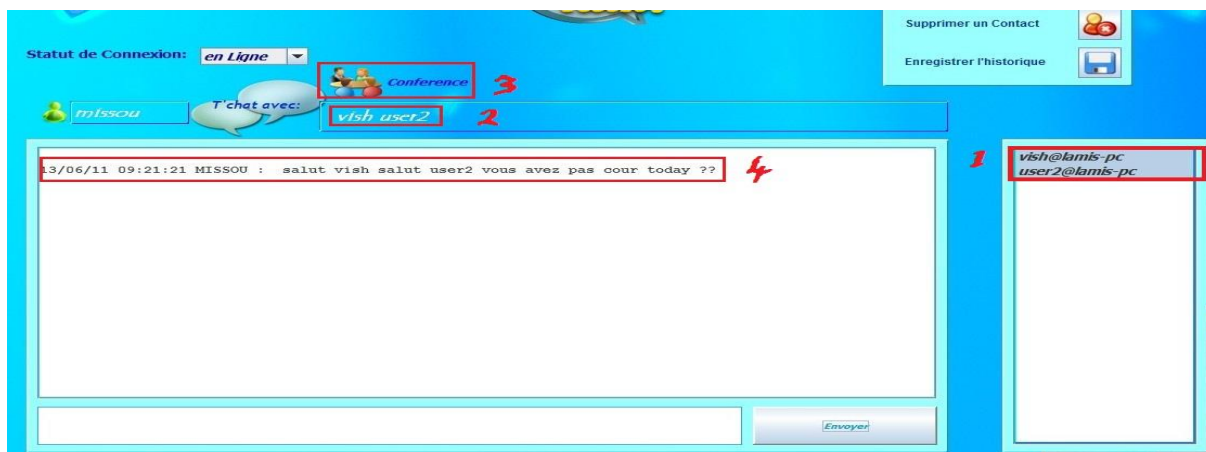
**4.8.1 Taper un message**



## LasiTalk ! Messagerie Instantanée.



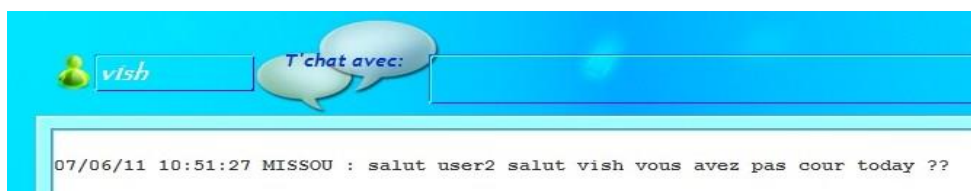
### 4.8.2 Conversation (une seule personne)



### 4.8.3 Conférence



#### 4.8.3.1 Conférence (message reçu)



#### 4.8.3.2 Conférence (message reçu)

La figure 4.8.1 : Pour envoyer un message, on sélectionne un contact dans la liste de contact, on tape notre message puis on l'envoie avec le bouton envoyer.



## **LasiTalk ! Messagerie Instantanée.**

Comme vous l'avez remarqué avec **LasiTalk !**, on peut parler avec :

- ✓ **Une seule personne**  : **Figure 4.8.2**

On sélectionne un contact dans la liste. On tape le message et on envoie le message. **LasiTalk !** Affiche le pseudo du contact avec qui on est en train de discuter **figure 4.8.1**, **LasiTalk !** Nous affiche aussi que c'est une conversation (avec une seule personne)

- ✓ **Plusieurs personnes au même temps (conférence)**  : **Figure 4.8.3**

On peut envoyer un même message à plusieurs personnes, en les sélectionnant dans la liste des contacts avec (ctrl +souris).



Exemple : Le message envoyé par (missou) dans la **figure 4.8.3** était reçu par (user2 et vish).

Comme on le voit dans la **figure 4.8.3.1** et **figure 4.8.3.2**.

**Nb:** un utilisateur peut envoyer des messages instantanés à un autre utilisateur, c'est-à-dire que même si le destinataire n'est pas connecté, on peut lui envoyer des messages. Les messages seront affichés chez lui lors de sa prochaine connexion sur **LasiTalk !**

### **4. Les fonctionnalités optionnelles de LasiTalk !**

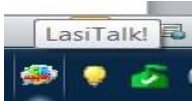
Après avoir terminé avec les besoins nécessaires de **LasiTalk !**, nous avons songé à ajouter d'autres fonctionnalités comme :

1. **Conférence**  : Expliquer dans la **figure 4.8.3**, on a pensé qu'un utilisateur peut vouloir discuter avec plusieurs personnes en même temps alors on a rajouté cette fonctionnalité dans l'envoi des messages sur **LasiTalk !**
2. **Icone système**  : On a pensé qu'un utilisateur peut fermer la fenêtre de **LasiTalk !** sans vouloir se déconnecter alors on a créé une icône système, qui représente un raccourci **LasiTalk !**, et qui contient un menu barre qui permet de naviguer sur les différentes interfaces de **LasiTalk !**




## LasiTalk ! Messagerie Instantanée.

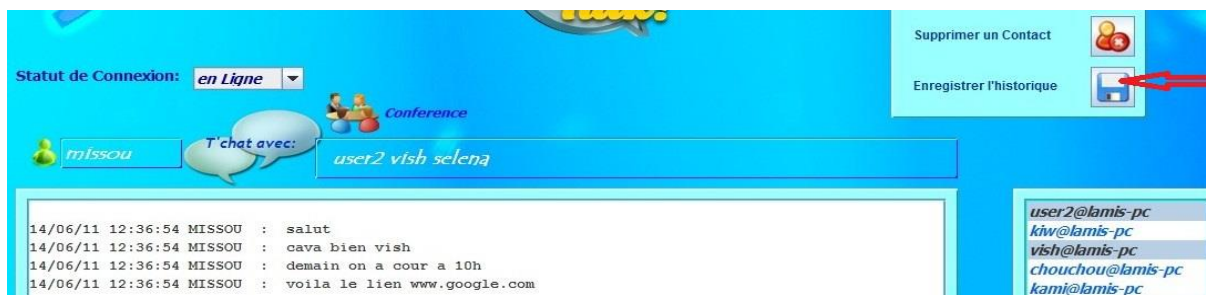
- ✓  Icône système pour LasiTalk !

- ✓  Une bulle d'aide pour LasiTalk !





- ✓  Grâce à ce menu barre, on peut accéder aux différentes fenêtres de LasiTalk !

- ✓  On a remplacé l'icône de java par l'icône LasiTalk !

**3. Historique :** nous avons pensé qu'un utilisateur peut vouloir enregistrer une conversation alors on a ajouté la fonctionnalité historique qui permet d'enregistrer une conversation dans des fichiers txt.



Le nom du fichier de la conversation est enregistré selon la date, l'heure et le nom de l'utilisateur qui a enregistré la conversation.

	missou14_06_11 12_48Conversation	14/06/2011 12:49	Document texte	1 Ko
	missou14_06_11 12_50Conversation	14/06/2011 12:51	Document texte	1 Ko
	simal	19/05/2011 13:50	Fichier PNG	80 Ko
	vish13_06_1	13/06/2011 12:05	Document texte	1 Ko

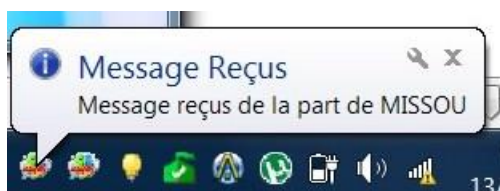




## **LasiTalk ! Messagerie Instantanée.**



4. **Notification** : Quand un message est reçu, une notification apparaît devant la barre système informant l'utilisateur qu'un message est reçu en lui précisant le nom de l'utilisateur qui a envoyé ce message.



5. **L'heure et la date** : l'une des caractéristiques principale de LasiTalk ! est l'envoi des messages instantanés, alors on a affiché l'heure et la date des messages envoyés et des messages reçus.

```
07/06/11 09:30:53 MISSOU : salut vish comment ça va ??  
07/06/11 09:30:53 MISSOU : salut user2 salut vish vous avez pas cour today??
```

6. **les bips sonores** : Un bip retentit, lors de la réception d'un message.
7. **Bulle d'aide** : Pour chaque bouton, on a mis une bulle d'aide pour que l'utilisateur ne se perd pas.





***LasiTalk ! Messagerie Instantanée.***

## Conclusion

Ce projet a été avant tout une synthèse de notions acquises durant les trois années précédentes et il nous a permis d'acquérir de nouvelles connaissances, dans le domaine du projet tout comme la programmation.

Le but de notre travail était de réaliser une application de messagerie instantanée en langage JAVA, nous estimons avoir atteint notre objectif, en utilisant les nouvelles technologies : langage de programmation (java), environnement de programmation (Eclipse, Netbeans), et surtout le protocole Jabber qui est un protocole conçu spécialement pour la messagerie instantanée avec son package Smack Api, son serveur Open fire, grâce à lui on a découvert de nouvelles méthodes du nouveau package et on espère que ce protocole se mondialisera partout dans le monde, car c'est une source sûre pour la messagerie instantanée.

Notre expérience a été très instructive et très importante pour le développement, approfondissement et enrichissement de nos connaissances, et on a profité durant cette découverte d'exprimer les techniques de traitements dans les cas réels en suivant le processus de développement tout en acheminant notre travail de l'étude préliminaire jusqu'à l'implémentation de notre projet.

Pour conclure, on peut dire que nous avons pu atteindre les objectifs visés, dans la limite du temps qui nous a été accordé, au début de notre mémoire et ce, en la réalisation d'une Application de Messagerie Instantanée en Java.

Enfin nous espérons, que ce modeste travail va plaire et on espère qu'il puisse aider les promotions à venir.





***LasiTalk ! Messagerie Instantanée.***

## **Bibliographie**

DVD Apprendre Java Les fondamentaux (Elephorm la formation par les nouveaux médias).

<http://sensorapp.net/?p=85>

<http://www.youtube.com/watch?v=0Re608vsi0A>

<http://prevert.upmf-grenoble.fr/Prog/Java/swing/JList.html>

<http://www.programmez.com/tutoriels.php?tutoriel=77&titre=Icône-dans-la-barre-de-système-avec-Java-6>

[http://forum.hardware.fr/hfr/Programmation/Java/java-afficher-heure-sujet\\_82635\\_1.htm](http://forum.hardware.fr/hfr/Programmation/Java/java-afficher-heure-sujet_82635_1.htm)