Rapport final

Application de vidéoconférence

Rapport présenté à

Pr. Dr. Kamel Adi

Par

Mathieu Gauthier
Yannick Gauthier

Dans le cadre du cours

Projet synthèse

1. Rappel du projet....3
1.1 Description....3
1.2 Objectif du projet....3
1.3 Définition....3
1.4 Échéancier....4
2. Analyse des logiciels existants....5
2.1 Contexte....5
2.2 Approche....5
2.3 Point de départ....5
2.4 Profil des utilisateurs....6
2.5 Environnement des utilisateurs....6
2.6 Types de transmission de flux multimédia....7
3.6.1 L'unicast....7
2.6.2 Le Peer to peer (d'égal à égal)....8
2.6.3 Le multicast....9
2.7 Les différents protocoles....10
2.7.1 Le protocole H.323....10
2.7.2 Types de MCU (multipoint control unit) disponibles....13
2.7.3 Types de Gatekeepers disponibles....14
2.7.4 Le protocole SIP (Session Initiation Protocol)....15
2.8 Solutions avec le protocole H323....17
2.8.1 CU-SeeMe de l'université de Cornell....17
2.8.4 Click to Meet....20
2.9 Solutions avec le protocole SIP....21
2.9.1 Marratech....21
2.9.3 Breeze Meeting....23
2.9.4 AccessGrid....24
Interopérabilité....25
2.11 Conclusion....28
3. Implémentation du logiciel....29
3.1 Langage de programmation utilisé....29
3.2 Environnement de développement utilisé....29
3.3 Plateforme utilisée....29
3.4 Les librairies existantes....30
3.5 Les Tests....31
3.6 Fonctionnement interne de l'application cliente....32
3.8 Guide des classes....34
3.9 Guide de l'utilisateur....58

1. Rappel du projet

1.1 Description

L'objectif de notre projet est donc d'explorer le domaine de la vidéoconférence pour l'enseignement à distance en réalisant une étude critique expérimentale des plateformes existantes. Ensuite, nous allons développer une application de vidéoconférence qui permettra à deux individus de se voir à l'aide d'une caméra et de se parler à l'aide d'un microphone. Nous allons concevoir une application cliente qui pourra être déployée sur chacun des postes clients. De plus, nous allons développer une application serveur qui permettra aux utilisateurs de se connecter à celui-ci, dans le but d'entrer en communication entre eux.

1.2 Objectif du projet

Tout d'abord, il faut savoir que le but du projet n'est pas de faire mieux que les applications de vidéoconférence qui sont sur le marché. Nous sommes conscients que ces applications sont développées depuis plusieurs années par plusieurs programmeurs et nous ne pourrons faire mieux. L'objectif principal est d'apprendre lors de la conception de notre projet. Apprendre en approfondissant nos connaissances par la recherche et par nos erreurs. Nous voulons consolider les connaissances acquises durant notre bac en informatique. Nous avons choisi ce projet parce qu'on y voyait un certain défi dans sa réalisation.

1.3 Définition

La vidéoconférence est un mode de communication qui utilise une caméra et un microphone pour transmettre l'image et le son à chacun des participants.

Elle permet à un groupe de plusieurs personnes situées à différents endroits géographiques d'avoir une conversation mêlant l'audio et la vidéo.

1.4 Échéancier

Tâches	Date de début	Date de fin
Élaborer les spécifications du projet	07/01/2008	13/01/2008
Analyser les applications de vidéoconférence déjà existante	14/01/2008	20/01/2008
Recherche des librairies et modules existants pouvant être utilisés	21/01/2008	27/01/2008
Déterminer le langage de programmation qui sera utilisé	21/01/2008	27/01/2008
Déterminer l'environnement de développement dans lequel sera développée l'application	21/01/2008	27/01/2008
Déterminer la plateforme sur laquelle seront exécutées les applications cliente et serveur	21/01/2008	27/01/2008
Conception de notre application, identification des différents modules, interaction entre les modules	21/01/2008	27/01/2008
Développer le module de gestion de la webcam	28/01/2008	10/02/2008
Développer le module de gestion du son	11/02/2008	24/02/2008
Développer l'application cliente	25/02/2008	09/03/2008
Développer l'application serveur	10/03/2008	23/03/2008
Tester l'application (la partie cliente et serveur).	24/03/2008	30/03/2008
Rédiger les documentations pour le projet	31/03/2008	…

2. Analyse des logiciels existants

2.1 Contexte

Le département d'informatique et d'ingénierie s'est récemment intéressé à l'enseignement à distance et souhaite à terme offrir des cours à distance dans ses programmes de certificat et de baccalauréat en informatique. Dans ce contexte, nous allons donc réaliser une étude critique expérimentale des plateformes existantes.

2.2 Approche

Nous avons analysé les différentes technologies qui étaient utilisées dans le domaine de la vidéoconférence. Comme point de départ, nous avons sélectionné les systèmes qui pouvaient s'appliquer à l'enseignement à distance, pour ainsi faire une étude plus approfondie de chacun d'eux.

Nous avons fait le test des logiciels qui semblait répondre le plus au critère d'enseignement à distance. Ces tests nous permettent de faire une étude encore plus profonde des logiciels qui sont présentement sur le marché.

2.3 Point de départ

Les recommandations que vous allez retrouver dans ce document sont basées sur plusieurs points :
-Le choix des systèmes a été fait selon les critères énoncés en classe par les enseignants responsables de la vidéoconférence dans le contexte d'un enseignement à distance.
-Nous avons déterminé un équipement matériel de base devant être requis par chacun des utilisateurs.
-Du côté réseautique, il est certain que les utilisateurs devront avoir une connexion suffisamment rapide pour subvenir à la bande passante d'une application de vidéoconférence.
-Nous n'avons pas seulement regardé les solutions « open sources », mais aussi les solutions payantes.
-Ce document ne traite pas des problèmes qui peuvent survenir à cause de l'infrastructure du réseau. Des « firewalls » et le « NAT » ou une connexion ayant une bande passante limitée peuvent causer des problèmes au logiciel serveur recommandé.

2.4 Profil des utilisateurs

Les utilisateurs sont pour la plupart des étudiants qui vont assister à des cours à partir de la maison. Certains enseignants auront aussi accès à ce système de vidéoconférence. Ainsi, nous pouvons en déduire que les étudiants n'auront pas le même type d'équipement sous la main. Certains d'entre eux n'auront possiblement pas les connaissances pour faire l'installation de logiciel ou encore, pour se connecter à un serveur. Dans le cas des enseignants, nous ne pouvons pas être certains que tous ont des connaissances assez poussées en informatique pour faire une utilisation judicieuse des logiciels. Le logiciel client devra être convivial, mais on ne peut le rendre parfait ou automatisé.

2.5 Environnement des utilisateurs

Pour l'étudiant :
Il est certain que les étudiants auront des ordinateurs différents, des Webcams différentes, des microphones différents et des connexions différentes. Il ne faut pas omettre que la qualité de l'équipement joue un rôle très important. La Webcam et le microphone devront être d'une certaine qualité si nous désirons une communication à un niveau acceptable. Le système d'exploitation que nous allons choisir pour être utilisés du côté client est Windows. C'est le système d'exploitation le plus utilisé actuellement. On ne peut se permettre de faire changer les utilisateurs de système d'exploitation, la tâche serait trop difficile. Ainsi, en sélectionnant le système d'exploitation le plus utilisé nous rejoignons le maximum de personne.

Côté réseautique, les utilisateurs ont des connexions ayant des débits différents. La seule caractéristique commune concernant les connexions des utilisateurs est l'accès à un réseau IP. Ainsi, nous allons devoir imposer un débit minimum pour une connexion dans lequel notre solution fonctionnera correctement.

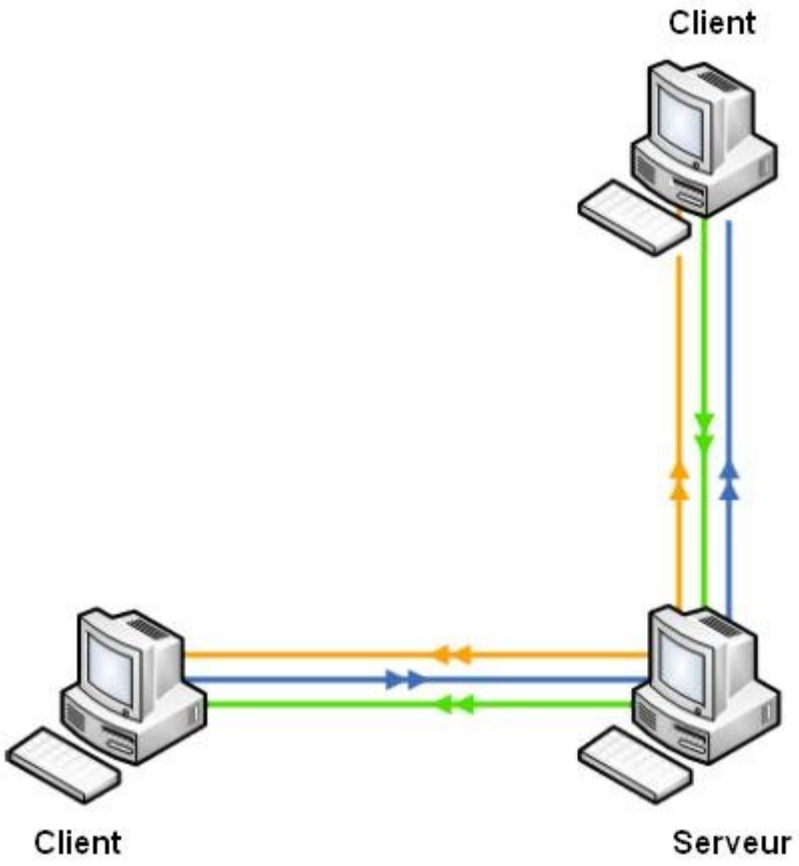
Pour l'enseignant :
Des salles seront spécialement aménagées pour permettre un enseignement en ligne de qualité. Le système d'exploitation sera Windows XP, qui est déjà installé dans la plupart des classes. Les équipements utilisés seront assez performants pour que la communication soit de qualité. Il est préférable que chaque classe possède un équipement similaire.

2.6 Types de transmission de flux multimédia

Trois possibilités sont disponibles sur un réseau TCP/IP pour transmettre des flux multimédias.

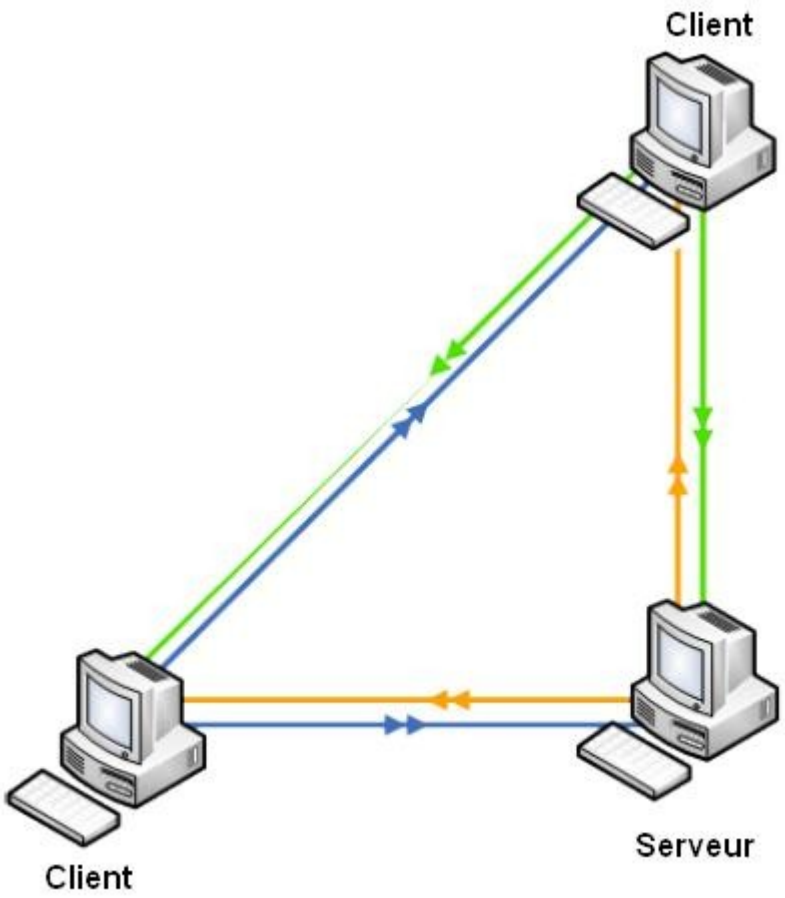
3.6.1 L'unicast

Tous les clients envoient leurs flux au serveur, qui les retransmet à son tour à tous les participants.
Avantages :
-Seul le serveur a besoin d'une bande passante importante.
-Simplification de la gestion des flux des clients côté serveur (play/stop sur les flux).
Inconvénient :
-Implique un temps de latence supplémentaire.



2.6.2 Le Peer to peer (d'égal à égal)

Chaque client envoie ses flux à tous les autres participants.
Avantage :
- Envoi direct, pas d'introduction de latence supplémentaire.
Inconvénients :
-Tous les clients ont besoin d'une bande passante importante.
-Nécessite une gestion des flux clients plus poussée (play/stop sur les flux).



2.6.3 Le multicast

Utilisation d'une adresse réseau spécifique pour la transmission des flux.

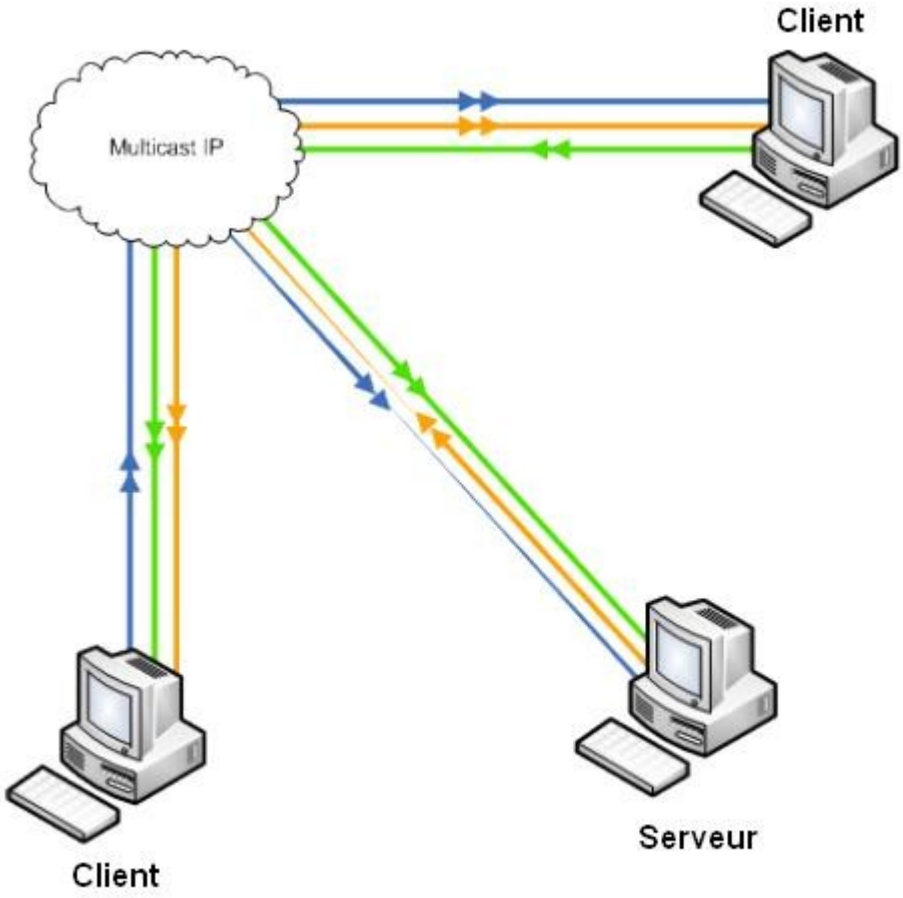
Avantages :

-Envoi direct, pas d'introduction de latence supplémentaire.

-La bande passante nécessaire est peu importante.

Inconvénients :

-Fonctionne uniquement en LAN.



2.7 Les différents protocoles

2.7.1 Le protocole H.323

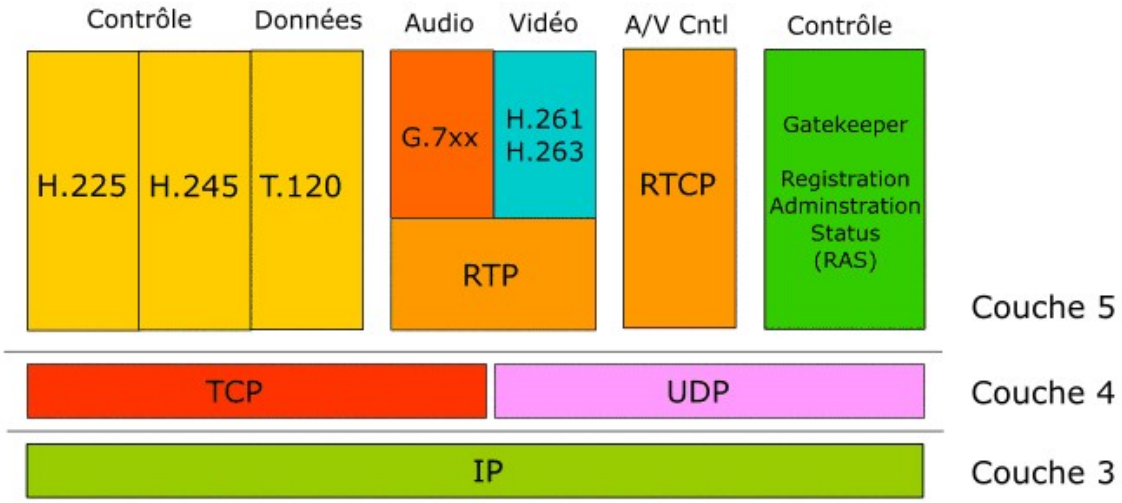
C’est un protocole très ancien, mais qui est encore utilisé dans le domaine de la vidéoconférence. Il reste quand même une très bonne solution assez performante.

En fait, c’est plus qu'un protocole, H.323 ressemble davantage à une association de plusieurs protocoles différents et qui peuvent être regroupé en trois catégories : la signalisation, la négociation de codec, et le transport de l’information.

Les messages de signalisation sont ceux que l’on envoie pour demander d’être mis en relation avec une autre personne.

La négociation est utilisée pour se mettre d’accord sur la façon de coder les informations qu’on va s’échanger. Il est important que les téléphones (ou systèmes) parlent un langage commun s’ils veulent se comprendre. Il serait aussi préférable, s’ils ont plusieurs alternatives de langages qu’ils utilisent le plus adaptées. Il peut s’agir du codec le moins gourmand en bande passante ou de celui qui offre la meilleure qualité. Le protocole utilisé pour la négociation de codec est le H.245

Le transport de l’information s’appuie sur le protocole RTP qui transporte la voix, la vidéo ou les données numérisées par les codecs. On peut aussi utiliser les messages RTCP pour faire du contrôle de qualité, voire demander de renégocier les codecs si, par exemple, la bande passante diminue. Pour les conversations de groupe, ce protocole nécessite un MCU et un Gatekeeper.



Élément requis

MCU (multipoint control unit)
La vidéoconférence à plusieurs nécessite de l'équipements supplémentaires par rapport à la vidéoconférence d'un à un. Tout d'abord, elle nécessite un « MCU ». Le MCU est un serveur central multimédia sur lequel chacun des clients se connecte pour joindre une session de vidéoconférence. Le MCU se charge de contrôler l'audio et le vidéo pour chacun des participants d'une session. Il est possible de cascader plusieurs MCU pour accroître la capacité en répartissant la charge de travail.

Le MCU est un logiciel informatique ou une machine servant à établir simultanément plusieurs communications de visioconférence ou de VoIP. Cette fonction permet à plusieurs « clients » de la conférence de se réunir dans une même « salle virtuelle » de discussion. On doit dans beaucoup de cas lier le MCU à un Gatekeeper.

Plusieurs MCU peuvent être mis en cascade pour permettre à un plus grand groupe de communiquer. Les MCU en cascade se partagent la tâche et permettent de rester performant même quand on a beaucoup de personnes qui communiquent en même temps.

Gatekeeper
De plus, le H.323 Gatekeeper est un autre serveur qui contrôle et gère les ressources pour une session de vidéoconférence.

Selon la recommandation H.323, un gatekeeper doit fournir les services suivants:

Traduction d'Adresse
Contrôle d'Admissions
Contrôle de Bande Passante
Gestion de Zone
Call Control Signaling
Autorisation d'Appel

Gestion de Bande Passante

De plus, le gatekeeper permet aussi de faire la gestion entre un client de vidéoconférence et un téléphone. On peut donc faire des appels téléphoniques si la configuration du gatekeeper est faite en ce sens.

Plateforme supportée

La plupart des applications clientes sont seulement disponibles pour les différentes versions de Windows. La plupart des applications clientes gratuites sont disponibles pour toutes les plateformes.

Convivialité

Les applications clientes les plus anciennes sont basées sur le protocole H.323. Ces applications ont passé par plusieurs versions et ont été graduellement améliorées pour en arriver maintenant. Donc, on se retrouve avec des applications qui sont vraiment conviviales. Les utilisateurs peuvent se servir de ces logiciels sans trop de difficulté. Quelques explications sur les fonctions du logiciel permettront aux utilisateurs de s'y retrouver avec relativement d'aisance. De plus, les nouveaux MCU offrent la possibilité de choisir entre différents styles de fenêtrage. Vous pouvez choisir de voir seulement la personne qui parle dans une grande zone ou toutes les personnes de la conversation dans plusieurs petites zones.

Interopérabilité

Il est possible de joindre une communication sur un MCU à partir d'applications différentes. Toutefois, il y a plusieurs restrictions. Les applications doivent utiliser les mêmes codecs audio et vidéo. De ce fait, il est préférable que tous les utilisateurs possèdent la même application cliente de vidéo-conférence.

Sécurité

La sécurité pour le protocole H.323 n'est pas toujours appliquée. Certaines applications clientes l'utilisent et d'autre non. Par contre, les applications les plus récentes supportent presque tout l'aspect de sécurité du protocole H.323. Souvent l'aspect de sécurité est seulement utilisé pour assurer l'enregistrement auprès du « gatekeeper ». Le flux de donnée audio/vidéo n'est souvent pas sécurisé. Par contre, plusieurs applications ont leur propre système de cryptages pour les flux de données audio/vidéo.

Fonctionnalités supplémentaires

Le protocole H.323 permet l'envoi de document séparément du flux de données audio/vidéo. Certaines applications implémentent cet aspect.

2.7.2 Types de MCU (multipoint control unit) disponibles

Logiciel libre

OpenMCU est le meilleur MCU libre. Très utilisé parce qu'il est évidemment gratuit. C'est un MCU qui est assez performant. Il nécessite un serveur qui est sous Linux. Le serveur doit être assez performant si on veut y connecter un grand nombre d'utilisateurs.

Logiciel Payant

-Solution de Polycom
-Solution de Radvision
-Solution de Codian

Ces 3 compagnies sont orientées sur la vidéoconférence. Leurs produits sont développés depuis plusieurs années et sont très performants. Elles offrent plus de performance que OpenMCU. Elles sont très utiles si un très grand nombre de personnes sont connectées au serveur MCU.

Solution matérielle

Plusieurs compagnies offrent des solutions matérielles qui sont les solutions les plus performantes et les plus sécurisés. Certaines de ces solutions intègrent même le gatekeeper.

Plusieurs compagnies très connues ont développé des MCU matérielles. Cette solution est très coûteuse, ce genre de système coûte de 3000$ jusqu'à 20 000$.

Parmi les compagnies qui développent des MCU matérielles, on note la présence de Cisco et de Nortel. Vous trouverez ci-bas quelques modèles de MCU offert par ces compagnies.
- Cisco IPVC 3511 MCU
- Cisco IPVC 3540 MCU
-Nortel link 3000 MCU

Note : Il en existe plusieurs autres.

2.7.3 Types de Gatekeepers disponibles

Logiciel libre

OpenGk est le meilleur gatekeeper libre. Il est un des plus utilisés, car il est gratuit. C'est un gatekeeper qui offre un bon niveau de performance. Il existe seulement pour les serveurs Linux.

Logiciel payant

Il existe certaines compagnies qui ont conçu des gatekeeper, mais ces solutions sont payantes. Ces logiciels sont très performants lorsqu'ils sont installés sur un serveur qui peut soutenir la charge de travail. Les 3 compagnies principales sont les mêmes qui ont créé les MCU. Il est recommandé d'utiliser ces Gatekeeper avec un MCU provenant de la même compagnie.

-Solution de Polycom
-Solution de Radvision
-Solution de Codian

Solution matérielle

Il y a quelques compagnies qui offrent des solutions matérielles. Ces solutions sont souvent les plus performantes lorsqu'elles sont bien installées. Ce type de matériel est très coûteux et se chiffre en millier de dollars.

- Cisco 2500
- Cisco 2600
- Cisco 7200

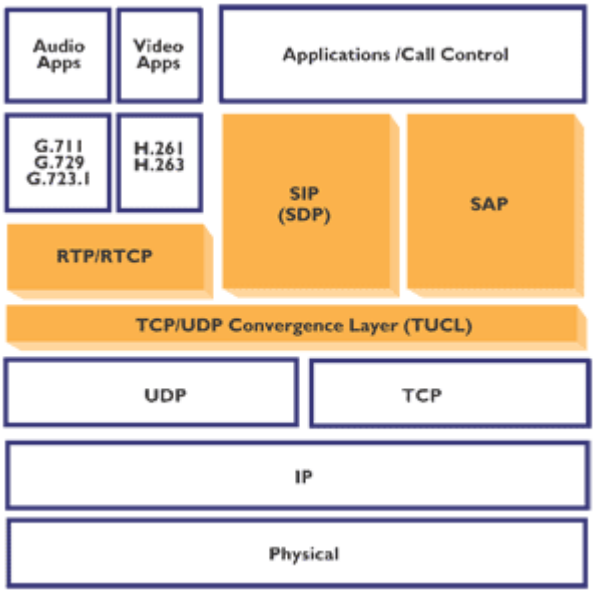
Note : Il en existe plusieurs autres.

2.7.4 Le protocole SIP (Session Initiation Protocol)

SIP est un protocole standard ouvert de télécommunications multimédias (son, image, etc.). Il est en 2007 le plus courant pour la téléphonie par internet (la VoIP). Il faut savoir que les applications clientes utilisant ce protocole sont appelées « user agent » (UA).

Le SIP n'est donc pas seulement destiné à la VoIP, mais aussi de nombreuses autres applications telles que la visiophonie, la messagerie instantanée, la réalité virtuelle ou même les jeux vidéo.

Session Initiation Protocol (dont le sigle est SIP) est un protocole normalisé et standardisé qui a été conçu pour établir, modifier et terminer des sessions multimédias. Il se charge de l'authentification et de la localisation des multiples participants. Il se charge également de la négociation sur les types de média utilisables par les différents participants en encapsulant des messages SDP (Session Description Protocol). SIP ne transporte pas les données échangées durant la session comme la voix ou la vidéo. SIP étant indépendant de la transmission des données, tout type de données et de protocoles peut être utilisé pour cet échange. Cependant, le protocole RTP (Real-time Transport Protocol) assure le plus souvent les sessions audio et vidéo. SIP remplace progressivement H.323.



Élément requis
Pour pouvoir utiliser l'application cliente correctement l'utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d'une Webcam pour la partie vidéo.

Pour supporter une communication à plus que deux utilisateurs, il est nécessaire d'installer un serveur de communication, qui est l'équivalent du serveur MCU avec le protocole H.323.

Plateforme supportée
La plupart des clients sont disponibles pour toutes les plateformes : (Windows, Linux, BSD, Mac OS). Cependant, la plupart de ces applications supportent seulement l'audio.

Convivialité
Le protocole SIP étant très récent, les applications le sont d'autant plus. Donc, on se retrouve avec des applications modernes qui sont vraiment conviviales. Ces applications sont faciles à installer, faciles à configurer et surtout faciles à utiliser.

Interopérabilité
Du fait que le protocole SIP est relativement récent, on se retrouve avec quelques problèmes de compatibilités entre les différents clients. Ces problèmes sont surtout causés parce que les codecs audio/vidéo sont différents d'un logiciel à un autre. Le serveur de communication n'accepte souvent qu'un type de client pour les problèmes de codecs mentionné ci-haut.

Sécurité
SIP supporte les registrations sécurisées. Par contre, il n'y a pas beaucoup d'application cliente qui sécurise le flux de données audio/vidéo.

Fonctionnalités supplémentaires
Le protocole SIP intègre du « peer to peer » pour l'échange de document. Plusieurs applications clientes incluent le « peer to peer ». De plus, plusieurs serveurs de communication supportent le partage d'application et de document.

2.8 Solutions avec le protocole H323

2.8.1 CU-SeeMe de l'université de Cornell

Le logiciel a été développé au départ sur Macintosh par l'université de Cornell, aux États-Unis à partir d'une commande publique de la National Science Fondation (Le CNRS américain). En 1994, le logiciel est déjà fonctionnel, et la NASA retransmet ses missions spatiales en direct sur son réflecteur. Ce logiciel est le plus ancien que l'on peut retrouver sur Internet.

Élément Requis

Pour pouvoir utiliser l'application cliente correctement l'utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d'une Webcam pour la partie vidéo.

De plus, chaque personne voulant faire parti de la conférence doit avoir préalablement installé le client sur son poste. Il doit aussi se connecter sur un serveur MCU qui lui est lié à un gatekeeper.

Plateforme supportée

Toutes les versions de Windows sont supportées, Mac Os et Linux.

Convivialité

Ce logiciel de vidéoconférence est développé depuis 1994, donc il a déjà passé par plusieurs versions. Il a été grandement amélioré depuis sa première version qui était seulement en noir et blanc. Ce logiciel est très facile à installer. Il peut être utilisé par des débutants qui ne s'y connaissent pas vraiment dans le domaine. Après quelques minutes, ils vont se retrouver facilement dans ce logiciel.

Interopérabilité

Ce logiciel n'est pas interopérable.

Sécurité

L'audio et la vidéo sont cryptés.

Fonctionnalités supplémentaires

Il permet de discuter textuellement dans une fenêtre distincte. Par contre, n'offre aucun partage de document et d'application.

2.8.2 Microsoft NetMeeting

NetMeeting est un des premiers logiciels de vidéoconférence, il a passé à travers de multiples versions.

Élément requis

Pour pouvoir utiliser l'application cliente correctement l'utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d'une Webcam pour la partie vidéo.

De plus, NetMeeting est intégré dans tous les systèmes d'exploitation Windows, donc il n'est pas nécessaire d'installer le client.
Pour pouvoir faire une conversation à plusieurs il est nécessaire d'installer un serveur MCU et un GateKeeper.

Plateforme supportée

NetMeeting fonctionne très bien avec tous les systèmes d'exploitation de Microsoft. Il fonctionne avec Windows 95 jusqu'à Windows Vista.

Convivialité

Les multiples versions de ce logiciel en font un logiciel assez complet et surtout très simple à utiliser. Un nouvel utilisateur s'y retrouve sans trop de difficulté.

Interopérabilité

NetMeeting n'est pas vraiment interopérable. Il peut communiquer via un serveur MCU avec GnomeMeeting qui lui est un logiciel sous Linux.

Sécurité

La sécurité est améliorée par le cryptage des données, l'authentification des participants et l'ajout d'un mot de passe.

Fonctionnalités supplémentaires

On y retrouve plusieurs fonctionnalités intéressantes comme le chat textuel, l'envoi de documents, le partage d'application et le mode tableau blanc. Le partage d'application permet à un utilisateur distant d'effectuer des opérations sur l'ordinateur de la personne avec laquelle elle communique. De plus, le mode tableau blanc permet aux personnes qui communiquent de dessiner sur un même tableau et d'en voir le résultat. Il est intéressant de savoir qu'il y a une version française de NetMeeting qui est disponible.

2.8.3 GnomeMeeting

GnomeMeeting est aujourd'hui la solution la plus simple et la plus aboutie pour pouvoir communiquer sous Linux, que ce soit par la voix, par l'image et par le texte même si son but premier est quand même la vidéoconférence. La communication avec une personne de votre choix ou avec plusieurs est aussi possible. L'affichage en plein écran est aussi possible (SDL indispensable). Ceci rend l'utilisation vraiment très agréable.

Élément requis

Pour pouvoir utiliser l'application cliente correctement l'utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d'une Webcam pour la partie vidéo.

De plus, on doit installer le client sur chacun des postes utilisateurs qui veulent joindre une conférence. Pour pouvoir faire une conversation à plusieurs il est nécessaire d'installer un serveur MCU et un GateKeeper.

Plateforme supportée

Linux seulement.

Convivialité

Très facile à utiliser. Toute la configuration peut s'effectuer à la souris, ou presque, vos éléments (cartes sons et webcam) sont normalement reconnus automatiquement, vous n'avez pratiquement rien à faire, sauf peut-être régler la qualité de l'image et du son.

Interopérabilité

GnomeMeeting n'est pas vraiment interopérable. Il peut communiquer via un serveur MCU avec NetMeeting qui lui est un logiciel sous Windows.

Sécurité

L'audio et la vidéo sont cryptés.

Fonctionnalités supplémentaires

Il n'est pas possible de faire du partage de document et d'application. Par contre, Il est possible d'effectuer une capture de l'image de votre correspondant (au format png).

2.8.4 Click to Meet

Click to Meet est une solution logicielle client/serveur qui permet à des groupes dispersés de se réunir en ligne et de collaborer de manière plus efficace à partir de leurs postes de travail. Click to Meet combine multipoint audio, vidéo et partage de données dans une seule et même application accessible depuis un simple navigateur web.

Élément requis

Pour pouvoir utiliser l'application cliente correctement l'utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d'une Webcam pour la partie vidéo. Chaque utilisateur voulant faire partie de la conférence devra préalablement installer un plug-in qui se joint à Internet Explorer. Il faut aussi installer le serveur MCU construit exclusivement pour Click to Meet.

Plateforme supportée

Seulement Windows.

Convivialité

Du fait que ce logiciel est intégré à Internet Explorer, il est facile de s'y retrouver. Les fonctionnalités sont très intuitives. Ainsi, les nouveaux utilisateurs s'y retrouveront sans aucun problème.

Interopérabilité

Il est possible de joindre la conférence à partir d'un téléphone si le GateKeeper est configuré en ce sens.

Sécurité

L'utilisateur doit préalablement se connecter pour joindre une conférence et pour aussi utiliser certaines fonctionnalités.

Fonctionnalités supplémentaires

Cette application utilise en arrière-plan le logiciel NetMeeting qui peut donc utiliser le partage de document et d'application. Il est aussi possible d'enregistrer une conférence.

2.9 Solutions avec le protocole SIP

2.9.1 Marratech

Le client *Marratech* est un logiciel gratuit, facile à installer sur votre ordinateur. *Marratech* donne accès à un environnement de travail en groupe qui offre une transmission voix sur IP haute fidélité, un tableau blanc interactif, la possibilité de partager des informations et des documents, de parler et de converser en groupe ou en privé et si on le désire, l'opportunité de se voir mutuellement à l'aide de caméras web.

Élément requis

Pour pouvoir utiliser l'application cliente correctement l'utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d'une Webcam pour la partie vidéo. Toutes les personnes voulant faire partie d'une conférence doivent posséder le client Marratech. Il faut savoir que Marratech a conçu son propre serveur de communication qui est l'équivalent d'un MCU. Donc, il est nécessaire d'installer ce serveur de communication pour rendre le tout fonctionnel.

Plateforme supportée

On retrouve le client pour Linux, Windows, Mac et solaris.
On retrouve le serveur pour Linux, Windows, Mac et solaris.

Convivialité

La compagnie Marractech a entièrement construit son application ce qui la rend riche en fonctionnalité. Malheureusement, l'accès aux fonctions n'est pas toujours très intuitif. Ce logiciel est très simple d'installation.

Interopérabilité

Pas disponible présentement.

Sécurité

Une protection (256 bits AES) est installée par défaut. Cette sécurité protège l'authentification, la voix, la vidéo et le mode tableau blanc.

Fonctionnalités supplémentaires

Il est possible de partager des documents et des applications. De plus, il y a un chat textuel qui est intégré à Maratech. Les utilisateurs peuvent aussi enregistrer les flux audio et vidéo.

2.9.2 Skype

C'est actuellement le logiciel de vidéoconférence le plus utilisé sur Internet. La qualité du son est excellente dans ce logiciel.

Élément requis

Pour pouvoir utiliser l'application cliente correctement l'utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d'une Webcam pour la partie vidéo.

De Plus, les utilisateurs voulant faire parti d'une même conversation doivent tous préalablement faire l'installation du client. Il est malheureusement impossible d'installer un serveur Skype sur son réseau. Les serveurs Skype sont publics et sont sur le réseau de la compagnie. De ce fait, il n'est pas possible de détourner le processus de connexion de Skype vers des serveurs privés. Ainsi, il ne peut pas être utilisé avec un MCU, ce qui rend cette solution peu intéressante.

Plateforme supportée

Windows, Max Os, Linux

Convivialité

Il est très facile d'installation et d'utilisation, c'est sûrement pourquoi il est utilisé par autant de personnes dans le monde. Il s'utilise facilement avec les pare-feu et les routeurs. Il est disponible en plusieurs langues, dont le français.

Interopérabilité

Il n'est pas interopérable du fait que les clients Skype peuvent seulement se connecter aux serveurs de la compagnie.

Sécurité

L'audio et la vidéo sont cryptés.

Fonctionnalités supplémentaires

On peut y transférer des fichiers, mais il n'est pas possible de faire du partage d'application. On doit aussi noter qu'il est possible d'appeler sur un téléphone conventionnel à l'aide de Skype.
Il est facile d'installation et d'utilisation. Il s'utilise facilement avec les pare-feu et les routeurs. Il est disponible en plusieurs langues, dont le français. Il est possible d'appeler sur des téléphones conventionnels. On peut y transférer des fichiers.

2.9.3 Breeze Meeting

Breeze Meeting est un logiciel de vidéoconférence créé pas la compagnie macromedia. Ce logiciel fonctionne avec un fureteur web. Il peut être intégré à presque tous les fureteurs web parce qu'il utilise le plug-in flash.

Élément requis

Pour pouvoir utiliser l'application cliente correctement l'utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d'une Webcam pour la partie vidéo. Tous les participants à une conférence doivent préalablement installer le plug-in flash de macromedia. Il faut aussi installer le plug-in breeze qui s'insèrera dans le fureteur web de votre choix. De plus, il faut installer un serveur de communication créé exclusivement pour Breeze Meeting.

Plateforme supportée

Le client peut être installé sur tous les systèmes d'exploitation Windows qui ont un fureteur supportant flash. Mac OS X avec mozilla ou safari avec l'application flash d'installée.

Le serveur de communication peut seulement être installé sur Windows 2000 serveurs ou Windows 2003 serveurs.

Convivialité

L'application est très conviviale et offre des caractéristiques intéressantes. Toutes les fonctionnalités sont très intuitives à utiliser.

Interopérabilité

Aucune interopérabilité actuellement présente.

Sécurité

Il est possible d'activer le cryptage des données audio et vidéo avec SSL (Secure Sockets Layer).

Fonctionnalités supplémentaires

Il est possible de faire le partage de document et d'application lorsque ces options sont activées sur le serveur de communication. Il est aussi possible de faire l'enregistrement d'une conférence.

2.9.4 AccessGrid

Ce logiciel est un ensemble de ressource multimédia de large format pour l'affichage. Il inclut plusieurs fonctionnalités pour les présentations et un environnement interactif.

Élément requis

Pour pouvoir utiliser l'application cliente correctement l'utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d'une Webcam pour la partie vidéo. On peut utiliser un projecteur parce que ce logiciel offre des options d'affichage sur les projecteurs.

De plus, il est nécessaire d'installer le client d'AccessGrid sur tous les postes qui veulent faire parti d'une conversation. On doit aussi installer le serveur d'AccesGrid pour que les clients puissent s'y connecter et faire une conférence.

Plateforme supportée

L'application cliente supporte :
Windows, Max Os, Linux et certaines variantes de BSD.
L'application serveur supporte :
Windows 2000 serveurs ou Windows 2003 serveurs.

Convivialité

L'installation du client est relativement facile. Ce logiciel est facilement configurable pour utiliser un projecteur. Les fonctionnalités sont assez intuitives pour être utilisé par des personnes non expérimentées.

Interopérabilité

Il est possible de se connecter au serveur avec d'autres types de client que celui qui a été développé par la compagnie.

Sécurité

Les certificats sont utilisés pour ce logiciel ce qui ajoute un certain niveau de sécurité. Par contre, par défaut L'audio et la vidéo ne sont pas cryptés.

Fonctionnalités supplémentaires

Il y a plusieurs fonctionnalités d'intégrer dans ce logiciel. La conversation textuelle est implémentée dans AcessGrid. Il est possible de partager des fichiers. On peut aussi y faire jouer des vidéos en direct.

2.9.5 Intelligere

C’est un logiciel de vidéoconférence qui fonctionne dans un fureteur Internet et qui a été conçu pour l’enseignement en ligne. Ce logiciel permet à plusieurs étudiants de visionner le cours d’un enseignant. Cette application est open source.

Élément requis

Pour pouvoir utiliser l’application cliente correctement l’utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d’une Webcam pour la partie vidéo.

Un serveur est installé et les clients se connectent à l’aide de leur fureteur web.

Plateforme supportée

Win2k et WinXp

Convivialité

Très facile à utiliser, on s’y retrouve vraiment facilement. C’est un logiciel assez complet et il reste tout de même très instinctif. Après quelques minutes d’utilisations, on connaît presque toutes les fonctionnalités de ce logiciel.

Interopérabilité

Aucune, du fait que ce logiciel est conçu pour fonctionner seulement avec des clients web.

Sécurité

Toutes les données transférées sont cryptées et il supporte MD5 pour l’authentification.

Fonctionnalités supplémentaires

C’est une application assez complète et surtout très intéressante. Permet le partage d’application, visionner des power points, mode tableau blanc, mode de question pour étudiant, plusieurs options de sécurités.

2.9.6 DIMDIM

C’est une petite révolution dans le monde de la vidéoconférence en ligne, surtout que ce logiciel est explicitement conçu pour faire des cours en ligne. Il permet d’avoir entre 0 et 120 internautes suivant le cours en ligne. Dimdim semble à terme avoir un gros potentiel, d’autant plus que des paquetages d’installation existent pour Moodle et s’installent facilement, comme des modules. Par contre, la jeunesse de ce projet se ressent au niveau des fonctionnalités : la vidéo et le son ne sont partageables que dans le sens professeur vers les étudiants et le partage d’applications. Cependant, il intègre plusieurs fonctionnalités, ainsi que le support des systèmes MacOS et Linux.

Élément requis

Pour pouvoir utiliser l’application cliente correctement l’utilisateur doit disposer pour la partie audio (des haut-parleurs ou écouteur, un microphone, une carte de son). Il doit aussi disposer d’une Webcam pour la partie vidéo.

Pour monter votre site de formation, il vous faudra: laroline : une application web qui permet de gérer des formations et créé le lien vers le serveur Dimdim. Dimdim : une application pour créer le serveur qui fait l'interface entre le diffuseur et les visionneurs. Le client a seulement besoin du « flash player ».

Plateforme supportée

Tous les systèmes d’exploitation ayant un fureteur récent.

Convivialité

Très conviviale et facile à utiliser, l’étudiant n’a rien besoin de faire, seulement se connecter avec son fureteur web.

Interopérabilité

Aucune

Sécurité

Toutes les données transférées sont cryptées.

Fonctionnalités supplémentaires

C’est l’application la plus complète que j’ai trouvée jusqu’à maintenant. Permet le partage d’application, visionner des PowerPoint, mode tableau blanc, mode de question pour étudiant, plusieurs options de sécurités. Cette application est open source.

2.10 Technologie actuellement en développement

Nous allons discuter de ce qui est développé actuellement et qui pourra ainsi être considéré dans le futur. Il y a plusieurs fonctionnalités et avancées technologiques qui sont présentement en développement. Nous allons discuter de ce qui se pointe à l'horizon dans un avenir très rapproché.

Plusieurs systèmes actuellement en développement ne se concentrent pas seulement sur l'audio, la vidéo, le chat textuel et le partage de fichiers. Ils intègrent des fonctionnalités de collaboration entre les membres d'une même discussion en temps réel. Ainsi, il est possible de travailler dans Word, PowerPoint, Excel, Access tous en même temps. Dans un mode de coopération qui comprend certaines restrictions pour que le tout se passe sans problème.

Présentement, il n'y a pas beaucoup de logiciel qui permet d'intégrer plusieurs caméras pour un participant d'une conversation. Il est important de savoir que plusieurs compagnies développent actuellement des produits qui vont permettre d'utiliser plus d'une caméra pour un utilisateur. Ce qui devrait ajouter un peu de concurrence sur le marché et ainsi faire réduire les prix parce qu'actuellement il n'y a pas beaucoup de logiciels qui offrent cette fonctionnalité.

De plus, certaines compagnies sont présentement en cours de développement de nouveaux codec qui vont intégrer le support haute définition pour les protocoles H323 et SIP. De ce fait, les conversations vont être plus claires et plus précises. Actuellement on est un peu limité dans la qualité de l'image, mais avec ces nouvelles avancées technologiques, il sera possible d'atteindre un bon niveau de qualité de l'image. Ces nouveaux codec intégreront aussi une bien meilleure qualité sonore.

On tente aussi actuellement de développer des nouveaux codec qui rendre le flux de données audio/vidéo plus léger. Le problème actuellement est lorsqu'on est plusieurs dans une conversation, on doit restreindre la zone d'affichage de l'image de chacun parce qu'il y a trop de données, ce qui est exigeant sur la bande passante d'une connexion.

2.11 Conclusion

Nous vous avons présenté ci-haut, plusieurs logiciels qui offrent des possibilités différentes de vidéoconférence pour son éventuelle utilisation comme plateforme d'enseignement à distance. Certains de ces logiciels sont très coûteux. Il y a aussi des solutions gratuites qui s'en sortent assez bien.

Premièrement, le logiciel que nous avons préféré est Skype. C'est un logiciel qui est gratuit et très facile d'utilisation. Malheureusement, il est actuellement impossible d'installer un serveur MCU à l'université et de s'y connecter. Skype exige une connexion sur ses propres serveurs. Donc, on peut éliminer cette solution pour le moment.

La solution gratuite que nous avons le plus appréciée est celle utilisant NetMeeting. Cette solution fonctionnait très bien et n'était pas vraiment difficile à installer. De plus, elle est intégrée à tous les systèmes Windows.

Donc, nous proposons l'utilisation de DIMDIM qui est spécialement conçu dans le cadre de l'enseignement en ligne. De plus, il s'intègre à Moodle ce qui le rend d'autant plus intéressant. C'est un logiciel open source vraiment très complet. C'est le logiciel de vidéoconférence le plus complet qui m'a été donné de tester. Il intègre plusieurs fonctionnalités très utiles pour l'enseignement en ligne. Il est de loin le logiciel que j'ai préféré. De plus, il permet à l'enseignant d'afficher ce qu'il voit à l'écran. Il est facile à utiliser pour le professeur et les étudiants.

De plus, nous croyons que les logiciels payants ont actuellement une longueur d'avance sur les autres. La qualité de l'image est dans plusieurs cas meilleure et offre surtout plus de fonctionnalité intéressante dans le cadre de l'enseignement en ligne. Marratech est un logiciel très performant qui pourrait être utilisé pour l'enseignement en ligne. Mais malheureusement, ce logiciel a un coût assez élevé.

Il ne faudrait pas mettre de côté le MCU matériel et le Gatekeeper matériel. Dans le cas où les solutions libres ne puissent pas subvenir à la demande d'un grand nombre d'utilisateurs connectés simultanément, il serait souhaitable de se tourner vers ces solutions. Malheureusement, ce type de matériel est très dispendieux.

3. Implémentation du logiciel

3.1 Langage de programmation utilisé

Nous avons choisi le langage c++, car c'est le langage dans lequel nous avons le plus de connaissances. De plus, les librairies que nous avons choisies pouvaient être intégrées par le langage c++. Le langage Java ne suivait pas de très loin, il contenait des librairies de vidéoconférence très intéressante, mais celle pour le c++ nous semblait plus simple à intégrer. Ensuite, nous ne voulions pas avoir à installer une machine virtuelle pour faire fonctionner notre application, le moins d'installation possible serait favorable à l'utilisateur.

3.2 Environnement de développement utilisé

Nous avions plusieurs choix sous la main. Nous avions des licences pour Borland c++ Builder 5 et pour Microsof visual studio .net. Les deux nous semblaient assez complets pour créer notre application. Ces deux environnements de développement sont très bien conçus et disposent de plusieurs fonctionnalités de débogage. Notre choix s'est finalement arrêté sur Borland c++ Builder 5 parce qu'il permettait de créer des fenêtres facilement. Nous n'aurions pas à tout programmer pour créer nos fenêtres. Tout ce qui était visuel s'intègre facilement avec ce compilateur. Ce qui à nos yeux, lui conférait un avantage sur le produit Microsoft.

3.3 Plateforme utilisée

- L'application cliente sera exécutée sous la plateforme Windows XP parce que Windows est la plateforme la plus utilisée actuellement et notre but est de donner accès à notre logiciel pour le plus de gens possible.

- L'application serveur sera exécutée sous Windows pour garder un certain standard avec l'application cliente qui s'y trouve déjà. Dans ce cas nous allons pouvoir utiliser les mêmes librairies pour les applications cliente et serveur. Nous avons fait ce choix pour nous permettre de faire une certaine réutilisation du code entre les applications cliente et serveur.

3.4 Les librairies existantes

Dans le cadre de notre projet, nous avons analysé les différents composants pouvant être intégrés à un logiciel de vidéoconférence. Nous vous présentons les différents composants que nous avons trouvés.

Nous avons fait le choix de la librairie **Vfw32**. C'est une librairie qui est utilisée par les systèmes d'exploitation Windows. Elle contient beaucoup de fonctions qui sont utilisées pour le multimédia de ces systèmes d'exploitation.

Tout d'abord, notre choix s'est arrêté sur cette librairie parce qu'elle contenait plusieurs fonctions que nous avons besoins, certaines fonctions pour la webcam et d'autre pour le son. De plus, cette librairie est très simple à utiliser et elle était très bien documentée sur le site de Microsoft.

De plus, nous utilisons la bibliothèque **VCL** de borland. Elle nous permet d'utiliser plusieurs composants visuels, comme les boutons et les zones de texte. Ce ne sont pas seulement des composants visuels, ce sont des composants très faciles à utiliser, on doit seulement les glisser sur la fiche pour les implémenter. Aussi, nous utilisons les composants de « socket » provenant de cette même bibliothèque. Ils sont utilisés pour notre application cliente et notre application serveur. Nous utilisons ces « sockets » pour faire de la transmission de données entre le client et le serveur.

Socket : Pour communiquer entre deux applications ou ordinateurs, il vous faut un téléphone, un socket. Un socket est attaché à un port (une porte au sens imagé). Vous ouvrez votre porte et attendez qu'un colis soit acheminé à celle-ci. Une fois le colis reçu, vous pouvez soit envoyer un autre colis ou bien fermer la porte, le port. Vous devez initialiser le socket, faire un lien avec le port, attendre pour un paquet, et fermer le socket. Pour avoir plus d'informations sur les protocoles de transfert TCP/IP ou bien sur les sockets en général, voir la section Références.

Le composant **indy** : C'est un composant qui contient tout ce qu'on a besoin pour capturer de la vidéo et il s'intègre à borland builder. Nous n'avons pas pris ce composant, ceux intégré à borland builder nous satisfaisait amplement pour ce que nous voulions faire.

3.5 Les Tests

Applications cliente :
Plateforme : Windows XP
Mémoire vive : 2 go
Processeur : core 2 duo E4500
Carte de son : Sound blaster audigy ZS
Webcam : Logitech clicksmart 510

Plateforme : Windows XP
Mémoire vive : 1go
Processeur : AMD Athlon 2500+
Carte de son : Onboard
Webcam : Ezonics

Nous avons fait tourner l'application cliente sur ces 2 systèmes avec la webcam allumée plusieurs fois pour une période d'environ 24 heures. Ce qui montre que lorsque l'application est en communication, elle est vraiment stable.

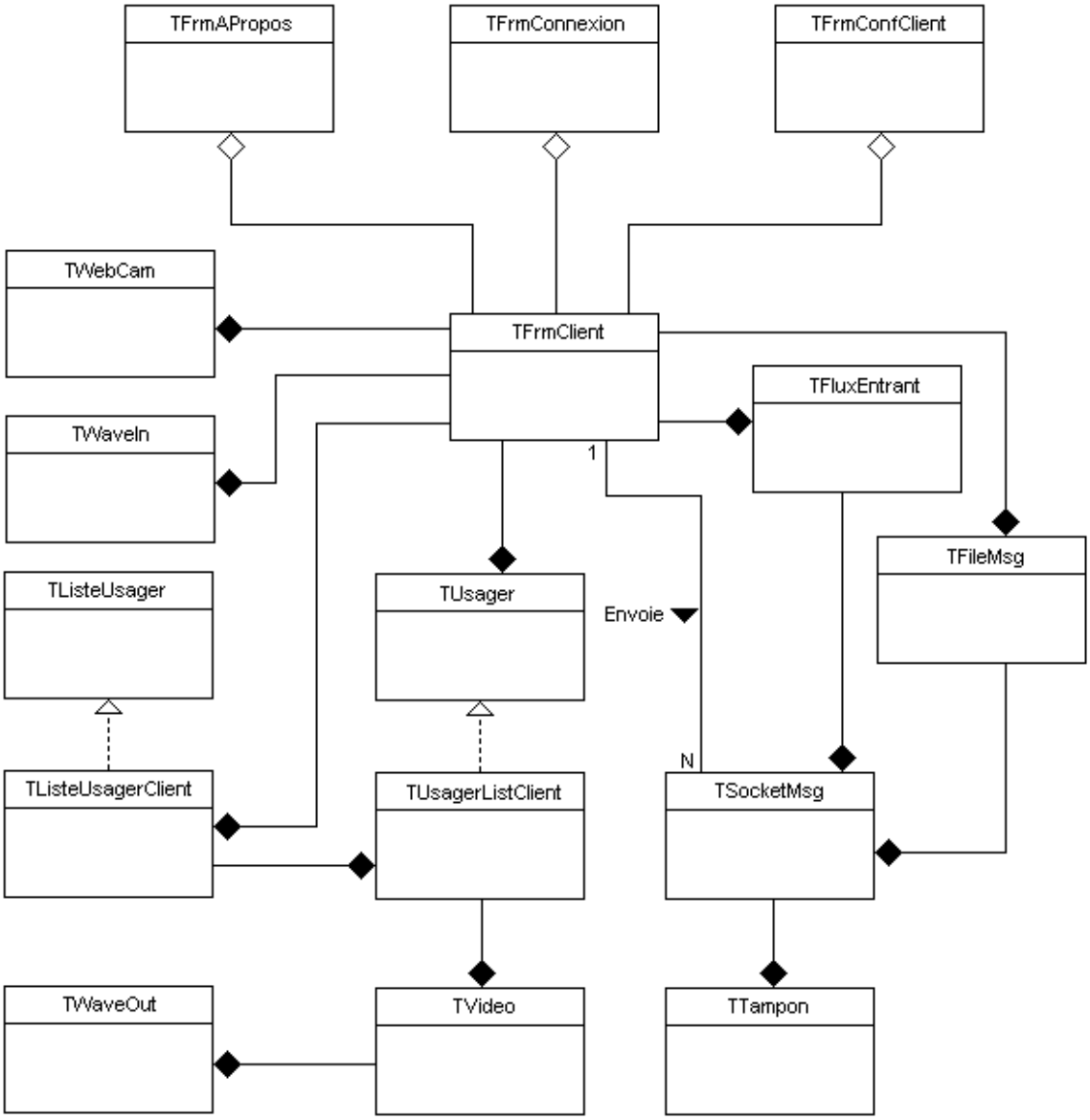
De plus, nous avons essayé de rendre l'application lente et à la limite qu'elle nous donne une erreur. Nous n'avons pas réussi. Nous avons fait fonctionner la webcam et envoyé du texte à répétition dans le « chat » textuel sans aucun problème. Nous en sommes venus à la conclusion que l'application était robuste sous Windows XP.

Application serveur :
Plateforme : Windows XP
Mémoire vive : 2 go
Processeur : core 2 duo E4500

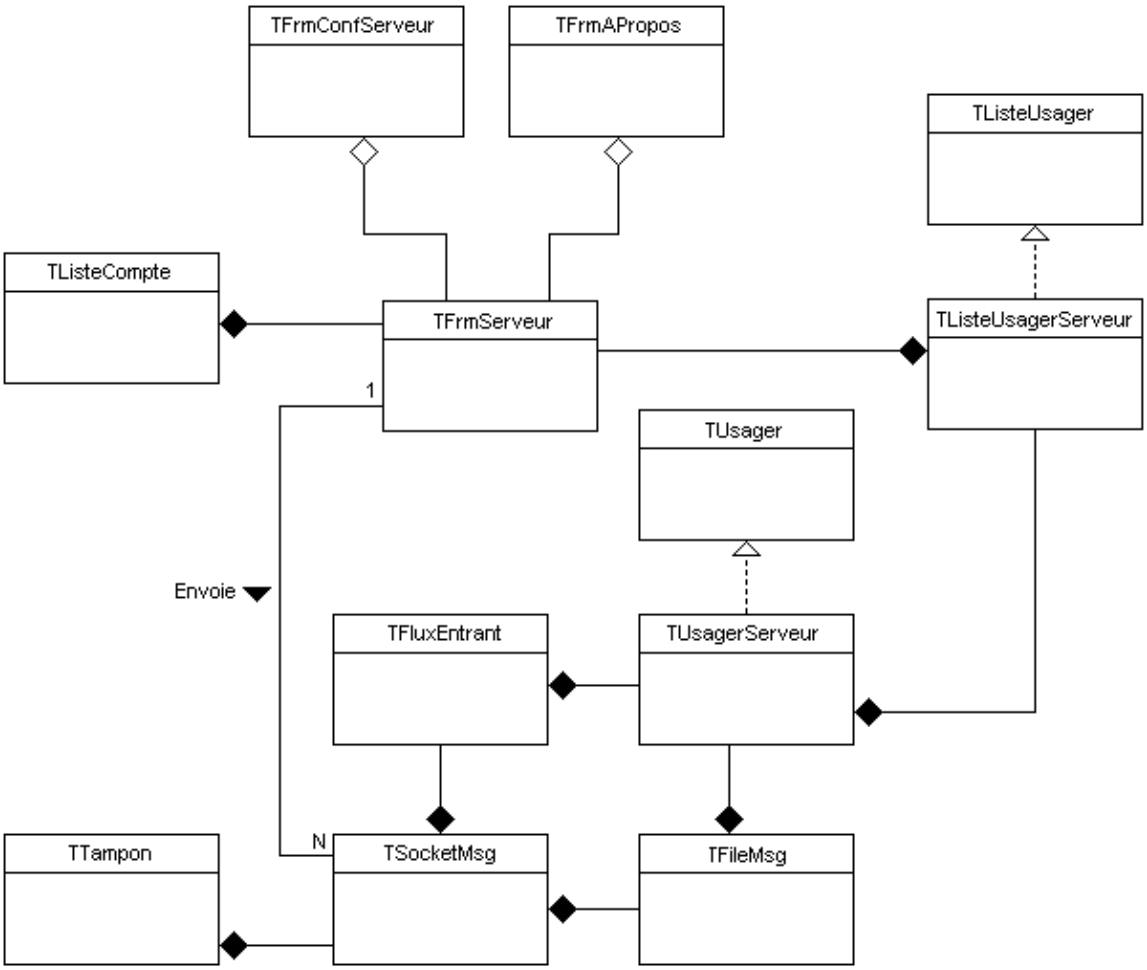
Plateforme : Windows XP
Mémoire vive : 1go
Processeur : AMD Athlon 2500+

Les tests de l'application serveur ont été effectués en même temps que ceux pour l'application cliente. Nous avons vraiment essayé de causer des problèmes à ces applications, sans aucun résultat. Nous pouvons donc affirmer que l'application serveur est robuste.

3.6 Fonctionnement interne de l'application cliente



3.7 Fonctionnement interne de l'application serveur



3.8 Guide des classes

TFileMsg
Fonction:
Cette classe permet de créer une file de messages qui essaie d'envoyer, par un socket, à intervalle régulier les messages qui y ont été ajoutés. Les messages sont retirés de la file s'ils ont pu être envoyés par le socket.
Méthotes:
Méthode: __fastcall TFileMsg(TCustomWinSocket* Socket); **Fonction:** Constructeur de la classe. **Paramètre(s):** Socket: Socket par lequel les messages seront envoyés. **Valeur de retour:** Aucune.
Méthode: __fastcall ~TFileMsg(); **Fonction:** Destructeur de la classe. **Paramètre(s):** Aucun. **Valeur de retour:** Aucune.
Méthode: void AddMsg(char* Donnees, int Longeur); **Fonction:** Permets d'ajouter dans la file, des messages qui devront être envoyés par le socket. **Paramètre(s):** Données : Message sous forme de données. Longueur : Longeur des données. **Valeur de retour:** Aucune.
Méthode: bool Vide(void); **Fonction:** Permets de savoir si la file est vide. **Paramètre(s):** Aucun. **Valeur de retour:** Renvoie **true** si la file est vide, sinon renvoie **false**.

TFluxEntrant

Fonction:

Classe fait la gestion d'un flux en permettant d'extraire les messages de celui-ci.

Méthotes:

Méthode:
TFluxEntrant();
Fonction:
Constructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
~TFluxEntrant();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void AjouterDonnees(void* Donnees, int LongeurDonnees);
Fonction:
Ajoute des données dans le flux en cours.
Paramètre(s):
Données : Données à ajouter dans le flux.
LongeurDonnees: Longeur des données à ajouter dans le flux.
Valeur de retour:
Aucune.

Méthode:
bool MessagePret(void);
Fonction:
Appeler cette méthode pour extraire un message du flux. Elle retroune **true** si un message à été extrait du flux. Elle retourne **false** dans le cas contraire. À chaque appel de cette fonction, le message courant est remplacé par le nouveau message.
Paramètre(s):
Aucun.
Valeur de retour:
true si un message a été extrait du flux. **false** dans le cas contraire.

Méthode:
TSocketMsg* GetMsgCourant(void);
Fonction:
Retourne le message courant extrait du flux par l'appel de la fonction MessagePret.

Paramètre(s):
Aucun.
Valeur de retour:
Message courant du flux de données.

TSocketMsg

Fonction:

Permets de créer un message pouvant être envoyé par un socket.

Méthotes:

Méthode:
TSocketMsg(int Type, int ExpediteurUID, void* Donnees, int LongeurDonnees);
Fonction:
Constructeur de la classe (utilisé si on veut construire un nouveau message).
Paramètre(s):
Type: Type de message.
ExpediteurUID: L'UID de l'expéditeur du message.
Données : Données du message.
LongeurDonnees: Longeur des données du message.
Valeur de retour:
Aucune.

Méthode:
TSocketMsg(void* Msg, int Longeur);
Fonction:
Constructeur de la classe (utilisé pour construire un message à partir d'un message reçu par un socket).
Paramètre(s):
Msg: Message sous forme d'un tableau d'octets.
Longueur : Longeur du message (en octet).
Valeur de retour:
Aucune.

Méthode:
~TSocketMsg();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void CopierMsg(void* Tampon, int LongeurTampon);
Fonction:
Copier les octets du message dans un tampon.
Paramètre(s):
Tampon: Tampon dans lequel on veut obtenir la copie du message.
LongeurTampon: Longeur du tampon.

Valeur de retour:
Aucune.

Méthode:
void* GetMsg(void);
Fonction:
Obtenir un pointeur vers le tampon d'octets du message.
Paramètre(s):
Aucun.
Valeur de retour:
Pointeur vers le tampon d'octets du message.

Méthode:
int GetLongeurMsg(void);
Fonction:
Obtenir la longueur, en octet, du tampon contenant le message.
Paramètre(s):
Aucun.
Valeur de retour:
Pointeur vers le tableau d'octets du message.

Méthode:
short GetDebut(void);
Fonction:
Obtenir les 2 premiers octets du message (les 2 premiers octets et les 2 derniers octets d'un message doivent être identiques, sinon le message n'est pas un message valide. Ces 2 pairs d'octets sont utilisés pour valider si c'est vraiment un message ou si le message est intact).
Paramètre(s):
Aucun.
Valeur de retour:
Un short qui représente les 2 premiers octets du message.

Méthode:
int GetType(void);
Fonction:
Obtenir le type du message.
Paramètre(s):
Aucun.
Valeur de retour:
Type du message.

Méthode:
int GetExpediteurUID(void);
Fonction:
Obtenir l'UID de l'expéditeur du message.
Paramètre(s):
Aucun.
Valeur de retour:
UID de l'expéditeur du message

Méthode:

int GetLongeurDonnees(void);

Fonction:

Obtenir la longueur des données du message.

Paramètre(s):

Aucun.

Valeur de retour:

Longueur des données du message.

Méthode:

void CopierDonnees(void* Tampon, int LongeurTampon);

Fonction:

Copier les données dans un tampon.

Paramètre(s):

Tampon: Pointeur vers un tampon dans lequel on veut copier les données.

LongeurTampon: Longueur du tampon.

Valeur de retour:

Aucune.

Méthode:

void* GetDonnees(void);

Fonction:

Obtenir un pointeur vers les données du message.

Paramètre(s):

Aucun.

Valeur de retour:

Pointeur vers les données du message.

Méthode:

short GetFin(void);

Fonction:

Obtenir les 2 derniers octets du message (les 2 premiers octets et les 2 derniers octets d'un message doivent être identiques, sinon le message n'est pas un message valide. Ces 2 pairs d'octets sont utilisés pour valider si c'est vraiment un message ou si le message est intact).

Paramètre(s):

Aucun.

Valeur de retour:

Un short qui représente les 2 derniers octets du message.

Méthode:

void SetExpediteurUID(int UID);

Fonction:

Changer le UID de l'expéditeur du message (les messages redirigés par le serveur aux clients devraient toujours appeler cette méthode pour savoir de quel usager provient le message).

Paramètre(s):

UID: Nouveau UID de l'expéditeur du message.

Valeur de retour:

Aucune.

Méthode:

AnsiString ToString(void);

Fonction:

Convertir le message en string.

Paramètre(s):

Aucun.

Valeur de retour:

Chaîne de caractères représentant le message.

Méthode:

FInfoUsager ToInfoUsager(void);

Fonction:

Convertir le message en FInfoUsager.

Paramètre(s):

Aucun.

Valeur de retour:

FInfoUsager représentant le message.

Méthode:

TJPEGImage* ToJpg(void);

Fonction:

Convertir le message en Image Jpg.

Paramètre(s):

Aucun.

Valeur de retour:

Image Jpg obtenue du message.

TTampon

Fonction:

Cette classe permet de créer un tampon de données qui est facilement manipulable.

Méthotes:

Méthode:

TTampon();

Fonction:

Constructeur de la classe.

Paramètre(s):

Aucun.

Valeur de retour:

Aucune.

Méthode:

~TTampon()

Fonction:

Destructeur de la classe.

Paramètre(s):

Aucun.

Valeur de retour:

Aucune.

Méthode:

void AddDonnees(void* Donnees, int Longeur);

Fonction:

Ajouter des données à la fin du tampon.

Paramètre(s):

Données : Pointeur vers un tampon contenant les données à ajouter.

Longueur : Longueur du tampon de données à ajouter.

Valeur de retour:

Aucune.

Méthode:

void* Get(int Position);

Fonction:

Obtenir un pointeur vers une position spécifiée dans le tampon de données.

Paramètre(s):

Position: Position de laquelle on veut obtenir un pointeur.

Valeur de retour:

Pointeur vers une position spécifiée dans le tampon de données.

Méthode:

void Supprimer(int Start, int Longeur);

Fonction:

Supprimer une partie du tampon à un endroit spécifié et d'une longueur spécifiée.

Paramètre(s):

Start: Position à partir de laquelle on veut supprimer des données dans le tampon.

Longueur : Nombre d'octets dont on veut supprimer dans le tampon.

Valeur de retour:

Aucune.

Méthode:

int GetLongeur(void);

Fonction:

Obtenir la longueur du tampon de données.

Paramètre(s):

Aucun.

Valeur de retour:

Longueur du tampon de données.

Méthode:

void CopierDonnnes(void* Donnees, int Longeur);

Fonction:

Copier des données du tampon à partir du début jusqu'à une longueur spécifiée.

Paramètre(s):

Données : Pointeurs vers un tampon de données pour copier des données du tampon.

Longueur : Longeur du tampon de données.

Valeur de retour:

Aucune.

TVideo

Fonction:

Cette classe permet de gérer les informations de la vidéo en cours (Son et image).

Méthotes:

Méthode:
TVideo();
Fonction:
Constructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
~TVideo();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void AjouterJpg(TJPEGImage* Jpg);
Fonction:
Remplace l'image courante de la vidéo.
Paramètre(s):
Jpg: Une image de type JPG représentant l'image courante de la vidéo en cours.
Valeur de retour:
Aucune.

Méthode:
TJPEGImage* GetJpg(void);
Fonction:
Obtenir l'image courante de la vidéo en cours.
Paramètre(s):
Aucun.
Valeur de retour:
Image de type JPG représentant l'image courante de la vidéo en cours.

Méthode:
void AjouterWave(char* Tampon, int LongeurTampon);
Fonction:
Ajouter un échantillon de type WAVE dans la vidéo en cours. L'échantillon est automatiquement ajouté à la file d'attente et sera joué quand les autres échantillons auront fini d'être joués.
Paramètre(s):
Tampon: Tampon de données contenant l'échantillon WAVE.

LongeurTampon: Longueur du tampon de données contenant l'échantillon.

Valeur de retour:

Aucune.

Méthode:

bool EnAffichage(void);

Fonction:

Connaître l'état de la vidéo en cours. Utilisé pour savoir si les images doivent être affichées. Elle retournera la valeur true si la fonction AjouterJpg a été au moins appelé une fois.

Paramètre(s):

Aucun.

Valeur de retour:

true si l'image doit être affichée, false dans le cas contraire.

Méthode:

void ArreterAffichage(void);

Fonction:

Arrêter l'affichage en cours. Après l'appel de cette méthode, la fonction EnAffichage retournera false tant et aussi longtemps que la fonction AjouterJpg n'aura pas été appelée.

Paramètre(s):

Aucun.

Valeur de retour:

Aucune.

Méthode:

bool EnMicro(void);

Fonction:

Connaître l'état du son de la vidéo.

Paramètre(s):

Aucun.

Valeur de retour:

true si le son doit être joué, false dans le cas contraire.

Méthode:

void DemarrerMicro(void);

Fonction:

Identifier que le logiciel joue le son du vidéo.

Paramètre(s):

Aucun.

Valeur de retour:

Aucune.

Méthode:

void ArreterMicro(void);

Fonction:

Identifier qu'on a terminé de jouer le son du vidéo.

Paramètre(s):

Aucun.

Valeur de retour:

Aucune.

TWaveIn

Fonction:

Cette classe permet de capturer le son en provenance du microphone de l'ordinateur.

Méthotes:

Méthode:
__fastcall TWaveIn();
Fonction:
Constructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
__fastcall ~TWaveIn();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void Start(void);
Fonction:
Démarrer la capture du son.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void Start(void);
Fonction:
Arrêter la capture du son.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void Reset(void);
Fonction:
Redémarrer la capture du son.
Paramètre(s):
Aucun.
Valeur de retour:

Aucune.

Méthode:
bool EstEnLecture(void);
Fonction:
Connaître l'état de la capture du son.
Paramètre(s):
Aucun.
Valeur de retour:
true si le son est capturé du micro, false dans le cas contraire.

TWaveOut

Fonction:

Cette classe permet de jouer des échantillons WAVE.

Méthotes:

Méthode:
__fastcall TWaveOut();
Fonction:
Constructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
__fastcall ~ TWaveOut ();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void __fastcall AjouterDonnees(char* Donnees, int LongeurDonnees);
Fonction:
Ajouter un échantillon de données WAVE à jouer. Les échantillons WAVE sont ajoutés dans une file d'attente. Un thread s'occupe de jouer successivement chacun des échantillons de la file.
Paramètre(s):
Données : Données de l'échantillon WAVE.
LongeurDonnees: Longueur des données de l'échantillon WAVE.
Valeur de retour:
Aucune.

TWebCam

Fonction:

Cette classe permet de capturer les images de la webcam.

Méthotes:

Méthode:
TWebCam(HWND hWnd, int DriverIndex, int Width, int Height,
void* VideoCallback);
Fonction:
Constructeur de la classe.
Paramètre(s):
hWnd: Handle de la fenêtre parent.
DriverIndex: Index du driver de la webcam dont on veut capturer les images.
Width: Largeur des images.
Height: Hauteur des images.
VideoCallback: Pointeur vers la fonction callback de récupération des images.
Valeur de retour:
Aucune.

Méthode:
~TWebCam();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void Start(void);
Fonction:
Démarrer la capture d'images.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void Stop(void);
Fonction:
Arrêter la capture d'images.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void GetJpg(TJPEGImage* Jpg);
Fonction:
Obtenir un pointeur sur la dernière image capturée.
Paramètre(s):
Pointeur qui obtiendra la dernière image capturée.
Valeur de retour:

Aucune.

Méthode:
void SetQualiteCompression(int Qualite);
Fonction:
Changer la qualité de compression des images capturées.
Paramètre(s):
Qualite: nouvelle qualité de compression des images (doit être entre 0 et 100).
Valeur de retour:
Aucune.

Méthode:
void SetDimension(int Width, int Height);
Fonction:
Changer la largeur et la hauteur de l'image.
Paramètre(s):
Width: nouvelle largeur des images capturées.
Height: nouvelle hauteur des images capturées.
Valeur de retour:
Aucune.

Méthode:
void SetPret(bool Pret);
Fonction:
Changer l'état pour dire qu'on a une nouvelle image d'arrivée.
Paramètre(s):
Pret: true pour dire qu'on a une nouvelle image, false dans le cas contraire.
Valeur de retour:
Aucune.

Méthode:
bool IsPret(void);
Fonction:
Obtenir l'état si une image est arrivée.
Paramètre(s):
Aucun.
Valeur de retour:
true si une image est arrive, false dans le cas contraire.

Méthode:
bool EnMarche(void);
Fonction:
Obtenir l'état pour savoir si la webcam est allumée.
Paramètre(s):
Aucun.
Valeur de retour:
true si la webcam est allumée, false dans le cas contraire.

Méthode:
int GetWidth(void);
Fonction:
Obtenir la largeur des images capturées.

Paramètre(s):
Aucun.
Valeur de retour:
Largeur des images capturées.

Méthode:
int GetHeight(void);
Fonction:
Obtenir la hauteur des images capturées.
Paramètre(s):
Aucun.
Valeur de retour:
Largeur des images capturées.

Méthode:
void SetDonnees(char* Donnees);
Fonction:
Donner les données de l'image capturée par la webcam (cette fonction est appelée par la fonction CallBack).
Paramètre(s):
Données : Données de l'image de la webcam.
Valeur de retour:
Aucune.

TListeUsager

Fonction:

Crer une liste d'usager.

Méthotes:

Méthode:
TListeUsager();
Fonction:
Constructeur de la classe. Appeler ce constructeur si la liste n'a pas besoin d'être dessinée (une liste non visible à l'usager).
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
~TListeUsager();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void RetirerParId(short Id);
Fonction:
Retire un usager de la liste par sa position dans la liste.
Paramètre(s):
Id: Position à laquelle on veut enlever un usager de la liste.
Valeur de retour:
Aucune.

Méthode:
void RetirerParUID(int UID);
Fonction:
Retire un usager de la liste par son numéro d'utilisateur unique.
Paramètre(s):
UID: Numéro d'utilisateur unique de l'usager dont on veut retiré de la liste.
Valeur de retour:
Aucune.

Méthode:
void Clear(void);
Fonction:
Efface tous les usagers de la liste.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
int Count(void);
Fonction:
Compter le nombre d'usagers dans la liste.
Paramètre(s):
Aucun.
Valeur de retour:
Nombre d'usagers dans la liste.

TUsager

Fonction:

Cette classe permet de créer des objets usager.

Méthotes:

Méthode:
TUsager(FInfoUsager Info);
Fonction:
Constructeur de la classe (utilisé quand on connaît les informations de l'usager).
Paramètre(s):
Info: Informations de l'usager.
Valeur de retour:
Aucune.

Méthode:
~TUsager();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void SetCompte(AnsiString Compte);
Fonction:
Changer le compte de l'usager.
Paramètre(s):
Compte: String qui représente le nouveau compte.
Valeur de retour:
Aucune.

Méthode:
void SetMotPasse(AnsiString MotPasse);
Fonction:
Changer le mot de passe de l'usager.
Paramètre(s):
MotPasse: String qui représente le nouveau mot de passe de l'usager.
Valeur de retour:
Aucune.

Méthode:
void SetPrenom(AnsiString Prenom);
Fonction:
Changer le prénom de l'usager.
Paramètre(s):
MotPasse: String qui représente le nouveau prénom de l'usager.
Valeur de retour:
Aucune.

Méthode:
void SetNom(AnsiString Nom);
Fonction:
Changer le nom de l'usager.
Paramètre(s):
MotPasse: String qui représente le nouveau nom de l'usager.
Valeur de retour:
Aucune.

Méthode:
void SetUID(int UID);
Fonction:
Changer l'UID de l'usager.
Paramètre(s):
MotPasse: String qui représente le nouveau UID de l'usager.

Valeur de retour:
Aucune.

Méthode:
void SetInfoUsager(FInfoUsager Info);
Fonction:
Changer toutes les informations de l'usager.
Paramètre(s):
Info: Structure qui représente les nouvelles informations de l'usager.
Valeur de retour:
Aucune.

Méthode:
AnsiString GetCompte(void);
Fonction:
Obtenir le compte de l'usager.
Paramètre(s):
Aucun.
Valeur de retour:
Compte de l'usager.

Méthode:
AnsiString GetMotPasse(void);
Fonction:
Obtenir le mot de passe de l'usager.
Paramètre(s):
Aucun.
Valeur de retour:
Mot de passe de l'usager.

Méthode:
AnsiString GetPrenom(void);
Fonction:
Obtenir le prénom de l'usager.
Paramètre(s):
Aucun.
Valeur de retour:
Prénom de l'usager.

Méthode:
AnsiString GetNom(void);
Fonction:
Obtenir le nom de l'usager.
Paramètre(s):
Aucun.
Valeur de retour:
Nom de l'usager.

Méthode:
int GetUID(void);
Fonction:
Obtenir l'UID de l'usager.

Paramètre(s):
Aucun.
Valeur de retour:
UID de l'usager.

Méthode:
FInfoUsager GetInfoUsager(void);
Fonction:
Obtenir toutes les informations de l'usager.
Paramètre(s):
Aucun.
Valeur de retour:
Informations de l'usager.

Méthode:
bool EstActive(void);
Fonction:
Obtenir l'état de l'usager. (L'usager n'est pas actif tant qu'il n'a pas envoyé ses informations au serveur. Un usager inactif ne peut pas recevoir de messages en provenance du serveur.).
Paramètre(s):
Aucun.
Valeur de retour:
true si l'usager est actif sinon false.

Méthode:
void SetActive(bool Active);
Fonction:
Changer l'état de l'usager (L'usager n'est pas actif tant qu'il n'a pas envoyé ses informations au serveur. Un usager inactif ne peut pas recevoir de messages en provenance du serveur.).
Paramètre(s):
Active: État de l'usager (true pour actif, false pour inactif).
Valeur de retour:
Aucune.

TUsagerListeServeur

Fonction:

Cette classe permet de créer une liste d'usager pour le serveur.

Méthotes:

Méthode:
TListeUsagerServeur();
Fonction:
Constructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
~TListeUsagerServeur();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
~TListeUsagerServeur();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void Ajouter(FInfoUsager Info, TCustomWinSocket* Socket);
Fonction:
Ajouter un usager à la liste d'usagers.
Paramètre(s):
Info: Information de l'usager à ajouter.
Socket: Socket associé à l'usager (pour l'envoi de message à cet usager).
Valeur de retour:
Aucune.

Méthode:
TUsagerServeur* GetParId(short Id);
Fonction:
Cherche un usager dans la liste d'usagers par son id.
Paramètre(s):
Id: Position de l'usager dans la liste.
Valeur de retour:
L'usager trouvé.

Méthode:
TUsagerServeur* GetParSocket(TCustomWinSocket* Socket);
Fonction:
Cherche un usager dans la liste d'usagers par son socket associé.
Paramètre(s):
Socket: Socket associé de l'usager.
Valeur de retour:
L'usager trouvé.

Méthode:
TUsagerServeur* GetParCompte(AnsiString Compte);
Fonction:
Cherche un usager dans la liste d'usagers par son compte.
Paramètre(s):

Compte: Compte de l'usager.
Valeur de retour:
L'usager trouvé.

Méthode:
TUsagerServeur* GetParUID(int UID);
Fonction:
Cherche un usager dans la liste d'usagers par son UID.
Paramètre(s):
UID: UID de l'usager.
Valeur de retour:
L'usager trouvé.

Méthode:
void RetirerParSocket(TCustomWinSocket* Socket);
Fonction:
Retire un usager de la liste d'usagers par son Socket associé.
Paramètre(s):
Socket: Socket associé de l'usager.
Valeur de retour:
Aucune.

Méthode:
void EnvoyerATous(TSocketMsg* Msg);
Fonction:
Envoyer un message à tous les usagers de la liste.
Paramètre(s):
Msg: Message à envoyer.
Valeur de retour:
Aucune.

Méthode:
void EnvoyerAuxAutres(int ExcepteUID, TSocketMsg* Msg);
Fonction:
Envoyer un message à tous les usagers de la liste à l'exception d'un seul.
Paramètre(s):
Msg: Message à envoyer.
Valeur de retour:
Aucune.

TListeUsager

Fonction:

Cette classe permet de créer une liste d'usager pour le client.

Méthotes:

Méthode:
TListeUsagerClient(TImageList* ImgLst, int LargeurVideo, int HauteurVideo);
Fonction:
Constructeur de la classe.
Paramètre(s):

ImgLst: Liste d'images contenant les images utilisées pour dessiner la liste des usagers.
LargeurVideo: Largeur de chaque vidéo dessiné dans la liste.
HauteurVideo: Hauteur de chaque vidéo dessiné dans la liste.

Valeur de retour:

Aucune.

Méthode:

~TListeUsagerClient();

Fonction:

Destructeur de la classe.

Paramètre(s):

Aucun.

Valeur de retour:

Aucune.

Méthode:

void Ajouter(FInfoUsager Info);

Fonction:

Ajouter un usager à la liste d'usagers.

Paramètre(s):

Info: Information de l'usager à ajouter.

Valeur de retour:

Aucune.

Méthode:

TUsagerClient* GetParId(short Id);

Fonction:

Cherche un usager dans la liste d'usagers par son id.

Paramètre(s):

Id: Position de l'usager dans la liste.

Valeur de retour:

L'usager trouvé.

Méthode:

TUsagerClient* GetParUID(int UID);

Fonction:

Cherche un usager dans la liste d'usagers par son UID.

Paramètre(s):

UID: UID de l'usager.

Valeur de retour:

L'usager trouvé.

Méthode:

TUsagerClient* GetParCompte(AnsiString Compte);

Fonction:

Cherche un usager dans la liste d'usagers par son compte.

Paramètre(s):

Compte: Compte de l'usager.

Valeur de retour:

L'usager trouvé.

Méthode:
Graphics::TBitmap* GetVideosImage(void);
Fonction:
Dessine la liste des usagers.
Paramètre(s):
Aucun.
Valeur de retour:
Liste des usagers sous forme d'une image JPG.

TUsagerServeur

Fonction:

Cette classe permet de créer un usager pour du côté serveur. Chaque usager possède un socket associé, ce qui permet d'envoyer des messages directement à l'un d'entre eux. De plus, chaque objet de type TUsagerServeur possède leur propre flux entrant, ce qui permet de gérer indépendamment chaque flux provenant de chaque usager.

Méthotes:

Méthode:
TUsagerServeur(FInfoUsager Info, TCustomWinSocket *Socket);
Fonction:
Constructeur de la classe.
Paramètre(s):
Info: Information de l'usager.
Socket: Socket associé à l'usager.
Valeur de retour:
Aucune.

Méthode:
~TUsagerServeur();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
TCustomWinSocket* GetSocket(void);
Fonction:
Obtenir le socket associé à l'usager.
Paramètre(s):
Aucun.
Valeur de retour:
Socket associé à l'usager.

Méthode:
TFluxEntrant* GetFlux(void);
Fonction:
Obtenir le flux provenant de l'usager.

Paramètre(s):
Aucun.
Valeur de retour:
Flux de l'usager.

Méthode:
void EnvoyerMsg(TSocketMsg* Msg);
Fonction:
Envoyer un message à l'usager par son socket associé.
Paramètre(s):
Msg: Message à envoyer.
Valeur de retour:
Flux de l'usager.

TUsagerClient

Fonction:

Cette classe permet de créer un usager pour du côté client. Chaque usager possède une vidéo permettant de gérer les images et les échantillons waves en provenance de cet usager, et ceci afin de reconstituer une vidéo à être jouée.

Méthotes:

Méthode:
TUsagerClient();
Fonction:
Constructeur de la classe.
Paramètre(s):
Info: Information de l'usager.
Valeur de retour:
Aucune.

Méthode:
~TUsagerClient();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
TVideo* GetVideo(void);
Fonction:
Obtenir la vidéo de l'usager.
Paramètre(s):
Aucun.
Valeur de retour:
Vidéo de l'usager.

TListeCompte

Fonction:

Cette classe permet de charger une liste de compte à partir d'un fichier txt. Chaque ligne du fichier txt représente un compte. Chaque compte doit être constitué de 4 chaînes de caractères séparés par des virgules. La première chaîne représente le compte de l'usager, la deuxième représente le mot de passe de l'usager, la troisième représente le nom de l'usager et la quatrième représente le prénom de l'usager.

Méthotes:

Méthode:
TListeCompte();
Fonction:
Constructeur de la classe.
Paramètre(s):
Info: Information de l'usager.
Valeur de retour:
Aucune.

Méthode:
~TListeCompte();
Fonction:
Destructeur de la classe.
Paramètre(s):
Aucun.
Valeur de retour:
Aucune.

Méthode:
void Charger(AnsiString Chemin);
Fonction:
Charger la liste des comptes à partir du fichier.
Paramètre(s):
Chemin: Chemin du fichier txt à charger.
Valeur de retour:
Aucune.

Méthode:
FInfoUsager ValiderCompte(AnsiString Compte, AnsiString MotPasse);
Fonction:
Validé si un usager ainsi que son mot de passe peuvent être trouvés dans la liste.
Paramètre(s):
Compte: Compte de l'usager.
MotPasse: Mot de passe associé à l'usager.
Valeur de retour:
Si l'usager est trouvé, les informations sur l'usager sont retournées, dans le cas contraire, une erreur d'exception est levée.

3.9 Guide de l'utilisateur

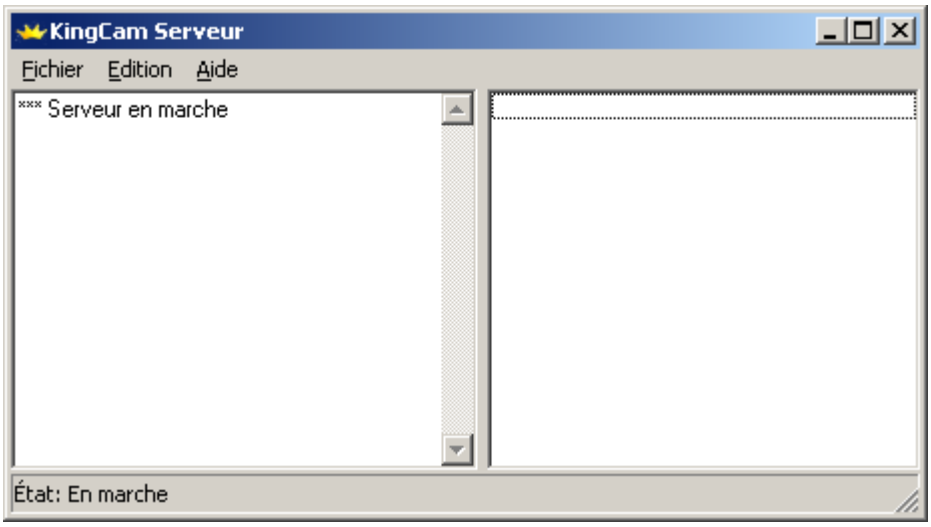
3.9.1 Application serveur

Il suffit seulement d'ouvrir l'application KingCam serveur, celle-ci se mettra automatiquement en marche.

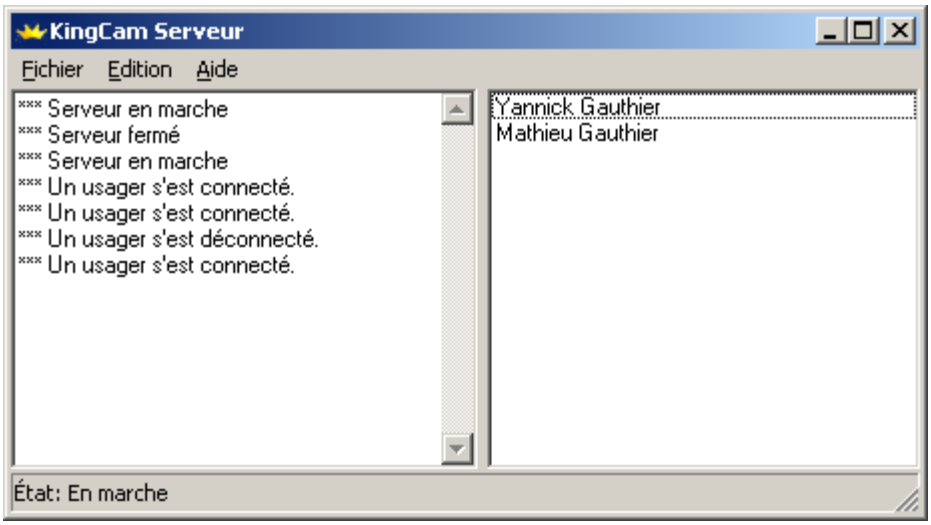
Redémarrage du serveur : Il est possible de redémarrer le serveur en faisant Fichier->Redémarrer

Arrêter le serveur : il est possible d'arrêter le serveur en faisant Fichier->Arrêter

Démarrer le serveur à la suis d'un arrêt: Il suffit de faire Fichier->Démarrer



État du serveur : Vous pouvez voir l'état du serveur en bas à gauche, l'état est soit « En marche » ou « fermé ».



Log du serveur : Le log affiche toutes les actions qui sont effectuées sur le serveur. Sur l'image ci-dessus on retrouve le log à gauche. Le log affiche ce qui suit :

- Lorsqu'on démarre le serveur
- Lorsqu'on arrête le serveur
- Lorsqu'on redémarre le serveur
- Lorsqu'un utilisateur se connecte
- Lorsqu'un utilisateur se déconnecte
- Les communications textuelles des utilisateurs
- Certaines erreurs liées à la connexion
- Certaines erreurs liées aux utilisateurs

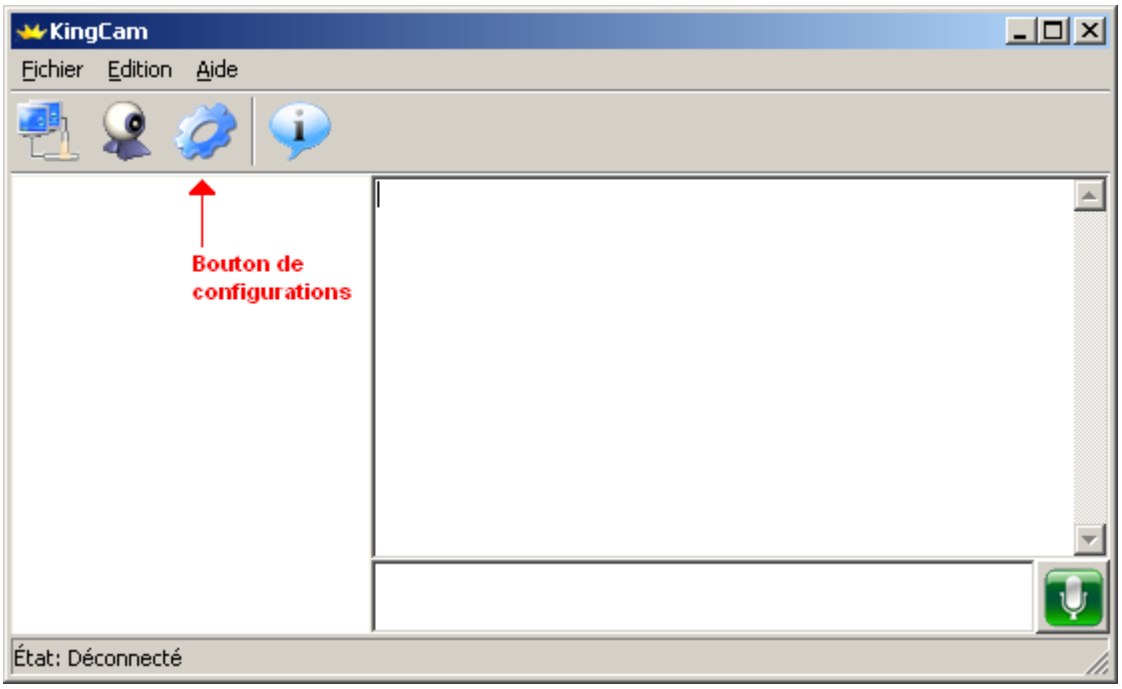
Liste des personnes connectées : La liste des personnes qui sont actuellement connectées sur le serveur se trouve à droite dans l'application serveur. Avec cette liste on peut savoir en tout temps qui se trouve sur le serveur.

Effacer le log : On peut effacer le contenu du log en allant dans Edition->Effacer le log.

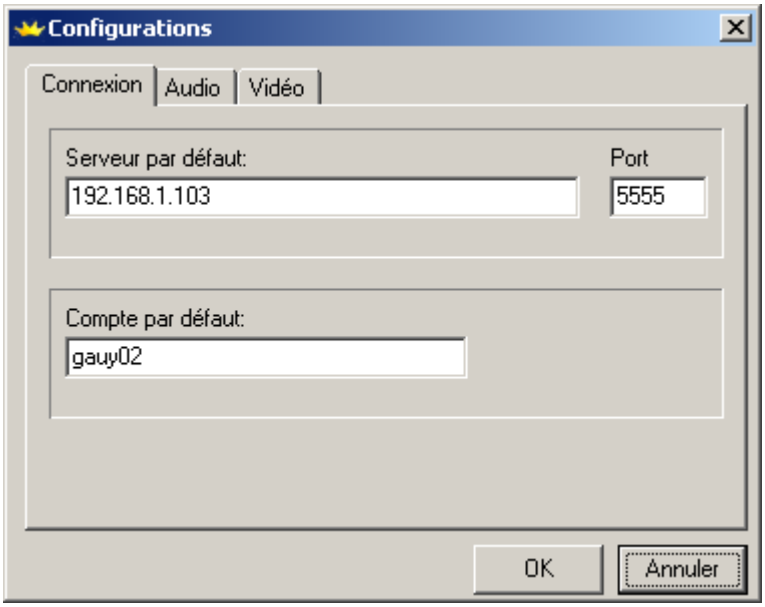
Port par défaut : Il est possible de changer le port par défaut sur lequel le serveur se connectera dans Edition->Configurations, section Port par défaut.

Démarrage automatique : Pour ne pas que le serveur démarre automatiquement lors de l'ouverture de l'application serveur il s'agit d'aller dans Edition->Configurations et décocher « Démarrage automatique ».

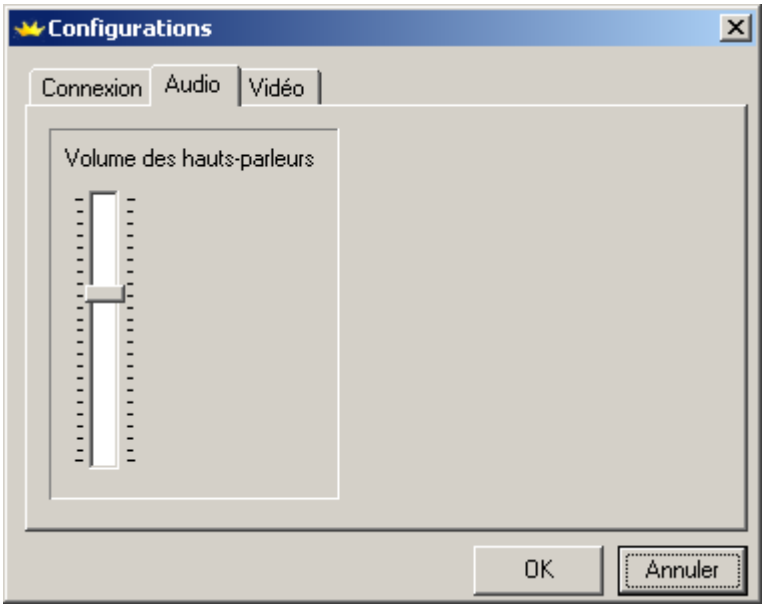
3.9.2 Application Cliente



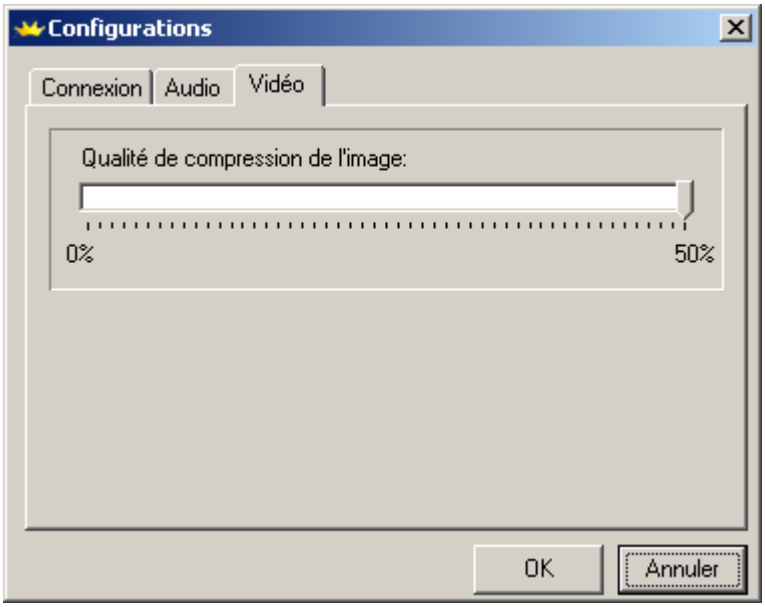
Configurer l'application : Pour accéder aux configurations pour l'application cliente vous devez aller dans le menu Edition->Configurations ou « appuyer sur le bouton de configurations. Par la suite, la fenêtre suivante s'ouvrira :



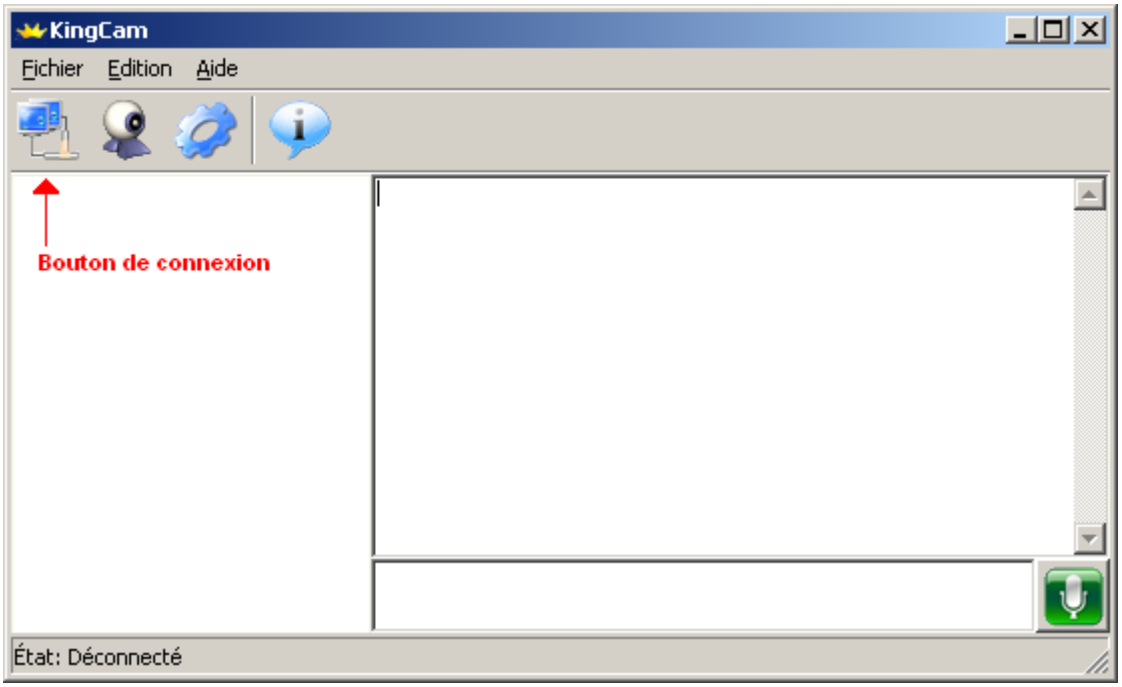
Onglet Connexion : Permet de sélectionner certains paramètres par défaut qui vont apparaître lorsque le client veut se connecter.



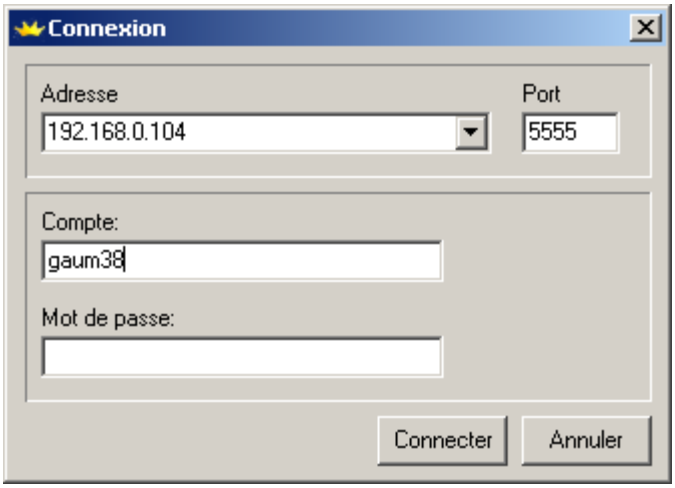
Onglet Audio : Permet tout simplement d'augmenter ou de réduire le volume des haut-parleurs ou des écouteurs.



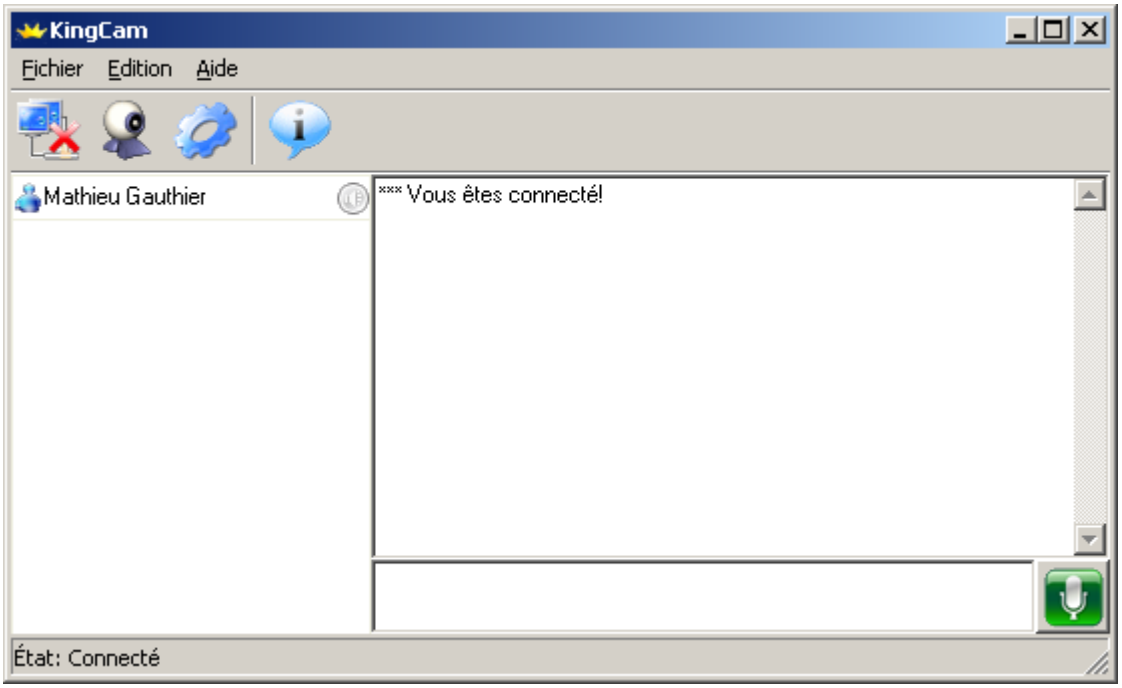
Onglet Vidéo : Permet de changer la qualité de compression de l'image. Lorsque la qualité de compression de l'image est moindre, le poids de l'image sera moindre, mais l'image sera moins belle. Modifier la compression permet de rendre l'application plus performante parce que l'application utilisera moins de bande passante.



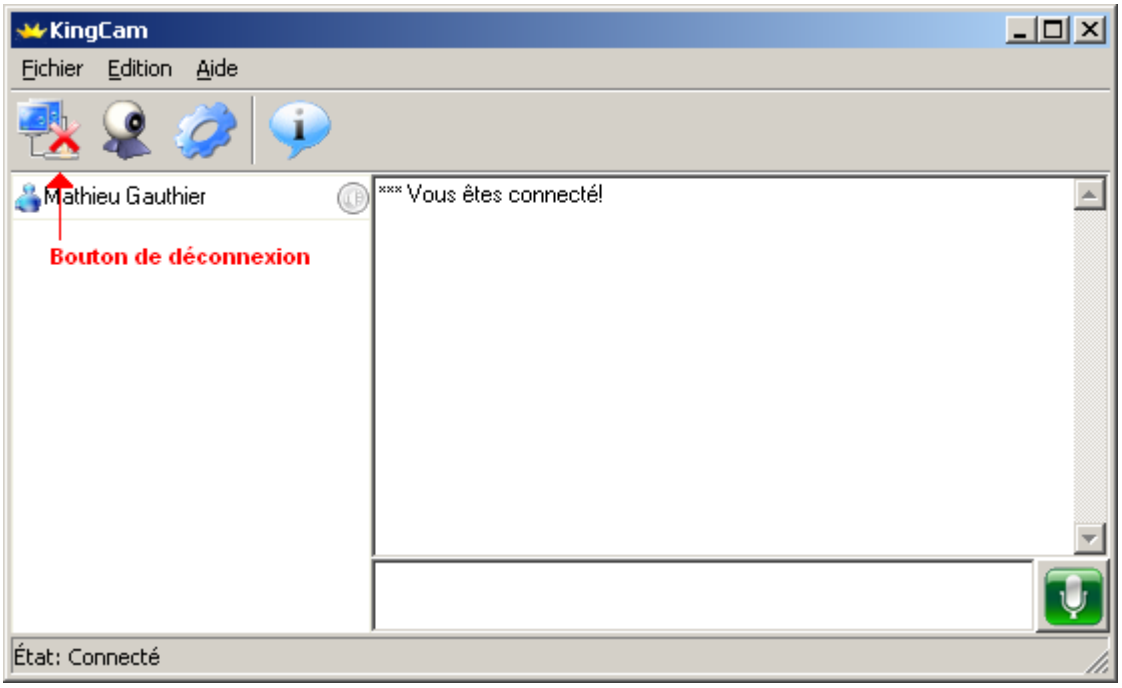
Se connecter au serveur : Pour se connecter au serveur, il suffit d'aller dans le menu Fichier->Connecter ou d'appuyer sur le bouton de connexion. Par la suite la fenêtre suivante s'ouvrira :



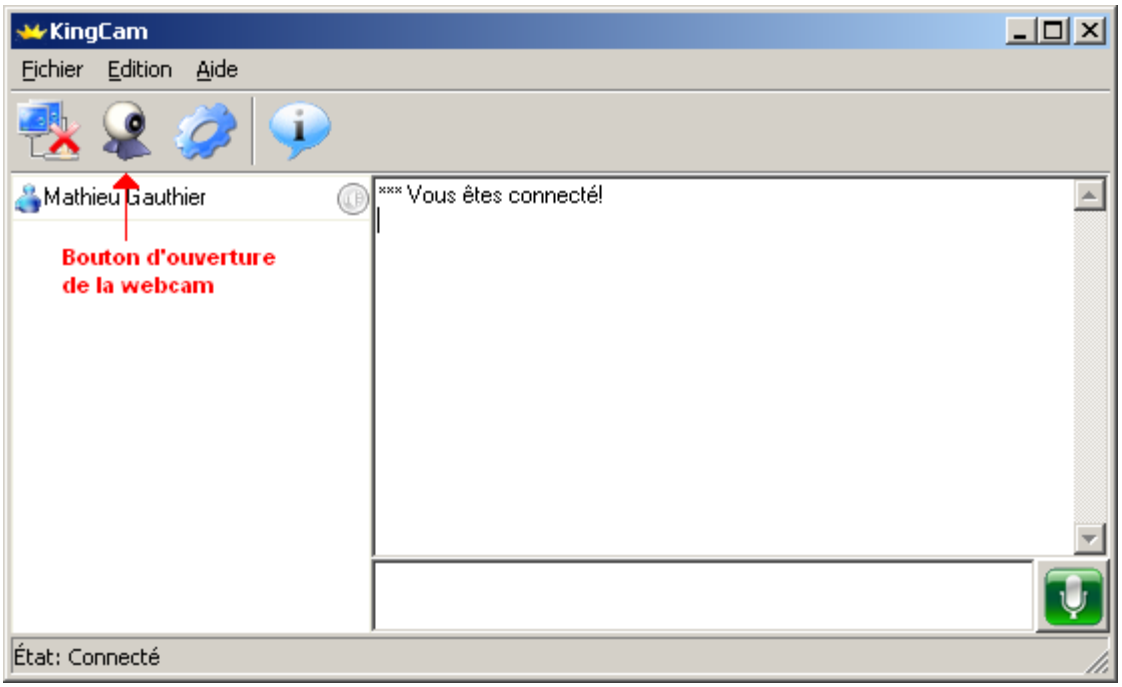
Vous devez entrer l'adresse d'un serveur KingCam ainsi que le port sur lequel il écoute. De plus, il est nécessaire d'entrer un compte utilisateur et un mot de passe (Les comptes sont stockés sur le serveur et sont créés par l'administrateur). Finalement, vous devez cliquer sur « Connecter ».



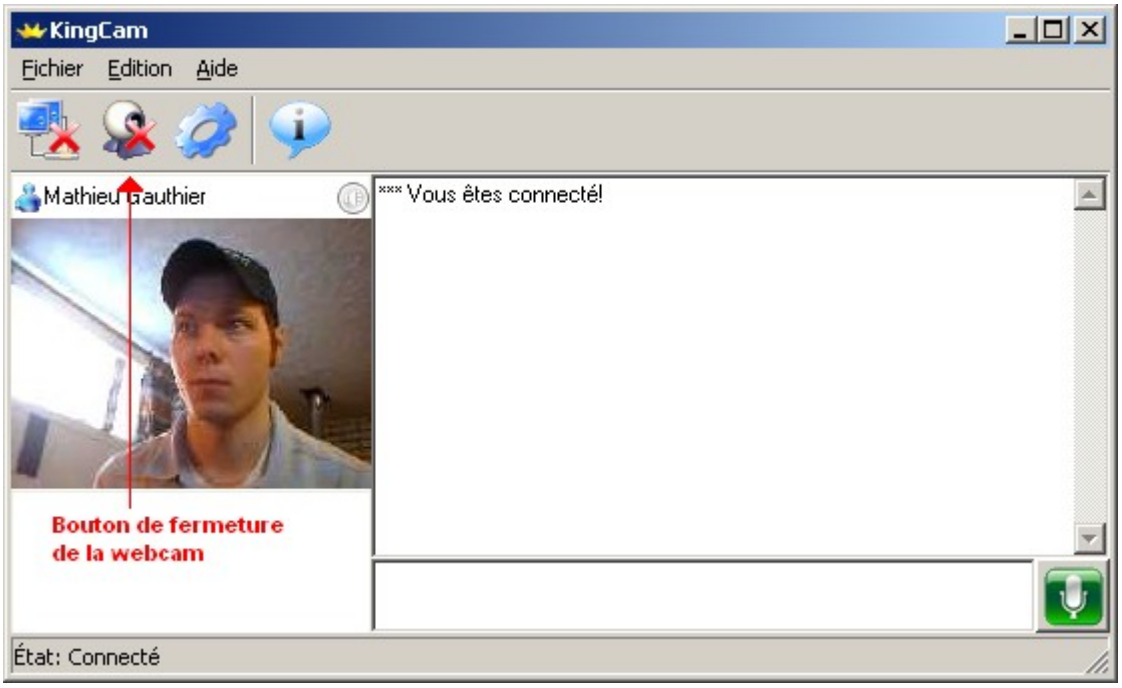
L'état du client : Lorsque vous êtes connecté, l'état du client qui se trouve au bas de l'application à gauche sera à « Connecté ».
De plus, lorsque vous êtes connecté votre nom apparaît dans la liste des utilisateurs. Pour cet exemple il s'agit de Mathieu Gauthier. L'état sera « En connexion » lorsque le client tente de se connecter.



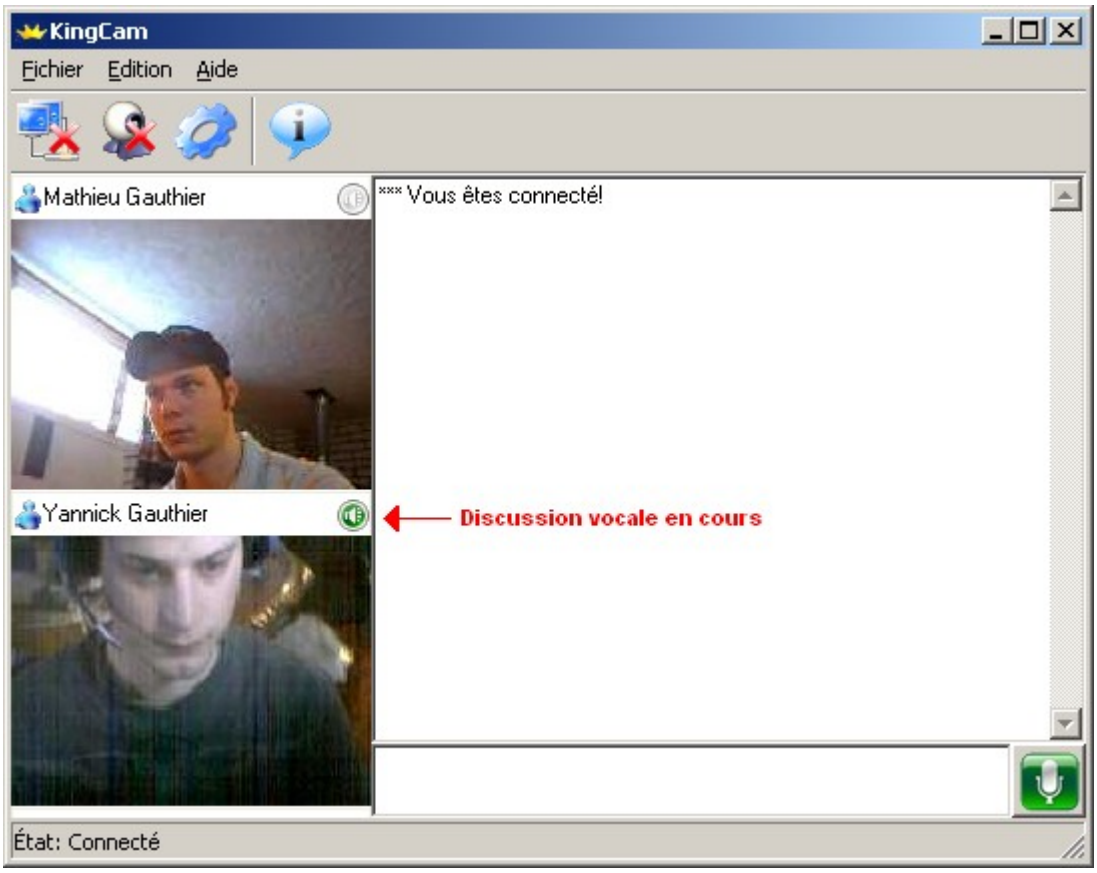
Se déconnecter du serveur : Pour se déconnecter du serveur il suffit d'aller dans le menu Fichier->Déconnecter ou d'appuyer sur le « Bouton de déconnexion ».



Ouverture de la webcam : Pour ouvrir sa propre webcam, il suffit de cliquer sur le « Bouton d'ouverture de la webcam ». L'utilisateur doit être connecté à un serveur pour faire l'ouverture de sa webcam.



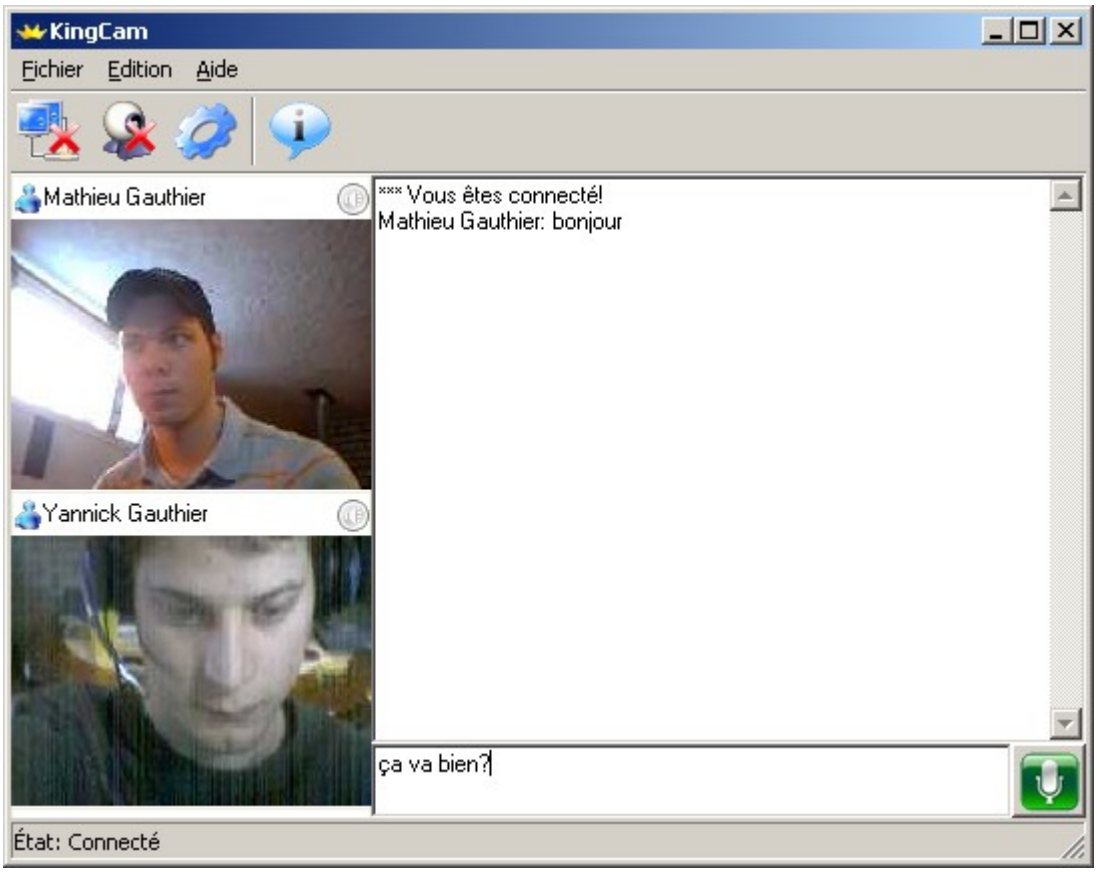
Fermeture de la webcam : Pour fermer sa propre webcam, il suffit de cliquer sur le « Bouton de fermeture de la webcam ».



Discussion vocale : Pour être en mesure de parler avec le microphone il s'agit d'appuyer sur le bouton vert en bas à droite. Pour parler, vous devez tenir le bouton tant et aussi longtemps que vous parlez, quand vous avez terminé, relâché tout simplement le bouton.

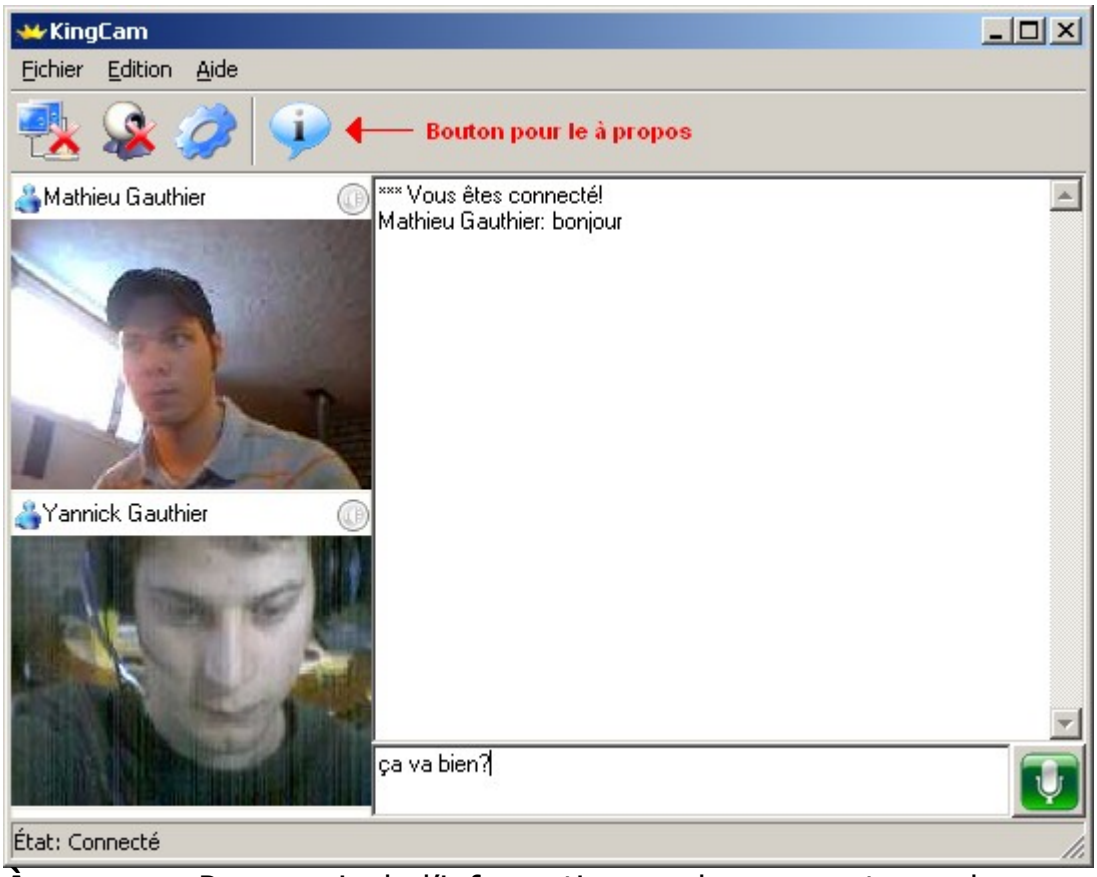
Lorsqu'une personne discute avec son microphone un voyant vert s'allume à côté de son nom. De cette manière, il est toujours possible de savoir qui parle.

Il faut noter qu'il est possible que plusieurs personnes parlent simultanément.



Discussion textuelle : Pour avoir une discussion textuelle, il s'agit d'écrire dans la zone texte et par la suite d'appuyer sur le bouton « entrer ».

Tout ce que les utilisateurs disent apparaîtra dans la fenêtre de dialogue. Il est possible de toujours savoir de qui provient le texte parce que le nom de la personne est toujours devant le texte.



À propos : Pour avoir de l'information sur les concepteurs de ce logiciel il s'agit d'aller dans le menu Aide->À propos ou cliquer sur le « bouton pour le à propos ».

Log : Le log contient beaucoup d'informations utiles pour les utilisateurs :

- Les conversations des utilisateurs
- Message de connexion
- Message de déconnexion
- Différents messages d'erreurs
- Nom d'utilisateur ou mot de passe invalide

Effacer le log : Il est possible d'effacer le contenu du log en se rendant dans le menu Edition->Effacer le log.

3.10 Références

http://www.uic.edu/depts/accc/itl/pubs/ipvc/index.html

http://myhome.hanafos.com/~soonjp/vidconf.html

http://myhome.hanafos.com/~soonjp/vclinux.html

http://www.marratech.com/

http://www.thorvinelectronics.com/video/index.html

http://www.1videoconference.com/

http://wapedia.mobi/fr/H.323

http://fr.wikipedia.org/wiki/Session_Initiation_Protocol

http://fr.wikipedia.org/wiki/SIP

http://www.indyproject.org/index.en.aspx

http://www.dimdim.com/

http://msdn2.microsoft.com/en-us/library/aa910189.aspx

http://msdn2.microsoft.com/en-us/library/ms713477%28VS.85%29.aspx

http://www.mangue.org/wiki/Capture_vid%C3%A9o_sous_MS-windows

http://www.planet-source-code.com/vb/scripts/ShowCode.asp?txtCodeId=4422&lngWId=3

http://www.windowsmobiledn.com/articles/multiplewaves.html