République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université SAAD DAHLEB Blida1

Faculté des Sciences

Département : d'Informatique

Projet de fin d'études pour l'obtention du diplôme de Master

Option : Systèmes Informatiques et Réseaux

Thème :

Développement d'une solution de communication basée sur la technologie *Web Real-Time Communication* (WEBRTC)

Réalisé par

AISSIOU Sonia

MANSOUR Hafsa

Soutenue le	22/06/2023
Encadreurs :	Mr. FARES Abdelhalim
	Mr. MELOUK Lyes
Promoteurs :	Mr. OULD KHAOUA Mohamed
***Jurys*:**	Mr. CHIKHI Fateh Nacim
	Mme. BACHA Sihem

Année universitaire : 2022-2023

Remerciements

Nous tenons tout d'abord à remercier le bon dieu de nous avoir donné santé, courage, volonté et foi pour réaliser ce travail.

Nous adressons nos remerciements à notre tuteur ***Mr. Ould Khaoua****, Professeur à L'université Saad Dahleb de Blida, pour ses prodigieux conseils avisés d'ordre pédagogiques et scientifiques, qui ont été d'une aide considérable.*

Et un grand remerciement à nos encadreurs du centre ***ICOSNET SPA*** ***Mr. AbdelHalim fares*** *responsable ingénierie et infrastructure VoIP et communication et* ***Mr. melouk Lyes*** *chef département VoIP, pour leur aide et leur suivi qui nous a permis d'avancer avec Sérénité tout au long du projet de fin d'étude.*

Sans oublier également d'adresser nos sincères remerciements à l'ensemble des membres du Jurys.

Nous adressons aussi toute notre gratitude à nos amis du ***club scientifique ITC*** *qui nous ont assistés par leurs conseils à réaliser notre projet.*

Enfin, ce travail n'aurait pu aboutir sans le soutien sans faille tout au long de nos études de nos parents, nous les remercions de tout cœur.

ملخص

يستكشف هذا البحث بالتفصيل التواصل القائم على تقنية WebRTC، وهي تقنية الاتصال في الوقت الحقيقي التي تعتمد على متصفحات الويب. يتيح للمستخدمين إنشاء اتصالات صوتية وفيديو ورسائل مباشرة من متصفحهم دون الحاجة إلى تثبيت برامج إضافية.

لضمان إنشاء وإدارة هذه الاتصالات، تلعب عمليات إرسال الإشارات (signalisation) دورًا حاسمًا. حيث تسمح للمشاركين بالتفاوض حول معلومات الاتصال مثل البروتوكولات وشفرات البيانات وخيارات الأمان وغيرها. يُستخدم بروتوكول إرسال الإشارات SIP بشكل شائع لتيسير هذا التواصل. كما أن مشفرات البيانات ضرورية لضغط وفك ضغط تدفقات الصوت والفيديو المتبادلة أثناء التواصل باستخدام WebRTC.

بالإضافة إلى ذلك، يعتمد WebRTC على نهج التواصل p2p، مما يتيح للمشاركين التواصل المباشر بين بعضهم البعض دون الحاجة إلى خادم مركزي. يعزز ذلك كفاءة التواصل والخصوصية، مع تقليل التأخير والاعتماد على البنى التحتية المكلفة. من خلال فهم هذه المبادئ الأساسية، يهدف هذا البحث إلى توفير نظرة شاملة للتواصل القائم على WebRTC وإمكاناته في تطوير حلول اتصال مبتكرة وفعالة.

الكلمات المفتاحية : WebRTC, SIP, P2P, Signalisation

Résumé

Ce mémoire explore en détail la communication basée sur WebRTC qui est une technologie de communication en temps réel basée sur les navigateurs web. Il permet aux utilisateurs d'établir des connexions vocales, vidéo et une messagerie depuis leur navigateur, sans besoin d'installer des logiciels.

Pour assurer l'établissement et la gestion de ces connexions, la signalisation joue un rôle essentiel. Elle permet aux participants de négocier les paramètres de communication tels que les protocoles, les codecs, les options de sécurité, etc. Le protocole de signalisation SIP est couramment utilisé pour faciliter cette communication. Les codecs sont également essentiels pour la compression et la décompression des flux audio et vidéo échangés lors des communications WebRTC.

De plus, WebRTC adopte une approche de communication de P2P, permettant aux participants de communiquer directement entre eux, sans passer par un serveur centralisé. Cela améliore l'efficacité et la confidentialité des communications, tout en réduisant la latence et la dépendance à l'égard d'infrastructures coûteuses.

En comprenant ces principes fondamentaux, ce mémoire vise à fournir une vue d'ensemble complète de la communication basée sur WebRTC et de son potentiel pour développer des solutions de communication innovantes et efficaces**.**

Mots clés : WebRTC**,** SIP, P2P, Signalisation

Abstract

This thesis provides a detailed exploration of WebRTC-based communication, a real-time communication technology built into web browsers. It enables users to establish voice, video, and messaging connections directly from their browser, without the need for additional software installations.

To ensure the establishment and management of these connections, signaling plays a crucial role. Signaling allows participants to negotiate communication parameters such as protocols, codecs, security options, and more. The SIP signaling protocol is commonly used to facilitate this communication. Codecs are also essential for compressing and decompressing audio and video streams exchanged during WebRTC communications.

Additionally, WebRTC adopts a peer-to-peer (P2P) communication approach, allowing participants to communicate directly with each other without relying on a centralized server. This enhances communication efficiency and privacy, while reducing latency and dependence on costly infrastructures. By understanding these fundamental principles, this thesis aims to provide a comprehensive overview of WebRTC-based communication and its potential for developing innovative and efficient communication solutions.

Keyboard: WebRTC, SIP, P2P, Signalisation

Liste des acronymes et abréviations

AOMedia	**:**	**A**lliance for **O**pen **M**edia
API	**:**	**A**pplication **P**rogramming **I**nterface
AVI	**:**	**A**udio **V**ideo **I**nterleave
CentOS	**:**	**C**ommunity **Ent**erprise **O**perating **S**ystem
CRM	**:**	**C**ustomer **R**elationship **M**anagement
DDoS	**:**	**D**istributed **D**enial **o**f **S**ervice
DNS	**:**	**D**omain **N**ame **S**ystem
DTLS	**:**	**D**atagram **T**ransport **L**ayer **S**ecurity
ESXi	**:**	**E**lastic **S**ky **X** **I**ntegrated
GIPS	**:**	**G**lobal **I**nvestment **P**erformance **S**tandards
H.264	**:**	**Advanced Video Coding**
H.323	**:**	**Protocole de communication multimédia sur des réseaux IP**
HTTP	**:**	**H**yper**T**ext **T**ransfer **P**rotocol
HTTPS	**:**	**H**yper**T**ext **T**ransfer **P**rotocol **S**ecure
ICE	**:**	**I**nteractive **C**onnectivity **E**stablishment
IETF	**:**	**I**nternet **E**ngineering **T**ask **F**orce
IP	**:**	**I**nternet **P**rotocol
Jansson	**:**	**Bibliothéque C pour la manipulation de donnée JSON**
JSON	**:**	**J**ava**S**cript **O**bject **N**otation
MacOS	**:**	**O**perating **S**ystem **developed by Apple Inc**
MGCP	**:**	**M**edia **G**ateway **C**ontrol **P**rotocol
MPEG4	**:**	**M**oving **P**icture **E**xperts **G**roup-4
NAT	**:**	**N**etwork **A**ddress **T**ranslation
Opus	**:**	**Open and royalty-free audio codec**
OS	**:**	**O**perating **S**ystem
P2P	**:**	**P**eer-**T**o-**P**eer
PHP	**:**	**H**ypertext **P**reprocessor
PSTN	**:**	**P**ublic **S**witched **T**elephone **N**etwork
QoS	**:**	**Q**uality **o**f **S**ervice
RAM	**:**	**R**andom **A**ccess **M**emory
RHEL	**:**	**R**ed **H**at **E**nterprise **L**inux
RTC	**:**	**R**eal **T**ime **C**ommunication
RTCP	**:**	**R**eal-Time **T**ransport **C**ontrol **P**rotocol
RTP	**:**	**R**eal-**T**ime **T**ransport **P**rotocol
SCTP	**:**	**S**tream **C**ontrol **T**ransmission **P**rotocol
SDES	**:**	**S**ession **D**escription **P**rotocol **S**ecurity **D**escriptions
SDP	**:**	**S**ession **D**escription **P**rotocol
SIP	**:**	**S**ession **I**nitiation **P**rotocol
SRTP	**:**	**S**ecure **R**eal-**T**ime **T**ransport **P**rotocol
SSH	**:**	**S**ecure **Sh**ell
SSL	**:**	**S**ecure **S**ockets **L**ayer
STUN	**:**	**S**ession **T**raversal **U**tilities for **NAT**
TCP	**:**	**T**ransmission **C**ontrol **P**rotocol
TLS	**:**	**T**ransport **L**ayer **S**ecurity

TURN	**:**	**T**raversal **U**sing **R**elays around **NAT**
UDP	**:**	**U**ser **D**atagram **P**rotocol
UIT	**:**	**U**nion **I**internationale des **T**élécommunications
VMware	**:**	**V**irtual **M**achine **ware**
VOIP	**:**	**V**oice **o**ver **I**nternet **P**rotocol
VP8	**:**	**video codec**
VP9	**:**	**Video codec**
VS	**:**	**V**isual **S**tudio
VVOIP	**:**	**V**ideo **V**oice **o**ver **I**nternet **P**rotocol
W3C	**:**	**W**orld **W**ide **W**eb **C**onsortium
Web	**:**	**W**orld **W**ide **W**eb
WebRTC	**:**	**Web R**eal-**T**ime **C**ommunication
WSS	**:**	**W**eb**S**ocket **S**ecure

Liste des figures

Table des matières

Introduction Générale

Ces derniers temps, l'internet a fait des progrès considérables et ses applications web sont devenues plus complexes. Avec l'augmentation spectaculaire du nombre d'utilisateurs et d'activités qui migrent en ligne, les exigences techniques et de sécurité sont désormais plus élevées. L'un des besoins essentiels de nombreuses applications est d'identifier et de classer les flux des réseaux ainsi que les protocoles.

La communication en ligne est devenue un élément indispensable de la vie moderne, en particulier à la lumière de la pandémie de Covid-19. L'adoption généralisée d'outils en ligne pour travailler, étudier et rester en contact avec ses proches a également ouvert de nouvelles voies de divertissement.

Une communication efficace est essentielle pour réussir dans le paysage commercial contemporain. Les entreprises qui privilégient leur capacité à interagir avec les clients, à résoudre rapidement les problèmes et à répondre à leurs besoins ont plus de chances de prospérer. À cette fin, il est primordial de faciliter l'accès aux moyens de communication sans faire peser sur les clients les coûts associés. C'est pourquoi de nombreuses entreprises proposent des numéros verts qui permettent aux clients de les joindre gratuitement.

1. Problématique

L'objectif premier de l'entreprise est de fournir à ses clients une solution de communication qui ne leur coûte rien tout en diminuant les dépenses liées aux appels entrants. Les numéros verts traditionnels sont souvent coûteux pour les entreprises car elles supportent les frais liés aux appels entrants, ce qui peut entraîner des dépenses importantes, en particulier pour les grandes entreprises qui reçoivent des appels fréquents.

L'accessibilité des numéros verts peut être limitée dans certains pays. Par conséquent, les entreprises peuvent ne pas être en mesure de communiquer efficacement avec les clients potentiels de ces pays. Cette situation crée des obstacles à la communication qui risquent de frustrer les clients internationaux et de les empêcher de contacter l'entreprise pour obtenir de l'aide ou poser des questions sans frais supplémentaires.

2. Objectif

Dans le but de résoudre ce problème, une solution de communication en ligne qui est innovante, fluide et économique avec les clients a été envisagée.
Il existe déjà de nombreux systèmes d'appels disponibles sur le marché. ils utilisent un protocole propriétaire pour la transmission des flux multimédia, en plus il nécessite l'installation d'une application mobile ou bureau pour accéder à des services tels que les appels téléphoniques, messages et visioconférences. Les utilisateurs de ces systèmes peuvent visualiser, enregistrer, commenter ou modifier les flux de contenu vidéo et audio à l'aide du Cloud.
Le WebRTC est une technologie de communication en temps réel basée sur les navigateurs web, il propose une alternative prometteuse aux numéros verts traditionnels. Il permet d'établir des connexions vocales et de partage de données directement entre les navigateurs des utilisateurs, sans la nécessité d'installation de logiciel tiers. Cela signifie que les clients peuvent

communiquer avec l'entreprise sans avoir à composer un numéro spécialisé ni à supporter des frais supplémentaires.

De plus, WebRTC offre une portée mondiale permettant aux clients de partout dans le monde d'accéder à la solution de communication de l'entreprise, il n y a plus de restrictions géographiques liées aux numéros verts traditionnels, ce qui élargit considérablement l'audience et le potentiel commercial de l'entreprise.

Dans notre étude, nous visons à développer une solution de communication novatrice en utilisant WebRTC intégrée à un serveur Asterisk. Cette solution permettra à l'entreprise de fournir à ses clients une communication gratuite, sans gaspillage d'unités et sans passer par les numéros verts traditionnels.

Cette solution permettra aux clients de communiquer directement avec l'entreprise ICOSNET via son site web sans avoir besoin d'un numéro de téléphone spécial ou d'une application tierce, les clients peuvent simplement cliquer sur un bouton de communication sur le site web de l'entreprise et être instantanément connectés à un représentant compétent offrant ainsi une expérience utilisateur fluide et transparente.

3. Structure du mémoire

Ce mémoire est structuré en quatre chapitres qui abordent différents aspects de la solution de communication basée sur WebRTC et intégrée à un serveur Asterisk. Voici un aperçu de la structure du mémoire :

- **Dans le premier chapitre**, nous aborderons les principes fondamentaux des réseaux de communication, généralités sur la VOIP et du WebRTC.
- **Dans le deuxième chapitre**, nous concentrons sur les API, la signalisation, les protocoles et les codecs…
- **Dans le troisième chapitre,** nous aborderons les schémas et les diagrammes UML et l'analyse du projet.
- **Enfin dans le dernier chapitre,** nous nous baserons sur les outils et les technologies utilisés, les étapes de développement, ainsi que des exemples et des captures d'écran.

Cette thèse présente une approche complète de la mise en place d'une solution de communication basée sur WebRTC et le serveur Asterisk. Elle guide le lecteur dans la compréhension de WebRTC, l'exploration de ses fonctionnalités, la familiarisation avec la modélisation et la conception de la solution et la compréhension des étapes de la réalisation réussie du projet.

Chapitre I : : Généralité sur la VOIP et WebRTC

Introduction

Au fil des décennies, différentes technologies ont été développées pour faciliter la transmission de l'information entre les individus. De la communication verbale à la communication visuelle, ces avancées technologiques ont ouvert de nouvelles possibilités et ont révolutionné notre façon de nous connecter les uns avec les autres. Parmi ces avancées, le WebRTC qui occupe une place prépondérante en offrant une solution de communication en temps réel basée sur le web.
Dans ce chapitre, nous allons explorer les principes fondamentaux sur la VOIP et sur le WebRTC.

1. VoIP

1.1 Définition

VoIP signifie *Voice over Internet Protocol*, comme son nom l'indique, la VoIP permet de transmettre des sons (en particulier la voix) dans des paquets IP circulant sur Internet.
L'idée originale derrière la VoIP est de transmettre un signal vocal en temps réel sur un réseau de données et de réduire le coût des appels interurbains, car les appels VoIP passeraient par des réseaux basés sur des paquets à un tarif forfaitaire, au lieu du réseau téléphonique public commuté traditionnel qui coûtait cher pour les appels longue distance, de plus en plus de personnes de différents groupes d'âge comptent désormais sur des produits et des outils VoIP pour passer des appels audio/vidéo afin de rester en contact avec leurs familles et leurs amis, car ils sont gratuits ou peux coûteux.

Depuis son invention, la VoIP a connue une croissance exponentielle, passant d'une application de laboratoire à petite échelle à l'outil mondial d'aujourd'hui avec des applications dans la plupart des domaines de l'entreprise et de la vie quotidienne. De plus en plus d'organisations passent du PSTN traditionnel aux solutions VoIP modernes.[3]

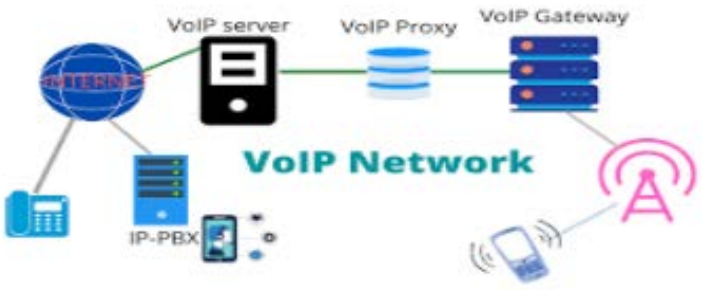

Figure I.1: Réseaux VOIP

Les systèmes de téléphonie basés sur la VoIP peuvent être déployés de différentes manières, que ce soit en utilisant des passerelles pour connecter le réseau IP au réseau téléphonique traditionnel, en utilisant des solutions logicielles sur des serveurs ou en utilisant des services basés sur le cloud.

1.2 VVOIP

VVOIP, ou Voice and *Video over* IP, est une extension de la VoIP qui intègre la transmission de la voix et de la vidéo sur les réseaux IP. Elle offre des possibilités de communication plus riches et interactives. Il permet aux utilisateurs de communiquer à la fois par la voix et par la vidéo en utilisant des applications et des services basés sur IP. Cela inclut les appels audio et vidéo individuels, les conférences audio et vidéo et la collaboration en temps réel.

Figure I.2: Réseau VVoIP

La VVOIP utilise des protocoles et des technologies spécifiques pour la transmission de la voix et de la vidéo sur les réseaux IP. Cela peut inclure des protocoles tels que le protocole SIP pour établir et gérer les sessions de communication, des codecs ainsi que des protocoles de transport tels que RTP pour la transmission en temps réel des données multimédias.

2. WebRTC

2.1 Définition

Web Real-Time Communication (WebRTC) est une technologie de communication en temps réel basée sur des standards web, qui permet aux utilisateurs de communiquer directement à travers leur navigateur, sans avoir besoin de télécharger ou d'installer des logiciels supplémentaires. Cette technologie offre de nombreux avantages, tels que la qualité de service élevée, la rapidité de mise en place, la sécurité et l'interopérabilité avec d'autres plateformes.
WebRTC est une nouvelle norme qui permet aux navigateurs de communiquer et d'échanger des données y compris l'audio/vidéo en temps réel en utilisant une architecture peer-to-peer. Cette évolution est perturbatrice dans le monde des applications web car elle permet pour la toute première fois aux développeurs web de créer des applications multimédia en temps réel sans avoir besoin de plugins propriétaires.[2]

Notre étude revêt une importance majeure dans le contexte actuel de la communication en ligne, en proposant une solution innovante et performante pour améliorer la qualité de la communication en ligne. En combinant une analyse approfondie de la technologie WebRTC avec une évaluation critique des publications disponibles, notre étude vise à proposer une solution de communication en ligne basée sur WebRTC fiable, performante et adaptée aux besoins des utilisateurs

2.2 Historique

WebRTC est une technologie open source développée par Google en 2011. Elle permet de réaliser des communications audio et vidéo en temps réel directement à partir d'un navigateur web, sans avoir besoin d'installer un logiciel ou une extension supplémentaire. Dans la figure suvant nous alons explorer l'historique du WebRTC :

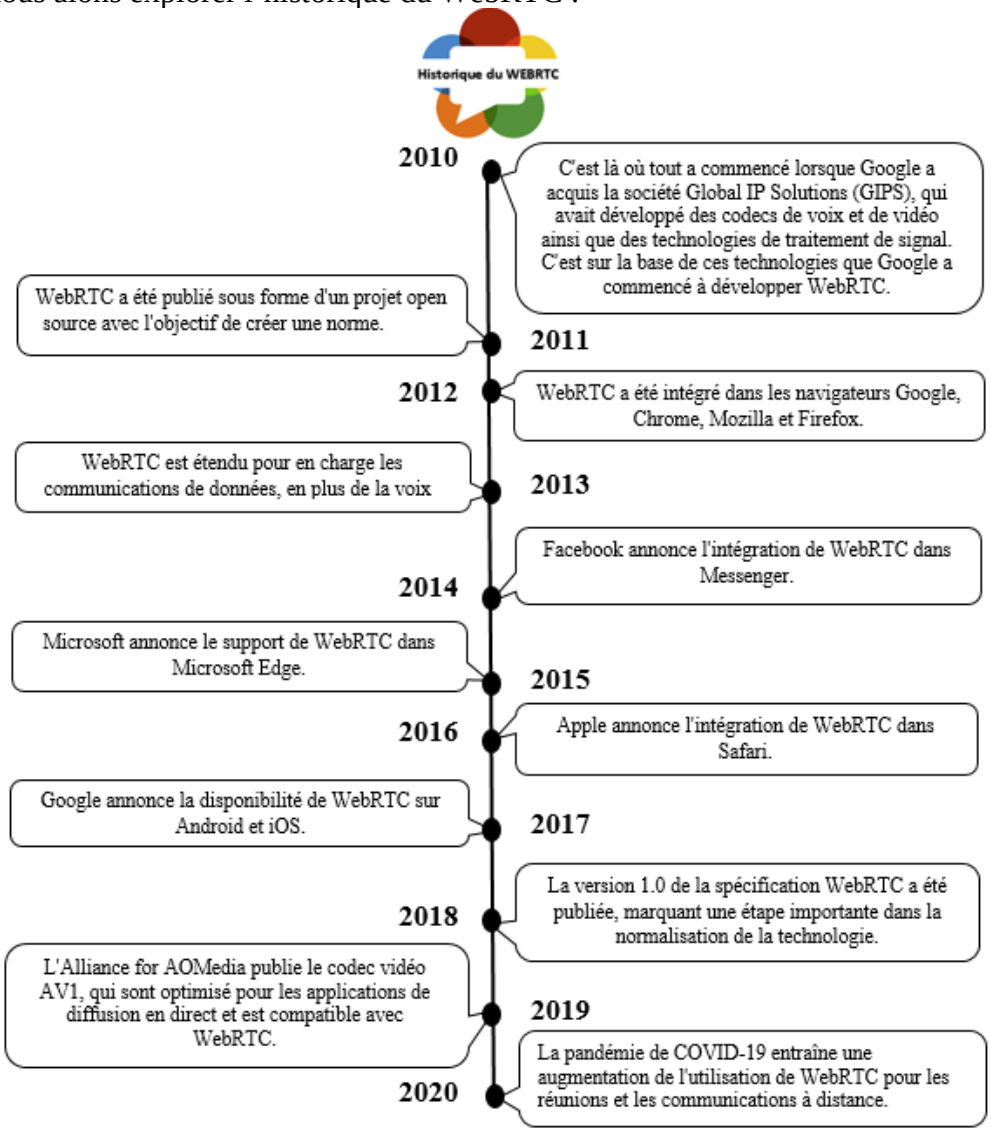

Figure I.3: Historique de WebRTC

Au fil des années, WebRTC a évolué pour devenir une technologie mature et largement utilisée pour les communications en temps réel sur le web. La disponibilité de WebRTC dans de nombreux navigateurs et sur de nombreuses plates-formes ont fait une solution pratique pour de nombreuses applications, allant des appels vidéo personnels aux applications professionnelles de diffusion.

2.3 Les avantages du WebRTC

- **Qualité de service élevée :** WebRTC offre une qualité de service élevée grâce à son support intégré pour les codecs vidéo et audio de haute qualité avec un taux de compression efficace, permettant une utilisation optimale de la bande passante, tels qu'Opus, VP8 et VP9. De plus, WebRTC utilise des techniques telles que le contrôle de la congestion et l'adaptation dynamique de la résolution pour assurer une expérience utilisateur fluide, même dans des conditions de bande passante limitée.

- **Sécurité :** WebRTC intègre des mécanismes de sécurité robustes, tels que le chiffrement de bout en bout, qui signifie que les données sont chiffrées sur l'appareil de l'émetteur et ne sont déchiffrées que sur l'appareil du récepteur empêchant toute interception ou modification des données en transit pour garantir la confidentialité et l'intégrité des données échangées lors des communications en ligne. WebRTC offre une protection contre les attaques DDoS et d'autres types d'attaques potentielles, grâce à des mécanismes tels que la validation des identités et le contrôle d'accès.

- **Interopérabilité :** WebRTC est une technologie standardisée et largement adoptée, ce qui garantit son interopérabilité avec d'autres plateformes et applications de communication en ligne.
  Les applications basées sur WebRTC peuvent communiquer avec succès avec une variété d'applications de communication, notamment les applications de visioconférence, de chat, de voix sur IP et de diffusion en direct.

- **Flexibilité :** WebRTC est une technologie flexible, qui permet une personnalisation et une adaptation facile aux besoins spécifiques de chaque application de communication en ligne. Les développeurs peuvent personnaliser WebRTC en utilisant des API et des bibliothèques open source, ce qui leur permet de créer des applications de communication personnalisées pour répondre aux besoins de leur entreprise ou de leurs utilisateurs.

- **Évolutivité :** WebRTC est une technologie évolutive, qui peut s'adapter à une large gamme de cas d'utilisation et de scénarios de communication en ligne, WebRTC est utilisé dans une variété de domaines, notamment la santé, l'éducation, les services financiers et les jeux en ligne. En outre, la technologie continue de se développer et d'évoluer pour répondre aux besoins changeants des entreprises et des utilisateurs finaux.

2.4 Exploration des applications de WebRTC

Visioconférence	Grâce à WebRTC, les entreprises peuvent offrir une expérience de visioconférence de haute qualité avec des fonctionnalités avancées telles que la collaboration en temps réel, le partage d'écran, l'enregistrement de la réunion, les annotations et bien plus encore. Les utilisateurs peuvent rejoindre les réunions à partir de n'importe quel appareil compatible avec WebRTC, sans avoir besoin de télécharger de logiciel ou d'installer de plug-in.
Voix sur IP	Les applications basées sur WebRTC offrent également une solution VoIP fiable et de haute qualité. Les appels audio en direct peuvent être effectués à partir de n'importe quel navigateur compatible avec WebRTC, sans aucune installation supplémentaire. WebRTC offre une qualité audio supérieure à celle des autres protocoles VoIP, avec une faible latence et une clarté supérieure.
Diffusion en direct	Les applications de diffusion en direct basées sur WebRTC offrent une expérience de visionnage en direct en temps réel, permettant aux utilisateurs de diffuser des événements en direct en ligne. Les utilisateurs peuvent regarder des événements en direct avec une faible latence et une qualité vidéo et audio élevée, sans interruption ni décalage. WebRTC permet également aux utilisateurs d'interagir en temps réel avec les diffuseurs, en permettant des commentaires en direct et des chats en direct.
Services de santé en ligne	WebRTC permet de faire des consultations vidéo en direct entre les patients et les professionnels de la santé. Les patients peuvent consulter un médecin ou un spécialiste à distance, sans avoir besoin de se déplacer. WebRTC offrent une qualité vidéo et audio élevée, tout en garantissant la sécurité et la confidentialité des données de santé des patients.
Jeux en ligne	WebRTC permet de faire des communications vocales en temps réel entre les joueurs. Elle offre une qualité audio supérieure, ce qui permet aux joueurs de communiquer clairement et de manière fluide pendant le jeu. Les fonctionnalités de partage d'écran offertes par WebRTC peuvent également être utilisées pour permettre aux joueurs de collaborer et de partager des astuces de jeu en temps réel.
Éducation en ligne	WebRTC permet de faire des cours en ligne en direct. Les étudiants peuvent participer à des cours en ligne en direct, avec des fonctionnalités avancées telles que le partage d'écran, la collaboration en temps réel, le partage de fichiers, le chat en direct ... Les enseignants peuvent également utiliser WebRTC pour organiser des sessions de formation à distance et des réunions de groupe.

2.5 Domaine d'implémentation du WEBRTC

La communication en temps réel joue un rôle central dans de nombreux domaines de notre société moderne. Le WebRTC trouve des applications dans divers domaines, notamment la métaverse, la médecine et la sécurité.

2.5.1 Domaine de la métaverse

Dans l'article [4] la métaverse est un monde virtuel en 3D, où les utilisateurs peuvent interagir avec des objets et des personnes virtuelles. Dans une métaverse éducative, les enseignants peuvent créer des environnements d'apprentissage virtuels où les étudiants peuvent interagir avec du contenu éducatif en 3D.

En intégrant WebRTC et en utilisant Mozilla hubs dans une métaverse éducative, les enseignants peuvent offrir une expérience d'apprentissage plus immersive en permettant aux étudiants de discuter en temps réel, de poser des questions et de travailler en collaboration. En utilisant la vidéo et l'audio, les enseignants peuvent également fournir une formation en direct à leurs étudiants, ce qui peut améliorer leur compréhension des concepts clés.

2.5.2 Domaine de la médicine

D'après l'article [5] qui vise à améliorer un large domaine de vue entourant le fauteuil roulant pour assurer une navigation en fauteuil roulant et une assistance efficace pour les lecteurs de fauteuils roulants.

Une caméra double fisheye est montée devant le fauteuil roulant pour capturer des images qui peuvent ensuite être diffusées sur internet. Un protocole de communication WebRTC a été mis en œuvre pour fournir un flux vidéo et de données efficace. Les résultats globaux montrent l'efficacité de notre approche proposée pour la vision à 360 degrés.

Dans étude a pris en compte les WebRTC qui sont implémentés chez l'expéditeur et le destinataire. Les résultats montrent que le temps de connexion de WebRTC est plus rapide avec moins de latence que les autres plateformes et réduit le délai de traitement et le temps de connexion. De plus, le débit de transmission de WebRTC est capable de s'adapter à la bande passante disponible du réseau.

2.5.3 Domaine de la sécurité

De nos jours, nous observons un développement rapide des systèmes de surveillance basés sur les drones, qui sont confrontés à de plus en plus de nouvelles tâches, telles que la haute résolution temporelle et la haute résolution spatiale des mesures, ou l'intelligence artificielle à bord.

Dans l'étude [6] WebRTC offre une transmission simultanée d'un flux vidéo en temps réel et du flux de données provenant de capteurs, et assure une sorte de protection du flux de données, ce qui conduit à préserver son caractère quasi temps réel et permet une communication contextuelle. Le prototype exemplaire du système a été évalué en termes de capacité à travailler avec des capteurs à réponse rapide, de capacité à travailler avec des résolutions temporelles et spatiales élevées, d'informations vidéo en temps réel dans des conditions de mauvaise visibilité et de préparation à l'IA.

Conclusion

Dans ce chapitre nous avons exploré les fondamentaux des réseaux de communication, la VoIP, la VVoIP et les généralités sur WebRTC.

Nous avons appris que les réseaux de communication ont évolué vers des infrastructures IP modernes, permettant une transmission plus rapide et plus économique des données.
La VoIP et la VVoIP ont révolutionné la communication en permettant la transmission de la voix et de la vidéo sur les réseaux IP, offrant des avantages tels qu'une meilleure qualité audio et vidéo et des coûts réduits. De plus, nous avons examiné les généralités sur WebRTC, une technologie qui permet la communication en temps réel directement à partir d'un navigateur web. Cela ouvre de nouvelles possibilités pour des applications de communication basées sur Le Web.

Chapitre II : Architecture et Fonctionnement du WebRTC

Introduction

L'architecture WebRTC repose sur un ensemble de composants interconnectés qui travaillent en harmonie pour offrir une expérience de communication fluide et transparente.
Voici une vue d'ensemble des principaux composants de l'architecture WebRTC, en mettant l'accent sur les API, la signalisation, les protocoles et les codecs.

En combinant ces différents aspects, l'architecture de WebRTC offre une solution puissante et accessible pour les applications de communication en temps réel via les navigateurs web, ouvrant la porte à une large gamme de possibilités pour les utilisateurs et les développeurs.

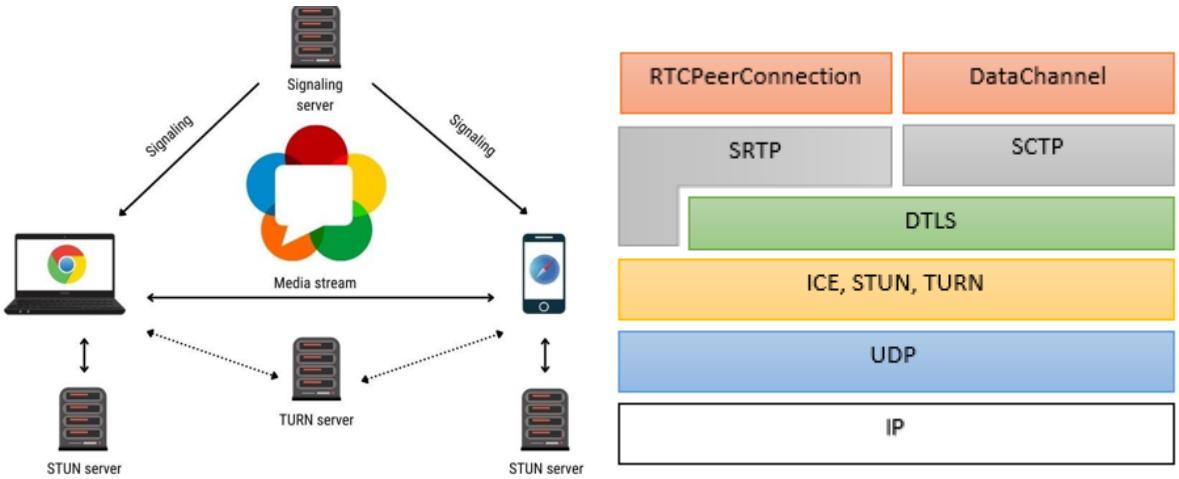

Figure II.4: L'architecture*[7]* & la pile de WebRTC

1. API WebRTC

***A**pplication **P**rogramming **I**nterface WebRTC* (API) WebRTC est une suite de technologies et d'interfaces de programmation qui permettent aux navigateurs web d'établir des connexions en temps réel, audio et vidéo, ainsi que de partager des données entre les utilisateurs sans avoir besoin de plugins ou de logiciels supplémentaires. Elle offre des fonctionnalités avancées de communication en temps réel directement intégrées dans les navigateurs web modernes.

L'API WebRTC permet aux développeurs de créer des applications de communication en temps réel sophistiquées, telles que des appels audio/vidéo, des conférences en ligne, des jeux multi-joueurs et bien plus encore. Elle offre une solution puissante et standardisée pour la communication directe entre les navigateurs, en évitant les coûts et les restrictions liés à l'utilisation de solutions tierces. Grâce à son intégration native dans les navigateurs modernes, l'API WebRTC offre une expérience utilisateur fluide, sécurisée et de haute qualité pour la communication en temps réel sur le web.

L'API WebRTC se compose de plusieurs composants clés qui fonctionnent ensemble pour permettre la communication peer-to-peer :

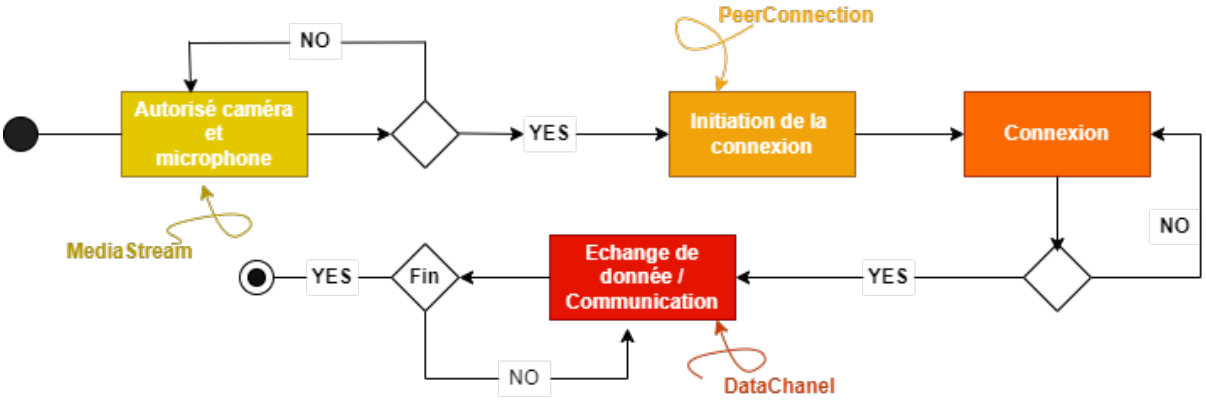

Figure II.5: API WebRTC

1.1 Media Stream

Media Stream est une représentation abstraite d'un flux audio et/ou vidéo. Il permet de contrôler le flux multimédia, que ce soit pour afficher le contenu à l'écran, l'enregistrer ou l'envoyer à un participant distant. Le *Media Stream* peut représenter un flux provenant d'une source distante (flux reçu) ou un flux local (flux envoyé) vers un point distant. Pour créer et utiliser un flux local, une application web doit demander l'accès à un périphérique d'enregistrement local tel qu'une webcam ou un microphone en utilisant la fonction *getUserMedia()*. L'application spécifie le type de média (audio ou vidéo) auquel elle souhaite accéder, et l'utilisateur donne son consentement en sélectionnant l'appareil dans l'interface du navigateur. L'application peut arrêter l'accès au flux en appelant la fonction *stop()* sur le *Local Media Stream* une fois qu'elle a terminé.

Media Stream se compose de zéro ou plusieurs objets *Media Stream Track*, qui comprennent un ou plusieurs canaux, et qui sont contenus dans une *Media Stream Track List*. Chaque *Media Stream Track* peut avoir un ou plusieurs canaux. Le canal représente la plus petite unité d'un flux multimédia, tel qu'un signal audio.

Les objets *Media Stream* ont une entrée et une sortie. Un *Local Media Stream* est un objet *Media Stream* généré par *getUserMedia()*, et qui a comme entrée source la caméra ou le microphone de l'utilisateur. La sortie décrit comment le consommateur utilise l'objet *Media Stream.*

1.2 Peer Connexion

Peer Connection permet à deux utilisateurs de communiquer directement de navigateur à navigateur. Représente une association avec un participant distant, qui est généralement une autre instance de la même application JavaScript exécutée sur l'extrémité distante.

La communication est coordonnée via un canal de signalisation. Une fois la connexion de l'abonné établie, les flux multimédias (objets *Media Stream* définis localement) peuvent être envoyés directement au navigateur distant.

Le mécanisme *Peer Connection* utilise le protocole ICE avec les serveurs STUN et TURN pour envoyer des médias en streaming UDP via des passerelles NAT et des pare-feu. ICE permet aux navigateurs d'obtenir suffisamment d'informations sur la topologie du réseau dans lequel ils sont déployés pour trouver le meilleur chemin utilisable. L'utilisation d'ICE fournit également des mesures de sécurité, car elle empêche les sites Web et les applications non fiables d'envoyer des données indésirables aux participants à la communication.

1.3 Data Channel

L'API *Data Channel* est conçue pour fournir des services de transport génériques qui permettent aux navigateurs Web d'échanger des données entre eux dans une communication bidirectionnelle p2p. Les travaux de normalisation de l'IETF ont abouti à la décision d'utiliser SCTP encapsulé avec DTLS pour les types de données non multimédias. L'encapsulation de SCTP sur DTLS sur UDP avec ICE fournit une fiabilité, une authentification de la machine et une intégrité de la transmission sécurisée.

L'API *Data Channel* est conçue pour être bidirectionnelle, avec un flux entrant et sortant de SCTP. En utilisant la fonction *CreateDataChannel*(), on peut configurer un nouveau *Data Channel* dans un bundle SCTP existant.

1. Signalisation

La signalisation dans WebRTC est le processus de communication entre les différentes parties impliquées dans une session de communication afin d'établir, modifier et terminer la connexion. Elle est utilisée pour échanger les informations nécessaires pour configurer les paramètres de communication tels que les adresses IP, les codecs pris en charge, les clés de chiffrement et les types de média.

L'étape de signalisation est la configuration de la conversation. Lorsque deux clients souhaitent communiquer, ils doivent disposer d'un canal d'information commun.
Si le client S souhaite communiquer avec le client H, voici les étapes de signalisation :

- Le client S ouvre une connexion d'écoute locale et envoie une requête au serveur de signalisation pour communiquer avec H. Cette requête héberge des informations sur les capacités du client (codecs média disponibles, version de l'API, etc.), ainsi que des identifiants de sécurité qui sont utilisés pour initier une connexion directe entre pairs.
- Le serveur de signalisation authentifie S, et transmet la demande à H.
- Le client H approuve/rejette la demande du client S.
    - Si la demande est approuvée, le client H ouvre une connexion d'écoute et envoie la réponse au serveur de signalisation.
- Le serveur de signalisation transmet la réponse au client S.
    - Si le client H a approuvé la demande, les clients S et H échangent des informations de communication directe. Cela inclut l'adresse IP où une interface d'écoute ouverte pour la communication,
    - Sinon si H décide de refuser l'appel, H peut envoyer une réponse de refus (réponse SDP) à S. Cette réponse peut inclure un message indiquant le refus de l'appel, éventuellement accompagné d'une raison spécifique.
- S reçoit la réponse de refus de H, le serveur peut afficher a S un message indiquant que l'appel a été refusé par H et donner la possibilité de prendre une autre action, comme rappelé ultérieurement ou essayer un autre contact.

Dans le cas où H accepte l'appel, les deux clients ouvrent un canal de communication direct l'un avec l'autre. Une condition préalable pour établir une communication WebRTC entre S et H est que les deux doivent utiliser une application ou un service Web qui implémente et supporte la communication WebRTC.
Cela peut être défini à l'aide d'un navigateur où les clients naviguent vers une adresse en ligne ou une page stockée localement, ou à l'aide d'une application native (par exemple, Java, C++) avec des ressources locales et distantes pré-stockées.

Après l'étape de signalisation vient la communication, une fois que les participants ont partagé leurs informations entre eux, un canal direct est établi et à partir de ce moment, les deux participants peuvent échanger des informations sans l'intervention du serveur de signalisation.

2. Protocoles WebRTC

WebRTC utilise plusieurs protocoles pour permettre la communication en temps réel entre les navigateurs web. Parmi ces protocoles clés on retrouve SIP, RTP/RTCP, STUN/TURN, TCP / UDP, TLS / DTLS et nous finirons avec l'architecture p2p.

Ces protocoles interagissent de différentes manières pour faciliter la transmission des flux multimédias et établir des connexions sécurisées.

2.1 Le protocole SIP

Session Initiation Protocol (SIP) est un protocole TCP/IP de couche application normalisé et standardisé par l'IETF (RFC 3261). Il a été conçu pour établir, modifier et terminer des sessions multimédia. Il prend en charge l'authentification et la localisation de multiples participants. S'il se charge de la négociation des médias, il laisse le soin à d'autres protocoles de transporter du texte, de la voix ou de la vidéo.[8]
SIP prend en charge six facettes de l'établissement et de la terminaison de communications multimédia :

- **Création des comptes :** il permet de créer les comptes des clients dans un fichier sip.conf.
- **Localisation de l'utilisateur :** détermination du système terminal à utiliser pour la communication.
- **Disponibilité de l'utilisateur :** détermination de la volonté de l'appelé à s'engager dans une communication.
- **Capacités de l'utilisateur :** détermination du support et des paramètres de support à utiliser.
- **Établissement de session :** "sonnerie", établissement des paramètres de session à la fois chez l'appelant et l'appelé.
- **Gestion de session :** y compris le transfert et la terminaison des sessions, la modification des paramètres de session, et l'invocation des services.

Lorsqu'il est combiné avec WebRTC, SIP joue un rôle essentiel dans l'établissement des sessions de communication entre les utilisateurs.

La signalisation SIP intervient pour faciliter la configuration de ces connexions, en établissant une session SIP entre les navigateurs. Cette session SIP permet d'échanger les informations de

signalisation nécessaires pour établir les paramètres de communication, tels que les adresses IP, les ports, les protocoles de transport, les codecs, etc.

2.2 Les protocoles TCP / UDP

➢ TCP

Transmission Control Protocol (TCP) est un protocole de transport orienté connexion qui envoie des données sous la forme d'un flux d'octets non structuré. En utilisant des numéros de séquence et des messages d'accusé de réception, TCP peut fournir à un nœud expéditeur des informations de livraison sur les paquets transmis à un nœud de destination. Lorsque des données ont été perdues en transit de la source à la destination, TCP peut retransmettre les données jusqu'à ce qu'une condition de temporisation soit atteinte ou jusqu'à ce que la livraison soit réussie. TCP peut également reconnaître les messages en double et les supprimer de manière appropriée. Si l'ordinateur expéditeur transmet trop rapidement pour l'ordinateur récepteur, TCP peut utiliser des mécanismes de contrôle de flux pour ralentir le transfert de données. TCP peut également communiquer des informations de livraison aux protocoles et applications de couche supérieure qu'il prend en charge. Toutes ces caractéristiques font de TCP un protocole de transport fiable de bout en bout.[9]

WebRTC utilise le protocole TCP pour la signalisation car il offre des garanties de fiabilité et d'ordre de livraison des messages. Il permet de s'assurer que les informations de configuration et de contrôle nécessaires à l'établissement de la connexion WebRTC sont correctement transmises et reçues entre les pairs.

Une fois la connexion TCP établie, les pairs utilisent cette connexion pour échanger des informations de contrôle et les paramètres des flux multimédias, tels que les codecs pris en charge, les adresses IP et les ports à utiliser pour l'échange des données multimédias.

Une fois les connexions de données établies, les flux audio et vidéo sont envoyés directement entre les pairs en utilisant les protocoles appropriés, tels que RTP pour la transmission des paquets multimédias et RTCP pour les rapports de contrôle et de surveillance de la qualité.

➢ UDP

User Datagram Protocol (UDP) est un protocole de transport utilisé dans les communications des applications en temps réel, Par exemple, la vidéoconférence en direct, le trafic vocal et les applications de jeu. UDP n'est pas un protocole fiable. Il ne se soucie pas de savoir si les données sont reçues ou non du côté récepteur. Il n'y a pas de mécanisme ACK ni de mécanisme de récupération d'erreur.[10]

Lorsqu'un utilisateur envoie des données audio ou vidéo, celles-ci sont divisées en petits paquets appelés datagrammes. Chaque datagramme est encapsulé dans un en-tête UDP qui contient des informations telles que les numéros de port source et de port de destination.

Les datagrammes UDP sont envoyés sur le réseau sans établir de connexion préalable avec le destinataire. Cela signifie que les datagrammes sont envoyés indépendamment les uns des autres et ne nécessitent pas de confirmation de réception. Ils sont simplement envoyés à l'adresse IP et au numéro de port de destination spécifiés.

Contrairement à TCP, UDP ne dispose pas de mécanismes intégrés de contrôle de flux pour réguler le débit des données. Cela signifie que les datagrammes sont envoyés aussi rapidement que possible, sans attente ni régulation du flux. Cela peut entraîner une congestion du réseau si la quantité de données envoyées dépasse la capacité du réseau.

UDP ne garantit pas la livraison des datagrammes ni leur ordre. Cela signifie que certains datagrammes peuvent être perdus en cours de transmission ou arriver dans un ordre différent de celui dans lequel ils ont été envoyés. Cette nature non fiable d'UDP est compensée par des mécanismes de correction d'erreurs et de résilience dans le cadre de WebRTC.

Lorsqu'un datagramme est perdu, il n'y a pas de mécanisme automatique pour le récupérer. Cependant, dans WebRTC, des mécanismes de correction d'erreurs tels que la duplication de paquets et le mécanisme de retransmission sont utilisés pour atténuer les effets des pertes de paquets. De plus, les codecs audio et vidéo utilisés dans WebRTC sont conçus pour être résilients aux pertes de paquets et fournir une qualité de communication acceptable même en présence de pertes.

2.3 Le protocole RTP / RTCP

Real-time Transport Protocol (RTP) est un protocole de transport qui permet de transporter les flux multimédias entre les participants d'une communication en temps réel.
Ce protocole assure une numérotation et un horodatage des paquets en encapsulant les données dans un nouvel entête. Il repose lui-même sur le protocole UDP, qui ne renvoie pas les trames perdues en cours de route, afin de privilégier l'enchaînement du son et des images, plutôt que l'intégrité des données.

Dans le contexte de WebRTC, RTP est utilisé pour acheminer les flux audio et vidéo entre les différentes parties impliquées dans une communication. L'émetteur (S) segmente les données audio/vidéo en paquets RTP, les marque avec des informations telles que la séquence, le timestamp et le type de média, puis les envoie à travers le réseau à destination du récepteur (H). Le récepteur reçoit les paquets RTP, les reconstitue dans l'ordre et les décode pour restituer l'audio/vidéo.

RTP est associé à un autre protocole appelé *Real-time Transport Control Protocol* (RTCP), qui est utilisé pour le contrôle et la surveillance de la qualité de la transmission RTP.
La fonction principale du RTCP est de fournir des commentaires sur la QoS dans la distribution des médias en envoyant périodiquement des informations statistiques telles que le nombre d'octets et de paquets envoyés, la perte de paquets, le changement de délai de paquet et le temps de retard dans les deux sens aux participants dans une session de diffusion multimédia. L'application peut utiliser ces informations pour contrôler la qualité des paramètres de service, pour ajuster les paramètres de la transmission et améliorer la qualité de l'expérience utilisateur.

2.4 Le protocole STUN / TURN

➢ STUN

Session Traversal Utilities for NAT (STUN) est un protocole utilisé dans WebRTC pour résoudre les problèmes de connectivité entre les pairs lors de l'établissement d'une communication en temps réel. Il permet de découvrir et de contourner les restrictions imposées par les pare-feu et les NAT présents sur les réseaux.
Un serveur STUN est utilisé par chaque homologué pour déterminer son adresse IP publique et est référencé par le cadre ICE lors de l'établissement de la connexion. Les serveurs STUN sont généralement accessibles au public et peuvent être utilisés librement par les applications WebRTC.

Lorsqu'un pair WebRTC souhaite établir une communication avec un autre pair, il commence par rechercher un serveur STUN disponible. Cette information peut être préconfigurée ou obtenue dynamiquement auprès d'un service de découverte de serveurs STUN.

Une fois le serveur STUN identifié, le pair envoie un message STUN (requête) au serveur STUN en utilisant le protocole UDP. Le message STUN contient des informations sur la connectivité du pair et demande au serveur STUN de lui fournir des détails sur son adresse IP et son port.

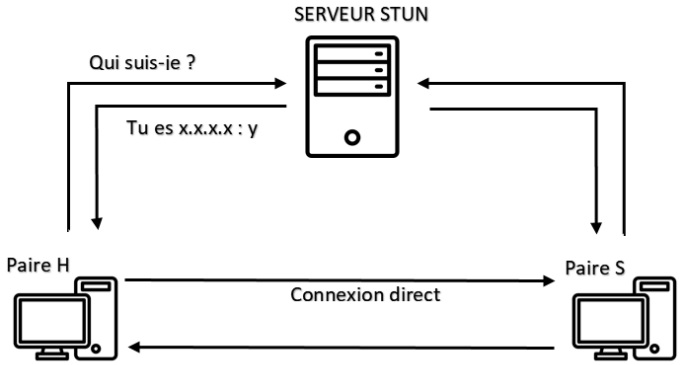

Figure II.6: Utilisation de protocole STUN

Le serveur STUN reçoit la requête du pair et répond avec un message STUN contenant les informations d'adresse IP et de port du pair. Ces informations peuvent inclure l'adresse IP publique, l'adresse IP locale et les ports utilisés pour la communication.

Le pair utilise les informations d'adresse IP et de ports fournis par le serveur STUN pour établir des connexions avec d'autres pairs. Il peut les transmettre à d'autres pairs via des mécanismes de signalisation tels que SIP ou SDP, permettant ainsi aux pairs de s'échanger les informations nécessaires pour établir des connexions directes.

Il convient de noter que STUN ne résout pas tous les problèmes de connectivité, en particulier lorsque les pare-feu ou les NAT sont très restrictifs, c'est là que TURN est nécessaires.

- **TURN**

Traversal Using Relays around NAT (TURN) est utilisé lorsque l'établissement d'une communication P2P échoue en raison de restrictions strictes imposées par les pare-feu ou les NAT. Lorsqu'un pair ne peut pas établir une connexion directe, une option de secours peut être fournie via un serveur TURN, il est utilisé pour acheminer le trafic à travers un serveur TURN.

En relayant le trafic entre homologues, la communication WebRTC peut être assurée, mais peut souffrir de dégradations de la qualité des médias et de la latence. Les serveurs TURN peuvent garantir un succès élevé dans la configuration des appels, quel que soit l'environnement de l'utilisateur final. Comme les données sont envoyées via un serveur intermédiaire, la bande passante du serveur est également consommée. Si de nombreux appels sont acheminés simultanément via le serveur, la bande passante devienne de taille considérable.

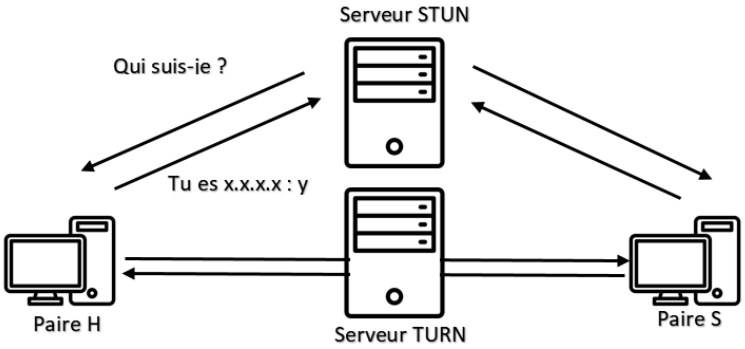

Figure II.7: Utilisation de protocole TURN.

Lorsqu'un pair WebRTC constate qu'il ne peut pas établir une connexion directe avec un autre pair, il doit rechercher un serveur TURN disponible. Il peut obtenir cette information à partir de sa configuration ou en utilisant des protocoles tels que STUN, pour interroger un serveur TURN.

Le pair établit une connexion avec le serveur TURN à l'aide du protocole TURN, cela implique l'échange d'un message de demande d'allocation de relais avec le serveur TURN.
Le serveur TURN alloue une adresse IP et un port relais pour le pair. Cette adresse est utilisée pour acheminer le trafic entre les pairs qui ne peuvent pas communiquer directement.

Le serveur TURN agit comme un relais en transférant les paquets de données entre les pairs. Il reçoit les paquets d'un pair, les décapsule, puis les ré-encapsule pour les envoyer au destinataire. Le processus est répété pour les paquets de données dans les deux sens.
Le protocole TURN prend en charge des mécanismes de sécurité pour protéger les communications. L'authentification est utilisée pour vérifier l'identité des pairs et empêcher les accès non autorisés aux serveurs TURN.

Une fois que la session de communication est terminée ou que les connexions directes deviennent possibles, le pair peut libérer les ressources du serveur TURN en envoyant un message de libération.

2.5 Les protocoles TLS / DTLS

➢ TLS

Transport Layer Security (TLS) est utilisé pour sécuriser la phase de signalisation et l'échange des informations nécessaires à l'établissement de la connexion entre les pairs.
Lorsqu'un appel WebRTC est initié, les pairs utilisent la signalisation pour échanger les informations nécessaires à l'établissement de la connexion, Une fois que les pairs ont échangé les informations de signalisation, ils utilisent le protocole TLS pour négocier les paramètres de sécurité de la connexion, tels que les algorithmes de chiffrement, les méthodes de vérification d'authenticité et les clés de session.

TLS permet l'authentification mutuelle entre les pairs pour vérifier leur identité. Cela se fait à l'aide de certificats numériques qui sont émis par des autorités de certification de confiance. Les certificats permettent de garantir que les pairs sont légitimes et de prévenir les attaques d'usurpation d'identité.

➢ DTLS

Datagram Transport Layer Security (DTLS*)* est une version adaptée de TLS conçue spécifiquement pour les protocoles de transport basés sur des datagrammes tels qu'UDP, qui est utilisé par WebRTC.
DTLS a tendance à avoir une latence plus faible, car il n'offre pas la même garantie de livraison fiable des paquets.
DTLS est plus flexible en termes de validation des certificats, car il est souvent utilisé dans des environnements où les certificats peuvent être auto-signés ou ne pas être vérifiés par une autorité de certification (CA).

DTLS est utilisé lorsque les deux clients ouvrent une connexion directe entre eux, pour un cryptage asymétrique de leur communication.
Lorsque les deux parties commencent une communication directe, chaque client envoie à l'autre un certificat TLS cette empreinte est utilisée pour garantir l'authenticité du certificat TLS une fois reçu.

DTLS est plus couramment utilisé dans des applications en temps réel, telles que la VoIP et la vidéoconférence, où la latence et la perte de paquets peuvent être mieux tolérées.
DTLS, en raison de sa conception sans connexion, peut offrir une latence et un débit plus faible.

3. Codecs Audio / Vidéo

Dans un système de téléphonie sur IP, après la numérisation de la voix, celle-ci se retrouve compressée, on parle à ce moment de codec. Il s'agit d'un circuit imprimé de compression du son et permet ainsi de réduire le volume de données échangées entre deux correspondants, on parle souvent de : Compression / Décompression. La voix numérisée est donc compressée selon l'un des formats de codec puis insérée dans un paquet IP.

Le codec joue un rôle important dans la mise en place d'une téléphonie IP, car premièrement la qualité sonore dépend du codec utilisé (Taux de compression), deuxièmement, en téléphonie sur IP, la bande passante peut varier selon les codecs utilisés.

3.1 G.711

Le G.711 est un format de codage standard utilisé pour la transmission des signaux audio sur les réseaux numériques. Il s'agit d'une norme de l'UIT qui est utilisée pour compresser et coder les signaux audio dans un format numérique. L'algorithme de codage fonctionne en compressant le signal audio en un flux binaire. Le flux binaire est ensuite divisé en trames de 8 bits de longueur. Les trames sont ensuite codées à l'aide d'un algorithme appelé modulation par impulsions et codage (PCM). [11]

Le G.711 est largement utilisé dans l'industrie des télécommunications parce qu'il s'agit d'une norme et qu'il est très fiable. Il s'agit également d'un algorithme de codage relativement simple et facile à mettre en œuvre. Le G.711 fournit également une bonne qualité audio qui est utilisé dans les applications VoIP.

On outre il est utilisé dans un large éventail d'applications, notamment la VoIP, la téléphonie par Internet et la vidéoconférence. Il est également utilisé dans les systèmes téléphoniques numériques.[11]

3.2 Opus

Opus est un codec audio polyvalent qui combine l'équilibre d'une compression de signal audio de haute qualité avec de faibles taux de retard. Sa flexibilité réside dans l'adaptation aux changements de capacité de bande passante du canal et la prise en charge de tout type d'encodage audio. De nos jours, le codec audio Opus est considéré comme le meilleur à tous égards parmi ses rivaux, car sa qualité est même supérieure au MP3 largement utilisé.[12]

Opus est l'un des deux codecs vocaux sélectionnés comme obligatoires à implémenter dans WebRTC (l'autre codec est G.711). En tant que codec, Opus est capable de prendre en charge la bande étroite et jusqu'à la pleine bande stéréo tout en utilisant des débits binaires faibles avec une résilience élevée.[13]

3.3 vp8

VP8 est annoncé par Google comme l'un des meilleurs formats vidéo ayant le meilleur débit de données de qualité d'image et la meilleure vitesse d'encodage. Le meilleur avantage de VP8 est qu'il s'agit d'un substitut libre de droits de H.264. Il s'agit d'un format spécifique pour l'encodage et le décodage d'une vidéo de haute qualité sous forme de fichier ou de flux binaire. C'est gratuit car Google a publié tous les brevets VP8 sous une licence publique libre de droits. Près de 90 % ou plus de toutes les sessions vidéo WebRTC utilisent VP8. Il est conçu pour fournir une vidéo de haute qualité pour le Web et les appareils mobiles vous avez envoyé.[14]

Ce format est reconnu dans de nombreuses applications industrielles telles que HTML5, Web Real-Time Communication et la lecture vidéo dans différents navigateurs.
Parmi ces avantages le VP8 est un codec vidéo gratuit, il est le format de fichier vidéo le plus progressif, il fait une amélioration de la résistance à la perte de trame et il fait un décodage vidéo à grande vitesse**.**[14]

3.4 VP9

Google a développé le codec VP9 en tant que norme de codage vidéo open source libre de droits en tant que successeur de VP8. Il a été conçu à l'origine pour compresser le contenu ultra HD sur YouTube, car il étend et améliore l'efficacité de codage de son prédécesseur.[14]

Parmi ces avantages il prend en charge une gamme complète de cas d'utilisation Web et mobile, il fait une compression à faible débit binaire à l'ultra-HD de haute qualité, avec une prise en charge supplémentaire d'encodage, il peut réduire les débits binaires vidéo jusqu'à 50% par rapport aux autres, il est préparé pour le streaming adaptatif et est utilisé par YouTube et d'autres fournisseurs de vidéos Web bien connus et il prend en charge les résolutions vidéo supérieures à 1080p sont modifiées via VP9 avec une compression sans perte. [14]

Conclusion

Dans le fonctionnement de WebRTC, les API, la signalisation, les protocoles et les codecs travaillent en étroite collaboration pour permettre une communication en temps réel. Les API fournissent une interface permettant aux développeurs d'accéder aux fonctionnalités de communication du navigateur, tandis que la signalisation facilite l'échange d'informations pour établir la connexion entre les pairs. Les protocoles, comme RTP, assurent le transport des données audio et vidéo, tandis que les codecs compressent et décompressent les flux multimédias.

Ces composants sont synchronisés de manière harmonieuse, les API étant utilisées pour configurer les flux, la signalisation échangeant les informations nécessaires, les protocoles transportant les données et les codecs en garantissant une transmission efficace. Cette synchronisation permet une expérience fluide et immersive lors des communications en temps réel via les navigateurs web.

Chapitre III : Conception de notre système

Introduction

Avant de programmer l'application et de se lancer dans l'écriture du code, il faut d'abord organiser la réalisation en définissant les modules et les étapes de la réalisation, Le but de ce chapitre est de comprendre le contexte du système, pour mieux clarifier les besoins. Nous essaierons donc d'exprimer les besoins sous forme de schémas.

Au cours de ce chapitre, nous examinerons l'architecture globale du système, détaillant les composants clés et leurs interconnexions. Ensuite, nous nous pencherons sur les diagrammes de cas d'utilisation, qui permettent de décrire les interactions entre les acteurs du système et ses différentes fonctionnalités. Enfin, nous explorerons les diagrammes de séquence, qui illustrent de manière séquentielle les actions entre les composants du système.

Tout cela pour visualiser et décrire le comportement de notre système dans différentes situations.

1. Architecture de notre système

L'architecture de notre projet repose sur une combinaison de plusieurs composants techniques qui travaillent en harmonie pour assurer le bon fonctionnement de la solution de communication basée sur WebRTC.

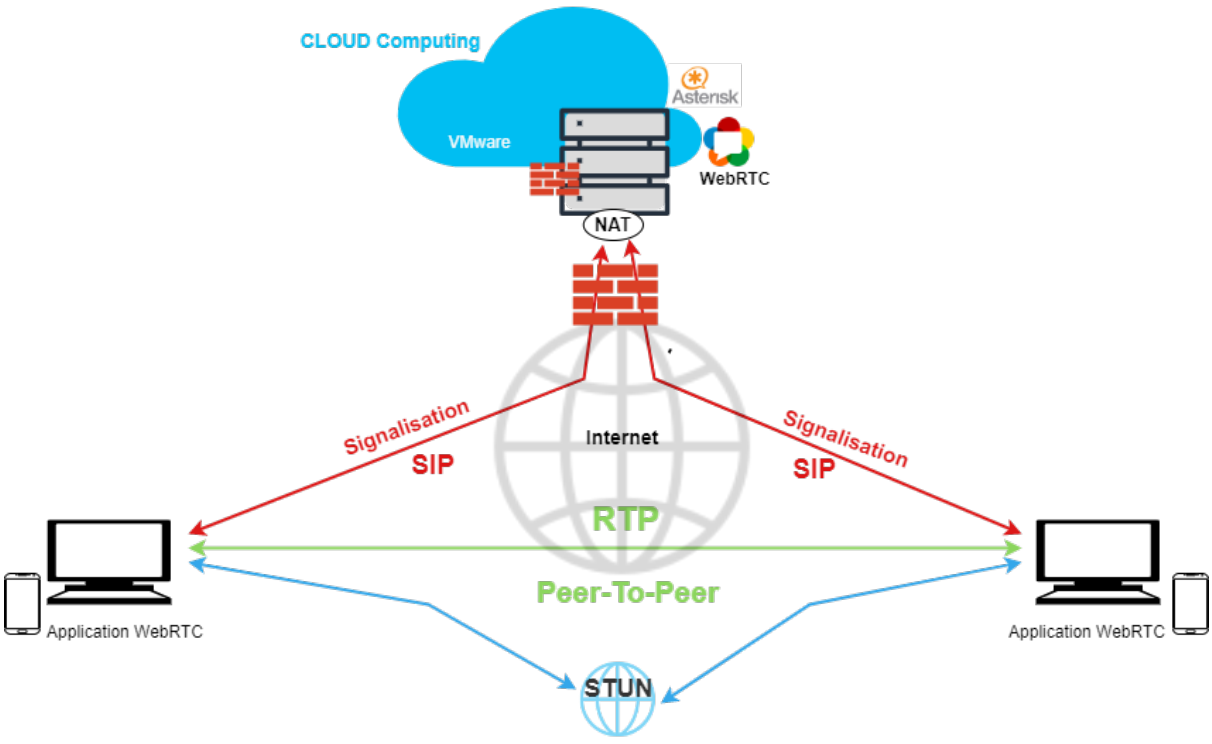

Figure III.8: Architecture de notre système

L'architecture de notre système repose sur un modèle P2P, où les utilisateurs communiquent directement entre eux sans passer par un serveur centralisé pour la transmission des flux média.

Le processus débute avec la signalisation, lorsqu'un utilisateur souhaite initier une session de communication, il envoie une requête de signalisation au serveur en utilisant le protocole SIP, cette requête contient des informations sur l'identité de l'utilisateur appelant et de l'utilisateur appelé. Les utilisateurs échangent leurs adresses IP et numéros de port via la signalisation pour établir une connexion directe entre eux. Cependant, la présence des dispositifs de sécurité tels que les pare-feu (logiciel et matériel) ainsi que les NAT peuvent compliquer l'établissement de cette connexion directe, et c'est pour cette raison que le protocole STUN est utilisé, il permet aux utilisateurs de découvrir leurs adresses IP et numéros de port publics en contournant les NAT.

Une fois que la signalisation est terminée et que les utilisateurs sont prêts à établir la communication, ils utilisent WebRTC pour établir un flux média P2P, cela signifie que les données audio et vidéo sont transmises directement entre les utilisateurs en utilisant le protocole RTP sans passer par le serveur de signalisation.

Notre architecture inclut également un environnement Cloud basé sur VMware, qui joue un rôle dans le déploiement et la gestion de l'infrastructure. Le Cloud fournit des ressources virtuelles pour exécuter les différents composants du système, offrant ainsi une flexibilité et une évolution significative.

2. Identification des acteurs et des besoins

La spécification des exigences ou la spécification des besoins et des acteurs est la phase de démarrage de toute application à développer dans laquelle nous identifierons les besoins de notre application, Ces besoins sont regroupés dans les schémas des cas d'utilisation, dont on doit couvrir principalement les besoins fonctionnels suivants :

Dans le cas d'un employé :

- S'authentifier.
- Envoyer des messages à un autre employé.
- Appeler un autre employé (Appel audio).
- Appeler un autre employé (Appel vidéo).

Dans le cas d'un client :

- Lancer un appel (audio).

3. Diagrammes de cas d'utilisation

Cette visualisation nous permettra de mieux comprendre les besoins et les attentes des utilisateurs, ce qui facilitera la conception et le développement de notre solution. Dans notre projet nous avons trois diagrammes de cas d'utilisation :

a. L'employé

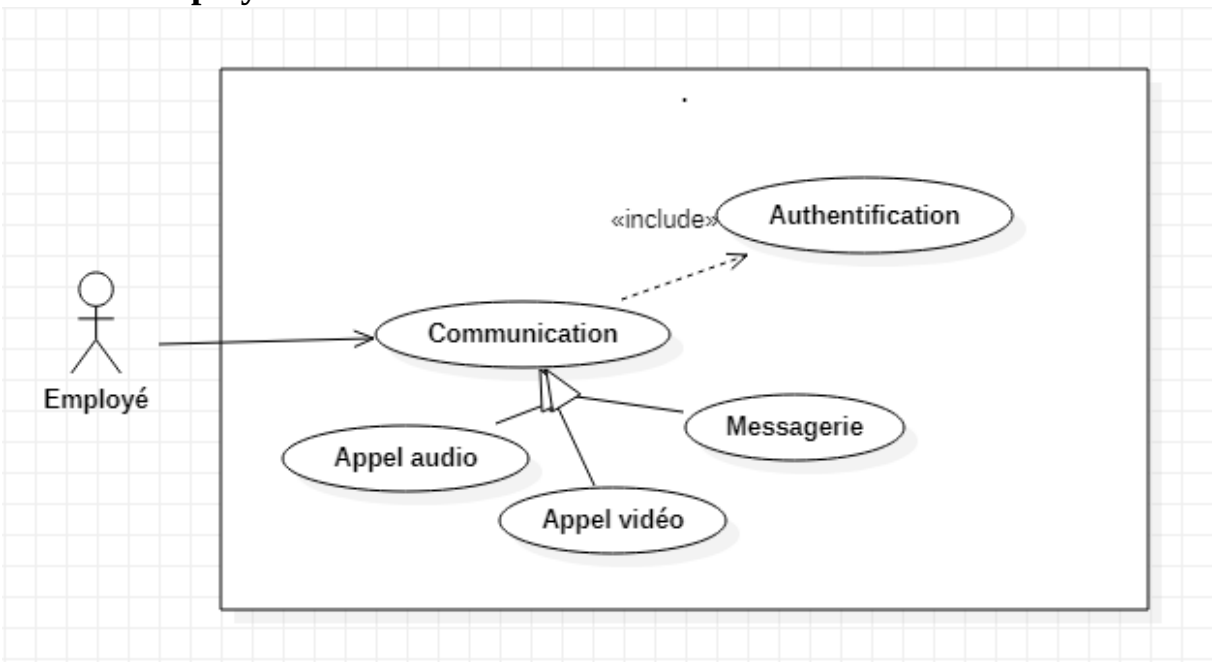

Figure III.9: Diagrammes de cas d'utilisation de l'employé

- ✓ Après être authentifier, l'employé à la possibilité de lancer un appel audio, vidéo ou de démarrer une conversation via une messagerie avec les autres employés.
- ✓ Lorsque l'employé S décide de lancer un appel avec H, il sélectionne le contact H avec lequel il souhaite communiquer.
- ✓ En fonction de l'option choisie (audio, vidéo ou messagerie), une demande d'autorisation appropriée est envoyée à l'employé H.
- ✓ Une fois que l'employé H accepte la demande, une connexion WebRTC est établie entre leurs navigateurs, permettant la communication en temps réel.

b. Client

Figure III.10: Diagrammes de cas d'utilisation du client

Après avoir consulté le site de l'entreprise ICOSNET, le client trouve une option pour lancer un appel audio en cliquant sur le bouton appel.

Une fois l'autorisation accordée, une connexion WebRTC est établie entre le navigateur du client et le téléphone IP de la réception.

Le client peut alors parler en direct avec la réception via un appel audio

4. Diagrammes de Séquence

Les diagrammes de séquence sont des outils de modélisation qui permettent de représenter visuellement le déroulement des interactions entre les différents acteurs et composants d'un système. Ils sont utilisés pour décrire et analyser le flux chronologique des messages échangés entre les entités lors de l'exécution d'un scénario spécifique. Dans notre projet nous avons deux diagrammes.

4.1 La connexion SIP

Pour le cas des employés, avant de pouvoir accéder à leur compte ils doivent d'abord se connecter. Ce diagramme de séquence montre comment l'employé envoie une requête de connexion au serveur, comment celui-ci répond avec une confirmation, et comment les échanges de messages continuent jusqu'à ce que la connexion soit établie.

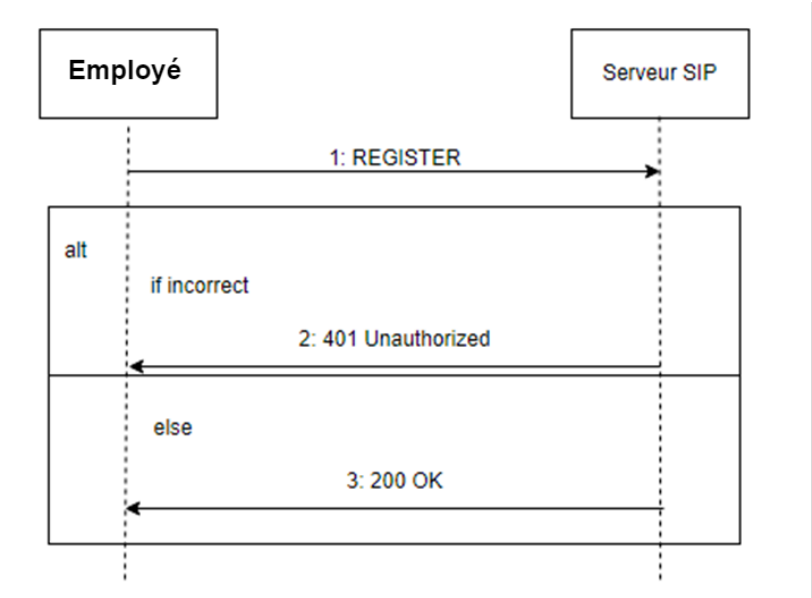

Figure III.11: Diagramme de séquence pour se connecter

- ✓ Le processus de connexion commence par l'employé qui envoie une requête REGISTER au serveur SIP pour se connecté à son compte pour pouvoir utiliser les services de communication.
- ✓ Dans le cas où les informations sont fausses, le serveur SIP répond avec le code de réponse 401 Unauthorized, pour indiquer que l'employé doit fournir les bonnes informations d'authentification avant de pouvoir terminer la connexion.
- ✓ Dans le cas où l'employé envoie les informations d'authentification correctes, le serveur SIP vérifie que les informations fournies sont valides et il renvoie une réponse 200 OK pour indiquer que la connexion est établie. Il est maintenant considéré connecté et peut utiliser les services et les fonctionnalités associés à son compte, tel que passer ou recevoir des appels.

4.2 L'appel SIP

Le deuxième diagramme de séquence, entre l'utilisateur S, le serveur SIP et l'utilisateur H (sachant que l'utilisateur H est la réception dans le cas d'un appel d'un client) lors de l'initiation, de l'établissement et de la Mise en fin d'un appel téléphonique. Il décrit le déroulement chronologique des messages tels que les requêtes INVITE, les réponses 100 Trying, 180 Ringing, 200 OK, les échanges de médias RTP/RTCP, et la requête BYE pour mettre fin à l'appel.

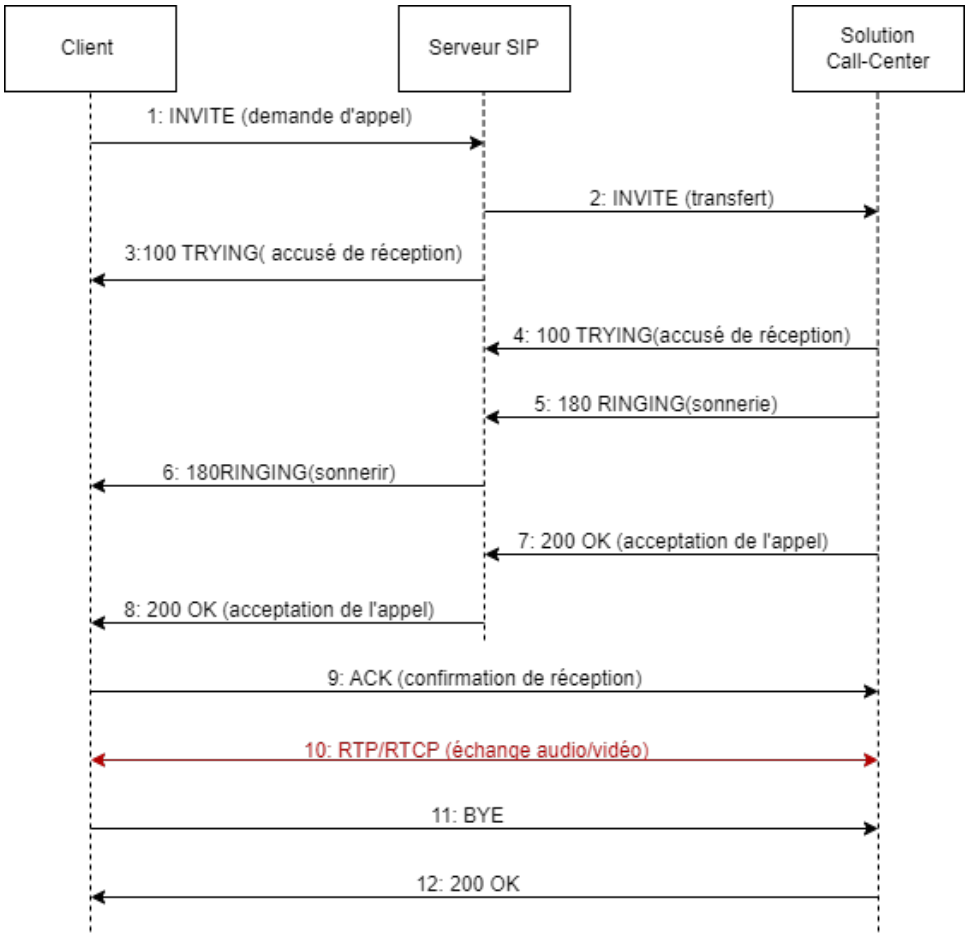

Figure III.12: Diagramme de séquence des appels SIP

1. INVITE (de l'utilisateur S au serveur SIP) : L'utilisateur S envoie une requête INVITE au serveur SIP pour initier un appel avec l'utilisateur H.

2. INVITE (du serveur SIP a l'utilisateur H) : Le serveur SIP relaie la requête INVITE vers l'utilisateur H pour établir la session d'appel.

3. 100 Trying (du serveur SIP à l'utilisateur S) : Le serveur SIP envoie une réponse 100 Trying à l'utilisateur S pour indiquer que la requête a été reçue et que le traitement est en cours.

4. 100 Trying (de l'utilisateur H au serveur SIP) : L'utilisateur H envoie également une réponse 100 Trying au serveur SIP pour indiquer que la requête a été reçue et que le traitement est en cours.

5. 180 Ringing (de l'utilisateur H au serveur SIP) : L'utilisateur H envoie une réponse 180 Ringing au serveur SIP pour indiquer que l'appel est en cours de sonnerie côté appelé.

6. 180 Ringing (du serveur SIP à l'utilisateur S) : Le serveur SIP relaie la réponse 180 Ringing à l'utilisateur S pour indiquer que l'appel est en cours de sonnerie côté appelé.

7. 200 OK (L'utilisateur H au serveur SIP) : L'utilisateur H envoie une réponse 200 OK au serveur SIP pour indiquer que l'appel a été accepté et que la session d'appel peut être établie.

8. 200 OK (du serveur SIP à l'utilisateur S) : Le serveur SIP relaie la réponse 200 OK à l'utilisateur S pour indiquer que l'appel a été accepté et que la session d'appel peut être établie.

9. ACK (de l'utilisateur S à l'utilisateur H) : L'utilisateur S envoie une requête ACK à L'utilisateur H pour confirmer la réception de la réponse 200 OK et établir la session d'appel.

10. RTP/RTCP (entre l'utilisateur S et L'utilisateur H) : Une fois la session d'appel établit, des échanges de flux de médias RTP et de contrôle RTCP ont eu lieu entre l'utilisateur S et l'utilisateur H pour la transmission des données audio/vidéo.

11. BYE (de l'utilisateur S à l'utilisateur H) : L'utilisateur S envoie une requête BYE a l'utilisateur H pour terminer l'appel.

12. 200 OK (de l'utilisateur H à l'utilisateur S) : L'utilisateur H envoie une réponse 200 OK à l'utilisateur S pour confirmer la réception de la requête BYE et terminer la session d'appel.

4.3 Messagerie SIP

Dans ce diagramme de séquence, les utilisateurs S et H se connectent directement l'un à l'autre sans l'utilisation d'un serveur central. Les messages sont échangés directement entre les deux utilisateurs. Voici un exemple d'échange de message entre un employé S et un autre H

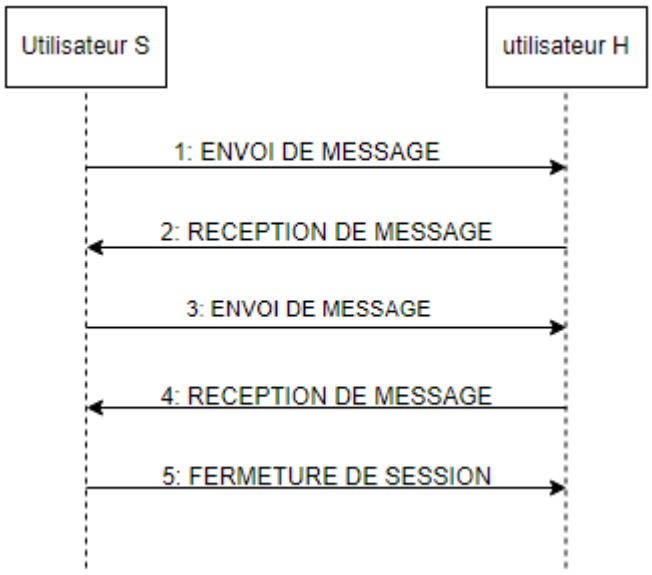

Figure III.13: Diagramme de séquence pour les messages

1. L'utilisateur S envoie un message à l'utilisateur H.
2. L'utilisateur H reçoit le message de l'utilisateur S.
3. L'utilisateur H envoie un message à l'utilisateur S.
4. L'utilisateur S reçoit le message de l'utilisateur H.
5. L'utilisateur S ou l'utilisateur H ferme la session de chat.

Conclusion

En conclusion, la partie conception de notre projet se distingue par une architecture bien définie et l'utilisation des diagrammes de cas d'utilisation et de séquence pour visualiser et comprendre les différentes facettes de notre système.

Les diagrammes de cas d'utilisation ont joué un rôle crucial dans notre conception, en nous permettant de décrire de manière détaillée les fonctionnalités spécifiques de notre solution. Grâce à cela, nous avons pu mieux comprendre les scénarios d'utilisation.

Les diagrammes de séquence ont également été d'une grande utilité pour modéliser les interactions entre les différents composants du système lors de l'exécution des tâches spécifiques. Ces diagrammes nous ont permis de visualiser le flux d'informations et de contrôler les étapes impliquées dans la communication basée sur WebRTC.

En combinant l'architecture du système, les diagrammes de cas d'utilisation et les diagrammes de séquence, nous serons en mesure de concevoir une solution de communication WebRTC robuste, répondant aux besoins fonctionnels et offrant une expérience utilisateur fluide et sécurisée. Cette approche de conception nous permettra d'avoir une vision claire du fonctionnement du système et de mettre en place les bonnes pratiques pour assurer son bon déploiement et sa maintenance.

Chapitre IV : Implémentation de notre système

Introduction

WebRTC est une technologie révolutionnaire qui permet aux navigateurs web modernes de communiquer directement, sans nécessiter l'installation logicielle supplémentaire, offrant ainsi une expérience utilisateur fluide et naturelle.

L'implémentation de ce projet permettra de mettre en place une solution de communication entre deux utilisateurs basée sur WebRTC nommé ***ICall***, permettant aux utilisateurs de passer des appels audio et vidéo en temps réel, quel que soit leur emplacement géographique. Pour cela nous étudierons les différentes étapes, depuis l'installation et la configuration d'Asterisk en tant que serveur de communication, l'intégration du protocole SIP, la gestion des comptes pour la signalisation des appels et l'utilisation de WebRTC pour la transmission audio et vidéo jusqu'au développement du site de communication.
En fin nous testerons notre solution de communication pour garantir son bon fonctionnement.

1. Environnement de travail

1.1 Environnement matériel

Pour réaliser ce projet nous aurons besoin de trois machines, elles peuvent être sous la forme de machines virtuelles :

- La première machine aura le rôle du serveur. elle devra donc exécuter un système d'exploitation CentOS pour permettre l'installation d'Asterisk et le support des modules pour le serveur web et le WebRTC.
- Les deux autres machines auront le rôle des clients. Pour fonctionner, ils auront uniquement besoin d'un navigateur, comme Google Chrome.

1.2 Environnement logiciel coté serveur

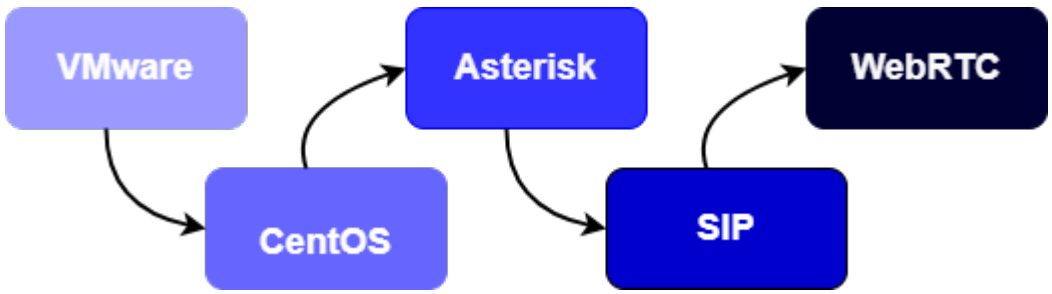

Figure IV.14: Environnement logiciel coté serveur

L'environnement logiciel côté serveur pour la mise en place de notre système peut être configuré en utilisant une VMware pour la virtualisation, CentOS comme système d'exploitation, Asterisk comme serveur de communication IP et les comptes SIP permettent l'identification et l'authentification des utilisateurs et des appareils tout en implémentant le WebRTC.

1.2.1 Installation d'une VMware

Une machine virtuelle est une instance logicielle d'un système informatique complet y compris du matériel, du système d'exploitation et des applications. Elle est isolée du système hôte sur lequel elle est exécutée et peut être configurée avec des ressources matérielles dédiées tel que la mémoire, le processeur et le stockage.

Figure IV.15: Interface du VMware ESXi

Dans notre projet nous avons travaillé avec VMware ESXi qui est un système d'exploitation spécialement conçu pour la virtualisation du serveur.

Voici les étapes pour créer une machine virtuelle :

- Télécharger et installer VMware.
- configurer les paramètres réseau, tels que l'adresse IP, le masque de sous-réseau, la passerelle et les serveurs DNS.
- Configurer le mot de passe administrateur.
- Créer une Machine virtuelle
- Configurer les paramètres de la machine virtuelle en fonction de nos besoins, en particulier la RAM et la taille du disque dur virtuel.

1.2.2 Installation du CentOS

Community Enterprise Operating System (CentOS) est une distribution Linux open-source basée sur les sources de la distribution *Red Hat Enterprise Linux* (RHEL). Il est reconnu pour sa stabilité et sa fiabilité, ainsi que pour ses performances élevées, il est conçu pour être compatible avec les mises à jour de sécurité et les correctifs de RHEL.

Travailler avec CentOS peut offrir plusieurs avantages pour notre projet de communication basé sur WebRTC, parmi les raisons pour lesquelles nous avons choisi CentOS c'est sa compatibilité avec Asterisk et SIP.

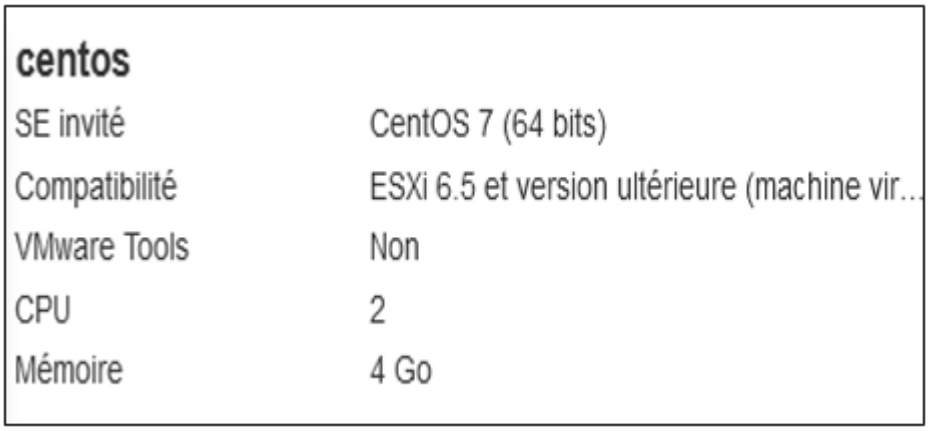

Figure IV.16: Paramètres du CentOS

Voici les étapes pour l'installation de CentOS :

- Créer une nouvelle machine virtuelle dans l'interface de gestion de VMware ESXi.
- Choisir un nom pour la machine virtuelle.
- Sélectionner 'Linux' comme type d'OS et 'CentOS 64-bits' comme version.
- Démarrer la machine virtuelle et commencer l'installation de CentOS en suivant les instructions de l'écran.
- Installer un logiciel Putty qui est un programme open source qui permet de se connecter à des serveurs à distance via des protocoles tels que SSH.
- Ouvrir une session SSH sur notre machine CentOS.
- Mettre à jour les packages système.

```
[root@localhost ~]# yum install wget nano git epel-release -y
```

Figure IV.17: Mettre à jour les package.

1.2.3 Installation du pare-feu

Un Pare-feu (*Firewall*) est un system de sécurité réseau qui surveille et filtre le trafic réseau entrant et sortant en s'appuyant sur des politiques de sécurité préalablement établies par l'entreprise. Un Pare-feu est essentiellement la barrière située entre un réseau interne privé et l'Internet public. Le principal objectif d'un Pare-feu est de permettre l'entrée de trafic non menaçant et d'empêcher l'entrée de trafic dangereux.[15]
Dans le cadre de notre projet, nous avons renforcé la sécurité de notre serveur en installant un pare-feu logiciel en plus de notre pare-feu matériel (de l'entreprise) déjà en place.
Le pare-feu logiciel que nous avons ajouté fonctionne au niveau de l'OS du serveur, ce qui nous permet d'exercer un contrôle plus granulaire sur les flux de données entrants et sortants.

Cette double couche de protection nous offre plusieurs avantages, le pare-feu matériel agit en tant que première ligne de défense en bloquant les attaques au niveau du réseau, telles que les

tentatives d'intrusions externes ou les scans de port, et le pare-feu logiciel se charge de surveiller et de filtrer le trafic à un niveau plus détaillé, offrant une protection supplémentaire contre les menaces qui pourraient contourner le pare-feu matériel.

Voici les étapes pour l'installation du pare-feu

- installation du pare-feu et son activation

```
[root@localhost ~]# yum install firewalld -y
[root@localhost ~]# systemctl start firewalld
```

Figure IV.18: Installation de pare-feu

- Configurer les règles de pare-feu, plus précisément ajouter les ports spécifiés à la zone publique en autorisant le trafic TCP ou UDP (les ports spécifiés sont 10000-20000/udp, 10000-20000/tcp, 8089/tcp, 443/tcp, 80/tcp et 22/tcp).

```
[root@localhost ~]# firewall-cmd –zone=public –add-port=10000
-20000/udp –permanent
[root@localhost ~]# firewall-cmd –zone=public –add-port=10000
-20000/tcp –permanent
[root@localhost ~]# firewall-cmd –zone=public –add-port=8089
/tcp –permanent
[root@localhost ~]# firewall-cmd –zone=public –add-port=443
/tcp –permanent
[root@localhost ~]# firewall-cmd –zone=public –add-port=80
/tcp –permanent
[root@localhost ~]# firewall-cmd –zone=public –add-port=22
/tcp –permanent
```

Figure IV.19: Configuration des règles du pare-feu

1.2.4 Installation de l'Apache

Apache est un serveur web open-source utilisé pour héberger des sites web et des applications web. Il est l'un des serveurs web les plus populaires, il est compatible avec différents systèmes d'exploitation tels que Linux, Windows et MacOS. Apache est connu pour sa sécurité et sa stabilité et offre une grande flexibilité pour la configuration et la personnalisation. Il est également extensible grâce à l'utilisation de modules tiers pour ajouter des fonctionnalités supplémentaires.

L'utilisation d'Apache dans notre projet permet de gérer les requêtes HTTPS, il est compatible avec les protocoles de sécurité, tels que SSL/TLS, permettant d'établir des connexions sécurisées avec les utilisateurs WebRTC, gestion des certificats SSL, flexibilité de l'architecture. Il garantit que les échanges de données entre les clients et le serveur se font de

manière cryptée, protège la confidentialité des informations sensibles et facilite la gestion des certificats SSL/TLS pour les connexions sécurisées.

Voici les étapes pour l'installation de l'Apache :

- Installer et démarrer le serveur web Apache sur CentOS

```
[root@localhost ~]# yum install httpd -y
[root@localhost ~]# systemctl start httpd
```

Figure IV.20: Installation du serveur web Apache.

- Installer un certificat.
- Accorder un nom de domaine au projet.
- Installez Jansson .

```
[root@localhost ~]#sudo git clone https://github.com/akheron
/jansson.git
[root@localhost ~]#sudo make
[root@localhost ~]#sudo make install
```

Figure IV.21: Installation de Jansson

1.2.5 Installation d'Asterisk et création des objets PJSIP

Asterisk est un logiciel de serveur de communication IP privé open source conçu pour les systèmes Linux, qui prend en charge une variété de protocoles y compris SIP, l'intégration d'Asterisk avec SIP permet d'établir, de gérer et de terminer des sessions de communication, tels que des appels audio et vidéo entre les utilisateurs.

PJSIP est une bibliothèque SIP open source largement utilisée et bien prise en charge par la communauté. PJSIP gère les aspects techniques complexes de la communication SIP tels que la négociation des codecs, le traitement des messages SIP et la gestion des sessions.

Dans le cas de notre projet, Asterisk semble répondre à nos exigences, il est open source, flexible, extensible et offre une large gamme de fonctionnalités, il est également très bien documenté et il peut être facilement intégré à d'autres systèmes et logiciels.

De plus, les objets PJSIP sont hautement configurables, ce qui nous permet d'adapter leurs comportements et de définir des paramètres tels que les codecs audio/vidéo pris en charge, les mécanismes d'authentification et les options de sécurité.

Voici les étapes pour installer Asterisk et pour créer des objets PJSIP dans notre projet :

- Installer PJSIP sur CentOS 7

```
#git clone https://github.com/pjsip/pjproject.git
#make
#sudo make install
```

Figure IV.22: Installation du PJSIP dans CentOS

Ces commandes permettent de cloner le référentiel Git de PJSIP, configurer le projet PJSIP et le compiler.

- Installer Asterisk 18 LTS sur CentOS 7

```
#sudo wget http://downloads.asterisk.org/pub/telephony/asterisk
/asterisk-18-current.tar.gz
```

Figure IV.23: Installation d'Asterisk 18 TLS

- Sélectionner les options de configuration pour Asterisk et parmi ces paramètres le choix des codecs.

```
[root@localhost asterisk-18.6.0]# make menuselect
```

Figure IV.24: La commande pour choisir les codecs

- Configurer le serveur HTTP intégré d'Asterisk et les paramètres comme les adresses IP, les ports utilisés pour les interfaces HTTP et l'état du support TLS.

```
[root@localhost ~] nano /etc/asterisk/http.conf
[general]
servername=Asterisk
tlsbindaddr=0.0.0.0:8089
bindaddr=0.0.0.0
bindport=8088
enabled=yes
tlsenable=yes
```

Figure IV.25: Le fichier http.conf

- Définir les paramètres de configuration pour les appels PJSIP

```
[root@localhost ~] nano /etc/asterisk/pjsip.conf
[system]
type=system
timer_t1=500
timer_b=32000
disable_tcp_switch=yes
```

Figure IV.26: La section system

[System] : Cette section définit des paramètres système globaux pour Asterisk, tel que l'environnement d'exécution du serveur, spécification de la valeur du timer, lors de la négociation des sessions SIP et avant de considérer un appel comme échouée.

```
[global]
type=global
max_initial_qualify_time=0
keep_alive_interval=90
contact_expiration_check_interval=30
default_voicemail_extension=*97
unidentified_request_count=3
unidentified_request_period=5
unidentified_request_prune_interval=30
mwi_tps_queue_high=500
mwi_tps_queue_low=-1
mwi_disable_initial_unsolicited=yes
send_contact_status_on_update_registration=yes
```

Figure IV.27: La section global

[global] : Cette section définit des paramètres globaux pour Asterisk, le type de configuration (global ou systèm), les délais pour les temporisations de sessions SIP, les intervalles pour les vérifications d'expiration des contacts, les paramètres de la messagerie vocale par défaut, ainsi que des options liées aux requêtes SIP non identifiées et aux notifications de messagerie vocale. Ces paramètres permettent de personnaliser le comportement du serveur HTTP d'Asterisk

```
[transport-wss]
type=transport
protocol=wss
bind=0.0.0.0:8089
local_net=10.10.0.9/16
local_net=10.116.0.6/20
external_media_address=178.128.149.185
external_signaling_address=178.128.149.185
allow_reload=yes
```

Figure IV.28: La section transport-wss

[transport-wss] : Cette section définit un transport de type WSS (*WebSocket Secure*) pour la communication SIP. Il spécifie le protocole WSS, l'adresse IP, les ports sur lesquels Asterisk écoutera les connexions WSS, l'adresse IP et le port de liaison du serveur Asterisk, les plages d'adresses IP locales autorisées, les adresses IP externes pour les médias et la signalisation, ainsi que l'option de rechargement dynamique des paramètres de transport.

- Création des comptes SIP

```
[webrtc-phones](!)
context=main-context
transport=transport-wss
allow=!all,opus,ulaw,alaw,vp8,vp9
webrtc=yes
```

Figure IV.29: Les paramètres de création des comptes SIP

[webrtc-phones] : Cette section crée un groupe de téléphones pour les utilisateurs WebRTC avec les paramètres suivants : *"context"* défini comme *"main-context"* pour acheminer les appels, *"transport"* configuré en *"transport-wss"* pour spécifier le type de transport utilisé, *"allow"* avec la valeur *"!all,opus,ulaw,alaw,vp8,vp9"* pour autoriser tous les codecs mentionnés, et *"webrtc"* est activé pour indiquer que ces téléphones sont des téléphones WebRTC. Ces configurations permettent de personnaliser le comportement des téléphones WebRTC dans l'environnement Asterisk.

```
[101](basic,webrtc)
username=101
callerid="Agent01" <101> secret=mlkys141

[103](basic,webrtc)
username=103
callerid="Agent03" <103> secret=mlkys143

[104](basic,webrtc)
username=104
callerid="Agent04" <104> secret=mlkys144


[105](basic,webrtc)
username=105
callerid="Agent05" <105> secret=mlkys145
```

Figure IV.30: les comptes SIP

(basic, webrtc) : Ces configurations permettent de créer des utilisateurs avec des informations d'identification des appels, tel que chaque utilisateur doit avoir un nom, un identifiant et un mot de passe.

- Ajouter la configuration STUN dans Asterisk

```
[root@localhost ~] nano /etc/asterisk/rtp.conf
rtpstart=10000
rtpend=20000
icesupport=yes
stunaddr=stun.l.google.com:19302
```

Figure IV.31: le fichier rtp.conf

- Copier le code du site dans un dossier dans Asterisk.

```
[root@localhost ~] git clone https://github.com/InnovateAsterisk
/Browser-Phone.git
```

Figure IV.32: Insertion du code de site

1.2.6 WEBRTC

Le WebRTC est un projet open source qui permet une communication entre navigateurs sans installer de plugin. L'architecture du WebRTC repose sur une structure en triangle entre un serveur et deux navigateurs. Le serveur permet de coordonner les échanges entre les deux navigateurs pour établir la connexion puis la contrôler

Une fois que la configuration de VMware, l'installation de CentOS, l'installation du pare-feu, d'Asterisk et la création d'objets PJSIP ont été réalisées, nous obtenons un environnement WebRTC fonctionnel. Tout ce qu'il nous reste à faire est de créer une interface client pour pouvoir utiliser notre serveur et effectuer une communication

1.3 Environnement logiciel coté application

1.3.1 Langage HTML

HTML signifie *Hyper Text Markup Language*, C'est un langage de description de données permettant de créer des pages web pouvant être lues dans des navigateurs, permet également de structurer sémantiquement et logiquement et de mettre en forme le contenu des pages, d'inclure des ressources multimédias dont des images, des formulaires de saisie, et des programmes informatiques, Il est souvent utilisé conjointement avec des langages de programmation (JavaScript) et des formats de présentation (feuilles de style en cascade). [16]

1.3.2 Langage CSS

CSS est un sigle qui désigne « *Cascading Style Sheets* » qui veut dire feuilles de styles en cascade, servent à mettre en forme des documents web, type page HTML ou XML. Par l'intermédiaire de propriétés d'apparence (couleurs, bordures, polices, etc.) et de placement (largeur, hauteur, côte à côte, dessus-dessous, etc.), le rendu d'une page web peut être intégralement modifié sans aucun code supplémentaire dans la page web. Les feuilles de styles ont d'ailleurs pour objectif principal de dissocier le contenu de la page de son apparence visuelle. [17]

1.3.3 Langage JS

JS signifie JavaScript, c'est un langage de programmation orientée objet de scripts employé dans les pages web interactives afin d'effectuer des contrôles sur les formulaires avant leur validation mais aussi il permet l'interaction des objets des pages web. ue JavaScript *open-source* qui implémente les protocoles SIP et WebRTC. Elle fournit une interface pour initier et recevoir des appels, ainsi que pour gérer les flux de médias.[18]

1.3.4 JSSIP

JSSIP est une bibliothèque JavaScript open-source qui implémente les protocoles SIP et WebRTC. Elle fournit une interface pour initier et recevoir des appels, ainsi que pour gérer les flux de médias.

1.3.5 Visual Studio

Visual Studio, également connu sous le nom de Microsoft Visual Studio et VS, est un environnement de développement intégré pour Microsoft Windows. C'est un outil pour écrire des programmes informatiques, des sites Web, des applications Web et des services Web. Il comprend un éditeur de code, un débogueur, un outil de conception graphique et un concepteur de schéma de base de données et prend en charge la plupart des principaux systèmes de contrôle

de révision. Il est disponible à la fois en version gratuite et en version commerciale payante. [19]

2. Présentation des fonctionnalités de l'application

2.1 Coté client

Dans notre projet pour offrir une expérience client gratuite, nous avons intégré un bouton distinctif sur le site de l'entreprise. Ce bouton permet aux visiteurs du site d'accéder à une interface permettant ainsi de lancer un appel directement à l'entreprise. Voici le site de l'entreprise :

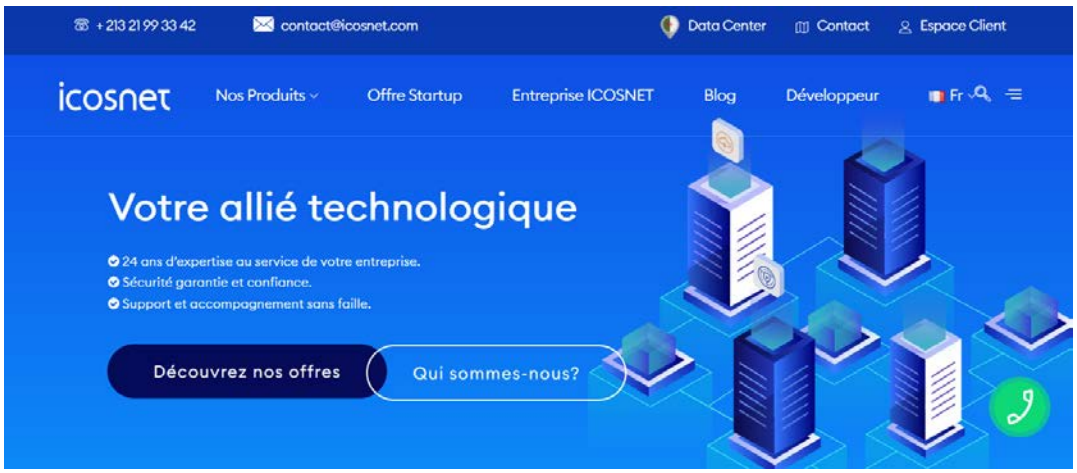

Figure IV.33: Page d'accueil ICOSNET

- Après avoir cliqué sur le bouton appel voici l'interface d'appel

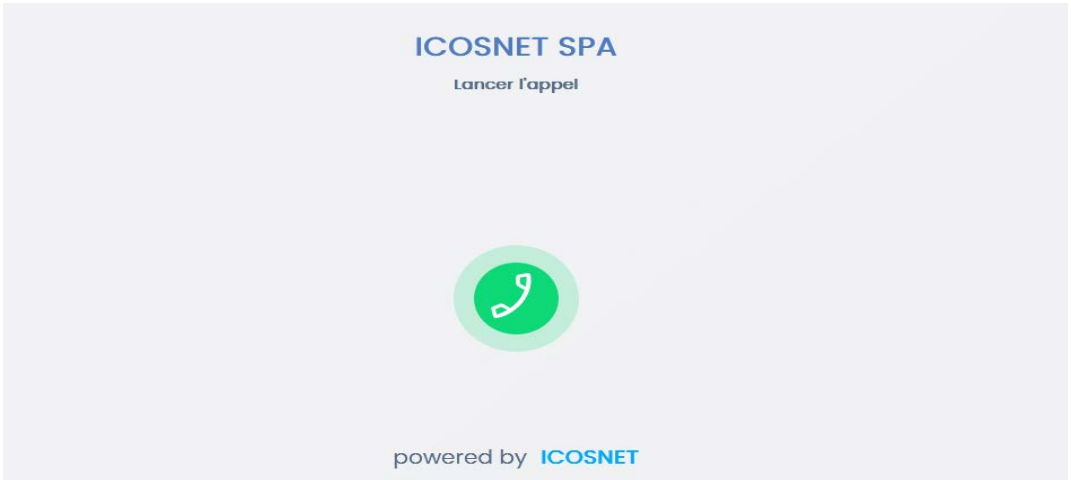

Figure IV.34: Interface de l'appel

- Quand la réception répond à l'appel du client voici l'interface qui s'affichera :

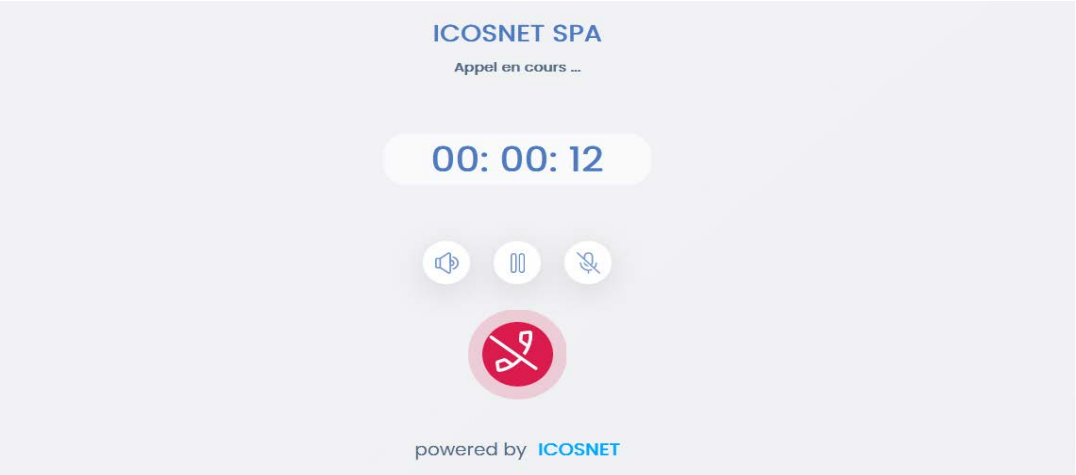

Figure IV.35: Appel entre le client et la réception

2.2 Coté Employé

Pour tester les autres fonctionnalités de notre serveur nous avons utilisé une application open source nommée *browser-phone* du projet Innovate-Asterisk, c'est un référentiel qui contient le code source d'une application de téléphone basée sur un navigateur. En clonant le Git dans le serveur, nous avons obtenu une copie locale de l'ensemble du code source de l'application.

- Pour que les employés puissent accéder à leur compte, il faudra d'abord s'authentifier.

Figure IV.36: Interface d'authentification

- Dans l'interface suivante les utilisateurs peuvent interagir avec le système et d'envoyer des messages à d'autres employés :

Figure IV.37: Interface des messages

- L'interface suivante permet aux employés d'initier des appels vocaux avec d'autres employés. Ils peuvent composer le numéro de téléphone et sélectionner le contact avec lequel ils souhaitent passer un appel. Ils peuvent ensuite utiliser les fonctionnalités fournies pour passer un appel, tel que la possibilité de mettre l'appel en attente, de transférer l'appel à un autre utilisateur et de régler le volume.

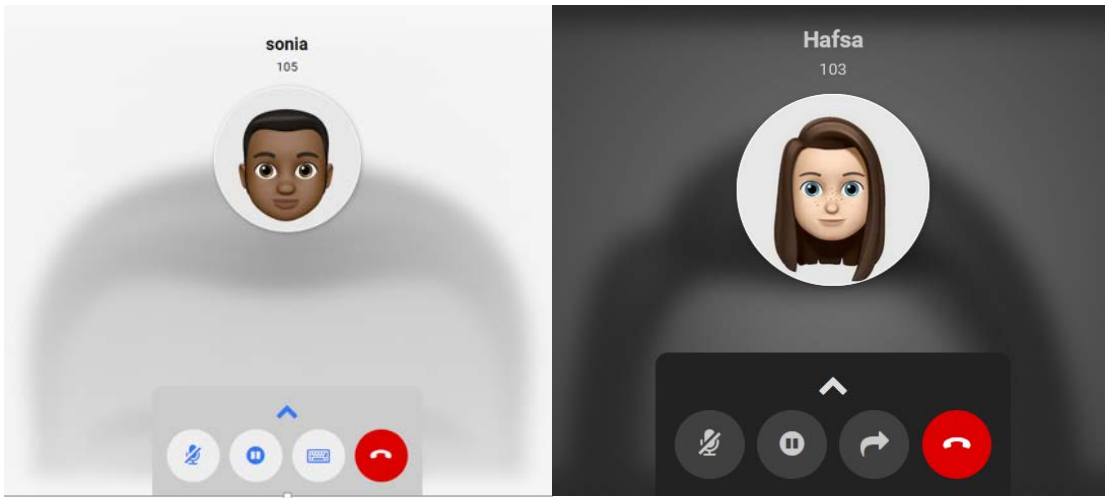

Figure IV.38: Interface de l'appel audio

- Cette interface permet aux employés d'initier des appels vidéo avec d'autres employés, leur permettant de communiquer en temps réel avec l'ajout de la composante visuelle.

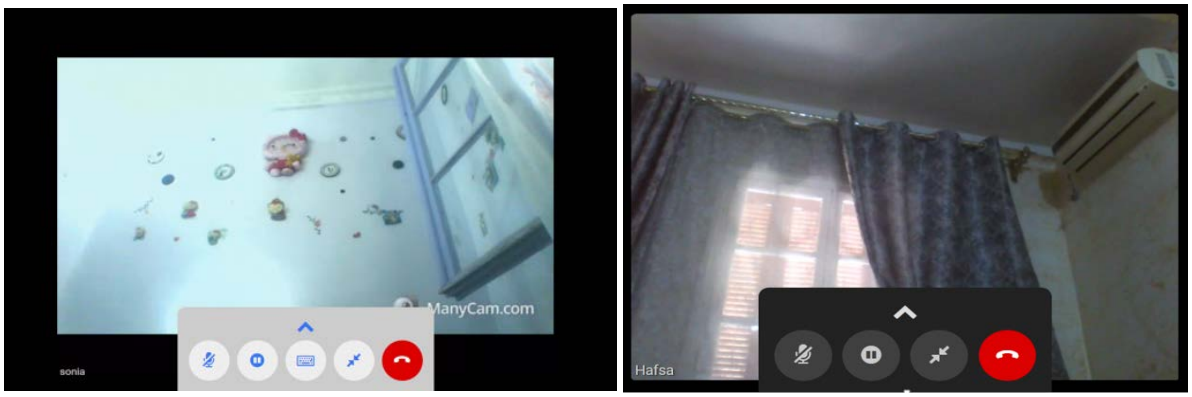

Figure IV.39: Interface de l'appel vidéo

- Afin de démontrer le bon fonctionnement de l'appel, une capture d'écran présentant les détails du déroulement de l'appel a été prise depuis le serveur Asterisk. Cette capture d'écran permet de visualiser les informations pertinentes et de confirmer le bon déroulement de l'appel.

```
[root@localhost ~] asterisk -rrrrrrrrrrrrr
Got RTP packet from 129.45.79.254:49742 (type 187, seq 831987, ts 841766176, len 888893)
Got RTP packet from 129.45.79.254:49742 (type 167, seq 831988, ts 841767136, len 808093)
Got RTP packet from 129.45.79.254:49742 (type 187, seq 831989, ts 841768896, len 888886)
Sent RTP packet from 41.188.218.125:54790 (via ICE) (type 189, seq 016833, ts 841994648, len 008073)
Got RTP packet to 129.45.79.254:49742 (via ICE) (type 187, seq 018488, ts 2832326490, Len 080078)
Sent RTP packet from 41.188.218.125:54790 (type 189, seq 882543, ts 2832368646, len 800081)
Got RTP packet to 129.45.79.254:49742 (via ICE) (type 107, seq 018409, ts 2832327360, len 000081)
Got RTP packet from 129.45.79.254:49742 (type 107, seq 032157, ts 842086176, len 000088)
Sent RTP packet to 41.188.218.125:54790 (via ICE) (type 109, seq 016835, ts 841996560, len 000075)
Got RTP packet from 129.45.79.254:49742 (via ICE) (type 187, seq 818416, ts 2832328320, len 800077)
Sent RTP packet to 41.188.218.125:54790 (type 189, seq 682544, ts 2832369686, len 000079)
```

Figure IV.40: Déroulement d'appel

Conclusion

Dans ce chapitre, nous avons présenté les aspects pratiques liés à la réalisation de notre projet, à savoir l'environnement matériel, logiciel coté serveur et coté application. Et nous avons testé les fonctionnalités de notre serveur à l'aide d'une application open source, suivi des capture d'écran du site de l'entreprise et des appels audio, vidéo et messages envoyés, ainsi que le chronogramme des taches accomplis durant le travail.

Conclusion générale

L'introduction de ce projet met en lumière un défi majeur auquel de nombreuses entreprises sont confrontées : la fourniture d'une solution de communication gratuite et économique pour leurs clients tout en réduisant les coûts associés aux appels entrants et en résolvant les problèmes liés aux numéros verts traditionnels coûteux et à la restriction géographique, cette solution ouvre de nouvelles perspectives pour les entreprises en termes de portée mondiale, d'efficacité et de satisfaction client.

En utilisant WebRTC les clients peuvent communiquer directement avec l'entreprise via son site web sans avoir besoin de numéros spécialisés ou d'applications tierces offrant ainsi une expérience utilisateur transparente et sans friction.

La réalisation de cette solution implique plusieurs étapes telles que la compréhension des principes fondamentaux des réseaux de communication la maîtrise des api, de la signalisation, des protocoles, des codecs ainsi que l'analyse et la conception du projet à travers des schémas et des diagrammes UML. Les outils et les technologies utilisés ainsi que les étapes de développement et les exemples de tests seront également abordés. En intégrant ces éléments, l'entreprise peut établir des connexions vocales directement entre les navigateurs des utilisateurs garantissant ainsi une communication en temps réel efficace et sécurisée. De plus la solution permet aux employés de l'entreprise de communiquer entre eux gratuitement favorisant ainsi la collaboration interne et l'efficacité opérationnelle.

Ce travail a été essentiel pour comprendre les défis de communication auxquels l'entreprise est confrontée, concevoir une solution innovante et la mettre en œuvre avec succès nous a permis d'ouvrir de nouvelles opportunités pour améliorer la communication avec les clients, réduire les coûts et renforcer la compétitivité de l'entreprise sur le marché.

Comme perspectif il reste encore à développer un site web personnalisé pour les employés de l'entreprise pour ne plus utilisé d'application open source, il faudra aussi développer une application mobile pour les Smartphones.

Enfin nous tenons à exprimer notre gratitude envers la société ICOSNET pour nous avoir offert l'opportunité de réaliser ce stage et de développer cette solution, nous sommes convaincus que notre travail contribuera positivement aux objectifs de la société et ouvrira de nouvelles perspectives dans le domaine de la communication en temps réel.

Référence

[1] « Laptop.communication Réseau Social Dans Les Réseaux Informatiques Mondiaux Clip Art Libres De Droits , Svg , Vecteurs Et Illustration. Image 11888394. », *123RF*. https://fr.123rf.com/photo_11888394_laptopcommunication-réseau-social-dans-les-réseaux-informatiques-mondiaux.html (consulté le 5 juin 2023).

[2] A. Kerdjadj. et I. A. Ziane., « Conception et réalisation d'une application web pour le partage de fichiers en peer 2 peer. », Thesis, Université Blida 1, 2015. Consulté le: 9 juin 2023. [En ligne]. Disponible sur: https://di.univ-blida.dz/jspui/handle/123456789/11084

[3] L. Sun, I.-H. Mkwawa, E. Jammeh, et E. Ifeachor, *Guide to Voice and Video over IP: For Fixed and Mobile Networks*. Springer Science & Business Media, 2013.

[4] Y.-H. Hu, K. Ito, et A. Igarashi, « Improving Real-time Communication for Educational Metaverse by Alternative WebRTC SFU and Delegating Transmission of Avatar Transform ». arXiv, 24 mars 2023. Consulté le: 3 juin 2023. [En ligne]. Disponible sur: http://arxiv.org/abs/2303.14071

[5] V. K. L. Ha, R. Chai, et H. T. Nguyen, « A Telepresence Wheelchair with 360-Degree Vision Using WebRTC », MDPI AG, 2020. Consulté le: 3 juin 2023. [En ligne]. Disponible sur: https://doaj.org/article/d2285d938cc046288381be248123487c

[6] A. Chodorek, R. R. Chodorek, et P. Sitek, « UAV-Based and WebRTC-Based Open Universal Framework to Monitor Urban and Industrial Areas », *Sensors*, vol. 21, nº 12, p. 4061, juin 2021, doi: 10.3390/s21124061.

[7] « [리액트] webRTC ». https://velog.io/@jsw4215/리액트-webRTC-recypumx (consulté le 5 juin 2023).

[8] F. Goffinet, « Architecture SIP », *Protocole SIP*, 1 janvier 2018. https://sip.goffinet.org/sip/architecture/ (consulté le 10 juin 2023).

[9] « TCP/IP Overview », *Cisco*. https://www.cisco.com/c/en/us/support/docs/ip/routing-information-protocol-rip/13769-5.html (consulté le 9 juin 2023).

[10] « UDP Protocol | User Datagram Protocol (UDP) ⋆ IpCisco ». https://ipcisco.com/lesson/udp-user-datagram-protocol/ (consulté le 9 juin 2023).

[11] « HTML (HyperText Markup Langage) : définition, traduction ». https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1203255-html-hypertext-markup-langage-definition-traduction/ (consulté le 9 juin 2023).

[12] « CSS - Définition ». http://glossaire.infowebmaster.fr/css/ (consulté le 9 juin 2023).

[13] « 404 », *IONOS Digital Guide*. https://www.ionos.fr/digitalguide/favicon.ico (consulté le 8 juin 2023).