Adaptabilité des flux multimédia appliquée au télé-diagnostic médical

Ronnie Muthada Pottayya

SPIM

Thèse de Doctorat

Adaptabilité des flux multimédia appliquée au télé-diagnostic médical

RONNIE MUTHADA POTTAYYA

SPIM

Thèse de Doctorat

UBFC

école doctorale SPIM sciences pour l'ingénieur et microtechniques

UNIVERSITE DE BOURGOGNE FRANCHE - COMTE

N° | X | X | X |

THÈSE présentée par

RONNIE MUTHADA POTTAYYA

pour obtenir le

Grade de Docteur de
l'Université de Franche-Comté

Spécialité : **Informatique**

Adaptabilité des flux multimédia appliquée au télé-diagnostic médical

Unité de Recherche :
Femto-st/DISC

Soutenue publiquement le 8 Décembre 2017 devant le Jury composé de :

THOMAS NOËL	Rapporteur	Professeur à l'Université de Strasbourg
LAURENT LEFÈVRE	Rapporteur	Chercheur HDR INRIA de Lyon
CHRISTOPHE NICOLLE	Examinateur	Professeur à l'Université de Bourgogne Franche-Comté
BENOIT HILT	Examinateur	Maître de conférences HDR à l'Université de Haute-Alsace, Mulhouse
JEAN-CHRISTOPHE LAPAYRE	Directeur de thèse	Professeur à l'Université de Bourgogne Franche-Comté
ÉRIC GARCIA	Co-Directeur de thèse	Docteur en Informatique, Ido-In/Covalia Interactive

À mes parents, Morgane et Hugo . . .

REMERCIEMENTS

Premièrement, je tiens à exprimer mes sincères et profonds remerciements à mon père qui est la vraie et réelle source de motivation pour moi. J'aurais tellement aimé qu'il soit là mais j'espère qu'il me voit et qu'il m'entend là où il est... Cette thèse, je la lui dédicace grandement, ainsi qu'à ma mère, à mon épouse Morgane, et à mon fils Hugo, aussi mes motivations de chaque jour. Hugo, qui est né peu avant ma soutenance et qui comble ses parents de joie.

Étant donné le caractère CIFRE de cette thèse, elle n'aurait pas pu être menée dans des conditions optimales sans l'aide de nombreuses personnes, qui tout au long de ces trois années, m'ont aidé pour que tout se déroule pour le mieux.

À mes directeurs de thèse, Jean-Christophe Lapayre, Professeur des Universités à l'Université de Bourgogne Franche-Comté de m'avoir accueilli au Département d'Informatique des Systèmes Complexes (FEMTO-ST/DISC) et Éric Garcia, Maître de Conférences et Directeur de la Business Unit Télémédecine, Ido-In, pour leurs précieux conseils, leur encadrement méticuleux, et surtout leur disponibilité. Pouvoir tous se concerter et discuter de la tournure de la thèse n'était jamais chose aisée, étant donné mes deux fonctions : d'Ingénieur Recherches au sein d'Ido-In et Doctorant. Mais ils ont toujours su se libérer et être disponibles pour me guider dans mes travaux de recherches. L'aboutissement de cette thèse n'aurait pas été sans leur aide précieuse.

J'exprime ma profonde reconnaissance à Thomas Noël, Professeur à l'Université de Strasbourg, et à Laurent Lefèvre, chercheur Habilité à Diriger les Recherches à l'INRIA de Lyon, pour avoir accepté de rapporter mon travail. Je remercie aussi Christophe Nicolle, Professeur à l'Université de Bourgogne Franche-Comté, et à Benoit Hilt, Maître de Conférences, Habilité à Diriger les Recherches à l'Université de Haute-Alsace à Mulhouse, pour avoir accepté de participer au jury et contribuer ainsi à l'aboutissement de ce travail.

Je tiens à exprimer mes sincères remerciements à tous ceux qui m'ont soutenu, et sans qui ce travail n'aurait pas pu être réalisé :

À Julien Vouillot, mon Directeur R& D chez Ido-In, pour m'avoir encouragé à effectuer cette thèse et m'avoir donné les moyens d'y arriver,

À l'ensemble des personnes du DISC, et mes collègues chez Ido-In, pour les bons moments passés ensemble, et les discussions plus ou moins philosophiques,

À Karla, Thomas, Jean-Louis, et Aline, mes meilleurs amis qui ont toujours été là et qui m'ont permis de tenir bon lors des périodes difficiles, merci aussi à eux pour leur grande aide pour ma soutenance de thèse,

À Monique, ma petite tante de France, pour m'avoir épaulé durant ces 8 années d'études,

À ma famille et mes amis pour leur soutien,

À ceux qui seront toujours présents dans mon coœur et à qui je dédie cette thèse

également,

Et une fois de plus, à Morgane, qui est devenue mon épouse au début de la thèse, et qui m'a donné le plus merveilleux des cadeaux, en fin de cette thèse, Hugo ! !

SOMMAIRE

INTRODUCTION

CONTEXTE DE LA THÈSE

CONTEXTE GÉNÉRAL

Les applications collaboratives se sont développées depuis quelques années avec les techniques de groupware : le mot clé est ”partage”. Elles permettent les échanges entre utilisateurs par l'intermédiaire de médias discrets (comme le texte, le dessin, ou encore les images fixes) et de médias continus (comme l'audio et la vidéo). Ces applications doivent pouvoir s'exécuter dans un contexte où l'hétérogénéité est reine. Avec l'évolution matérielle (terminaux, processeurs, ...) qui s'accompagne de l'augmentation des capacités réseaux, les plateformes de travail à distance se développent.

FIGURE 1 – Nouvelles Plateformes de Télédiagnostic

Et dans le domaine de la santé, c'est la e-Santé et la télé-médecine (cf figure 1) qui prennent leur essor. Ces travaux de recherche s'inscrivent dans la cadre d'une thèse CIFRE au sein de la société IDO-In du groupe Maincare Solutions qui propose une solution logicielle de collaboration en temps réel autour de la consultation et du diagnostic à distance. La suite Covotem^{TM} offre des fonctionnalités de télé-consultation et de télé-expertise. Elle permet d'effectuer à distance avec le patient des diagnostics, par exemple dans le domaine des accidents vasculaires cérébraux, mais également la réalisation de réunions de concertations pluridisciplinaires, entre spécialistes et ce afin de consulter l'avis des différents experts. Dans cet environnement basé sur le flux d'informations, le traitement des données patient est plus que primordial. Les médias nécessaires à un

diagnostic doivent être rapidement disponibles tout en conservant une parfaite intégrité. Le maintien de cette disponibilité nécessite que le système puisse s'adapter aux fluctuations des réseaux, à l'hétérogénéité des terminaux, ...

OBJECTIFS DE CES TRAVAUX

Une plateforme de télédiagnostic médical doit intégrer différents médias (conférence temps réel video, image, ...) en garantissant une continuité de service. Mais le contexte distribué est dynamique (variation dans le temps par exemple des bandes passantes) et hétérogène au niveau des réseaux, des terminaux et des profils utilisateurs. Compte tenu de tous ces paramètres évolutifs, il est nécessaire que ces systèmes distribués soient adaptables.

Ainsi, ces applications doivent posséder la capacité de s'adapter à leur environnement, à leur contexte : elles doivent faire intervenir l'utilisateur le moins possible dans la configuration du système. Il est nécessaire que ces systèmes permettent d'appréhender le contexte et de s'y adapter : le contexte est la combinaison de données centrées sur l'utilisateur (par exemple, ses préférences utilisateur en fonction de son/ses circonstances actuelles) et des ressources/données centrées système (par exemple, les paramètres et les contraintes des terminaux et réseaux utilisés).

Entre la collecte des données de contexte et l'adaptation se situe un module de prise de décision qui permettra une automatisation (ou semi-automatisation) de l'adaptation. Afin d'exprimer des évènements issus de l'observation du contexte ainsi que les conditions permettant d'identifier la situation dans laquelle se trouve un système, différents raisonnements logiques existent : la logique des propositions, la logique du premier ordre, l'inférence bayésienne, l'inférence fréquentiste, la loi binomiale, le raisonnement non-monotone, le raisonnement flou, ... Dans le cadre de ces travaux, le côté décisionnel sera primordial.

Nous souhaitons proposer un nouveau middleware qui permettra dans le cadre de la suite Covotem^{TM}, produite par la société IDO-In du groupe Maincare Solutions, l'adaptation des outils de télédiagnostic médical pour en assurer la continuité de service.

PLAN DU MÉMOIRE

Comme le montre la figure 2 nos travaux ont permis d'aborder plusieurs domaines de recherche, de l'adaptabilité, à la prise de décision, en passant par la notion d'applications distribuées multimédia.

La première partie de ce document est consacrée à notre état de l'art qui s'articule en deux chapitres.

Le premier chapitre est naturellement orienté sur l'adaptabilité et l'adaptation. Nous définirons tout d'abord le domaine de l'informatique ubiquitaire ou ambiante, et les problèmatiques impliquées par la mobilité dans ces sytèmes. Puis dans un deuxième temps, nous étudions les recherches menées sur l'adaptabilité de ces systèmes.

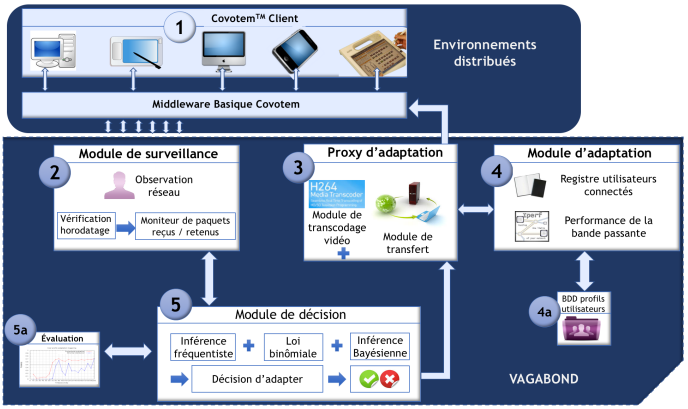


FIGURE 2 – Un nouvel intergiciel pour l'adaptation de flux multimédias

Le chapitre deux permet de développer d'une part la notion d'environnements distribués (cibles de nos adaptations 1) et d'autre part les mécanismes de décision. Ce chapitre débute par une étude comparative entre les différentes approches de la sensibilité au contexte. Une fois le contexte défini, Il est important d'étudier les mécanismes de prise de décision qui permettront de déclencher d'éventuelles adaptations en fonction du contexte. Ainsi différentes logiques, lois et raisonnements pourront être utiles, et seront étudiés dans la fin de ce chapitre.

La deuxième partie de ce mémoire constitue notre contribution.

Tout d'abord, le chapitre trois fait l'objet de la présentation de l'architecture globale de notre intergiciel, *VAGABOND* (en anglais *Video Adaptation framework, crossing security GAteways, Based ON transcoDing* cf figure 2) qui permet d'adapter la vidéo en utilisant des techniques de transcodage et en fonction de la bande passante disponible, et est capable de passer à travers des barrières de sécurité telles que des pare-feus et des proxies web. Les différents modules de notre intergiciel sont définis.

Le chapitre quatre est consacré à la description plus précise des fonctionnalités de l'intergiciel VAGABOND et de leurs mécanismes. Plus précisément, il s'agit des algorithmes et des techniques employés au sein des modules de la plateforme Vagabond. Le 1 représente les clients connectés de la suite Covotem^{TM}. Le 2 correspond au module de surveillance de l'état du réseau qui est embarqué sur les clients. Les données collectées et issues de la phase d'observation sont ensuite transmises au module de décision (5) qui est chargé d'évaluer (cinquième point bis sur le diagramme) l'état du réseau actuel. Les lois de probabilité telles que l'inférence fréquentiste, la loi binômiale, et l'inférence bayésienne seront utilisées. De ces évaluations en résultent des décisions d'adaptations. Ces décisions sont transmises au proxy d'adaptation 3 qui est chargé de transcoder (décodage et encodage dans un nouveau format) les trames de vidéo dans un format plus adapté Chaque proxy d'adaptation déployé est rattaché à un serveur d'adaptation unique 4. Enfin, une base de données regroupant les profils utilisateurs des professionnels de santé est également rattachée à ce module (quatrième point bis).

Le dernier chapitre de la contribution expose les implémentations et les résultats obtenus lors de l'utilisation de notre intergiciel *VAGABOND*. Nous montrons dans ce chapitre que le module de décision présent dans l'intergiciel est bien réactif quant aux changements de l'état du réseau et ainsi nous démontrons que les décisions que ce dernier

prend sont conformes à nos attentes en termes de réactivité et d'applications de règles d'adaptation. Nous présentons, en pseudo-code, les différents algorithmes qui ont été implémentés. Enfin, nous exposons des résultats (des courbes) obtenus lors des phases d'expérimentations. Nous expliquons, de par ces courbes, comment les données issues des phases d'observations sont prises en compte et comment plusieurs lois de probabilités mathématiques sont appliquées en vue d'une éventuelle adaptation.

Nous terminons ce document en synthétisant les différentes contributions que nous avons proposées étant donnés les objectifs que nous nous étions fixés pour ce travail. Nous finalisons ce chapitre par différentes perspectives de recherche que soulève notre travail et que nous envisageons pour des travaux futurs.

QUELQUES INDICATIONS POUR LA LECTURE DE CE RAPPORT

Le plan est organisé sur quatre niveaux : Parties, Chapitres, Sections et Sous-Sections. Il est utile de savoir que la Partie II, présentant les contributions et résultats, est indépendante de la première Partie.

Les résultats de nos travaux ont pour la plupart été publiés. La liste des publications personnelles est donnée avant la bibliographie placée en fin du document.

I

ÉTAT DE L'ART

Comme nous l'avons indiqué dans l'introduction de ce document, ces travaux de recherche ont été réalisés au sein d'une entreprise spécialisée dans les outils de télédiagnostic collaboratif. Il s'agit de proposer à distance un ensemble de média partagés (image, texte, vidéo, audio, téléconférence, ...) qui permettront à des médecins distants d'élaborer des diagnostics conjoints. Mais l'environnement distribué nécessaire à la mise en place de telles pratiques est, au niveau des terminaux, des réseaux et des profils utilisateurs, non seulement hétérogène mais en plus dynamique (dans le sens qui évolue au cours du temps). Pour conserver toute leur efficacité, ces environnements doivent ainsi devenir évolutifs : ils doivent être adaptés.

Dans le cadre de nos travaux sur l'adaptation des flux multimédia plusieurs domaines de recherche sont concernés. En effet, dans un contexte distribué particulier il peut être nécessaire d'adapter les média en fonction de critères qui seront les paramètres d'une prise de décision d'adapter.

Ainsi cette première partie d'état de l'art s'articule sur deux chapitres, le premier étant naturellement orienté sur l'adaptabilité et l'adaptation, et le deuxième développant d'une part la notion d'environnements distribués (cibles de nos adaptations) et d'autre part les mécanismes de décision (décisions qui sont prises pour adapter).

1

L'ADAPTABILITÉ DES SYSTÈMES UBIQUITAIRES

Au sein de ce chapitre, nous définirons tout d'abord le domaine de l'informatique ubiquitaire ou ambiante, et les problèmatiques impliquées par la mobilité dans ces sytèmes.

Et dans un deuxième temps, nous étudions les recherches menées sur l'adaptabilité de ces systèmes.

1.1/ POSITIONNEMENT DES TRAVAUX

1.1.1/ LE CONTEXTE DES APPLICATIONS RÉPARTIES

Nous avons développé nos travaux dans le cadre des applications de télédiagnostic collaboratif. Un système collaboratif est dit réparti : il met en jeu des éléments logiciels collaborants qui s'exécutent sur plusieurs machines reliées par un réseau de communication. Grâce au succès de l'Internet, ces applications ont connu un essor considérable qui a engendré une augmentation de la complexité des fonctions métier qu'elles fournissent ainsi que des propriétés de qualité de service associées. Ces dernières [Cam94] peuvent porter sur différents critères tels que la disponibilité des services offerts, la sécurité et l'intégrité des actions effectuées et des données manipulées, les temps de réponses fournis aux usagers, ...

En conséquence, toutes ces propriétés attendues des systèmes d'information impactent et rendent plus complexes toutes les phases du cycle de vie des applications réparties. Si l'on reprend le cycle de vie logiciel tel qu'il a été défini par W. Scacchi dans [Sca02], et qui est aussi connu comme le modèle en V, on s'aperçoit qu'à toutes les étapes la notion d'adaptation est omniprésente (cf figure 1.1).

Ainsi, ces différentes phases se voient modifiées afin d'incorporer une phase d'adaptation. De la phase de conception à celle du développement, les développeurs sont contraints de s'adapter sans cesse aux exigences et aux technologies qu'ils utilisent. Ils implémentent dès le départ des mécanismes d'adaptation qui faciliteront la phase de déploiement. Néanmoins, tous les mécanismes de déploiement sont difficilement prédictibles voire même impossible à prédire. Une équipe de déploiement doit s'adapter à l'environnement de déploiement, aux hébergeurs, aux systèmes d'exploitation des serveurs, aux contraintes de sécurité, ... L'adaptation sera également présente une fois

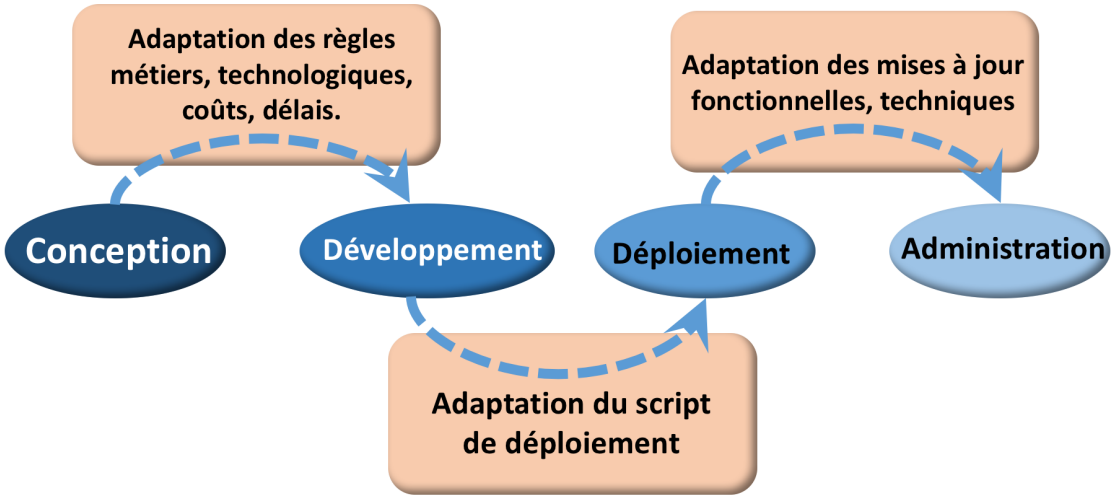


FIGURE 1.1 – Cycle de vie des applications réparties

l'application déployée. L'administration des applications réparties nécessite le plus souvent des mises à jour permanentes, que ce soit au niveau fonctionnel ou au niveau technique. Un administrateur d'une application distribuée doit donc s'adapter à toutes ces exigences.

Dans ce domaine des applications réparties, les terminaux mobiles sont de plus en plus utilisés. Ces nouveaux objets communicants et mobiles sont présents partout et introduisent donc une forte instabilité dans les environnements d'exécution des plateformes sensibles au contexte. L'ordinateur personnel qui était vu, jusqu'à présent, comme étant le seul objet informatisé ou encore comme l'assistant numérique universel se retrouve désormais entouré de ces objets qui font que cette vision de l'ordinateur personnel disparaît. En 1991, Mark Weiser [Wei99] a proposé la notion d'**informatique ambiante**, une informatique omniprésente. Cette notion est désormais plus connue sous le nom d'**informatique ubiquitaire**. Il précisa qu'il s'agit d'une informatique présente en tous lieux, à tous instants et en toutes choses. Dans les applications ubiquitaires, des mécanismes d'adaptation sont développés et mis en place et ce de la manière la plus transparente possible. Ces applications doivent posséder la capacité de s'adapter à leur environnement, à leur contexte : elles doivent faire intervenir l'utilisateur le moins possible dans la configuration du système afin d'être utiles et au service de l'homme et non l'homme au service de la machine.

Ainsi, la sensibilité au contexte (en anglais, *context awareness*) et l'adaptation sont devenues deux points clés lors du développement d'environnements ubiquitaires. En effet, ces applications doivent permettre d'appréhender le contexte et de s'y adapter : le contexte est la combinaison de données centrées sur l'utilisateur (par exemple, ses préférences utilisateur en fonction de son/ses circonstances actuelles) et des ressources/données centrées système (par exemple, les paramètres et les contraintes des terminaux et réseaux utilisés) [Ber06, Inv06, Per14, Fer15].

1.1.2/ POURQUOI ADAPTER ?

L'informatique ubiquitaire est la troisième ère de l'histoire de l'informatique [Kru16], qui succède à l'ère des ordinateurs personnels et celles des *mainframes*. Durant l'ère des *mainframes*, un ordinateur de forte capacité était utilisé collectivement par plusieurs personnes. Dans l'ère suivante, celle des ordinateurs personnels, un ordinateur appartient et est utilisé exclusivement par une seule personne. Dans la troisième ère de l'informatique, l'utilisateur a en plus de son ordinateur personnel à sa disposition une gamme de petits appareils tels que le smartphone ou l'assistant personnel, et leur utilisation fait partie de sa vie quotidienne (figure 1.2).

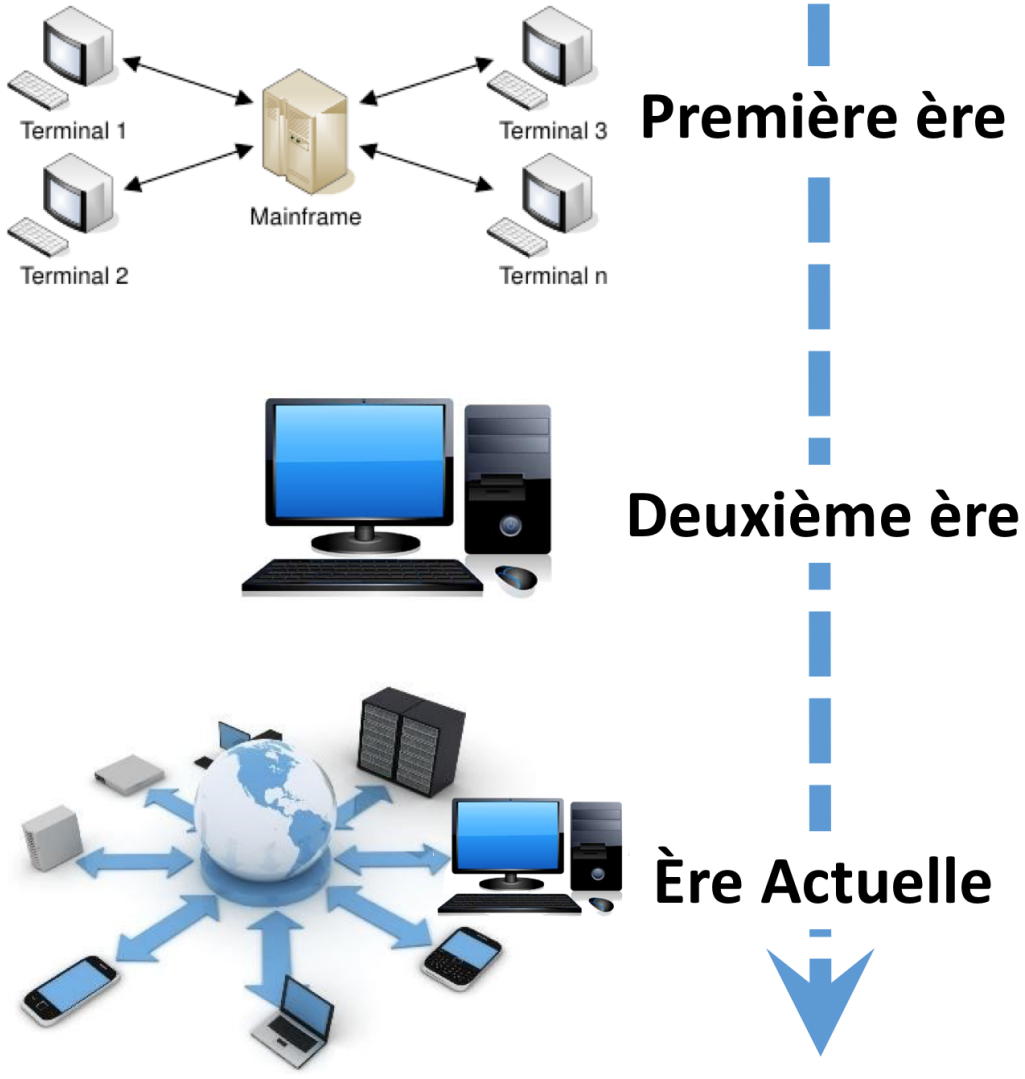


FIGURE 1.2 – L'ère des systèmes informatiques

Par conséquent, les applications pouvant dynamiquement s'adapter à leur environnement deviennent nécessaires. De nombreux systèmes ont été proposés pour l'informatique omniprésente : ces applications logicielles distribuées et mobiles requièrent de nouvelles fonctionnalités pour supporter l'adaptation à différents contextes d'utilisation pour lesquels elles doivent s'adapter (dispositifs multiples, environnements physiques, mobilité,

forte variabilité des conditions d'utilisation de l'application, . . .).

Afin que les terminaux mobiles puissent opérer à différents niveaux, les systèmes collaboratifs deviennent ainsi de plus en plus présents dans notre vie. Que ce soit à la maison ou au travail, les plateformes collaboratives permettent de s'affranchir de la notion de distance. Et, le plus souvent, ces applications nécessitent la mise en place de mécanismes d'adaptation pour garantir leur bon fonctionnement.

Comme par exemple, lorsque nous parlons d'un tableau blanc dans un environnement collaboratif, tous les participants doivent avoir les mêmes informations sur leurs écrans. En 1989, Mark Stefik [Ste87] parlait déjà de la notion de tableau blanc et du travail collaboratif dans son article sur le projet *"Colab"*. Il décrivait ce projet comme étant orienté autour du concept *WYSIWIS (What You See Is What I See)*. Une fenêtre privée correspond à un éditeur de texte individuel. Une fenêtre publique correspond à un tableau blanc partagé par plusieurs utilisateurs dans un système collaboratif. Cette notion est toujours présente dans les applications collaboratives intégrant un système de tableau blanc.

1.2/ L'INFORMATIQUE UBIQUITAIRE (*Ubiquitous Computing)*

1.2.1/ LES ARCHITECTURES DISTRIBUÉES

Dans les systèmes informatiques, différentes approches d'architectures logicielles sont envisageables (figure 1.3). Chacune d'entre elles ayant ses intérêts et étant plus ou moins bien adaptée à différentes situations. Nous pouvons distinguer trois grandes familles d'architectures que sont les systèmes centralisés, décentralisés, et distribués qui regroupent les systèmes partiellement décentralisés.

Ces derniers peuvent être classés comme suit :

- Dans une approche centralisée, une entité centrale, généralement appelée un serveur, gère les autres éléments et informations de l'infrastructure, qui sont généralement appelés les clients. Il s'agit d'une architecture client-serveur. Ce type de système distribué est connu pour provoquer un goulot d'étranglement qui est un point sensible du système.

- Une approche partiellement décentralisée met en avant certains composants du réseau qui jouent un rôle plus important que d'autres. En général, les composants les plus importants sont ceux ayant pour but la mise en relation des différents éléments de l'infrastructure ou encore des mécanismes de découverte. Ces composants jouent le rôle d'annuaire.

- Enfin les architectures totalement décentralisées (*full distributed*) forment un réseau non structuré. L'idée, pour un système de communication, est que toute entité (individu, association, organisation, . . .) puisse être une partie d'un réseau qui n'a pas d'autorité principale, et que ces autorités puissent échanger entre elles. Un partage et une réduction de coûts entre les différentes entités est ainsi obtenu avec ces architectures, ce qui permet de conserver une bonne robustesse grâce à l'absence d'entité centrale au rôle décisif. En conséquence, ces architectures sont facilement extensibles et offrent un meilleur passage à l'échelle. En agrégeant toutes

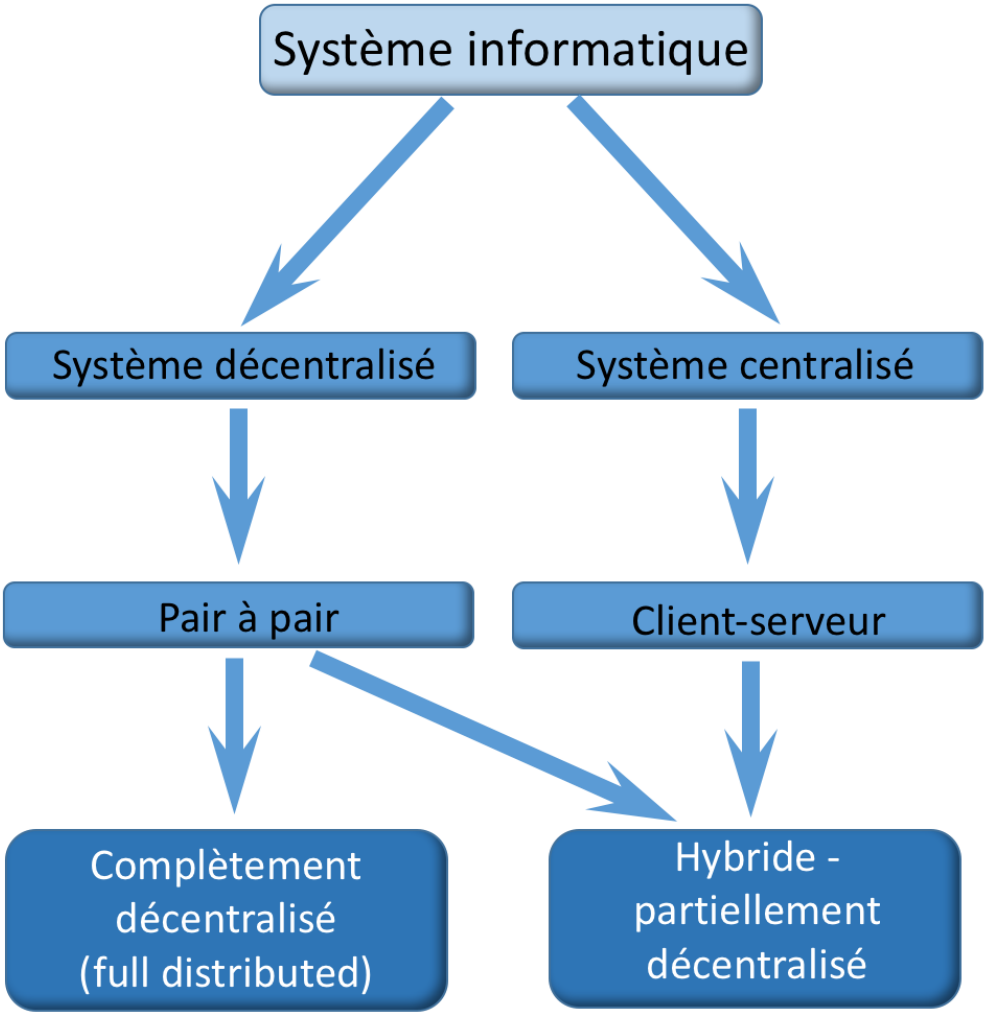


FIGURE 1.3 – Architectures des systèmes informatiques distribués

les ressources disponibles, la puissance dans de tels systèmes est aisément obtenue. Bien que ces derniers restent des systèmes autonomes et présentent bon nombre d'avantages, leurs déploiements dans des réseaux restreints freinent leur utilisation. De plus, les systèmes décentralisés sont plus compliqués à mettre en œuvre et la maintenance est coûteuse.

1.2.2/ LES SYSTÈMES INFORMATIQUES UBIQUITAIRES

Un système d'information ubiquitaire est dit *"multi"* [Kha13]. Ainsi la caractérisation suivante est proposée pour un système d'informations ubiquitaire :

- Multi-utilisateurs : dans un même espace, de nombreux utilisateurs peuvent être amenés à utiliser les mêmes ressources et à communiquer de manière concurrente,
- Multi-dispositifs : de nombreux objets physiques potentiellement informatisés et communicants peuvent nous entourer,

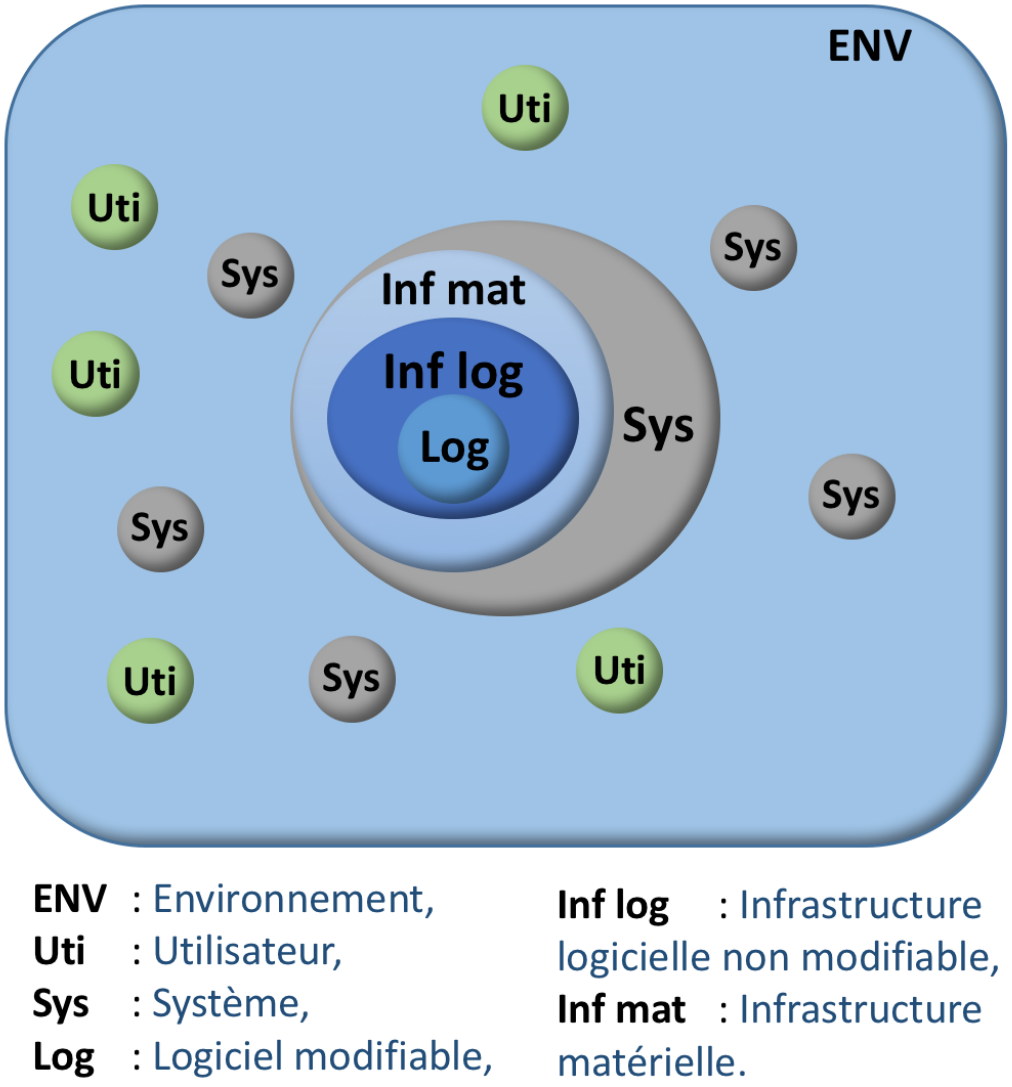


FIGURE 1.4 – L'environnement des systèmes ubiquitaires

- Multi-environnements : les dispositifs et les utilisateurs peuvent être mobiles et peuvent changer d'emplacement et donc d'environnement,
- Multi-applications : de nombreuses applications reposant sur des objets communicants peuvent être conçues.

A. Russ et al., dans [Rus08], expliquent que les acteurs dans un système d'information ne sont plus exclusivement des humains mais, pour une grande partie, des agents virtuels qui s'échangent des données suivant certaines stratégies et procédures préprogrammées. L'informatique ambiante repose donc sur un ensemble d'entités en interaction et sont incluses dans l'environnement (cf figure 1.4). Un environnement est l'ensemble des objets, des personnes et des systèmes informatiques du monde physique. Les entités sont de deux catégories :

- Être vivant : habitée par la vie possédant des capacités sensori-motrices et elle peut prendre des décisions et faire des choix en fonction de ses goûts et de ses affinités,

- Système informatisé : par exemple, des objets physiques comme des capteurs qui remontent des données au système.

Pour résumer

L'informatique ubiquitaire est un nouveau domaine de recherche et est une ère de convergence où une gamme d'objets physiques communiquent *discrètement* de manière non intrusive et presque invisible à travers un tissu de réseaux hétérogènes. Ces objets connectés, qui font partie de notre vie quotidienne, enrichissent leur connaissance de nos faits et gestes, de nos habitudes et ainsi anticipent les demandes. La mobilité et l'adaptation dynamique des systèmes ubiquitaires seront des traits dominants de ces systèmes.
Dans la littérature, on parle également d'informatique ambiante, informatique diffuse, informatique invisible, . . . Il s'agit d'une informatique qui s'adapte à l'homme et pour y arriver un système ubiquitaire se doit d'être présent partout et en temps réel. D'où la notion d'ubiquité. L'informatique ubiquitaire devrait rendre plus familier et instinctif l'outil informatique et en faciliter l'utilisation dans de nombreux domaines tels que la formation ou encore la médecine.

1.2.3/ LES CONTRAINTES DE CES SYSTÈMES

Les systèmes ambiants se caractérisent tout d'abord par la mise en œuvre de dispositifs et d'objets de la vie courante. Une première difficulté pour ces objets et ces dispositifs, pour interagir entre eux, reste l'hétérogénéité. Cette dernière peut se situer à différents niveaux. Par exemple, nous parlerons de réseaux hétérogènes ou encore de systèmes d'informations hétérogènes avec différents systèmes d'exploitation et donc de divers langages de programmation. Nous parlerons également d'utilisateurs hétérogènes suivant leur rôle et leurs connaissances.

La contrainte d'hétérogénéité implique que les infrastructures logicielles ubiquitaires doivent évoluer dynamiquement avec les apparitions et les disparitions des terminaux se trouvant dans l'environnement. La mobilité est une cause de l'évolution de ces objets et dispositifs. L'apparition et la disparition étant des mesures d'économie d'énergie ou encore dues à des pannes (perte de réseau par exemple). Divers protocoles de communication peuvent éventuellement être utilisés par les multiples entités hétérogènes qui peuvent apparaître ou disparaître à tout instant. La forte dynamicité, présente dans les systèmes ubiquitaires, implique que la mobilité reste un défi majeur pour ces systèmes. En effet, il s'agit de détecter les apparitions ou les disparitions le plus rapidement possible afin d'engager des actions et mettre à jour le système ambiant en conséquence.

Le but ultime de l'informatique ubiquitaire est de mettre en œuvre des outils informatiques accessibles à n'importe quel moment et indépendamment de l'emplacement où se situe un terminal. Cette approche est considérée comme étant réactive. Des outils proactifs sont nécessaires pour l'élaboration de logiciels ubiquitaires afin de prendre en compte

une multitude de terminaux dans un environnement ubiquitaire. Actuellement, le nombre de langages et d'outils pour mettre au point ce type de système reste très limité et ralentit le processus le développement de ces systèmes. L'idée est de cacher la complexité de l'environnement en isolant les applications de leur gestion explicite des protocoles, des accès mémoire distribuée, de la duplication de données, des erreurs de communications, . . .

Le nomadisme numérique, la mobilité que nous venons de décrire, implique de nouveaux défis pour les développeurs des systèmes ubiquitaires. Cette mobilité peut aussi bien être physique que logique. Les applications doivent être en mesure de passer d'un appareil à un autre tout en fournissant l'accès aux données et en maintenant les états de passage (applications de type "follow-me") [Bah12]. La mobilité a également introduit la notion de la prise en compte du contexte (*context awareness*) [Dey01b, Per14, Rah15, Mit15], qui permet de fournir des informations et des services dans le cas d'une disponibilité limitée ou intermittente.

Les intergiciels (ou *middleware*) peuvent aider à surmonter les problèmes d'hétérogénéités des architectures, des systèmes d'exploitation, des technologies réseau, ou encore des langages de programmation : ces systèmes demeurent le plus souvent centralisés, ce qui implique une gestion des *goulots d'étranglements*, ainsi qu'une optimisation des distances et des temps de communication. En revanche, l'utilisation d'un *framework* qui est un environnement comprenant des APIs (*Application Programming Interface*), des interfaces utilisateurs, et des outils qui simplifient le développement et la gestion d'applications dans un domaine bien spécifique se révèle être une meilleure solution. Il est également possible d'utiliser des *frameworks* pour le développement d'intergiciels et ainsi construire des applications efficaces sur ces intergiciels.

L'évolutivité, l'hétérogénéité, l'intégration, l'invisibilité, la sensibilité au contexte, et la gestion du contexte sont tous les défis à relever, selon D. Saha et A. Mukherjee, dans [Sah03], lorsque l'on définit un système ambiant ou ubiquitaire. Pour T. Kingberg et A. Fox [Kin02], deux caractéristiques clés doivent être prises en compte : l'intégration physique et l'interopérabilité spontanée. R. Want et T. Pering, dans [Wan05], proposent la gestion de l'énergie, la découverte, l'adaptation des interfaces utilisateurs et la géolocalisation informatique. M. Modahl et al., dans [Mod06], proposent une taxonomie pour la construction d'un intergiciel dans une infrastructure logicielle appelée *UbiqStack* : cet intergiciel comprend l'enregistrement et la découverte, les services et la souscription, le partage et le calcul, la gestion de contexte, le stockage de données et le streaming.

De tous ces différents systèmes et propositions issus de la littérature, la sécurité est un élément qui apparaît peu, voire pas du tout. Un système est sécurisé s'il existe des mesures permettant d'assurer la continuité de services, l'intégrité des données et la confidentialité. Des mécanismes de sécurité existant dans les systèmes distribués peuvent être utilisés mais ces mécanismes doivent être des processus légers afin de préserver la spontanéité des interactions et la limitation des périphériques, comme par exemple, les périphériques nomades qui possèdent souvent une puissance de calcul très limitée.

1.2.4/ LES PROBLÈMES LIÉS À LA MOBILITÉ DANS CES SYSTÈMES

Comme nous l'avons expliqué dans le paragraphe 1.2.3, le défi de la mobilité s'est ajouté à la liste des défis de l'informatique ubiquitaire : il découle naturellement de l'explosion

du développement de terminaux de nouvelle génération, tournés vers le nomadisme, et en parallèle l'amélioration permanente des technologies réseaux. De nos jours, l'informatique prend résolument le chemin de l'ubiquité, rendant obsolète la vision de l'ordinateur personnel restreint à un bureau. Comme l'indique S. Abolfazli et al., dans [Abo12] et confirmé sur le site [Gar11], la popularité des smartphones a considérablement augmenté et les statistiques montrent leur tendance à surpasser les dispositifs fixes.

Mais les systèmes informatiques mobiles sont très contraints par rapport aux systèmes fixes. Ces contraintes sont inhérentes à la mobilité, et ne sont pas seulement des conséquences de la technologie actuelle. Les périphériques mobiles sont pauvres en ressources (mémoire, CPU, stockage, . . .) par rapport aux périphériques statiques. Compte tenu des différents paramètres comme l'autonomie, la puissance de calcul, la taille, l'ergonomie associés à ces nouveaux périphériques mobiles (hormis les ordinateurs portables), ils seront toujours moins performants que les ordinateurs.

Une exigence clé des systèmes informatiques mobiles est la capacité d'accéder aux données critiques indépendamment de leurs emplacements. Il en va de même pour les données contraintes au temps réel, comme pour les données produites lors d'une vidéoconférence. Les données partagées des systèmes ainsi qu'éventuellement les bases de données partagées doivent être mises à la disposition des programmes en cours d'exécution sur les ordinateurs mobiles. Dans l'article [Chu14a], les auteurs mettent en évidence des travaux de recherche sur l'informatique ubiquitaire. En particulier, ils s'attardent sur le domaine de la médecine comme par exemple les travaux de EY. Jung et al. [Jun13]. Les auteurs font valoir que dans le cas de l'informatique ubiquitaire, nous recherchons de nouveaux types de contenu, la plupart générés par l'utilisateur, qui peuvent être recherchés, organisés et consommés sur de nombreux périphériques et dans de nombreux formats.

Les défis concernant la convergence des technologies et des contenus nécessitent de nouveaux algorithmes, de nouveaux paradigmes d'application, de nouvelles méthodes d'interaction ainsi que de nouveaux services de personnalisation : des algorithmes adaptatifs, des outils pour la découverte d'informations, des moteurs de recherches intelligents qui permettent de fournir des informations convergentes capables d'être délivrées au bon endroit, au bon moment et avec le bon niveau de détails. Par exemple, dans le cas d'une téléapplication dans le domaine de la neurologie, suivi d'un AVC (Accident Vasculaire Cérébral) : l'urgentiste sur site devra avoir un accès rapide aux bases de données médicales décrivant les symptômes d'un AVC confirmé, ainsi que l'accès au dossier médical du patient afin de connaître ses antécédents, sa sensibilité aux médicaments, par exemple, afin de lui donner les premiers soins le plus rapidement possible.

Il en résulte que l'exigence de l'accès aux données partagées implique une interdépendance entre les différents éléments d'un système informatique mobile. Ce dernier est une technologie qui fournit un service basé automatiquement sur l'information perçue sur la situation dans des environnements personnels et omniprésents. Dans ces systèmes, la nécessité d'être robuste face aux pannes de réseau et les sites distants exige que les terminaux soient aussi autonomes que possible [Chu14b]. Ainsi la mobilité exacerbe la tension entre l'autonomie et l'interdépendance qui est caractéristique des systèmes distribués. Idéalement, la mobilité devrait être totalement transparente pour les utilisateurs. Ainsi, cette transparence soulage les utilisateurs de la nécessité d'être constamment au courant des détails de leur environnement informatique (par exemple afin de se reconnecter ou d'adapter manuellement sa connexion). La nécessité d'adapta-

tion pour faire face aux changements de l'environnement doit être initiée par le système plutôt que par les utilisateurs [Rah14].

1.3/ L'ADAPTATION

1.3.1/ LES PRINCIPES DE L'ADAPTATION

FIGURE 1.5 – Adaptation par modification de la taille d'une image

Le terme adaptation est utilisé dans différents domaines et peut donc avoir plusieurs significations. Par exemple, en médecine, il s'agit de l'ensemble des phénomènes qui permettent à l'œil de percevoir des objets de moins en moins lumineux. Ou encore en biologie, ce terme possède plusieurs sous-significations, il s'agit principalement d'un changement survenu chez un individu animal ou végétal, à une lignée ou à une espèce, et qui augmente leurs chances de survie et de reproduction dans le milieu où ils vivent. Une deuxième définition dans ce domaine est un état général d'un organisme auquel un certain milieu est seul favorable, ou plus favorable que tout autre.

Plus généralement, adapter revient à dire que nous ajustons une chose à une autre. Par exemple, nous pouvons modifier la taille d'une image (figure 1.5) ou encore le nombre d'images par seconde (figure 1.6) dans le but de réduire la bande passante utilisée par une application de vidéoconférence.

En informatique, le terme adaptation fait couramment référence à un processus dans lequel un système d'interaction adapte son comportement selon chaque utilisateur basé sur les informations acquises de l'utilisateur et de son environnement [Wan07, Faj15]. Nous pouvons définir l'adaptation comme la capacité d'harmoniser l'application avec son

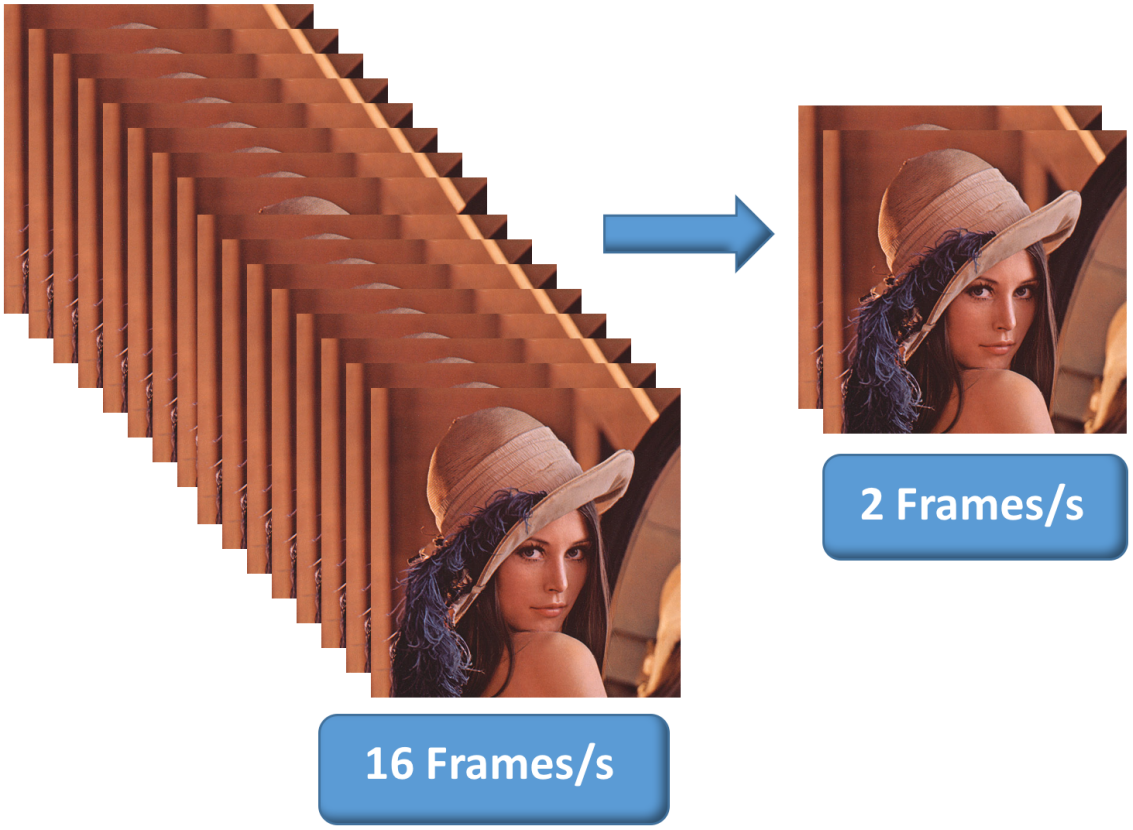


FIGURE 1.6 – Adaptation par modification du nombre d'images par seconde

environnement [Riv00]. Plusieurs types d'adaptations existent dans le domaine de l'informatique :

- L'adaptation au réseau prendra en compte la bande passante, la topologie du réseau, le délai, la gigue et les taux d'erreurs dans le but de déclencher une action.
- L'adaptation aux terminaux prend en compte les caractéristiques matérielles d'un terminal.
- L'adaptation à l'utilisateur prend en compte les préférences, les compétences, les rôles et la mobilité [Van08].
- Enfin, un autre type d'adaptation consiste à adapter une application aux équipements qui sont utilisés avec cette dernière (la taille de l'écran par exemple).

Bien évidemment, tous ces types d'adaptation peuvent et sont souvent employés dans le but de déclencher une seule adaptation globale. Par exemple, une application peut prendre en compte les caractéristiques d'un réseau mais seulement s'adapter aux préférences utilisateur. Dans ce cas, l'adaptation au réseau est implicite. Il y a un compromis réel entre les besoins et les contraintes d'un système. Tous les aspects nécessaires au fonctionnement des applications collaboratives temps réel doivent être pris en compte et le but ultime d'une adaptation est de fiabiliser et d'optimiser le transfert de données. D'autres types d'adaptation peuvent être présents dans différents processus.

L'ADAPTATION LORS DE LA CONCEPTION ET DU DÉVELOPPEMENT DES APPLICATIONS

La conception et le développement d'une application suivent un cycle décrit dans la figure 1.7) avant la livraison finale [Yeo01, Erg15]. Ce processus passe par une phase de faisabilité, d'analyse, de design, et les développeurs réalisent les différents composants du futur système. Un système ne peut pas être conçu en ayant pris en compte et en ayant prédit toutes les demandes actuelles et futures des utilisateurs. En d'autre terme, un système optimal et/ou complètement configurable est difficile voire impossible à concevoir. Et c'est un défi de comprendre les exigences des utilisateurs pour deux raisons principales [Abr04] :

- le groupe potentiel d'utilisateurs n'est pas forcément connu à l'avance mais nécessite d'être analysé par rapport aux futurs scénarios d'utilisations. Ces groupes d'utilisateurs nécessitent d'être analysés car la vision de tout un chacun évolue,

- les visions du futur système doivent être projetées dans un futur et non sur l'expérience des utilisateurs actuels. Donc, les utilisateurs actuels ne seront pas précis concernant leurs demandes du futur système.

L'une des principales tâches de la conception des applications orientées utilisateurs est de négocier et de faciliter la communication entre les utilisateurs et les développeurs. Et ainsi adapter au mieux une solution informatique à des exigences utilisateurs. Toutefois, malgré diverses techniques existantes sur ce type de processus, de nombreuses applications actuelles nécessitent une adaptation constante due à leurs expositions à des situations changeantes.

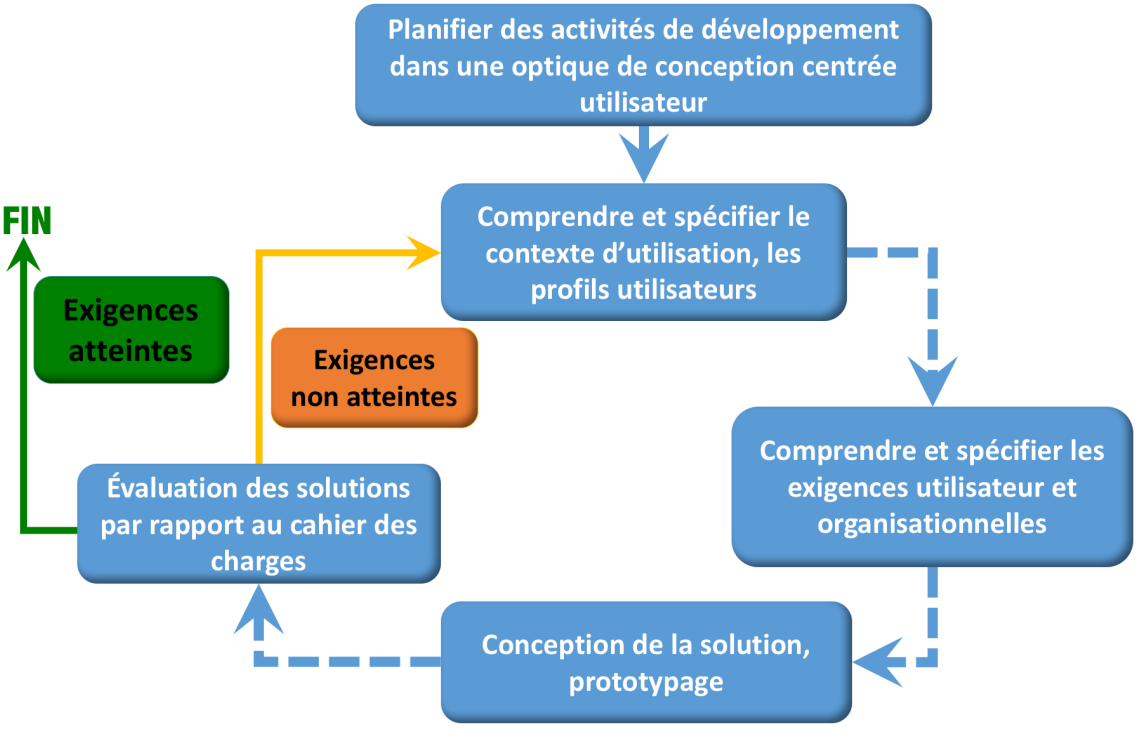


FIGURE 1.7 – Le cycle de conception centrée utilisateur

L'adaptation par transcodage des données

Dans ce type d'adaptation deux techniques sont souvent employées. Une première consiste à changer le média des données [And14]. Un transcodeur est utilisé pour extraire le contenu sémantique des données qui lui sont fournies pour produire de nouvelles données avec le même contenu sémantique mais sous une autre forme. Par exemple, lorsqu'un périphérique ne dispose pas d'un haut-parleur ou encore quand il s'agit d'une application pour les malentendants [Hea13], pour lire un flux audio un algorithme de reconnaissance vocale peut être utilisé dans le but de transcoder le flux audio en un flux texte.

Une deuxième technique consiste à changer le format des données [Nar14]. Ceci permet de conserver le contenu d'un média identique en utilisant un format plus approprié. Ce type d'adaptation est souvent utilisé lorsque le format de données à transmettre ne convient pas à la transmission en question ni au terminal récepteur ou à l'utilisateur. Une page HTML peut ne produire que le texte brut en omettant la feuille de style CSS. La présentation est modifiée mais les données restent identiques. Le changement de format peut être nécessaire et est conseillé lorsqu'un récepteur ne bénéficie pas des ressources nécessaires de l'application ou des codecs requis.

Ces deux techniques impliquent un coût lors du traitement des données au niveau de l'émetteur ou un composant intermédiaire mais permettent de diminuer fortement la quantité de données traitées par le récepteur.

L'adaptation au cycle de vie d'une application

Même si certains logiciels sont implémentés en étroite collaboration avec les utilisateurs finaux et que le degré d'acceptation par ces mêmes utilisateurs est honorable, le produit fini doit fournir une agilité à s'adapter à des conditions changeantes (par exemple, taille de l'écran). L'environnement opérationnel changera, les tâches pour l'application seront différentes en cours d'utilisation que pendant la conception, les utilisateurs finaux seront multiples et de modes d'utilisations hétérogènes et leurs compétences et attentes évolueront [DeL13]. Il est impossible pour les développeurs d'anticiper toutes les modifications possibles et surtout si c'est un domaine qu'ils connaissent très peu, comme par exemple le domaine de la médecine ou encore de la finance.

Les conditions changeant de manière très dynamique au cours de l'utilisation, ceci décale le processus de personnalisation de la phase de développement aux phases d'utilisation et de l'exploitation [Fug14]. Pour toutes ces raisons, les développeurs implémentent des techniques d'adaptation dans le système conçu dans le but de réagir rapidement aux différents changements d'environnements et de conditions du nouveau système. L'étude de l'état de l'art nous conduit à l'élaboration de la figure 1.8 qui résume les différentes contraintes de l'adaptabilité à prendre en compte lorsque nous cherchons à mettre en place des mécanismes d'adaptation dans une application.

Du point de vue de l'utilisateur, ses préférences, son rôle, sa priorité, ses compétences peuvent être pris en compte lors d'une adaptation. Les préférences se décomposent en plusieurs catégories dont nous pouvons noter principalement la langue de l'utilisateur et les mécanismes d'adaptation à ses préférences.

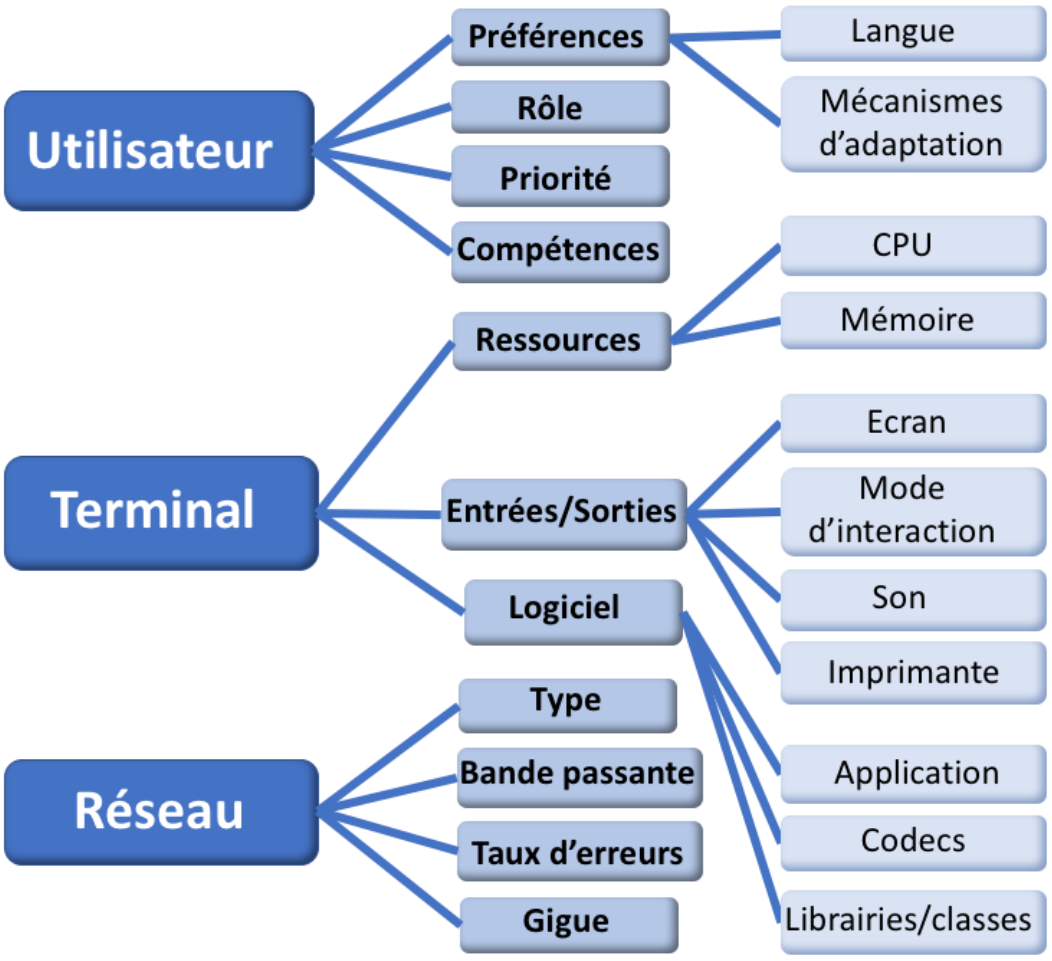


FIGURE 1.8 – Les contraintes de l'adaptabilité

Parmi les facteurs qui peuvent influencer une adaptation, nous distinguons les terminaux et leurs caractéristiques qui sont des facteurs déterminants dans les mécanismes d'adaptation. En effet, leurs ressources comme par exemple la vitesse de calcul ou la mémoire influeront sur l'adaptation des données. Les entrées et sorties telles que le mode d'affichage, le mode d'interaction, et la représentation physique des informations, peuvent aussi être prises en compte. Enfin, il est aussi primordial de prendre en compte les applications logicielles (systèmes d'exploitation, codecs, librairies/classes disponibles) qui entourent le système cible de l'adaptation.

Notre troisième axe concerne le réseau dans lequel se situe un système. Le type de réseau, la bande passante disponible, le taux d'erreurs, et la gigue détermineront le type d'adaptation et l'opportunité d'une adaptation. Les travaux que nous avons menés sont principalement dans cet axe et le type de réseau ainsi que la bande passante disponible seront les éléments déclencheurs d'une éventuelle adaptation.

L'ADAPTATION AU DÉBIT

L'adaptation au débit est souvent présente dans des applications transmettant des données sur des réseaux hétérogènes. Lorsque, dans une transmission, il existe un client connecté sur un réseau aux capacités réduites en matière de bande passante, il est alors nécessaire de réduire le flux généré par l'application. Par exemple, dans les applications de streaming, les flux vidéo sont souvent pré-encodés pour une transmission éventuelle. L'application ne connait pas à l'avance les caractéristiques du réseau qui sera utilisé. L'adaptation permettra à ces flux d'être adaptés dynamiquement selon la qualité du réseau sous-jacent en termes de bande passante disponible et les variations qui peuvent survenir (figure 1.9). Cette technique implique souvent une baisse de débit associée à des qualités visuelles dégradées [Nar14].

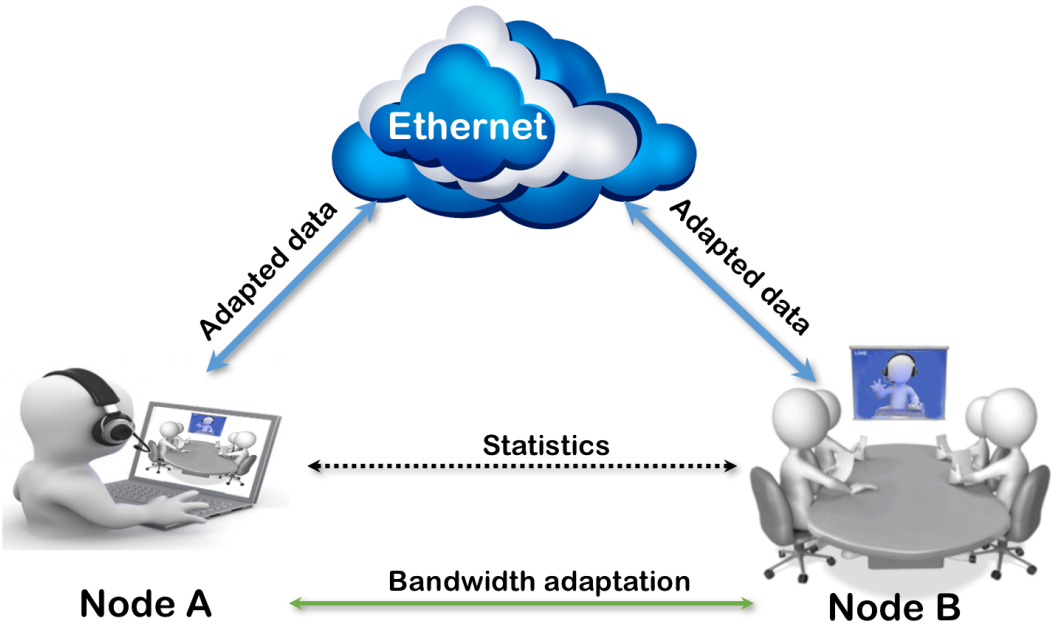


FIGURE 1.9 – L'adaptation au débit dans une transmission vidéo

L'ADAPTATION DES PROTOCOLES RÉSEAUX

L'adaptation doit également tenir compte des réseaux. Lorsque les données transitent sur plusieurs types de réseaux, le protocole utilisé pour leur transport peut s'avérer inefficace ou incompatible avec un des réseaux sous-jacents (IP multicast, par exemple, n'est pas supporté dans tous les domaines [RFC3170]). De plus, beaucoup d'architectures sécurisées n'autorisent l'utilisation que d'un nombre limité de protocoles comme HTTP par le biais d'un pare-feu.

Le premier type d'adaptation dans ce domaine consiste à changer de protocole de transport [Fox98]. Les données sont désencapsulées et réencapsulées dans un flux transmis avec un nouveau protocole. Par exemple, un flux de données diffusé peut être transmis en TCP si le réseau auquel il est connecté ne supporte pas le protocole UDP.

Le deuxième type d'adaptation consiste à utiliser un protocole spécifique pour pouvoir

passer soit un réseau, soit un pare-feu. Les données transmises avec un premier protocole sont encapsulées à l'intérieur d'un flux transmis avec un second protocole [Jur04]. Par exemple, un flux de données transmis avec le protocole RTP doit traverser un pare-feu n'autorisant que le protocole HTTP. Dans ce cas, tous les paquets RTP seront encapsulés en HTTP pour pouvoir passer le pare-feu.

1.3.2/ ADAPTATION ET ADAPTABILITÉ

Le problème de l'informatique ambiante est la haute variabilité de l'environnement qui demeure la principale raison d'adaptation [Dou04]. De ce fait, il est important que les applications s'adaptent à l'environnement qui les entoure. L'adaptation logicielle peut prendre plusieurs formes :

- D'une part, nous parlerons d'un système adaptable lorsqu'un utilisateur peut interagir avec le système, le modifier, le personnaliser.

- D'autre part, un système adaptatif identifie une situation et s'y adapte ; le déclenchement d'une telle adaptation peut être d'origine humaine ou encore d'un certain nombre d'observations de la part du système lui-même, on parle alors de système auto-adaptatif [Van08].

Dans le langage français, les termes *"système adaptable"* et *"système adaptatif"* se résument en un seul terme : adaptabilité. Même si en français les deux termes se traduisent de la même façon et signifient la même chose, en anglais il existe une différence.

Ainsi, l'adaptation se décompose en deux termes anglais *"adaptivity"* et *"adaptability"*, tous deux traduits en français en adaptabilité :

- Le terme anglais *"adaptivity"* indique un système capable de s'adapter automatiquement à ses utilisateurs par rapport aux conditions changeantes, donc un système adaptatif [Van08].
- Le terme *"adaptability"* fait référence aux utilisateurs qui peuvent potentiellement paramétrer le système par eux-mêmes. Nous parlerons dans ce cas d'un système adaptable [Van08].

Les systèmes adaptatifs et adaptables sont complémentaires les uns aux autres. Ces deux méthodes renforcent la relation entre les besoins de l'utilisateur et le comportement du système une fois le développement du système terminé. Ainsi, le système est maintenu flexible pendant toute son utilisation. Selon R. Oppermann et R. Rasher, dans [Opp97], les termes *"adaptivity"* et *"adaptability"* sont deux caractéristiques d'un système qui le rendent capable de s'adapter et de modifier son interaction avec l'utilisateur.

Pour A. Kobsa [Kob04], le terme *"adaptivity"* fait référence à la sélection et la présentation du contenu effectuées par le système en relation avec les préférences utilisateurs tandis que le terme *"adaptability"* signifie que l'utilisateur est en mesure de consciemment personnaliser une application. A. Kobsa fait valoir que la majorité des applications logicielles permettent aux utilisateurs de modifier certaines caractéristiques manuellement pour indiquer leurs préférences, tandis que peu d'applications sont en mesure de reconnaitre les besoins des utilisateurs et d'y répondre de manière automatique. A. Battou [Bat10] se réfère, quant à lui, pour l'adaptabilité à la capacité des systèmes d'apprentissage adaptatifs à adapter automatiquement le processus d'apprentissage aux exigences et aux préférences spécifiques d'un apprenti particulier. Dans le cas des systèmes d'apprentis-

sage, les auteurs dans [Gka08, Gka10] proposent l'utilisation de patrons de conception pour la création d'objets d'apprentissage d'adaptation.

Dans la littérature, l'adaptation sera également différenciée suivant qu'elle est réactive ou proactive [Akk07]. Une adaptation réactive est déclenchée lors de la découverte d'un contexte pertinent tandis qu'une adaptation proactive prépare de nouvelles actions pour les appels à venir lors de la détection d'un contexte pertinent. D'autres approches plus complexes permettent d'obtenir un autre type d'adaptation proactive, il s'agit alors d'anticiper la détection d'un contexte pertinent grâce à un historique des contextes observés ou par un mécanisme d'apprentissage (système de règles, inférences bayésiennes, Intelligence Artificielle, . . .).

1.3.3/ LES QUATRE W'S

Lorsque nous cherchons à adapter un système, la règle des quatre W's [Inv09] peut nous aider à trouver exactement ce que nous voulons adapter. En effet, les systèmes que nous considérons peuvent changer, par le biais d'une adaptation, leurs structures et/ou leurs comportements. La règle des quatre W's caractérise la nature du changement selon les quatre axes suivants :

- Le *Why* : Pourquoi avons nous le besoin de changer ? Cet axe rend explicite la nécessité du changement. Du point de vue de l'ingénierie logicielle, ce changement est toujours réalisé pour répondre aux exigences du cahier des charges. Ce changement peut être dû au fait que les exigences ont évolué ou alors que le système ne se comporte pas correctement selon les exigences énoncées. Ces exigences peuvent être fonctionnelles ou non fonctionnelles.

- Le *What* : Quelle est la partie du système qui est affectée par le changement ? Se référant à des modèles architecturaux, les changements peuvent affecter la structure et/ou le comportement d'une application. Pour la structure, de nouveaux composants peuvent être ajoutés ou retirés. Pour le comportement de l'application, les composants peuvent modifier leur fonctionnalité et les connecteurs peuvent modifier leurs protocoles d'interaction.

- Le *when* : Quand appliquer une adaptation ? Cet axe permet de capter le moment pendant la durée de vie du système dans lequel un changement se produit et nécessite une adaptation. Cela ne signifie pas que le changement se produit nécessairement au moment de l'exécution.

- Le *what/who* : Par qui ou par quoi un changement est-il survenu ? Cet axe implique de surveiller le système afin de recueillir des données pertinentes pour les évaluer et prendre une décision sur les changements alternatifs pour ensuite effectuer le changement réel.

1.3.4/ LES TECHNIQUES DE L'ADAPTATION

Comme nous l'avons montré dans la section précédente, plusieurs formes sont envisageables pour l'adaptation logicielle. Lorsqu'un utilisateur peut interagir avec le système et

par conséquent le modifier ou le personnaliser, nous parlons d'un système adaptable. Un système adaptatif s'adapte à une situation identifiée. Le déclenchement de cette adaptation peut avoir pour origine un certain nombre d'observations (réseau de capteurs, système d'apprentissage sur un phénomène bien précis) ou encore une intervention humaine. Nous parlons de systèmes auto-adaptatifs pour les systèmes totalement autonomes pour lesquels l'intervention humaine n'est pas requise. Un système adaptable peut être vu comme de l'adaptation statique tandis qu'un système adaptatif peut être vu comme de l'adaptation dynamique [Akk07].

Les applications sont exécutées sur des infrastructures complexes, hétérogènes, et hautement entrelacées sur lesquelles des évènements multiples peuvent survenir. Certaines applications peuvent être redémarrées (interruption, prise en compte de nouveaux paramètres, puis exécution à nouveau) afin d'appliquer les changements nécessaires. Mais, il existe des systèmes critiques qui ne peuvent être redémarrés, comme par exemple une application d'urgence médicale. Dans ce cas, les applications doivent adapter leur comportement lors de l'exécution afin de prendre en compte les nouvelles conditions de l'environnement. L'étude de l'état de l'art sur les techniques de l'adaptation [Mck04, Cet09, Alf11] nous a permis de distinguer deux techniques permettant de réaliser l'adaptation dynamique logicielle : l'approche paramétrée et la reconfiguration architecturale.

- L'approche paramétrée est utilisée afin de modifier des variables qui agissent directement sur l'exécution de l'application. Avec cette approche, le paramétrage des composants déjà existants est possible mais cela ne permet pas d'en ajouter de nouvelles. Toutefois, de part sa simplicité de mise en œuvre, elle est souvent la plus utilisée.
- La reconfiguration architecturale permet de changer des composants d'une application par d'autres afin de répondre au mieux à certains éléments qui sont pris en compte dans le nouvel environnement.

Dans les deux cas, les composants peuvent être des algorithmes, ou des boites noires. Une raison de plus, pour les concepteurs des systèmes d'informations de choisir l'approche de la programmation par composants.

Différentes autres techniques peuvent également être utilisées pour réaliser une adaptation logicielle.

L'APPROCHE MOP (*Meta Object Protocol*)

Bien que les intergiciels soient maintenant bien adaptés, il est crucial que les standards restent sensibles aux nouveaux défis tels les *groupwares*, le multimédia, le temps réel et la mobilité croissante. De tels défis exigent de nouvelles approches de l'ingénierie des plateformes d'intergiciels. Par exemple, les applications multimédia nécessitent un support très spécifique en termes de protocoles de communication et de gestion de ressources. Le concept de *réflexion* a été introduit pour la première fois par Smith [Smi82] qui a introduit l'hypothèse sur la *réflexion* explique : *dans la mesure où un processus de calcul peut être construit pour raisonner sur un monde extérieur comme un processus tiers (l'interprète) agissant formellement sur les représentations de ce monde, ce même processus de calcul pourrait aussi être fait pour raisonner sur lui-même comme un processus tiers (l'interprète) manipulant formellement les représentations de ces propres opérations et structures*.

L'importance de cette hypothèse est qu'un programme peut accéder, raisonner et modifier sa propre interprétation lors de son exécution. L'accès à l'interpréteur (*interpreter*) est fourni par le MOP (*Meta Object Protocol*) qui définit les services disponibles par des méta-classes à un méta-niveau. Des exemples d'opérations disponibles au méta-niveau comprennent la modification de la sémantique du passage des messages et l'insertion, avant ou après, des actions autour des invocations de méthodes. L'accès au méta-niveau est fourni par un processus de *réification* qui rend un aspect de la représentation interne explicite et donc accessible du programme (*introspection*). Le processus inverse est l'*absorption* (souvent appelée l'*intercession*) où un aspect du méta-système est modifié ou remplacé [Bla09].

BC. Smith propose, dans [Smi82], de connecter un grand corpus de recherche à l'application de la réflexion. Initialement, ce travail a été limité au domaine de la conception de langage de programmation et, plus récemment, à des systèmes distribués. La motivation principale d'un système réflexif est de fournir un moyen basé sur des principes (par opposition à un moyen ad hoc) afin de réaliser une ingénierie ouverte. Par exemple, la réflexion peut être utilisée pour inspecter le comportement interne d'un langage ou d'un système. En exposant l'implémentation sous-jacente, il devient simple d'insérer un comportement supplémentaire pour surveiller l'implémentation, par exemple, des moniteurs de performance ou des moniteurs de qualité de service. La réflexion peut également être utilisée pour adapter le comportement interne d'un langage ou d'un système (comme par exemple, l'insertion d'un objet filtrant les paquets dans un réseau afin de réduire l'utilisation de la bande passante d'un flux de communication) [Bla09].

L'APPROCHE PAR L'UTILISATION DE PROXY

En général, un proxy est un système intermédiaire qui agit pour un client en recevant des données depuis une source (par exemple un serveur), les traite et les redirige vers le client [Sha86]. Des proxies sont utilisés pour représenter des objets et peuvent ainsi rediriger des appels de méthodes vers différentes instances. Par exemple, un proxy web peut aussi être capable de filtrer et recompresser les données de manière à économiser de la bande passante ou mettre en cache des données pour permettre un accès plus rapide à celles-ci.

Des services proxy peuvent être utilisés dans le contexte de l'ubiquité pour adapter les flux de communication ou les applications de manière à correspondre aux services disponibles. De plus, des proxies peuvent être installés en frontière d'un réseau fixe, autorisant ainsi l'utilisation de différents protocoles dans les domaines fixes et mobiles. Plusieurs architectures [Joh01, Jar11, Roh16] utilisent un proxy pour la réalisation de l'adaptation. Leur objectif commun est d'adapter dynamiquement les données à l'aide du proxy.

L'APPROCHE PAR LA NOTION DE CONTENEUR

Un conteneur est l'entité dans laquelle sont implémentées les règles métier pour être déployées sur un serveur. Par exemple, il peut s'agit d'un fichier de type *"war"* (Web application Archive). Le conteneur peut héberger plusieurs entités implémentant les moteurs pricipaux des règles métier. Ces moteurs contiennent des fichiers *".class"* du langage de programmation Java par exemple. L'implémentation des entités peut être faite sur la base

de différents paradigmes tels que la programmation orientée objet, la programmation orientée aspect (POA), l'architecture orientée service, ...

La relation entre le conteneur et l'implémentation de la logique de base peut être décrite par une entité de configuration qui sera nécessaire pour le déploiement et éventuellement pendant l'exécution d'une application [Mar14]. Le conteneur contrôle les interactions du composant avec l'extérieur. Il peut également gérer ses composants internes. Il peut donc être réutilisé dans le but de paramétrer ou de reconfigurer un assemblage de composants pour modifier le comportement d'une application.

L'APPROCHE PAR LA PROGRAMMATION ORIENTÉE ASPECT (POA)

Le développement de logiciels orienté aspect (POA) est une approche de l'ingénierie logicielle qui permet l'identification explicite, la séparation, et l'encapsulation des préoccupations qui traversent la modularisation primaire d'un système. Les fonctionnalités fondamentales ne peuvent pas être découpées en sous-fonctionnalités et donc elles ne peuvent pas être efficacement structurées en modules, en utilisant d'autres techniques de développement bien connues telles que le développement orienté objet.

Par conséquent, des fonctionnalités non issues des règles métiers se retrouvent dispersées dans tout le système et se mêlent avec les fonctionnalités fondamentales, qui elles sont issues des règles métiers du système. Même si les fonctionnalités non issues des règles métiers proviennent des exigences non fonctionnelles comme le traçage de l'exécution d'une application, la sécurité, la persistance, et l'optimisation, elles englobent aussi des fonctions qui ont souvent leurs logiques comportementales réparties sur plusieurs modules.

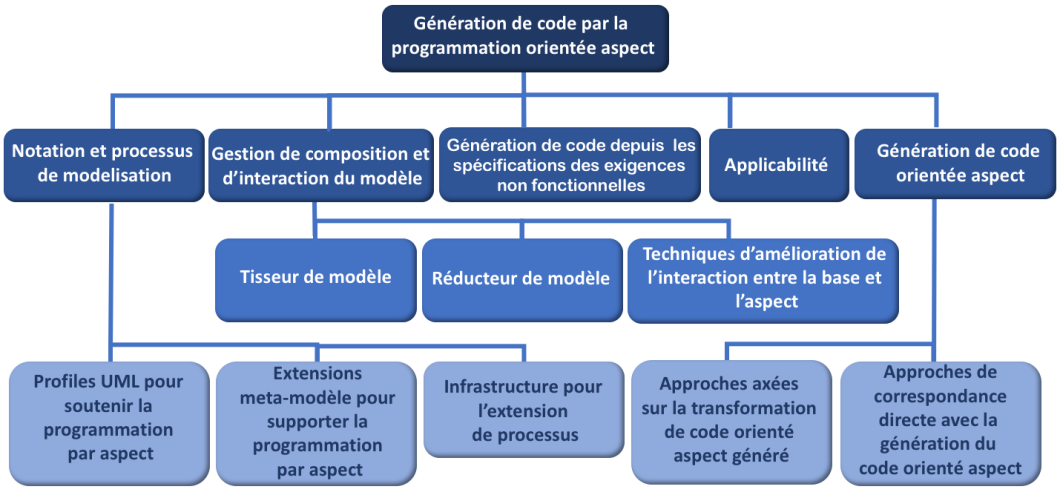


FIGURE 1.10 – Classification de la modélisation et de la génération de code orienté aspect

En utilisant la programmation orientée aspect, ces préoccupations sont identifiées, modélisées et mises en œuvre indépendamment des principales fonctionnalités du système. Une fois séparées de cette manière en modules, ces fonctionnalités nécessitent un mécanisme de composition (appelé le *tissage* pour contrôler où et quand le comportement des compositions est appliqué [Meh13]). Ainsi les aspects peuvent être des

comportements qui se *"tissent"* avec l'application existante. Ainsi ceci permet d'ajouter ou de supprimer dynamiquement à l'exécution des fragments de code relatifs à des préoccupations transverses [Kic97]. Une classification de la programmation orientée aspect a été proposée dans [Meh13].

Le modélisation des notations et processus inclut ou bien une étude unique ou une composition détudes. Des diagrammes UML, des méta-modèles ou encore des infrastructures existantes peuvent être utilisés dans cette phase.

La gestion de composition et d'interaction du modèle permet de définir le *tisseur* du modèle. Cette phase permet également de définir des techniques d'amélioration de l'interaction entre la base (ce qui a été défini) et l'aspect (ce qui sera étudié et implémenté).

Les exigences non fonctionnelles, comme les aspects recoupent souvent plusieurs composants d'un système logiciel. Par conséquent, la spécification des exigences non fonctionnelles est un domaine de l'ingénierie logicielle qui est très pertinent pour la conception d'applications orientées aspect. Par ailleurs, le domaine d'application du logiciel (*l'applicabilité*) déterminera si la programmation par aspect offre des avantages sur les autres types de programmation.

Enfin, la classification de génération de code orienté aspect permet une approche explicite en vue de générer du code orienté aspect à partir de modèles. Par exemple, un mécanisme permettant la conversion entre un modèle orienté aspect et un code peut être utilisé.

La capacité d'un système à s'adapter implique une certaine modularité. En effet, un système qui ne respecterait pas cette propriété devrait être entièrement changé. À l'inverse s'il est conçu comme un assemblage de modules, il devient alors possible d'en remplacer uniquement certains.

Notons que ces adaptations peuvent être ou bien dynamiques ou bien statiques, et ainsi ou bien se dérouler à l'exécution ou bien lors de la phase de chargement ou de compilation d'une application. Dans le cadre de l'informatique ubiquitaire, des adaptations dynamiques sont souhaitables.

Pour résumer

SYNTHÈSE

L'adaptabilité exige de mettre en place des mécanismes pour garantir l'intégrité des données à tous les niveaux et par tous les utilisateurs et ce quels que soient leurs environnements. Il s'agit de la capacité du système à prévoir et à vérifier que les échanges d'informations se font sans aucun problème et que les interactions entre les utilisateurs ne sont pas entravées par des contraintes issues de problèmes de configuration ou de congestion dans les canaux de communication. Avec l'ère de l'informatique ubiquitaire, pour les environnements collaboratifs, les données adaptées représentent un accès rapide pour une utilisation optimale. Ces données sont adaptées en fonction des contraintes du système et des besoins. Une adaptation se doit d'être la plus efficace possible car dans les environnements collaboratifs, des terminaux mobiles peuvent être utilisés et ces derniers sont souvent limités en puissance de calcul et/ou se trouvent sur des réseaux de communication moins performants. Dans un environnement collaboratif, un individu prend part aux actions de l'environnement. Cet individu peut se trouver dans un groupe qui travaille dans un même but. Le rôle de chaque individu du groupe peut alors évoluer au cours du temps. Chaque individu dispose d'une vue.
Les systèmes collaboratifs sont bâtis autour du concept *What You See Is What I See* (WYSIWIS). L'échange d'informations est assuré par l'espace de communication. L'espace de production est l'espace commun utilisé par les utilisateurs afin d'agir sur des données communes. L'espace de coordination est chargé de gérer la coopération entre les utilisateurs. Deux modes de fonctionnement sont distingués : le mode asynchrone et le travail en session (synchrone). La collaboration en temps réel vise à simuler la réalité, et la notion de réunion virtuelle impose la participation active. Les systèmes collaboratifs sont basés sur un métamodèle mais qui est non final et flexible. Ils sont conçus selon des contraintes technologiques, budgétaires, et les compétences et habitudes des concepteurs. La haute variabilité de l'environnement dans lequel se trouve un système collaboratif demeure la principale raison de l'adaptation. De ce fait, il est important que les applications s'adaptent à ce qui les entoure, à leur environnement. D'une part, un système adaptable permet à un utilisateur d'interagir avec le système et par ce biais de le modifier, de le personnaliser. D'autre part, un système adaptatif identifie une situation et s'y adapte : le déclenchement d'une telle adaptation peut être d'origine humaine ou encore d'un certain nombre d'observations de la part du système lui-même (système auto-adaptatif) [Van08].
Une adaptation réactive est déclenchée lors de la découverte d'un contexte pertinent tandis qu'une adaptation proactive prépare de nouvelles actions pour les appels à venir lors de la détection d'un contexte pertinent. D'autres approches plus complexes permettent d'obtenir un autre type d'adaptation proactive, il s'agit alors d'anticiper la détection d'un contexte pertinent grâce à un historique des contextes observés ou par un mécanisme d'apprentissage (comme un système de règles, d'inférences bayésiennes, . . .).

Dans le chapitre suivant, nous détaillerons la notion de contexte. Cette notion est souvent couplée aux termes adaptation et adaptabilité. Afin de pouvoir déclencher une adaptation, un système doit pouvoir analyser son environnement. Il doit pouvoir calculer et déduire l'adaptation nécessaire en fonction de l'évolution du contexte .

2

LES CIBLES ET LES MÉCANISMES DE L'ADAPTATION

Ce chapitre présente tout d'abord la notion de contexte, ainsi que ses différentes définitions issues de la littérature.

Dans la deuxième partie de ce chapitre, nous étudions les mécanismes logiques existants et pouvant être utilisés lors de la conception d'un intergiciel de prise en compte du contexte.

2.1/ LA NOTION DE CONTEXTE DANS LES ENVIRONNEMENTS DISTRIBUÉS

Le contexte n'est pas un concept nouveau en informatique. Dès les années soixante, les chercheurs en systèmes d'exploitation, théorie des langages et intelligence artificielle exploitaient déjà cette notion. Avec le développement de la notion d'adaptation dans les systèmes distribués, le terme de *"contexte"* a été redécouvert et est placé au cœur de plusieurs travaux de recherche dans le domaine, sans qu'il n'y ait une définition consensuelle claire et définitive.

La notion de contexte est en effet très importante dans un environnement qui souhaite intégrer de l'adaptation. Nous retrouvons ce principe dans plusieurs travaux de recherche. Ainsi dans cette thèse, nous étudions diverses définitions de la notion de contexte. Cette notion, associée à l'informatique ambiante, est apparue pour la première fois en 1993 dans les travaux de Mark Weiser [Wei93] comme l'ensemble des informations à prendre en compte en vue d'une adaptation. D'un point de vue informatique, le contexte est composé de l'ensemble des infrastructures matérielles/logicielles, d'un environnement et de l'ensemble d'entités ambiantes non informatisées comme les utilisateurs. Il s'agit des éléments présents dans l'environnement d'un système informatique ambiant comme nous l'avons montré dans la figure 1.4 de la section 1.2.

Issu du latin "contextus" qui signifie "assemblage" ou encore "contextere" qui signifie "tisser avec", le terme "contexte" englobe un ensemble de concepts. Le dictionnaire Larousse définit ce terme comme étant un *ensemble de conditions naturelles, sociales, culturelles dans lesquelles se situe un énoncé, un discours*. Et plus spécifiquement, dans le domaine de l'informatique, ce dictionnaire le définit comme étant *un ensemble*

d'informations caractérisant l'état de l'unité centrale d'un ordinateur à tout moment de l'exécution d'un programme. Ces définitions partageant l'idée d'ensemble d'informations associées implique une certaine notion temporelle.

2.1.1/ LA DÉFINITION DU CONTEXTE

Dans la littérature, plusieurs chercheurs ont proposé des définitions du contexte. Un des plus connus est le Docteur Anind Dey des laboratoires de recherches de Berkeley. En 1999 A. Dey et al., dans [Dey99], proposent la définition suivante pour la notion de contexte :

Définition

Dans [Dey99], Le contexte se compose de n'importe quelle information qui peut être utilisée pour caractériser la situation d'une entité. Une entité est une personne, un lieu, ou un objet en rapport avec une interaction homme-machine, y compris l'utilisateur et l'application.

Les mêmes idées se retrouvent à nouveau dans [Pas98, Dey00]. J. Pascoe définit le contexte comme étant n sous-ensembles d'états physiques et conceptuels ayant un intérêt pour une entité particulière (notion de pertinence).

Il évoque que les définitions données par Schilit et Theimer [Sch94] sont basées sur des exemples et ne peuvent pas être utilisées pour identifier de nouveaux contextes. Il ajoute également que les définitions données par PJ. Brown [Bro95], D. Franklin et J. Flachsbart [Fra98], T. Rodden *et al.* [Rod98], R. Hull *et al.* [Hul97], et A. Ward *et al.* [War97] utilisent des synonymes, tels qu'un environnement et une situation, pour décrire le terme contexte. Ces définitions ne permettent pas le concept de dynamicité, d'évolution du contexte.

Il est intéressant de souligner le caractère multiple de la définition d'A. Dey par rapport en particulier à la définition de Pascoe [Pas98] et ainsi qu'aux autres précédemment citées. Tous les aspects importants de toutes les situations sont impossibles à énumérer de manière exhaustive car ils changeront d'une situation à une autre. Par exemple, dans certains cas, l'environnement physique peut être important, tandis que dans d'autres cas, il ne sera pas utilisé dans la représentation du contexte : si l'on veut adapter à l'utilisateur et pas à l'aspect hardware.

Par la suite, les différents auteurs ont repris la définition d'A. Dey et y ont ajouté les éléments manquants tels que la notion temporelle, la sécurité, et les éléments de l'environnement physique, ... La définition d'A. Dey tend à montrer que le contexte n'est pas constitué que de données indépendantes collectées, mais qu'il s'agit de connaissances. Il y a de nombreuses données inutiles que l'on peut récupèrer d'un contexte, mais pour utiliser ce contexte dans le cadre de l'adaptabilité, seulement certaines seront intéressantes à conserver.

En 2004 Paul Dourish, dans [Dou04], propose une étude de l'état de l'art autour de la notion du contexte. Dans les années 2000, l'émergence de l'informatique ubiquitaire comme nouveau paradigme de conception d'applications pose des défis importants pour

la conception d'interfaces homme-machine. Pour l'auteur, durant cette période, la notion de contexte a engendré une confusion considérable autour de cette notion : qu'est-ce que cela signifie, qu'inclut-il, quel est son rôle dans les systèmes interactifs. Il apparaît que ces questions ont déjà trouvé des réponses par A. Dey dans sa définition autour de la notion du contexte. Paul Dourish ajoute que le contexte peut être vu comme étant une forme d'information (un paramètre) stable et définissable. L'activité d'un utilisateur (l'interaction homme-machine) a lieu à l'intérieur du contexte dans l'environnement de l'utilisateur. Les actions d'un utilisateur et l'environnement du contexte sont deux éléments bien distincts.

J. Strassner *et al.*, dans [Str08], proposent le modèle DEN-ng qui structure ces connaissances du contexte. Les informations sont définies pour un contexte particulier, qui permet à différentes informations de s'associer pour définir d'un contexte spécifique (préférences utilisateur, l'état du réseau, consommations CPU/mémoire, ...). Ainsi les auteurs proposent la définition suivante du contexte : "Le contexte d'une entité est une collection de mesures et de connaissances déduites qui décrivent l'état et l'environnement dans lequel une entité existe ou a existé". Cette définition introduit ainsi la notion de temps dans sa définition.

Proposant également un modèle pour structurer les informations du contexte, T. Winograd [Win01] décrit le contexte comme étant un ensemble d'informations structuré et partagé, qui évolue. La nature des informations, de même que l'interprétation qui en est faite, dépend de la finalité. Les informations peuvent être soit passives ou actives. Dans [Che03], les auteurs voient le contexte comme étant un ensemble d'états et de paramètres qui, ou bien détermine le comportement d'une application, ou bien dans lequel un évènement de l'application intéressant pour l'utilisateur se produit. Dans cette définition, apparaît la notion d'informations passives qui décrivent le changement de comportement ainsi que les informations actives qui déclenchent ce changement de comportement et qui peuvent déterminer une éventuelle adaptation.

2.1.2/ LA MODÉLISATION DU CONTEXTE

La modélisation du contexte est devenu un domaine de recherche à part entière et plusieurs travaux sont menés autour de cette thématique. Dès 1994, pour désigner un système doté d'un modèle de contexte, B. Schilit et N. Adams [Sch94] introduisent l'expression de *prise en compte du contexte*. Selon les auteurs, le contexte inclut l'identité des personnes et des objets, la localisation mais aussi les modifications pouvant intervenir sur ces objets. Sa définition se porte sur les changements de l'environnement physique, des ressources de calcul et de l'utilisateur. Puis, cette définition est reprise par N. Ryan *et al.*, dans [Rya99]. Dans ces deux travaux, la définition s'intéresse aux contextes des applications informatiques et au contexte des systèmes.

Dans toutes les définitions sur la notion de contexte, que l'on rencontre dans la littérature, certains auteurs se focalisent sur l'environnement dans lequel se trouve un système et les utilisateurs de ce dernier.

Par exemple, D. Thévenin et Joëlle Coutaz, dans [The99], ont défini le contexte comme étant un triplet *(plateforme, environnement, utilisateur)* :

- La plateforme (ou le support informatique ou encore une application) se trouve dans un environnement virtuel et physique.

- L'environnement est quant à lui l'ensemble des entités présent à un instant t et qui se compose d'objets, de personnes, d'évènements, de périphériques.

- Et enfin cet environnement peut avoir un impact sur le comportement du système ou de l'utilisateur.

Dans la même état d'esprit, les travaux de P. Brown se focalisent particulièrement sur l'utilisateur, et plus exactement sur l'environnement de l'utilisateur [Bro95]. Dans la suite de ses travaux P. Brown [Bro97], toujours axé sur l'utilisateur, a développé de nouveaux éléments tels que l'heure, la température, la saison, l'identité et la localisation de l'utilisateur.

Dans [Abo00], les auteurs identifient les *4 W's (Who/What, Where, When, Why)* comme étant le minimum d'informations nécessaires à la description du contexte. Les *4 W's* correspondent aux 4 premières questions à se poser lorsque l'on souhaite modéliser le contexte.

Sur le même principe S. Ahn et D. Kim, dans [Ahn06], ajoutent que le contexte peut être vu comme étant un ensemble d'évènements interdépendants avec des relations logiques et temporelles entre eux. Un évènement se produit lorsqu'une condition dans une zone ciblée (une partie) de l'environnement dans lequel se trouve une application est respectée et donc déclenchée. Dans un certain nombre de systèmes conçus par l'homme, tels que les réseaux de communication et d'ordinateurs, les systèmes informatiques, les unités centrales d'ordinateurs elles-mêmes, l'essentiel de l'enchaînement dynamique des tâches provient de phénomènes de synchronisation, et d'exclusion mutuelle ou encore de compétition dans l'utilisation de ressources communes, ce qui nécessite une politique d'arbitrage ou de priorité. Il est alors primordial, dans de tels systèmes, de définir des transformations qui sont déclenchées par des évènements ponctuels. Dans les travaux de S. Ahn, deux catégories d'évènements sont utilisées :

- Les évènements discrets : il existe plusieurs définitions d'un évènement discret . Un évènement est défini comme discret lorsqu'il existe un début et une fin. Il est possible de modéliser ces évènements au travers d'automates d'états finis. Par exemple, le mouvement d'un train d'un point A vers un point B se représente par deux évènements : celui du départ et celui de l'arrivée. Et le mouvement en soi est l'intervalle de temps écoulé entre les deux. Dans cet exemple, les sommets sont les points de départ et d'arrivée et l'arc représente le temps écoulé. Le terme d'évènement discret est cependant la plupart du temps utilisé dans le sens restreint, ce qui suggère que le système a été analysé comme une suite d'opérations (arrivée, temps écoulé, ressource utilisée, séparation, reconfiguration, ...). Toutefois, le mot "discret" réfère au fait que la dynamique est composée d'évènements, qui peuvent être des débuts et fins de tranches d'évolution continue mais qui seuls sont intéressants dans la mesure où les fins conditionnent de nouveaux débuts. En d'autres termes, un évènement se produit instantanément et cause la transition de l'état d'une valeur (discrète) à une autre valeur. Il peut être identifié avec une action spécifique qui a été prise (envoi de la requête au serveur). Un évènement est souvent représenté par le symbole e. Lorsqu'un système est affecté par différents types d'évènements, il est possible de définir un ensemble d'évènements E dont les éléments sont tous ces évènements.

- Les évènements continus : contrairement aux évènements discrets qui ont un espace d'état discret et où les transitions d'états sont déclenchées par des évènements. Les évènements continus possèdent un espace d'état continu. Les transitions d'états sont déclenchées par le temps. Par exemple, le lancement d'un calcul sur un serveur avant de renvoyer la réponse à un client est considéré comme étant un évènement continu car d'une part le temps de calcul sur le serveur n'est pas connu à l'avance et d'autre part le calcul et l'envoi de la réponse au client ne sont pas deux éléments distincts. En effet, la fin du calcul provoque l'envoi de la réponse. Autrement dit, une instance d'évènement t d'une durée d, précède l'instance à $t + d$ du même évènement.

Le contexte étant modélisé, il s'agit d'étudier la manière dont les différents systèmes utiliseront les informations de contexte. G. Abowd, A. Dey *et al.*, dans [Abo99], ont critiqué les approches issues de [Sch94] et de [Rya99] en indiquant que ces définitions sont trop spécifiques et ne peuvent être utilisées pour identifier si un système est sensible au contexte ou non. Dans ce même article, les auteurs définissent un système sensible au contexte de la manière suivante :

Définition

Définition issue de [Abo99]

Un système est conscient de son contexte s'il utilise son contexte pour fournir des informations et/ou des services pertinents à l'utilisateur : suivant la tâche de l'utilisateur.

Réflexions sur la notion de contexte

Nos recherches bibliographiques sur la notion de contexte nous ont conduits à une définition volontairement originale de Patrick Brézillon et al., dans [Bre02], qui définit : **"il n'y a pas de contexte sans contexte"**. Le contexte n'existe pas en tant que tel. Il émerge à partir d'un contexte existant, ou se définit pour une finalité précise. Dans notre cas, il s'agira d'adapter les applications aux évolutions du monde qui les entourent. Cette définition est sujette à des ambiguïtés et peut soulever des interrogations. Cela signifierait qu'un contexte, nommé C_2, est défini à partir d'un autre contexte, nommé C_1. Dans ce cas, les questions sont : d'où provient le profil initial C_1 ? À partir de quoi a t-il été composé ? De quoi est-il composé ? Et donc implicitement quels sont les composants statiques et les composants dynamiques.

Il existe de très nombreuses définitions plus récentes mais la plupart d'entre elles utilisent cette différenciation des aspects statiques et dynamiques. Par exemple dans [Mit15], les auteurs définissent ces deux composants du contexte. Un contexte statique ne change pas fréquemment tandis qu'un contexte dynamique change dans le temps et est difficile à prédire. Un contexte statique peut inclure les préférences utilisateur et les recommandations de sécurité. Un contexte dynamique peut inclure la position de l'utilisateur, la vélocité, la charge réseau, la charge de la batterie, l'utilisation mémoire/CPU, la présence et rapport signal sur bruit, ...

Dans les environnements réels, le contexte peut être hautement dynamique et stochastique, c'est à dire, qu'il peut changer très rapidement et est incertain. Il peut être imparfait, peut émettre un ensemble de caractéristiques temporelles, il peut avoir des représentations alternatives, il peut être distribué et peut ne pas être disponible à un instant précis [Bet10].

Nous pouvons également ajouter deux termes qui reviennent de manière récurrente : il s'agit de l'environnement et de la situation.

En conclusion, le contexte est une notion obligatoirement définie par rapport à une ou plusieurs entités auxquelles il se rapporte. Après avoir défini un système ambiant, le contexte doit donc s'établir à partir du choix des entités privilégiées ou de références. Avec l'émergence de l'informatique ambiante, la prise en compte du contexte a vu le jour et a été réellement utilisée depuis les années 1990. L'accent mis sur l'informatique ubiquitaire a évolué depuis les applications dites "stand-alone" vers l'informatique mobile, les applications web, et l'informatique ubiquitaire au cours de la dernière décennie. Toutefois, il est à noter que la prise en compte du contexte est devenue plus populaire avec l'introduction du terme "informatique ubiquitaire" par Mark Weiser dans son article *"The Computer for the 21st Century"* [Wei99].

De toutes les définitions vues de la prise en compte du contexte dans cet état de l'art, celles de G. Abowd, A. Dey et al. [Abo99] et de J. Pascoe [Pas98] sont largement acceptées par la communauté scientifique [Per14]. Ainsi, les intergiciels sensibles au contexte doivent supporter l'acquisition, la représentation, la délivrance, et la réaction [Dey01a].

2.1.3/ L'UTILISATION DU CONTEXTE DANS LES APPLICATIONS DISTRIBUÉES

Après avoir défini un système ambiant, le contexte doit donc s'établir à partir du choix des entités privilégiées ou de référence. Par exemple soit une entité de référence nommée Bob, le contexte de Bob est l'ensemble des entités du système appelées à interagir avec Bob. Soit une entité informatisée, son contexte sera alors composé de son infrastructure et de l'environnement dans lequel elle évolue : lui-même composé des autres systèmes et entités dont les utilisateurs. L'informatique ambiante impose donc de considérer de multiples utilisateurs ou encore de multiples dispositifs [Sou02]. Ces entités peuvent être hétérogènes, qu'il s'agisse de plateformes ou encore de dispositifs. Les utilisateurs peuvent être mobiles et il est nécessaire d'avoir un système extensible et capable de prendre en compte les fortes variations qui peuvent intervenir dans leurs environnements. La dynamicité de l'environnement doit être prise en compte afin d'être réactif le plus rapidement possible pour une utilisation optimale. Ainsi le défi de l'informatique ambiante est d'être capable de s'adapter dynamiquement au contexte.

L'adaptation est présente dans la vie d'un logiciel, de son développement à son installation ou encore à son exécution : c'est ce dernier cas qui nous intéressera plus particulièrement. Lorsque nous parlons de la prise en compte du contexte, nous parlons essentiellement d'une phase de perception, d'une phase d'évaluation et d'une phase d'adaptation [Bet10]. Les phases de perception et d'évaluation permettent de décider quand appliquer une adaptation. Cette dernière peut être décidée ou bien par un utilisateur ou bien lorsqu'un certain nombre de conditions (qui peuvent être autres que temporelles) sont remplies. Les procédés de la prise en compte du contexte pour l'adaptation

sont définis selon trois axes d'étude :

- Le premier axe concerne le modèle du contexte choisi,
- Le deuxième axe repose sur la décomposition fonctionnelle des différentes étapes de traitement des informations pour la prise en compte du contexte,
- enfin, le troisième axe, plus original, étudie le comportement dynamique des mécanismes de prise en compte du contexte au regard de l'évolution des éléments qui le composent.

Ces procédés doivent être attachés à un modèle qui soit utilisable en phase de conception (*design time*) comme au cours de l'exécution (*runtime*) :

- En phase *design time*, le modèle du contexte doit aider les concepteurs d'applications à prévoir les contextes auxquels leurs applications devront être capables de s'adapter,
- En phase *runtime*, le modèle de contexte doit fournir des mécanismes efficaces pour détecter les changements de contexte dans le but de déclencher une adaptation de l'application à ces changements.

De nombreux intergiciels commerciaux pour l'informatique ambiante ont été proposés et ont ainsi été peu publiés. L'intergiciel (middleware) est la couche applicative servant d'intermédiaire entre plusieurs applications, mais aussi entre applications et systèmes, et ceci principalement de manière distribuée. Cette couche offre des services de haut niveau principalement liés aux communications. Ces intergiciels doivent pouvoir assurer un niveau d'adaptation par la prise en considération d'un grand nombre d'éléments qui peuvent être sensibles au contexte (utilisateur, terminal, environnement, ...). Le comportement de ces applications interactives doit être en corrélation avec les capacités matérielles et logicielles de l'ensemble des composants du contexte (la diversité des terminaux mobiles, la diversité des préférences d'utilisateurs, ...). Généralement le type d'adaptation de l'information au contexte est déterminée selon l'acquisition d'information du contexte, et en découle une adaptation qui est déclenchée.

Les sous-sections, qui suivent, présentent les intergiciels les plus connus dans la littérature utilisant différentes techniques d'adaptation au contexte.

L'intergiciel Context Toolkit [Dey01a]

Context Toolkit a été développé en vue de faciliter le développement et le déploiement d'applications sensibles au contexte. L'architecture de cet intergiciel est un précurseur dans le domaine de la gestion de contexte en environnement ubiquitaire [New03]. Il a été développé dans le but d'aider la construction d'applications sensibles au contexte. Il introduit trois abstractions principales, la récupération de données d'un contexte, des techniques de raisonnement afin d'interpréter les données capturées et enfin la prise de décision. C'est un canevas logiciel orienté objet empruntant aux interfaces hommes-machines (IHM) les concepts de programmation événementielle et de *widget* pour la collecte du contexte des ressources. Le choix de l'analogie avec le concept de *widget* du

domaine des IHM s'explique par la possibilité des deux modes d'interaction "*observation*" et "*notification*", et la mise en œuvre des outils génie logiciel de la conception et de la programmation orientée objet, notamment l'encapsulation et la réutilisation. Les *widgets* mémorisent l'historique des données brutes collectées et les mettent à disposition des clients. Comme montre la figure 2.1 :

- L'agrégateur est le médiateur avec l'application.
- Le canevas propose les autres fonctionnalités de la gestion de contexte.
- L'interpréteur compose et abstrait les informations de contexte.
- Le service contrôle les actions de l'application sur le contexte.
- Le *discoverer* agit comme un serveur de noms ou comme un registre.

Dans la philosophie de ce canevas logiciel, les fonctions d'interprétation et d'agrégation sont à programmer dans des blocs monolithiques : un agrégateur et un interpréteur par client, et ce quel que soit le nombre de *widgets* et le niveau d'abstraction demandé par l'application.

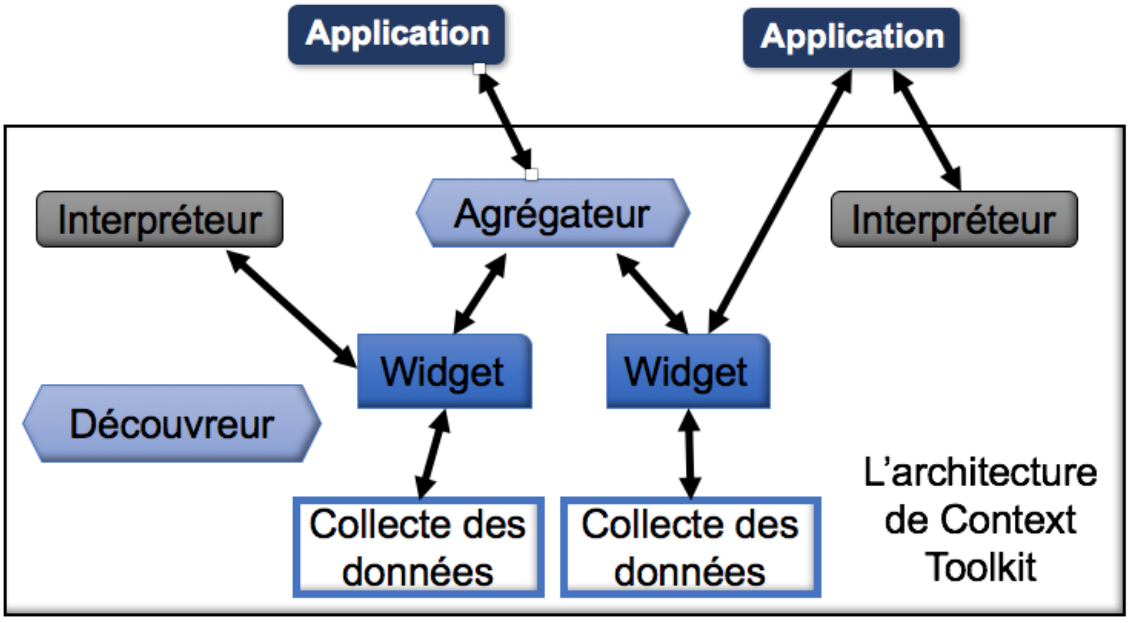


FIGURE 2.1 – L'architecture de l'intergiciel Context Toolkit

Par conséquent, *Context Toolkit* met en œuvre les patrons de conception *"Architecture en couches"*, *"Publication/notification"* et *"Nommage"*. Il permet les modes *"Observation* et *Notification"*, ainsi que *"l'extensibilité"* avec la définition de classes abstraites et la gestion de l'historique des informations de contexte, mais uniquement au niveau des collecteurs.

Notons que la gestion des ressources système consommées par les traitements d'inférence est explicitement laissée de côté par les auteurs, de même que la distribution des informations de contexte.

L'intergiciel Aura [Gar02]

Aura est un système orienté tâche et basé sur une architecture distribuée mettant l'accent sur les différents dispositifs informatiques utilisés par des utilisateurs humains. L'objectif est de lancer un ensemble d'applications appelées *Personal Aura* dans tous les appareils afin de gérer les tâches d'un utilisateur d'une manière sensible au contexte en garantissant l'interopérabilité. Comme les effets de la loi de Moore font que les systèmes informatiques deviennent moins chers et plus abondants, un nouveau problème se pose : le goulot d'étranglement n'est pas la capacité d'un disque dur ou la puissance de calcul d'un processeur mais l'attention humaine limitée. L'attention humaine fait référence à la capacité d'un utilisateur à participer à ses tâches principales, en ignorant les distractions générées par le système telles que les mauvaises performances et les échecs. Cet intergiciel vise à minimiser les distractions sur l'attention d'un utilisateur, en créant un environnement qui s'adapte automatiquement au contexte et aux besoins de l'utilisateur.

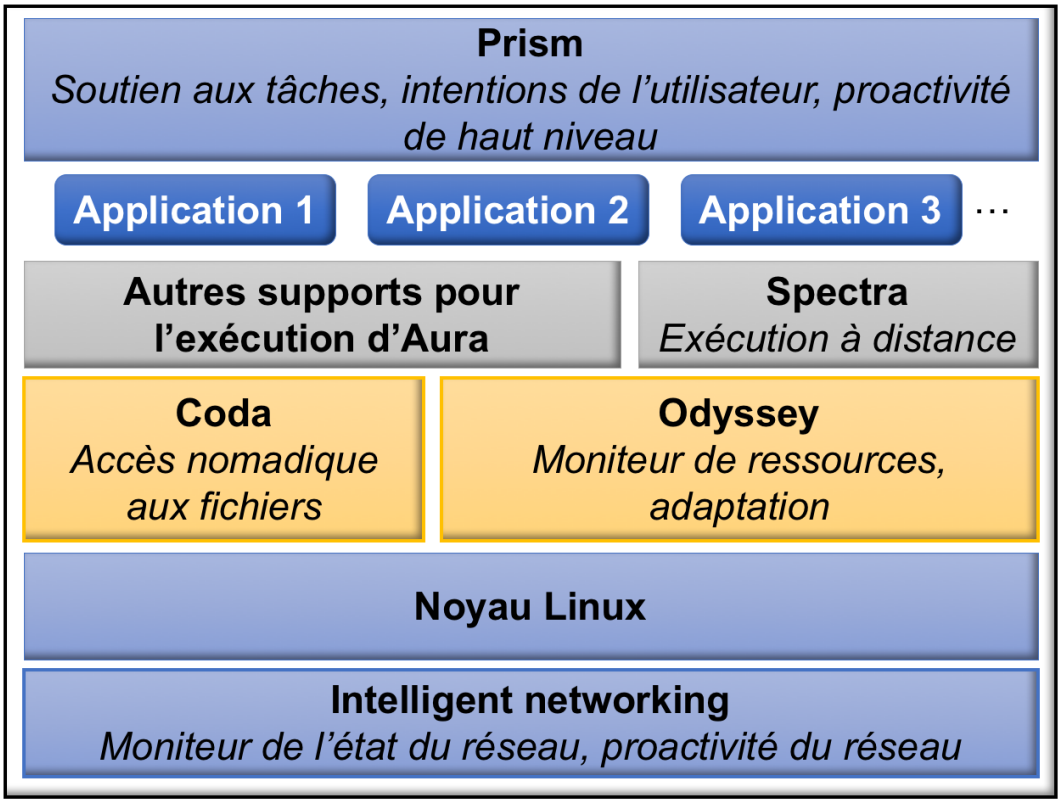


FIGURE 2.2 – L'architecture de l'intergiciel *Aura*

Aura est spécifiquement conçu pour les environnements informatiques omniprésentes impliquant la communication sans fil, les ordinateurs portables, les systèmes embarqués, et les espaces connectés. L'attention humaine est une ressource particulièrement rare dans de tels environnements, parce que l'utilisateur est souvent préoccupé par la marche, la conduite, ou d'autres interactions du monde réel. En outre, l'informatique mobile pose des défis à l'informatique ubiquitaire tels que la connectivité à bande passante intermittente et variable ou encore la durée de vie d'une batterie. Pour atteindre ces objectifs ambitieux, la recherche dans *Aura* couvre tous les niveaux du système : à partir du matériel, par le biais du système d'exploitation, aux applications et aux utilisateurs finaux. Derrière cette

diversité de préoccupations, *Aura* applique deux grands concepts :

- La "*proactivité*", qui est la capacité d'une couche du système à anticiper les demandes d'une couche supérieure. Dans beaucoup de systèmes actuels, chaque couche réagit simplement à la couche supérieure.
- La faculté d'auto-adaptabilité : les couches s'adaptent en observant les demandes faites sur elles et en ajustant leurs performances et l'utilisation de leurs ressources en fonction.

L'architecture d'*Aura* est présentée dans la figure 2.2, y compris les composants et leurs relations logiques. Le texte en italique indique le rôle de chaque composant. *Odyssey* prend en charge la surveillance des ressources et l'adaptation au niveau applicatif, et *Coda* fournit un support pour l'accès aux fichiers nomades, temporaires, et avec une bande passante adaptative. *Spectra* est un mécanisme d'exécution à distance adaptatif qui utilise le contexte pour déterminer la meilleure façon d'exécuter l'appel à distance. *Prism* est une couche du système qui est responsable de la capture et de la gestion des utilisateurs. Il se situe au-dessus de la couche applicative et fournit un soutien de haut niveau pour la proactivité.

L'intergiciel CARISMA [Cap03]

CARISMA (*Context-Aware Reflective mIddleware System for Mobile Applications*) se concentre sur les systèmes mobiles qui sont extrêmement dynamiques. L'adaptation est le principal objectif de CARISMA. Deux catégories d'adaptation sont définies dans cet intergiciel, l'adaptation passive et l'adaptation active. La catégorie passive définit des actions que l'intergiciel entamera lorsque certains évènements spécifiques se produiront (mise en veille après un temps d'inaction par exemple). La catégorie active permet de maintenir des relations avec les services utilisés par l'application, les politiques, et les configurations de contexte. Ce type d'information indique à l'application comment s'adapter à différentes conditions environnementales et à l'utilisateur.

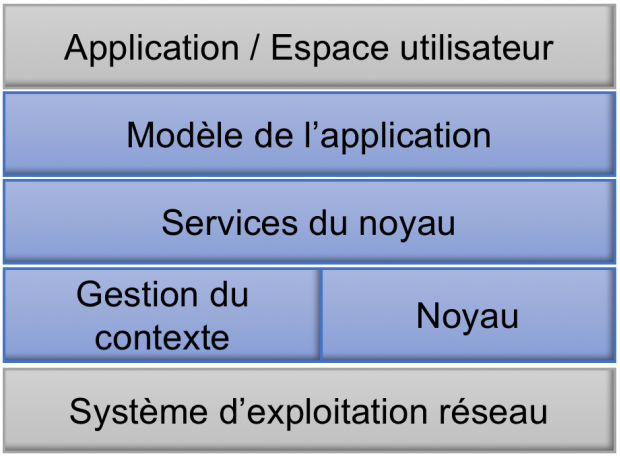


FIGURE 2.3 – L'architecture de l'intergiciel CARISMA

L'architecture CARISMA est composée de quatre éléments principaux, comme le montre la figure 2.3. Le composant de base (*Core component*) fournit des fonctionnalités de

base, telles que le support pour la communication asynchrone, découverte de services, ... Le composant de gestion de contexte (*Context Management component*) est responsable de l'interaction avec des capteurs physiques et le suivi des changements de contexte. Les services (*Core Services*) prennent soin de répondre à des demandes de services avec des niveaux de qualité de service (QoS) définis par l'application. Le modèle d'application (*Application Model*) définit un cadre standard pour créer et exécuter des applications sensibles au contexte.

L'intergiciel Amigo [Sac05]

Amigo est un projet *open source* européen dirigé par Philips. Cet intergiciel a été entièrement développé en Java et en DotNET. Il gère la découverte ainsi que la gestion de nouvelles entités et de nouveaux dispositifs en utilisant les protocoles SLP (Service Location Protocole) et UPnP (Universal Plug and Play). Cet intergiciel interopérable vise à permettre l'intelligence ambiante dans l'environnement d'un réseau domestique en permettant l'intégration transparente en réseau des appareils et des services d'application connexes (l'électronique grand public, la domotique, les ordinateurs et les mobiles) au sein du système d'origine.

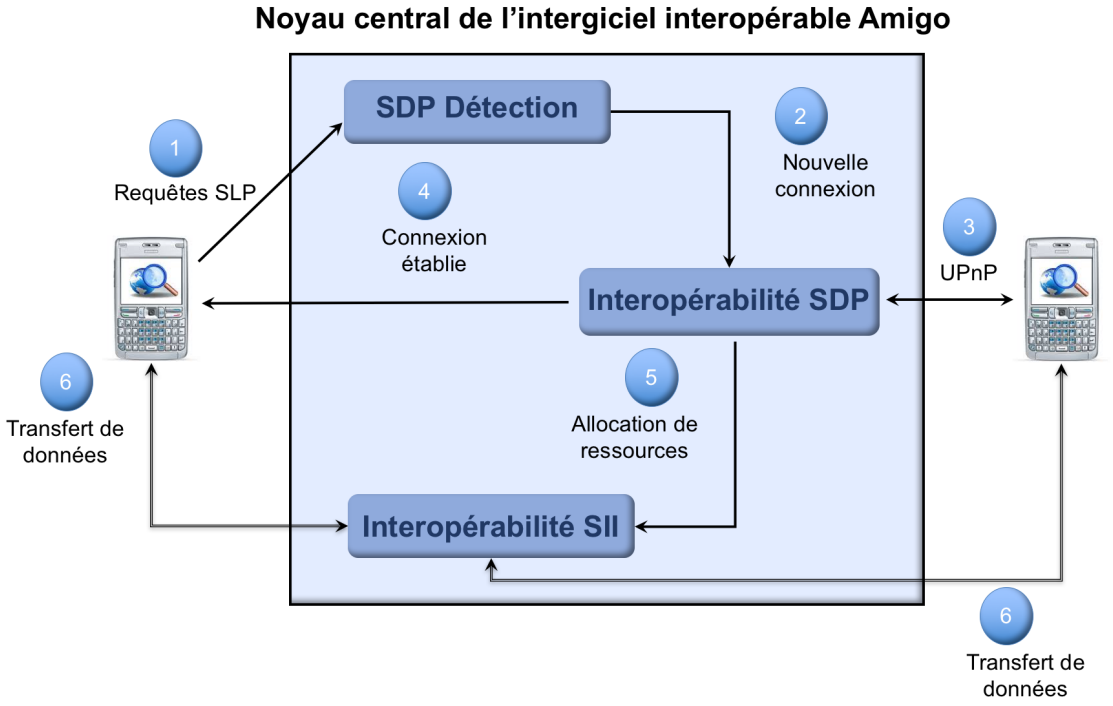


FIGURE 2.4 – L'architecture de l'intergiciel Amigo

L'architecture d'Amigo (voir figure 2.4) est spécialement conçue pour réaliser un système d'accueil en réseau ouvert qui intègre dynamiquement des dispositifs hétérogènes au fur et à mesure de leur apparition. L'architecture est adaptable aux informations relatives à l'utilisateur et aux services et permet une interopérabilité qui prépare des contenus hors-ligne pour les différentes classes de périphériques (lumière, son, gestion d'électricité, gestion d'eau, ...).

Les fonctionnalités principales de l'intergiciel reposent sur :

- La détection avec le protocole *Service Discovery Protocol (SDP)* : ce module est aussi appelé le SDI pour *Service Discovery Protocol Detection and Interoperability*. Il gère la découverte et la détection de services (par exemple, le protocole *Service Location Protocol (SLP)* ou encore le *Simple Session Description Protocol (SSDP)* (module SDP Détection dans le schéma 2.4) et garantit l'interopérabilité entre ces protocoles (module Interopérabilité SDP dans le schéma 2.4)). Il ne faut pas confondre ce protocole avec le protocole *Session Description Protocol (SDP)* qui est utilisé par les services en réseau pour la diffusion et la demande de services,
- L'interopérabilité entre les services avec le protocole *Service Interaction Interoperability (SII)* : ce protocole permet l'interaction entre les services (par exemple, l'interaction entre le service *Remote Method Invocation (RMI)* et le service *Simple Object Access Protocol (SOAP)*) indépendamment des protocoles d'interaction spécifiques utilisés par les services en réseau (module Interopérabilité *SII* dans le schéma 2.4).

L'intergiciel CoBrA [Che05]

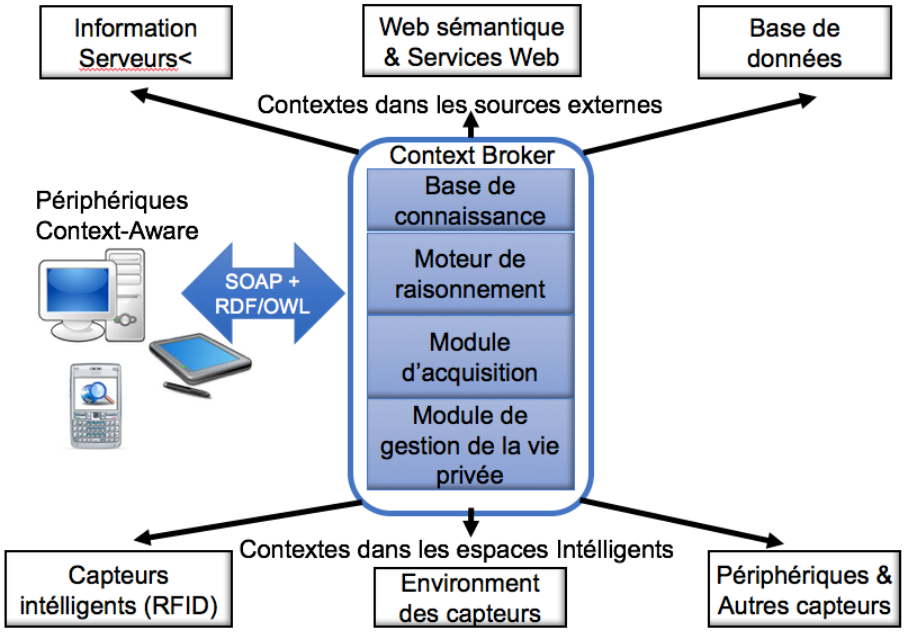


FIGURE 2.5 – L'architecture de l'intergiciel CoBrA

Context Broker Architecture (CoBrA) est une architecture à base d'agents pour supporter les systèmes sensibles au contexte dans les espaces connectés (par exemple, les salles de réunion connectées, les maisons connectées, et les véhicules connectés). Au centre de cette architecture se trouve un agent intelligent, le *Context Broker* qui maintient un modèle de contexte commun et partagé au nom d'une communauté d'agents, de services, et des dispositifs dans l'espace. Le *Contexte Broker* offre des protections de confidentialité pour les utilisateurs dans l'espace en appliquant les règles de stratégie qu'elles définissent. Il permet aux utilisateurs de définir la politique de confidentialité afin de contrôler le partage et l'utilisation de leurs informations sur leurs situations (par

exemple : où sont-ils ? Qui sont-ils ? Que font-ils ?). L'architecture de CoBrA est présentée sur la figure 2.5.

CoBrA est essentiellement axé sur des lieux de rencontres connectés. CoBrA aborde deux questions principales : le soutien des dispositifs informatiques mobiles à ressources limitées et répondre aux préoccupations sur la vie privée de l'utilisateur. Les informations contextuelles sont modélisées en utilisant le langage OWL (*Web Ontology Language*) qui est un standard sémantique web du consortium W3C. Un *Context Broker* comprend les quatre composants fonctionnels suivants : base de connaissances de contexte (fournit un stockage persistant pour des informations de contexte), moteur de raisonnement contextuel (effectue un raisonnement sur des informations de contexte sauvegardées), module d'acquisition de contexte (récupération du contexte à partir des sources de contextes), et le module de gestion de politiques (gère les politiques, par exemple, qui a accès à quelle donnée). Même si l'architecture est centralisée, plusieurs *Context Brokers* peuvent travailler ensemble par le biais d'un coordinateur (*Context Federation*). La connaissance d'un contexte est représentée dans le *Resource Description Framework* (RDF) par un serveur Jena ([Apa15]).

L'intergiciel proposé dans le projet SOCIETIES [Lim14]

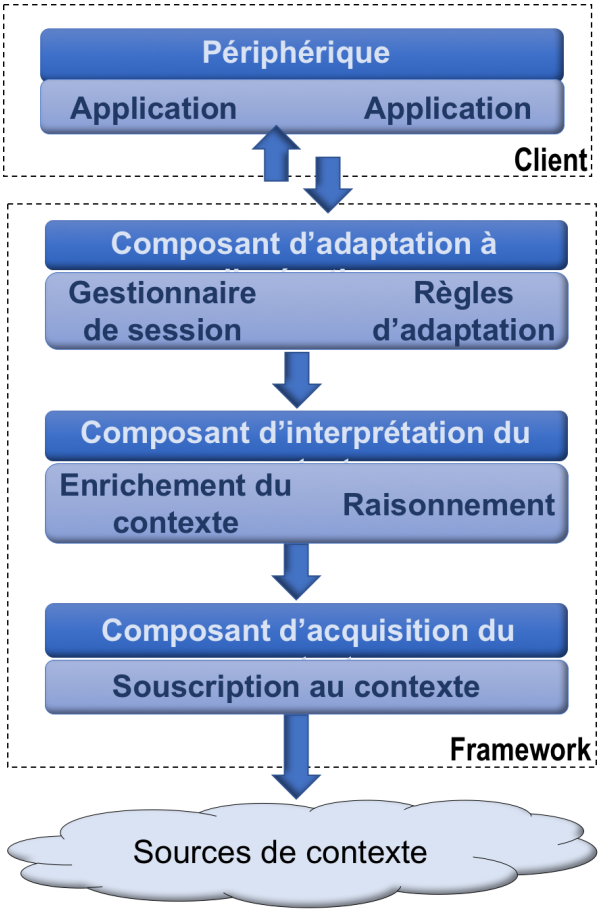


FIGURE 2.6 – L'architecture globale de l'intergiciel proposé dans le projet SOCIETIES

[Lim14] *et al.* ont proposé tout récemment une architecture globale pour la prise en compte du contexte. Ce principe est présent dans la plupart des intergiciels que nous avons étudiés. En effet, la règle primaire qui est la perception de l'information, l'interprétation des informations collectées, et le déclenchement des actions afin de prendre en compte ces informations est présente. La figure 2.6 montre un schéma d'ensemble de l'architecture proposée dans [Lim14].

Le module chargé de faire l'acquisition du contexte est la première couche de l'intergiciel. Ce module est responsable de faire la collecte des informations pertinentes du contexte (positionnement géographique, disponibilité actuelle, ressources actuelles, informations professionnelles, ...) et les prépare pour la couche suivante. Les informations recueillies sont classées en long terme (poste, centres d'intérêts, compétences, ...) ou court terme (positionnement géographique, disponibilité, ...), dépendant des caractéristiques temporelles. L'interprétation du contexte est faite par le composant responsable de la modélisation de la connaissance et des processus. Les données collectées sont représentées sous forme de graphe dans l'intergiciel. Chaque nœud représente une personne, les arcs représentant les relations existant entre les utilisateurs, basées sur des informations court et long terme du contexte. Les relations entre les personnes sont associées avec le degré de similarité et sont décrites comme les poids des arcs.

Le composant d'adaptation à l'exécution est responsable pour la gestion des sessions de collaboration et effectue des actions par le biais de règles d'adaptation. Les règles d'adaptation sont exprimées avec des clauses conditionnelles. Des règles automatiques, telles que *l'invitation d'un utilisateur*, *la création de nouvelle session*, ou encore *la suppression d'un utilisateur non-actif*, existent dans le module d'adaptation.

Parmi tous les intergiciels que nous avons étudiés dans cet état de l'art, aucun ne prend en compte les variations des entités du contexte. En effet, dans un contexte il existe un environnement, et dans cet environnement il existe des entités. Ainsi, dans notre travail, nous avons pu constater que dans les intergiciels étudiés lorsqu'une entité est prise en compte, l'évolution de cette dernière n'est souvent pas surveillée, voire pas du tout. Dans la plupart des cas, une entité est découverte, et est ensuite adaptée pour être prise en compte comme un élément pertinent du contexte. Nous tenterons donc de proposer dans la suite de ce document une nouvelle approche qui est l'adaptation *en continu* d'une entité des éléments de réseau tels que la gigue, les accès hétérogènes à d'autres réseaux, les variations dynamiques de la bande passante, ainsi que les déficiences aléatoires d'un réseau. La raison est que ces intergiciels se trouvent souvent dans un réseau local dans lequel ces problèmes surviennent rarement. Aucun de ces intergiciels ne propose une étude *en continu* sur une des entités du contexte : une entité peut émettre un flux de données *en continu* ou *occasionnellement*. Nos travaux, développés dans la partie contribution de cette thèse, ont pour but principal la conception d'un intergiciel permettant de prendre en compte des éléments pertinents du contexte et de suivre les évolutions des éléments pertinents dans des réseaux plus vastes.

Après la première étape de perception des paramètres de contexte, il s'agit d'évaluer les données recueillies afin de prendre des décisions et d'appliquer des règles d'adaptation. Les méthodes de raisonnement que nous avons étudiées dans les intergiciels existants sont basées sur des systèmes de règles ou de mécanismes de type *"action-réaction"*. Il n'existe aucun intergiciel avec un véritable système d'apprentissage capable d'apprendre sur un élément pertinent de son contexte et de déclencher une adaptation en conséquence. Ainsi, le chapitre qui suit présente quelques mécanismes existants qui peuvent être utilisés dans ces modules décisionnels.

2.2/ MÉCANISMES DE PRISE DE DÉCISION

Dans ces deux premiers chapitres d'état de l'art, nous avons étudié les systèmes ubiquitaires, l'adaptation, et enfin au début du présent chapitre la notion de contexte.

Si l'on veut gérer l'adaptation au contexte d'un système ubiquitaire, il sera nécessaire de déclencher des actions en fonction de paramètres de contexte. Il est donc important d'étudier les mécanismes de prise de décision qui permettront de déclencher d'éventuelles adaptations. Ainsi différentes logiques, lois et raisonnements pourront être utiles.

La logique est une branche fondamentale des mathématiques qui permet d'établir la valeur de vérité de propositions et de construire des raisonnements mathématiques. Une proposition logique (ou assertion) est une affirmation formée d'un assemblage de symboles et de mots, portant sur des objets mathématiques, à laquelle est attribuée la valeur vraie ou la valeur fausse.

Afin d'exprimer des évènements issus de l'observation du contexte ainsi que les conditions permettant d'identifier la situation dans laquelle se trouve un système, différents types de logiques existent. Plus que la seule prise en compte de l'infrastructure pertinente, il s'agit ici d'ajouter des informations opportunistes pour la prise de décision. C'est-à-dire, de considérer les changements apparaissant dans le contexte et y être au maximum réactif. Ainsi plusieurs types de logiques plus ou moins élaborées et complexes peuvent être mises en œuvre.

2.2.1/ LA LOGIQUE DES PROPOSITIONS

La logique "*classique*" permet d'exprimer des énoncés auxquels est attribuée une valeur dite de vérité. Un énoncé est soit vrai, soit faux mais pas les deux et cette valeur de vérité ne change pas au cours du temps. Il s'agit d'une des logiques les plus simples. Une formule de la logique des propositions contient juste des variables et des connecteurs logiques, pas de quantification, pas de fonction, pas de prédicat.

Par exemple : $A \vee B \rightarrow C$

Dans la logique propositionnelle, sont étudiées les relations entre des énoncés, appelées propositions. Ces relations peuvent êtres exprimées par l'intermédiaire de connecteurs logiques qui permettent, par composition, de construire des formules syntaxiquement correctes, composées au moyen de conjonction, disjonction (inclusive), implication, équivalence et négation.

La syntaxe

S'intéresser à la syntaxe de la logique propositionnelle, c'est considérer les formules qui sont "bien écrites". Pour cela, est défini un alphabet, dit autrement un ensemble de symboles, avec :

- un ensemble $V = \{p, \quad q, \quad r, \quad ... \quad \}$ dénombrable de lettres appelées **variables propositionnelles**. Il s'agit des propositions atomiques correspondant à des énoncés non décomposables tels que par exemple :

"10 est divisible par 2",

- les **constantes** vrai et faux,
- un ensemble (fini) de **connecteurs logiques** : $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\Leftrightarrow$
- les parenthèses : (,)

Parmi les propositions composées à l'aide de cet alphabet, il s'agira d'utiliser des expressions logiques bien formées, dit autrement les formules suivant les règles ci-dessous :

- toutes les propositions atomiques, $p, q, r, ...$, sont des expressions bien formées,
- si A est une expression bien formée, alors $\neg A$ est une expression bien formée,
- si A et B sont deux expressions bien formées, alors :
  $(A \wedge B), (A \vee B), (A \rightarrow B)$,
  et $(A \Leftrightarrow B)$ sont des expressions bien formées,
- il n'y a pas d'autres expressions bien formées que les précédentes.

Par exemple, $(((\neg p \Leftrightarrow q) \vee \neg(r \wedge s)) \rightarrow q)$ est une expression bien formée, tandis que par exemple $p\neg qr \rightarrow t(\Leftrightarrow$ ne l'est pas.

La sémantique

S'intéresser à la sémantique de la logique propositionnelle, c'est déterminer la valeur de vérité d'un énoncé. Il s'agit de l'interprétation d'une formule : il s'agit plus concrètement d'affecter une valeur $vrai$ ou $faux$ à chacune des variables propositionnelles qui la compose et d'en déduire la valeur de vérité de la formule selon la sémantique des connecteurs. Pour une formule à n variables, il y a 2^n valeurs de vérité distinctes de ces variables.

2.2.2/ LA LOGIQUE DU PREMIER ORDRE

La logique propositionnelle ne permet d'écrire que des constructions simples du langage, consistant essentiellement en des opérations booléennes sur les propositions. Il est possible, grâce à elles, d'étudier dans un cadre formel la valeur de vérité de formules relativement peu expressives. Toutefois, elle ne permet pas de tenir compte des solutions possibles entre des individus d'un énoncé.

Insuffisante pour représenter des procédés de langage effectivement utilisés en informatique linguistique ou en mathématiques, elle sert néanmoins de base à la construction de systèmes formels plus expressifs. La logique du premier ordre ajoute donc à la logique des propositions des quantificateurs, des relations (qui peuvent être d'arité variable), des fonctions ainsi que des constantes. La logique du premier ordre est la logique des formules usuelles, avec la contrainte que les variables représentent toutes des objets du même type. Elle est par nature plus expressive que la logique des propositions, et permet de représenter ces types de connaissances relatifs à des environnements complexes. Elle est construite à partir de la logique propositionnelle et s'inspire du langage naturel pour définir des objets, des fonctions et des relations.

La structure

Un énoncé en logique du premier ordre, est composé d'un ensemble de symboles plus riche qu'en logique des propositions :

- un ensemble (dénombrable) de constantes $\{a, \quad b, \quad c, \quad ...\}$;
- un ensemble (dénombrable) de variables $\{x, \quad y, \quad z, \quad ...\}$;
- un ensemble (dénombrable) de fonctions $\{f, \quad g, \quad h, \quad ...\}$;
- un ensemble (dénombrable) de prédicats, ou relations $P, \quad Q, \quad ...$;
- des connecteurs logiques, $\neg, \quad \wedge, \quad \vee, \quad \rightarrow, \quad ...$, ainsi que les parenthèses '(' et ')' ;
- les quantificateurs universel $\forall$ et existentiel $\exists$.

Les termes

Un terme est une expression logique qui renvoie à un objet. Les constantes, ainsi que les variables, sont des termes. Un terme composé est construit à l'aide d'une fonction, par exemple $f(x) \quad ou \quad g(y)$.

Les expressions

Une expression en logique des prédicats se construit comme une expression en logique des propositions : un prédicat joue un rôle analogue à une proposition dans le sens où il est vrai ou faux selon les objets qu'il met en relation. Par exemple :

- $P(x_1, \quad ..., \quad x_n)$ est une formule atomique,
- $t_1 = t_2$ est une formule atomique,
- si F est une formule, alors ¬F est une formule,
- si F et G sont des formules, alors $(F \wedge G)$, $(F \vee G)$, $(F \rightarrow G)$, ... sont des formules,
- si F est une formule et x une variable, alors $\forall x.F$ et $\exists x.F$ sont des formules.

Les quantificateurs

Le quantificateur universel $\forall$ exprime le fait que tous les éléments d'un ensemble d'objets sur lequel s'exprime un prédicat vérifient ce prédicat, c'est-à-dire $\forall x.P(x)$ est vrai revient à considérer que $P(a_1) \wedge ... \wedge P(a_n)$ est vrai, si $a_1, ..., a_n$ est le domaine de x.

De la même manière, le quantificateur existentiel $\exists$ exprime le fait qu'au moins un des éléments d'un ensemble d'objets sur lequel s'exprime un prédicat vérifie ce prédicat,

c'est-à-dire $\exists x.P(x)$ est vrai revient à considérer que $P(a_1) \vee ... \vee P(a_n)$ est vrai, si $a_1, ..., a_n$ est le domaine de x. La sémantique de la logique des prédicats est sensible à l'ordre des quantificateurs n'étant pas anodin.

Il existe des liens entre $\forall$ et $\exists$. Les lois de *de Morgan* sont définies ainsi pour les quantificateurs :

$$\neg\forall x.F \equiv \exists x.\neg F$$
$$\neg\exists x.F \equiv \forall x.\neg F$$
$$\forall x.F \equiv \neg\exists x.\neg F$$
$$\exists x.F \equiv \neg\forall x.\neg F$$

2.2.3/ L'INFÉRENCE BAYÉSIENNE

L'inférence bayésienne est la démarche logique permettant de calculer ou réviser la probabilité d'une hypothèse. Cette démarche est régie par l'utilisation des règles strictes de combinaison des probabilités, desquelles dérive le théorème de Bayes. Dans la perspective bayésienne, une probabilité n'est pas interprétée comme le passage à la limite d'une fréquence, mais plutôt comme la traduction numérique d'un état de connaissance (le degré de confiance accordé à une hypothèse).

L'inférence bayésienne est fondée sur la manipulation d'énoncés probabilistes. Ces énoncés doivent être clairs et concis afin d'éviter toutes confusions. L'inférence bayésienne est particulièrement utile dans les problèmes d'induction. Les méthodes bayésiennes se distinguent des méthodes dites standards par l'application systématique de règles formelles de transformation des probabilités.

Il existe seulement deux règles pour combiner les probabilités, et à partir desquelles est bâtie toute la théorie de l'analyse bayésienne. Ces règles sont les règles d'addition et de multiplication. La règle d'addition est la suivante :

Équation 1

Règle d'addition

$$p(A \bigcup B \mid C) = p(A \mid C) + p(B \mid C) - p(A \bigcap B \mid C)$$

La règle de multiplication est la suivante :

Équation 2

Règle de multiplication

$$p(A \bigcap B) = p(A \mid B)p(B) = p(B \mid A)p(A)$$

Le théorème de Bayes peut être dérivé simplement en mettant à profit la symétrie de la règle de multiplication :

Équation 3

Théorème de Bayes

$$p(A \mid B) = \frac{p(B|A)p(A)}{p(B)}$$

En théorie des probabilités, le théorème de Bayes énonce des probabilités conditionnelles : étant donné deux évènements A et B, le théorème de Bayes permet de déterminer la probabilité de A sachant B, si l'on connaît les probabilités de A, de B, et de B sachant A. Le théorème de Bayes permet d'inverser les probabilités. C'est-à-dire que si l'on connaît les conséquences d'une cause, l'observation des effets permet de remonter aux causes.

La notion d'évidence

La notation d'évidence en inférence bayésienne est souvent attribuée à Irving John Good [Goo74]. I.-J. Good était un mathématicien britannique qui a travaillé en tant que cryptographe à *Bletchley Park* avec Alan Turing sur la machine cryptographique allemande, *Enigma*. Cependant, Good en a attribué la paternité à Alan Turing.

Dans la pratique, quand une probabilité est très proche de 0 ou de 1, il faut observer des éléments considérés eux-mêmes comme très improbables pour la voir se modifier. L'évidence est définie ainsi :

Équation 4

Expression de l'évidence

$$Ev(p) = \log \frac{p}{(1-p)} = \log p - \log(1-p)$$

Pour mieux fixer les choses, l'évidence est souvent exprimée en décibels (dB), avec l'équivalence suivante :

Équation 5

Expression de l'évidence exprimée en décibels

$$Ev(p) = 10 \log_{10} \frac{p}{(1-p)}$$

Si le logarithme en base 2 est utilisé, l'évidence est exprimée en bits :

Équation 6

Expression de l'évidence utilisant le logarithme en base 2.

$$Ev(p) = \log_2 \frac{p}{(1-p)}$$

2.2.4/ L'INFÉRENCE FRÉQUENTISTE

Les fréquentistes définissent la probabilité en tant que la fréquence sur du long terme d'une certaine mesure ou observation. Le fréquentiste dit qu'il n'y a qu'une seule vérité et les mesures échantillonnent des instances bruyantes de cette vérité. Plus les données sont collectées, et meilleure sera l'identification de cette vérité.

Dans l'interprétation fréquentiste, les probabilités sont discutées uniquement à partir d'expériences aléatoires bien définies (ou d'échantillons aléatoires). L'ensemble de tous les résultats possibles d'une expérience aléatoire est appelé l'espace d'échantillon (en anglais, *sample space*) de l'expérience. Un évènement est défini comme étant un sous-ensemble de l'espace d'échantillon à examiner. Pour tout évènement donné, une seule des deux possibilités peut tenir : elle se produit ou non. La fréquence relative de l'occurrence d'un évènement, observée dans un certain nombre de répétitions de l'expérience, est une mesure de la probabilité de cet évènement. Ceci est la base de la conception des probabilités dans l'interprétation fréquentiste.

Ainsi, si n_t est le nombre total d'essais et n_x est le nombre d'essais où l'évènement a eu lieu, la probabilité $P(x)$ de l'évènement qui se produit sera évaluée par la fréquence relative comme suit :

Équation 7

La fréquence relative

$$P(x) \approx \frac{n_x}{n_t}$$

De toute évidence, comme le nombre d'essais augmente, la fréquence tend vers une meilleure approximation par rapport à une *"fréquence réelle"*. Une revendication de l'approche fréquentiste est que dans le long terme, comme le nombre d'essais tend vers l'infini, la fréquence relative converge finalement à la vraie probabilité :

Équation 8

Convergence de la fréquence relative

$$P(x) = \lim_{x \to \infty} \frac{n_x}{n_t}$$

L'exemple typique de ce type de logique est le jeté répété d'une pièce de monnaie pour déterminer si la fréquence à long terme est pile ou face : cette fréquence à long terme converge à la vérité. Une pièce de monnaie est jetée dix fois. Sept fois sur dix le côté *pile* est obtenu. En utilisant l'inférence fréquentiste, la probabilité d'avoir un côté *face* sera :

$$P(head) = \frac{7}{10} = 0.7$$

En résumé, l'inférence fréquentiste permet de calculer une probabilité à condition que les données soient des échantillons reproductibles aléatoires et que les paramètres sous-jacents restent constants au cours de ce processus reproductible. Il ne peut être utilisé pour donner une probabilité sur une hypothèse (pas de notion d'antériorité ou de postériorité). Elle ne nécessite pas d'*a priori* [Moo12].

EXEMPLE D'APPLICATION DES INFÉRENCES BAYESIENNE ET FRÉQUENTISTE ET L'INFÉRENCE

L'exemple suivant montre un cas concret d'utilisation des deux types d'inférence. Nous prendrons comme exemple leur utilisation sur la détection d'une mauvaise connexion Internet. Voici les données d'hypothèses par exemple issues des études statistiques précédemment menées (cf table 2.1) :

- 1% des utilisateurs ont une mauvaise connexion Internet. Ce qui signifie que 99% des utilisateurs ont une bonne connexion Internet.
- 80% des mécanismes d'adaptation détectent efficacement une mauvaise connexion : 20% ne le détecte pas dans ce cas.
- Dans la détection de mauvaise connexion est introduire une marge d'erreur : 9.6% des mécanismes d'adaptation détectent une mauvaise connexion alors qu'il n'y en a pas.
  Donc, 90.4% des mécanismes d'adaptation ne feront pas d'erreur lors de la détection d'une mauvaise connexion.

Nous pouvons résumé ces données dans le tableau 2.1 :

	Mauvaise connexion (MC) 1%	**Bonne connexion (BC) 99%**
Détection d'une mauvaise connexion	80%	9,6%
Non détection d'une mauvaise connexion	20%	90,4%

TABLE 2.1 – Tableau récapitulatif des données de l'hypothèse

Il s'agit de déterminer la probabilité d'avoir effectivement une mauvaise connexion selon les données d'hypothèse.

Le théorème de Bayes indique que la probabilité d'un évènement est calculé par rapport à l'évènement souhaité divisé par la probabilité de toutes les possibilités. Dans l'exemple, l'évènement souhaité est la probabilité d'avoir effectivement une mauvaise connexion, ce qui donne une probabilité de 1% * 80% = 0,008 (ces calculs sont récapitulés dans le tableau 2.2).

	Mauvaise connexion (MC) 1%	Bonne connexion (BC) 99%
Détection d'une mauvaise connexion	1% * 80% = 0.008	99% * 9,6% = 0.09504
Non détection d'une mauvaise connexion	1% * 20% = 0.002	99% * 90,4% = 0.89496

TABLE 2.2 – Tableau récapitulatif des données calculées à l'aide des inférences bayésiennes

Or, il se peut qu'il y ait une bonne connexion mais un mécanisme de détection de la connexion Internet qui détecte une fausse mauvaise connexion. Cela donne une probabilité de 99% * 9,6% = 0,09504 pour ce type d'évènement.

La probabibilité totale de toutes les possibilités d'avoir une mauvaise connexion est donc de 0,008 + 0,09504 = 0,10304. L'évènement souhaité (la probabilité de la détection d'une mauvaise connexion, est ainsi :

$$\text{Probabilité effective} = \frac{0{,}008}{0{,}10304} \approx 7,8\%$$

Ce calcul peut également être schématisé à l'aide d'un arbre. La figure 2.7 donne un exemple de l'arbre de calcul de notre exemple.

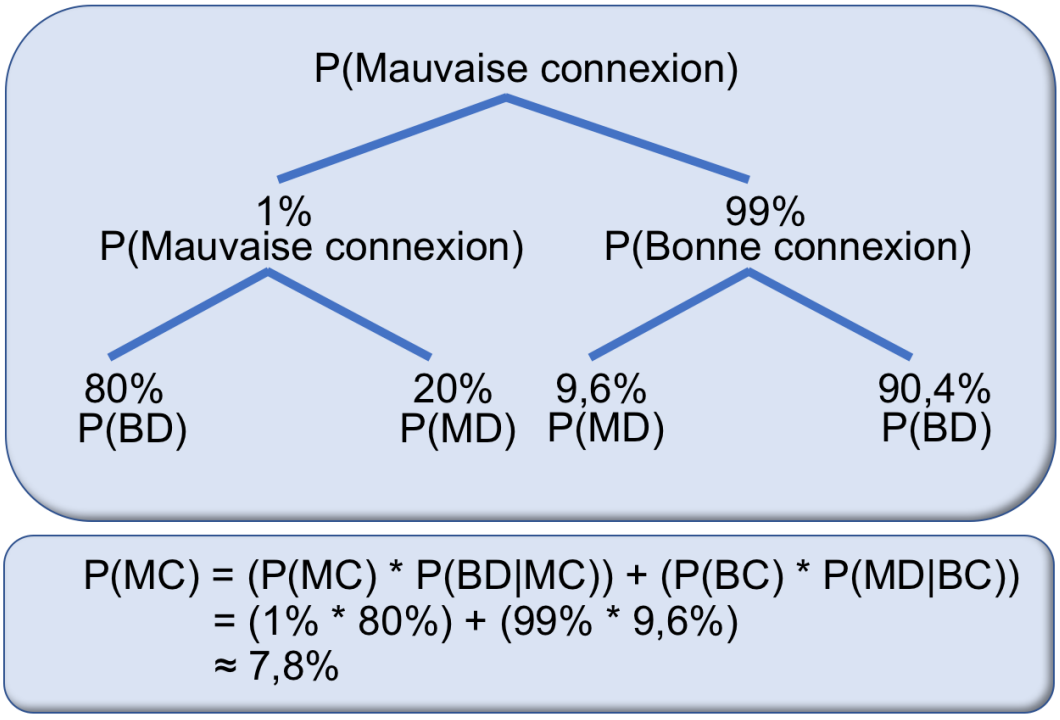


FIGURE 2.7 – Arbre représentant le calcul de la probabilité avec le théorème de Bayes

L'inférence fréquentiste, quant à elle, prend comme résultat 1%. Ce qui signifie qu'il y a 1% de chance d'avoir une mauvaise connexion.

2.2.5/ LA LOI BINOMIALE

En mathématiques, une loi binômiale de paramètres n et p correspond au modèle dans lequel on renouvelle n fois de manière indépendante une épreuve de Bernoulli de paramètre p (expérience aléatoire à deux issues possibles, généralement dénommées respectivement "succès" et "échec", la probabilité d'un succès étant p, celle d'un échec étant $q = 1 - p$). On compte alors le nombre de succès obtenus à l'issue des n épreuves et on appelle X la variable aléatoire correspondant à ce nombre de succès.

L'univers $X(\Omega)$ désigne l'ensemble des entiers naturels de 0 à n. La variable aléatoire suit une loi de probabilité définie par :

Équation 9

Loi binômiale

$$p(k) = P(X = k) = \binom{n}{k} p^k q^{n-k}$$

Où :

$$\binom{n}{k} = C_n^k = \frac{n!}{k!(n-k)!}$$

Cette loi de probabilité s'appelle la loi binômiale de paramètre $(n; p)$ et se note $B(n; p)$.

2.2.6/ LE RAISONNEMENT NON-MONOTONE

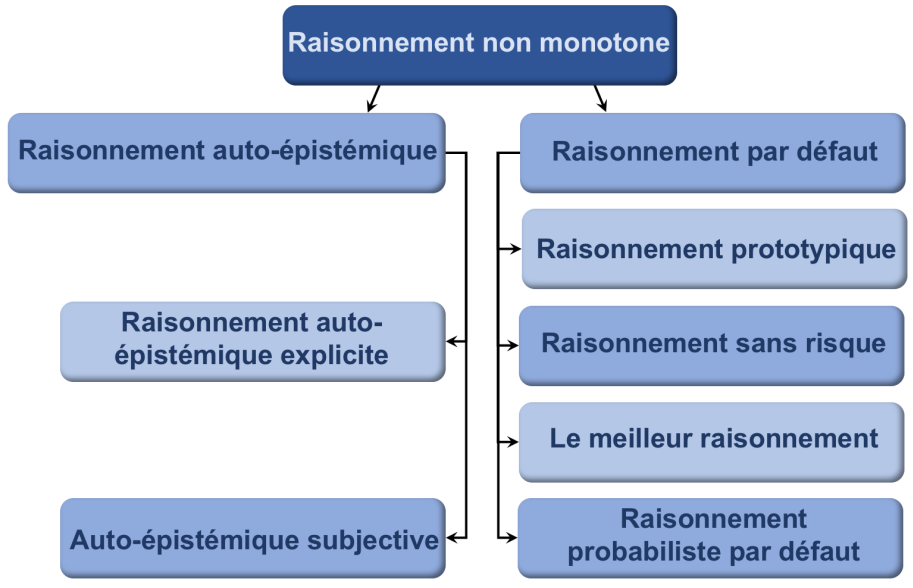


FIGURE 2.8 – Classification de Moore

Le raisonnement non-monotone est un processus d'inférence basé sur des informations partielles ou de véracité incertaine et dont les conclusions peuvent être rétractées avec

l'ajout de nouvelles informations : les conclusions ne se croisent pas nécessairement de façon monotone avec l'ajout de nouvelles connaissances.

En 1965, Robert Moore [Moo85] a produit une typologie des raisonnements non-monotones basée sur l'observation :

- information incomplète,
- représentation incomplète d'une information complète.

Puis la classification de Moore scinde ainsi l'ensemble des raisonnements non-monotones en deux grandes classes :

- le raisonnement par défaut,
- le raisonnement auto-épistémique.

La première classe se base sur des faits vrais en général pour tirer des conclusions. Par exemple : "en général les oiseaux volent. Titi est un oiseau, donc il vole."

La deuxième classe permet schématiquement d'obtenir une conclusion C de la manière suivante : "Si C était faux, nous le saurions (d'une manière ou d'une autre : nous l'aurions démontré, vérifié, appris, ...). Ce n'est pas le cas, donc C est une conclusion valide".

La classification de Moore est en fait beaucoup plus précise. Les figures 2.8 et 2.9 reproduisent et expliquent cette classification.

Type de raisonnement	Explication d'une conclusion *C*
Raisonnement auto-épistémique subjectif	Selon mon opinion subjective, j'aurais su si *C* était fausse. Parce qu'il me manque cette information, *C* doit donc être vraie.
Raisonnement auto-épistémique explicite	Selon certaines conventions explicites, j'aurais su si *C* était fausse. Parce qu'il me manque cette information, *C* doit donc être vraie.
Raisonnement prototypique	*C* décrit une situation typique. Ainsi, il y a de bonnes chances que *C* tienne.
Raisonnement sans risque	Si j'accepte non *C*, et que cela s'avère être fausse, les conséquences seraient fatales. Donc, si je dois choisir entre *C* and non *C*, je dois accepter *C*.
Le meilleur raisonnement	Compte tenu des éléments de preuve disponibles, *C* est la meilleure estimation à faire.
Raisonnement probabiliste par défaut	En supposant que les valeurs probabilistes sont suffisamment élevées, il est raisonnable de déduire *C*.

FIGURE 2.9 – Explications de la classification de Moore

2.2.7/ LE RAISONNEMENT FLOU

La logique floue est une extension de la logique booléenne qui permet la modélisation des imperfections des données et se rapproche dans une certaine mesure de la flexibilité du raisonnement humain. Elle a été créée par *Lotfi Zadeh* en 1965 en se basant sur sa théorie mathématique des ensembles flous [Zad65], qui est une généralisation de la théorie des ensembles classiques. En introduisant la notion de degré dans la vérification d'une condition, permettant ainsi à une condition d'être dans un autre état que vrai ou faux, la logique floue confère une flexibilité très appréciable aux raisonnements qui l'utilisent, ce qui rend possible la prise en compte des imprécisions et des incertitudes. Un

des intérêts de la logique floue pour formaliser le raisonnement humain est que les règles sont énoncées en langage naturel.

Un ensemble flou est un ensemble dont la fonction d'appartenance prend des valeurs dans l'intervalle $[0, 1]$ et non plus simplement 0 ou 1. Les fonctions de transfert permettent de définir l'appartenance à un groupe, elles sont définies en fonction du problème. Un système de règles floues permet de décrire sous forme de règles linguistiques une fonction de transfert. Et par conséquent l'appartenance à un groupe selon un certain degré de validité. En logique floue, on peut décrire une eau encore un peu froide et qui commence à être tiède.

Le flou est lié à la forme de la connaissance : son imprécision n'est donc pas de nature probabiliste. Par exemple, dire "l'âge de cette personne est autour de 30 ans" ne présume en rien de la probabilité de l'âge effectif de la personne. Il est possible de mieux voir la distinction entre imprécision et probabilité en pondérant cette assertion : "je suis sûr que l'âge de cette personne est autour de 30 ans" dans laquelle à la fois une imprécision (sur la valeur de l'âge) et une certitude (sur le fait que cet âge soit autour de 30 ans) sont présentes. Une autre solution est : "l'âge de cette personne est autour de 30 ans, avec une probabilité de 0.2" dans laquelle une connaissance floue ("autour de 30 ans") est relativisée par une probabilité de véracité.

La logique floue s'attache donc à une certaine forme de connaissance (avec imprécision) et propose un formalisme rigoureux permettant d'inférer de nouvelles connaissances. En cela, elle est complémentaire de la théorie des probabilités.

2.2.8/ UTILISATION DE CES LOGIQUES DANS LA PRISE DE DÉCISION D'ADAPTATION

Ces différents mécanismes et lois de probabilité permettent l'étude appliquée et continue sur des phénomènes tels que des flux de données continus car nous voulons évaluer le caractère probable d'un évènement, c'est à dire une valeur qui nous permettra de représenter le degré de certitude de l'évènement pour pouvoir adapter notre application. Il est toujours possible d'associer à une variable aléatoire une probabilité et de définir ainsi une loi de probabilité. Lorsque le nombre d'épreuves augmente indéfiniment, les fréquences observées pour le phénomène étudié tendent vers les probabilités et les distributions observées vers les distributions de probabilité ou loi de probabilité. Identifier la loi de probabilité suivie par une variable aléatoire donnée est essentiel car cela conditionne le choix des méthodes employées pour répondre aux problématiques d'adaptation par exemple.

Le fait d'avoir des flux de données continus dans l'architecture, que nous proposerons dans notre contribution, nous conduit donc à l'utilisation des logiques mathématiques appliquées à l'informatique. Les logiques des propositions et du premier ordre nous permettent de formaliser nos propositions en langage mathématique. La loi binomiale, l'inférence bayésienne, et l'inférence fréquentiste nous permettent d'évaluer le caractère probable de la tendance de nos flux de données et ainsi de représenter le degré de certitude quant aux différentes actions à mener sur ces derniers. L'inférence fréquentiste nous permet d'avoir un taux de réussite sur nos échantillonnages et est utilisée comme repère quant au déclenchement d'un calcul de probabilité, seul l'échantillon actuel est pris en compte. La loi binomiale modélise le nombre de succès obtenus lors de la répétition indépendante de plusieurs expériences aléatoires identiques. Pour chaque expérience

appelée épreuve de Bernoulli, l'utilisation d'une variable aléatoire prend la valeur 1 lors d'un succès et la valeur 0 sinon. Dans notre architecture, lorsque l'inférence fréquentiste donne un taux de succès inférieur à 0.5, cette loi est utilisée afin de réviser l'hypothèse de cette dernière. L'inférence Bayésienne nous permet de calculer ou de réviser la probabilité de l'hypothèse. Plus généralement, elle se base sur l'*"a priori"*, le présent pour calculer le futur. Elle sera utile lorsque la loi binomiale donnera un taux de résultat inférieur à 0.5. Le raisonnement non-monotone permet de situer le type de raisonnement que nous émettons sur l'étude des flux continus présents dans l'architecture. Enfin, le raisonnement flou nous permet d'introduire des degrés d'incertitude dans l'ensemble $[0, 1]$ et non simplement 0 ou 1. Ce type de raisonnement est complémentaire de la théorie des probabilités.

Pour résumer

SYNTHÈSE

Le contexte est l'environnement temporaire dans lequel se trouve un système d'information à un instant t. L'environnement est composé d'entités. Les entités sont tout ce qui entoure le système d'information (infrastructure matérielles/logicielles, utilisateurs, évènements, ...). L'environnement évolue dynamiquement au cours du temps. Un système d'information crée son propre contexte, qu'il entretient et exploite tout cours de son cycle de vie (notion temporelle).
L'informatique ubiquitaire impose de considérer de multiples utilisateurs ou encore de multiples dispositifs. Les entités peuvent être hétérogènes, qu'il s'agisse de plateformes ou de dispositifs. Les utilisateurs peuvent être mobiles et il est nécessaire d'avoir un système extensible et capable de prendre en compte les fortes variations qui peuvent intervenir dans leurs environnements. La dynamicité de l'environnement doit être prise en compte afin de réagir le plus rapidement possible pour une bonne utilisabilité. Les phases de perception, et d'évaluation permettent de décider quand appliquer une adaptation. L'adaptation peut être décidée par un utilisateur ou alors lorsqu'un certain nombre de conditions (qui peuvent être autres que temporelles) sont remplies. Ainsi, de nombreux intergiciels (Amigo, Context Toolkit, Aura, CARISMA, CoBrA, ...) ont été proposés dans la littérature pour la prise en compte du contexte. Ces intergiciels doivent pouvoir assurer un niveau d'adaptation par la prise en considération d'un grand nombre d'éléments qui peuvent être sensibles au contexte (utilisateur, terminal, environnement, ...).
Afin d'exprimer des évènements issus de l'observation du contexte ainsi que les conditions nous permettant d'identifier la situation dans laquelle se trouve un système, différents raisonnements logiques existent. Nous les avons rappelés dans la dernière section de ce chapitre : la logique des propositions, la logique du premier ordre, l'inférence bayésienne, l'inférence fréquentiste, la loi binomiale, le raisonnement non-monotone, le raisonnement flou, ... Dans le cadre de nos travaux, le côté décisionnel sera primordial.
Comme l'a montré l'état de l'art, le *"quand"* est une question primordiale dans la décision d'adaptation. Il s'agit de savoir à quel moment durant l'exécution d'une application, une nouvelle stratégie doit être mise en place. Pour la prise en compte d'un élément du contexte les raisonnements logiques cités dans la dernière partie de cet état de l'art peuvent être utilisés. La logique des propositions et la logique du premier ordre sont utiles afin de formaliser les observables issus du contexte en langage mathématique. L'inférence bayésienne permet de réviser ou de calculer la probabilité d'une hypothèse et est utile lorsque la loi binomiale n'est pas suffisante pour vérifier la véracité de l'hypothèse sur des évènements stochastiques. L'épreuve de Bernoulli (qui se solde uniquement par succès ou échec), donne lieu au schéma de Bernoulli (répétition n fois des épreuves de manière indépendante), et la loi binomiale qui correspond au nombre de succès à l'issue du schéma de Bernoulli. Le raisonnement non-monotone permet de situer la classe de raisonnement (raisonnement probabiliste, meilleur raisonnement, raisonnement par défaut, ...) et enfin, le raisonnement flou permet l'introduction de l'incertitude (par exemple, l'eau dans ma tasse est presque chaude) et est complémentaire de la théorie des probabilités.

Cette première partie était consacrée à nos états de l'art. Les notions d'adaptation et d'adaptabilité ont été définies, et ce dernier chapitre a présenté une étude de l'état de l'art autour de la notion de contexte. Les différents raisonnements logiques ont ensuite été rappelés : ils sont utililisés en particulier pour la prise de décision.

La deuxième partie de ce document est consacrée à notre contribution. Et plus particulièrement, le chapitre suivant permet une discussion sur les travaux issus de la littérature et la justification de notre contribution. Ainsi seront présentés la nouvelle architecture proposées pour répondre specifiquement aux problématiques de la télémédecine, les informations du contexte qui devront nécessairement être prises en compte par cet intergiciel, les traitements de ces informations (pour une prise de décision), et enfin la finalité d'une adaptation dans le système.

II

CONTRIBUTION

Après cette première partie consacrée à l'état de l'art, la deuxième partie de ce document est consacrée à la présentation de nos contributions.

Une des problématiques principales dans le cadre des logiciels de télémédecine est la variation de la bande passante dans les réseaux hétérogènes. En effet, l'étude de la bande passante est très importante afin de permettre une adaptation proactive. Par exemple, l'étude des variations dynamiques de la bande passante permet de prendre la décision du moment propice de l'envoi à un serveur d'une série volumineuse d'images médicales qui ne sature pas la bande passante totale disponible. Il est également possible d'adapter un flux en continu à la bande passante disponible dans un réseau. D'autres paramètres entreront ligne de compte comme le type de terminal, le profil utilisateur,... afin de trouver une stratégie d'adaptation la plus précise possible ou encore l'application des préférences utilisateur au moment opportun.

Les chapitres qui constituent cette deuxième partie sont structurés de la manière suivante :

- Le chapitre 3 présente l'architecture globale de notre intergiciel, VAGABOND (en anglais *Video Adaptation framework, crossing security GAteways, Based ON transcoDing*) qui permet d'adapter la vidéo en utilisant des techniques de transcodage et en fonction de la bande passante disponible, du type de terminal, du profil utilisateur, ..., et est capable de passer à travers des barrières de sécurité telles que des pare-feus et des proxies web, qui sont très couramment rencontrés dans les systèmes d'information de santé.

- Le chapitre 4 présente en détails les composants principaux de l'intergiciel : en particulier, les différents types de logiques présentés dans la section 2.2. sont utilisés pour la prise de décision.

- Enfin, le chapitre 5 est consacré aux implémentations, évaluations et discussions qui ont été menées sur notre intergiciel.

3

NOTRE NOUVEL INTERGICIEL VAGABOND

L'adaptabilité s'appuie sur la capacité des systèmes à pouvoir s'adapter aux besoins des applications qu'ils doivent héberger. Selon les types de service à fournir il est possible de calculer par anticipation les besoins et donc d'ajuster les propriétés des systèmes. Mais dans le contexte des environnements collaboratifs dans lesquels tous les acteurs de l'environnement peuvent être un élément du système, il est très difficile, voire impossible de prévoir le comportement général, et de définir une politique d'adaptation globale. De plus, ces environnements collaboratifs sont, de plus en plus, composés de terminaux hétérogènes et mobiles pouvant changer radicalement de topologie, de capacités et de mode d'accès, ... en très peu de temps (liés à la notion de systèmes ubiquitaires). Lorsque s'ajoute à cela la dimension temporelle des données qui doivent être synchronisées, il est nécessaire de fournir des services d'adaptation des données et de coordination des échanges afin de garder la cohérence de celles-ci.

Ce chapitre présente l'architecture de l'intergiciel que nous proposons. Comme nous avons pu le mettre en évidence dans l'état de l'art, il est très difficile, voire impossible de prendre en compte de manière exhaustive l'ensemble des éléments du contexte dans lequel se trouve un système informatique. Dans ces travaux, nous nous sommes focalisés principalement sur les données contraintes au temps-réel. Ainsi, nos mécanismes d'adaptation sont étudiés pour ces types d'échanges. Avant de pouvoir appliquer ces mécanismes, nous avons défini un nouvel intergiciel qui a été étudié afin de pouvoir être déployé spécialement dans le milieu hospitalier dans lequel les systèmes de sécurité sont extrêmement exigeants : les architectures réseaux y sont telles que seul le protocole *Transmission Control Protocol (TCP)* pour le transport des paquets IP est habilité à transiter et ce uniquement sur les ports 80 et 443. Le port 80 est souvent assimilé au protocole *HyperText Transfer Protocol (HTTP)* et le port 443 au protocole *HyperText Transfer Protocol Secure (HTTPS)*. Mais ces ports peuvent également être utilisés pour d'autres usages à condition que TCP soit le protocole de transport.

Le protocole *User Datagram Protocol (UDP)* étant prohibé dans ces réseaux pour des questions de sécurité. En effet, le protocole UDP est soumis à divers types d'attaques dont la plus connue est l'attaque par inondation de paquets UDP qui est une attaque par déni de services (DoS). L'idée est d'envoyer un grand nombre de paquets UDP à un hôte distante sur des ports aléatoires. Le résultat est que l'hôte vérifiera l'application qui est en écoute sur un port particulier, verra qu'il n'y a aucune application en écoute sur le port et répondra que la destination est injoignable avec le protocole *Internet Control Mes-*

sage Protocol (ICMP). Ainsi, avec un grand nombre de paquets UDP, le système attaqué sera forcé d'envoyer un grand nombre de paquets ICMP, le rendant ainsi injoignable par d'autres clients.

Face à de telles exigences, il faut développer des applications en mode connecté qui puissent d'une part intégrer des mécanismes d'adaptation et d'autre part qui puissent prendre en compte les délais que peut engendrer le protocole TCP lorsque des données contraintes au temps-réel sont manipulées comme c'est le cas avec les applications de vidéoconférence. Il est également nécessaire de prendre en compte toutes les fluctuations de débit qui peuvent exister sur un réseau, surtout lorsqu'il s'agit d'un réseau sans fil ou d'un réseau de données cellulaire. L'adaptation devient alors un principe incontournable dans des applications traitant des données contraintes au temps-réel.

3.1/ L'ARCHITECTURE GLOBALE DE VAGABOND

3.1.1/ RAPPEL DU CONTEXTE COVALIA

La plateforme Covotem™ permet aux professionnels de santé d'entrer en relation autour de données médicales afin d'échanger des idées et réaliser un diagnostic rapide et sûr. Pour ce faire, Covotem™ garantit un transfert sécurisé des données médicales, et est accompagné du matériel audio nécessaire à l'organisation de vidéoconférences de qualité, ainsi que de caméras haute définition pilotables à distance. Covotem™ peut de plus être utilisé pour répondre à certains actes médicaux en particulier :

- Prise en charge des Accidents Vasculaires Cérébraux (AVC [1]). L'application permet à un neurologue de diagnostiquer à distance afin de déterminer rapidement le traitement nécessaire à pratiquer au sein de l'hôpital distant.
- Suivi des plaies en dermatologie : à partir d'une tablette ou d'un smartphone, Covotem™ permet la prise de photos de plaies au chevet du patient et l'envoi en temps réel à un dermatologue.
- Réunions de concertation pluridisciplinaires (RCP) : partage d'imagerie médicale, d'examens du patient, de rapports de consultation ou tout autre document à distance accompagné d'une communication audio et vidéo (en vidéoconférence).

La plateforme Covotem™ est aujourd'hui largement déployée en France y compris dans les DOM-TOM (liste non-exhaustive) : Lorraine, Martinique, Guadeloupe, Midi-Pyrénées, esanté Luxembourg, Franche-Comté, Basse Normandie, Haute Normandie, Centre, Auvergne, Languedoc Rousillon, Rhône Alpes, Guyanne, Réunion, Hôpitaux de Paris... Développée en Java, elle fonctionne sur une base clients/serveur. Chaque serveur installé au sein d'un centre hospitalier contient des espaces de collaborations qui sont des espaces dans lesquels les utilisateurs peuvent interagir et travailler à distance. Pour se connecter à un espace de collaboration, un utilisateur télécharge un fichier de type Java Network Launching Protocol (JNLP). C'est un format de fichier associé à la technologie Java Web Start. Il s'agit de pouvoir déployer facilement des applications Java à partir d'un simple navigateur web.

Lorsqu'un utilisateur est connecté à un espace de collaboration, il peut choisir d'entrer dans différentes salles de travail. À son entrée dans une salle, il est considéré comme

1. Accident Vasculaire Cérébral - Attaque cérébrale due à un trombus ou à une hémorragie

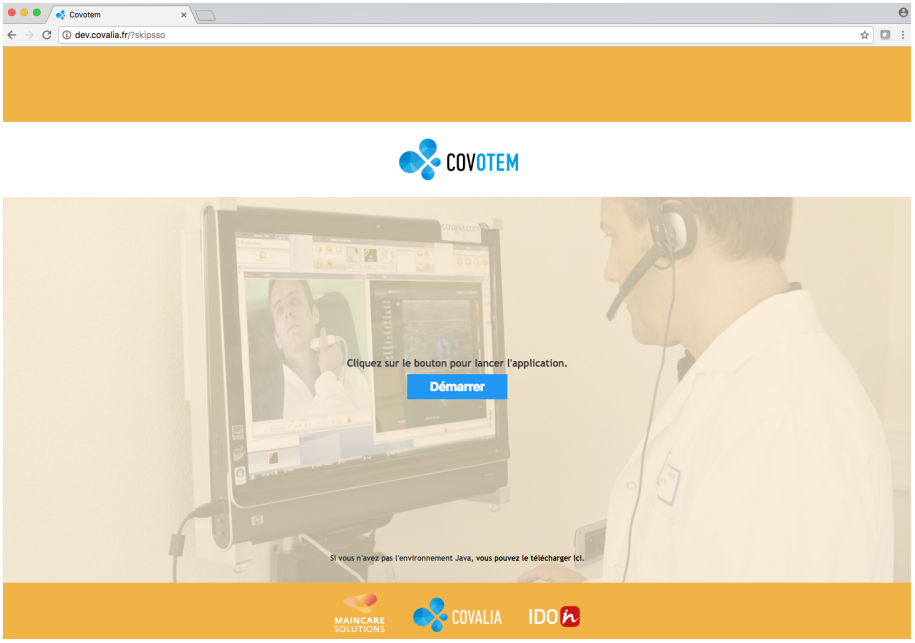


FIGURE 3.1 – Connexion d'un client Covotem^{TM}

étant en réunion. Il peut ainsi collaborer avec d'autres participants de la salle. En réunion, les utilisateurs peuvent effectuer de la vidéoconférence, échanger des documents en temps réel, partager leurs écrans, ... Un utilisateur peut également travailler en dehors d'une salle, il est alors défini comme étant hors-réunion. Il existe donc deux modes dans l'application : le mode synchrone (en réunion) et le mode asynchrone (hors réunion).

La plateforme Covotem^{TM} comporte plusieurs modules :

- La vidéoconférence,
- Le chat,
- La photo,
- L'éditeur de rapport,
- Le partage ou la capture,
- L'imagerie,
- La vidéo,
- Le lancement des applications externes,
- ...

Dans le cadre de cette thèse, nous nous intéressons au module de vidéoconférence, et plus précisément à l'adaptation de la vidéo appliquée au domaine de la télémédecine. Le module de vidéoconférence permet de visualiser les vidéos (webcams, caméra IP ou codecs SIP) des autres participants de la réunion. Il contient différents modes d'affichage : mosaïque ou large. Le mode mosaïque permet de visualiser l'ensemble des flux vidéo des participants de la réunion. Le mode large quant à lui se concentre sur l'affichage d'un seul flux vidéo. Si une caméra IP motorisée est affichée en mode large, il est possible de la contrôler à distance : orientation, zoom, définition de positions, ... Tous les participants à la réunion ont la possibilité d'intervenir sur une caméra. De plus, il est possible d'enregistrer une vidéo ou de prendre une photo à partir d'un flux vidéo d'une caméra IP ou d'une webcam. Ce module a été complètement délocalisé afin d'intégrer l'intergiciel

VAGABOND.

3.1.2/ VUE GLOBALE DE LA PLATEFORME

L'architecture de la plateforme VAGABOND est présentée dans la figure 3.2. Le mot VAGABOND est un acronyme en anglais de *Video Adaptation framework, crossing security GAteways, Based ON transcoDing*. Il s'agit d'une architecture spécialement conçue pour prendre en compte le contexte d'une session de vidéoconférence de professionnels de santé et pour permettre l'adaptation des paramètres de la session à ce contexte. Compte tenu du contexte CIFRE de cette Thèse, nous nous sommes focalisés principalement sur les données contraintes au temps-réel dans le cadre de la télémédecine. Ainsi, nos mécanismes d'adaptation sont spécifiquement étudiés pour ces types d'échanges.

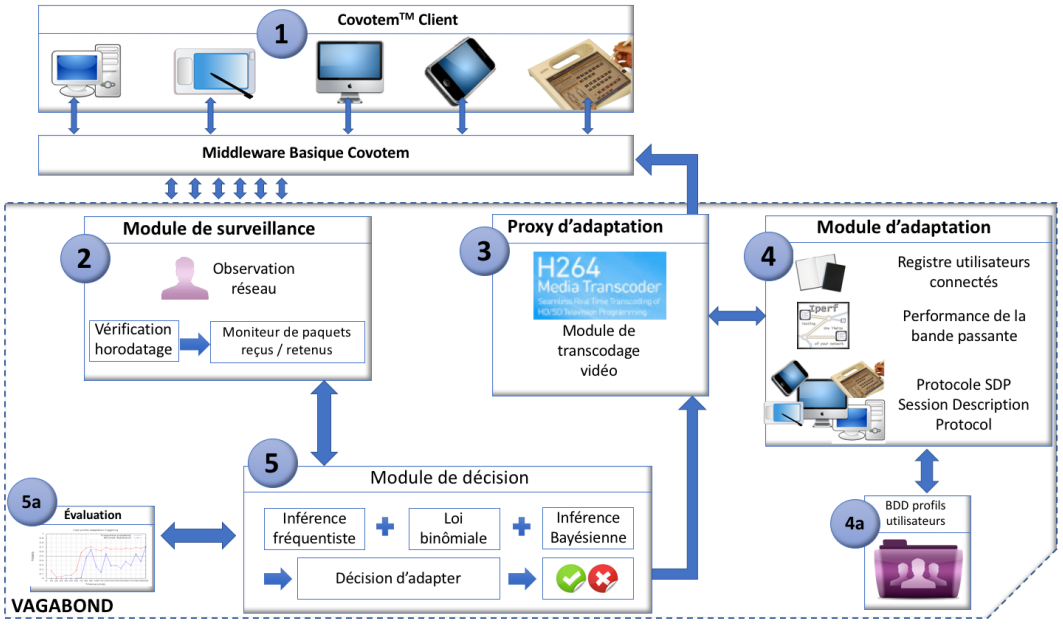


FIGURE 3.2 – L'architecture globale de la plateforme VAGABOND

Cette architecture est de type client-serveur et des mécanismes d'adaptation sont présents côté serveur et côté client. Sur la figure 3.2 le repère 1 représente les clients de l'application : il s'agit ici des clients covotem (cf section3.1.1).

Le repère 2 correspond au module de surveillance qui fait partie de l'architecture de VAGABOND. Ce module est embarqué sur les clients et surveille l'état du réseau. Concrètement, il est chargé de vérifier l'horodatage de chaque paquet vidéo entrant. Un moniteur de paquets est ensuite chargé de comptabiliser les nombres de paquets retenus et rejetés. Les critères et les décisions prises à ce niveau seront détaillés dans le chapitre 4 suivant : en définissant selon quels critères un paquet vidéo est soit accepté, soit refusé.

Ces données collectées et issues d'une phase d'observation sont ensuite transmises au module de décision 5 qui est chargé de faire des évaluations quant à l'état du réseau actuel en se basant sur des études statistiques de ces données difficilement prédictibles. En outre, des lois de probabilité telles que l'inférence fréquentiste, la loi binômiale, et l'inférence bayésienne seront utilisées. De ces évaluations seront définies des règles

d'adaptation. Ces dernières sont transmises au proxy d'adaptation qui est chargé (en tenant compte entre autres des paramètres des terminaux, des profils utilisateurs, ...) de diffuser les nouveaux paramètres de la vidéoconférence en cours.

Le proxy d'adaptation (3) est chargé de transcoder (décodage et encodage dans un nouveau format) les trames de vidéo : soit dans un format adapté au terminal du client (en particulier au début du cycle), soit en fonction des choix du module d'adaptation. À noter que le transcodage se fera dans une résolution inférieure ou égale à l'originale. Tous les flux échangés se font avec le protocole TCP et sont chiffrés en AES. Chaque proxy d'adaptation déployé est rattaché à un serveur d'adaptation unique. Sur le schéma 3.2, ce serveur est noté module d'adaptation (4). Ce module possède un registre unique répertoriant tous les proxies d'adaptation lui étant rattachés, un registre répertoriant tous les utilisateurs connectés à l'intergiciel. Il possède également un serveur IPerf (voir la section 4.2) qui est utilisé à l'étape d'établissement d'une connexion entre un client et le module d'adaptation de VAGABOND. Tous les échanges entre ce serveur et les clients se font par le biais de Webservices de type REST. Enfin, une base de données regroupant les profils utilisateurs des professionnels de santé est également rattachée à ce module (en 4b).

3.1.3/ LES ÉCHANGES ENTRE LES COMPOSANTS DE LA PLATEFORME

Les échanges entre un client et le serveur central d'adaptation se font par le biais de Webservices REST. REST est l'acronyme de *Representational State Transfer*. Il s'agit d'un style d'architecture orienté service (ou ressource) et non d'un simple protocole. Les architectures suivant l'architecture REST sont souvent appelées RESTful. L'architecture REST respecte un découpage entre client serveur qui peuvent évoluer indépendamment (contrairement à l'architecture des Webservices SOAP *Simple Object Access Protocol* dans laquelle les clients et les serveurs sont liés). Les appels entre les entités sont sans état, c'est à dire qu'ils ne dépendent pas d'un contexte conservé sur le serveur. Les requêtes sont simples et sous forme URI (*Uniform Resource Identifier*) avec des verboses HTTP (comme GET / POST / PUT / DELETE ...), des entêtes de requête pour décrire les informations envoyées. Le choix de REST est dû au fait que le protocole SOAP rencontre un problème d'interopérabilité que l'architecture REST résout.

De plus, les Webservices de type REST utilisent un URI (*Uniform Resource Identifier*) pour permettre de joindre des ressources distantes. Comme par exemple une page web qui ne change pas de nom, mais dont l'adresse IP peut changer de manière transparente à l'utilisation de la page web. Il y a donc une véritable flexibilité d'évolution avec les Webservices REST. En revanche, dans le cas de SOAP il y a un lien par contrat. Ce dernier est émis par le serveur et les clients se doivent de respecter ce contrat (un Webservices Descrition Language *WSDL*). Si le WSDL change au niveau du serveur, les clients doivent s'adapter pour respecter le nouveau contrat exigé par le serveur. Le couplage entre un serveur et les clients dans le cas de SOAP est très fort.

Les échanges entre un client et un proxy d'adaptation se font uniquement en TCP sur le port d'écoute du proxy d'adaptation. Généralement, il s'agit du port 80 ou du port 443 car comme nous l'avons expliqué dans les rappels sur le contexte de ce travail en début de ce chapitre, il s'agit des ports qui sont ouverts dans les établissements de santé. À titre d'illustration, le diagramme 3.2 expose les types d'échanges entre clients et serveurs.

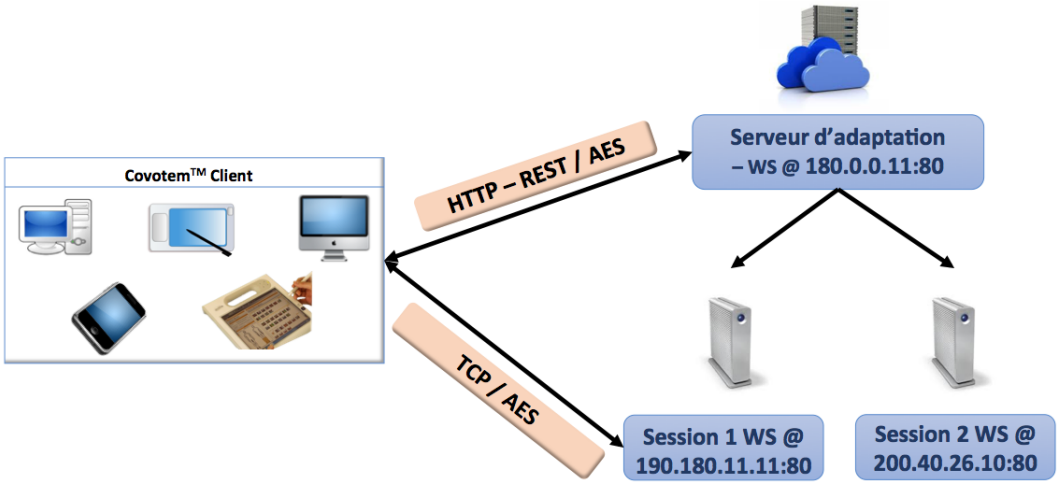


FIGURE 3.3 – Types d'échanges entre un client et les serveurs

Tous les flux échangés avec le serveur central d'adaptation et les proxies d'adaptation sont chiffrés à l'aide du standard AES (*Advanced Encryption Standard*) d'une clé de 128 bits. AES est le standard issu du concours lancé en 1997 par la NIST (*National Institute of Standards and Technology*). C'est le standard par défaut du gouvernement Américain. Il existe une attaque connue contre ce système de chiffrement [Jac11]. Cependant, malgré la rapidité de cette technique d'attaque, elle reste très complexe et difficilement réalisable en utilisant les technologies actuelles. Avec un trillion de machines, chacune pouvant tester un milliard de clés par seconde, cela prendrait plus de deux milliards d'années pour récupérer une clé AES 128 bits. L'attaque n'aurait donc pas, pour l'instant, d'implication pratique sur la sécurité des données des utilisateurs. Toutefois, elle met fin au mythe du chiffrement AES, considéré auparavant comme incassable.

3.1.4/ EXEMPLE DE SCÉNARIO D'UTILISATION

Afin de bien expliquer le mode de fonctionnement de notre plateforme, nous présentons dans cette section un exemple de scénario d'utilisation et de mise en œuvre de l'intergiciel VAGABOND. Dans cet exemple, nous utilisons trois clients comme le montre la figure 3.4 : un poste neurologue N, et des postes de deux internes I_1 et I_2. Il s'agit d'un contexte de télédiagnostic en neurologie.

Notons que dans une telle session à distance, l'expert neurologue doit avoir un visuel et entendre le patient car il devra suivre un protocole dans lequel un ensemble de gestes spécifiques seront demandés au patient et le neurologue analysera et donnera un score en fonction de gestes et de réponses audios. Les sites distants sur lesquels les internes sont présents n'ont besoin que d'entendre l'expert. Ils peuvent l'entendre avec un son de qualité moindre (du 8000Hz au lieu de 48000Hz par exemple). Le schéma simplifié, 3.4 expose cet exemple.

Toutes les étapes d'adaptation sont faites automatiquement sans une quelconque intervention humaine. Le scénario d'utilisation se résume suivant les étapes suivantes :

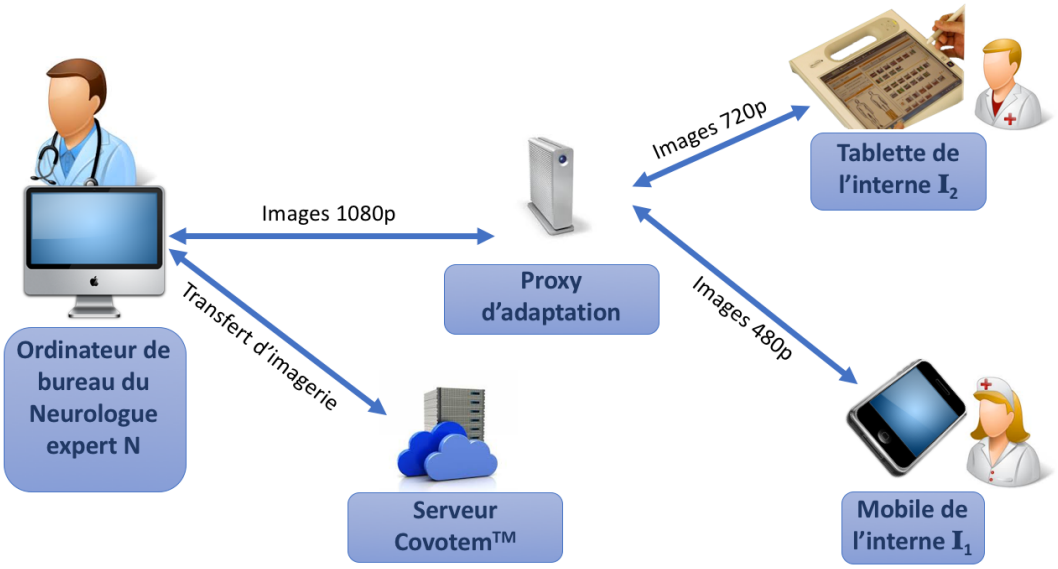


FIGURE 3.4 – Un exemple simplifié de scénario d'utilisation

- Le poste du client N est celui de l'expert en neurologie. Les postes de I_1 et I_2 étant ceux des internes demandant une aide distante au diagnostic.
- Le poste du client N est un ordinateur de bureau, celui de I_1 est un mobile et enfin celui de I_2 une tablette.
- Le poste du client N dispose d'une caméra haute résolution de 1080p et d'un écran de haute résolution de 1080p. Celui de I_1 possède un écran avec une résolution maximale de 320*480 pixels et d'une caméra de 480p. L'écran de I_2 possède une résolution maximale de 1024*768 pixels et dispose d'une caméra de 720p.
- Lorsque la session de vidéoconférence débute, la bande passante est suffisante pour soutenir une session avec des définitions maximales des matériels. Tous les clients se voient et s'entendent.
- Le client N envoie son flux d'origine compressé avec le standard H.264 à son proxy d'adaptation. Ce dernier, avant l'envoi aux autres participants de la session se charge dans un premier temps de transcoder ce flux (1080p en 720p et 480p). Après la phase de transcodage, il envoie ces flux aux autres participants (I_1 et I_2) de la session.
- À un instant t de la session, le neurologue sur machine N décide d'envoyer des images médicales DICOM du scanner cérébral du patient étudié à I_1 et I_2 (cet examen est composé de 300 coupes, avec une taille globale de 500 Mo).
- Ainsi, au cours de l'envoi de ces images de scanner, la bande passante disponible pour l'expert est réduite : la session avec les autres participants I_1 et I_2 devant se poursuivre de manière pérenne.
- Le module de surveillance et le module de décision chez les clients I_1 et I_2 détectent des latences dans la réception des paquets vidéo.
- Étant donné que nous nous situons dans un contexte de neurologie, les internes I_1 et I_2 disposent de profils utilisateurs qui indiquent un besoin de session de audioconférence optimale. Ces profils ont été téléchargés au début de la session, de la base de données.
- Les clients (les machines) sur lesquelles sont connectés les internes I_1 et I_2 envoient une demande automatiquement (suite aux règles émises par le module de

décision) auprès du client émetteur sur lequel se situe le neurologue N afin que ce dernier modifie ses paramètres (audio, vidéo) de vidéoconférence. Les clients de I_1 et I_2 adaptent également leurs paramètres.
- Les transferts depuis le client de N peuvent se poursuivre tout en continuant la session de vidéoconférence en mode dégradé par exemple en diminuant le nombre de médias (le client de N désactive automatiquement sa vidéo mais les utilisateurs sur les clients I_1 et I_2 continuent d'entendre le neurologue. Le client N émettra une qualité audio dégradée : 8000Hz au lieu de 48000Hz.
- À tout instant, par exemple à la fin des envois en provenance du client N, le module de décision émettra de nouvelles règles et l'adaptation des flux changera. Un retour vers la phase initiale est possible si le module de surveillance perçoit une amélioration nette de la bande passante disponible du poste N.

3.1.5/ LES CHOIX D'IMPLÉMENTATION DE VAGABOND

Nous avons retenu l'architecture client/serveur pour nos développements. L'intergiciel VAGABOND a été développé en Java, car d'une part, l'application Covotem™ est implémentée entièrement en Java et d'autre part, l'avantage majeur de ce langage, outre qu'il soit organisé et gratuit, est qu'il est portable. Les Webservices REST sont implémentés en Java avec l'API JAX-RS en utilisant *Jersey*. L'API JAX-RS est un ensemble d'outils permettant de développer les Webservices REST. JAX-RS permet de développer des applications Web facilement et est inclus dans la suite de Java EE6. *Jersey* est une bibliothèque *open source* qui implémente la JSR-311 (*Java Specification Request*). La JSR-311 est la référence pour l'implémentation des architectures RESTful. De plus, Jersey met à disposition une *API* qui permet aux développeurs d'étendre les capacités de Jersey. Enfin, REST, par comparaison à SOAP, permet une indépendance entre un client et un serveur. Concernant le chiffrement de nos flux, nous utilisons la bibliothèque cryptographique *Bouncy Castle*. Il s'agit d'une collection d'APIs qui sont utilisées à des fins de chiffrement. La collection est écrite en langage Java et C# . Le standard AES est utilisé et est connu, à ce jour, pour être un standard de sécurité informatique inviolable en temps réel.

3.2/ LES DIAGRAMMES DE SÉQUENCE

Le diagramme de séquence est un des diagrammes UML (*Unified Modeling Language*). Il est une représentation intuitive lorsque l'on souhaite modéliser des interactions entre des entités. Ils permettent à l'architecte/designer de créer au fur et à mesure sa solution. Cette représentation intuitive est également un excellent vecteur de communication au sein de l'équipe d'ingénierie. Les diagrammes de séquence sont également utiles dans la phase de test.

Les traces d'exécution d'un test peuvent en effet être représentées sous cette forme et servir de comparaison avec les diagrammes de séquence réalisés lors des phases d'ingénierie. Les principales informations contenues dans ce type de diagramme sont les messages échangés entre les lignes de vie, et présentés dans un ordre chronologique.

Les diagrammes de séquence permettent d'exposer toutes les activités de l'intergiciel VAGABOND. Ils ont pour vocation d'expliquer les mécanismes de notre système. Ils ont

également permis de mettre en évidence certaines lacunes de traitements et inefficacités (goulots d'étranglement, redondances, blocage, ...) qui ont ainsi pu être corrigées lors de la conception de l'intergiciel.

3.2.1/ LE DIAGRAMME DE SÉQUENCE DE LA CONNEXION À UNE SESSION

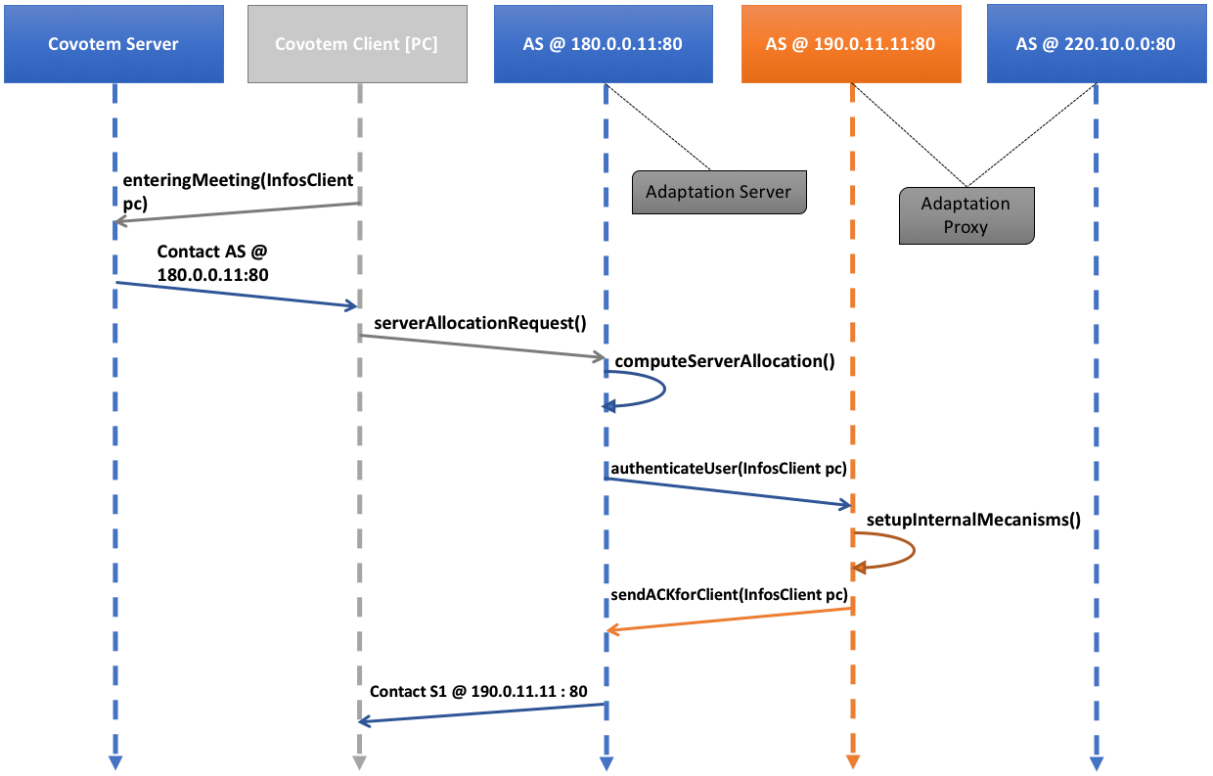


FIGURE 3.5 – Le diagramme de séquence de la connexion d'un client

Le diagramme de séquence, de la figure 3.5, présente la délocalisation du module de vidéoconférence de l'application Covotem^{TM} vers l'intergiciel VAGABOND. Lors de la connexion d'un client Covotem^{TM} dans une salle virtuelle, le serveur de Covotem^{TM} demande à ce dernier de contacter le serveur central de VAGABOND.

De par les caractéristiques matérielles et réseau d'un client, le serveur central attribuera à ce dernier un proxy d'adaptation. Tous les échanges se feront avec ce proxy d'adaptation attribué par le serveur central de VAGABOND.

3.2.2/ LE DIAGRAMME DE SÉQUENCE DE SESSION EN COURS ENTRE DIFFÉRENTS CLIENTS

Les deux pages suivantes exposent le diagramme de séquence illustrant un scénario complet entre trois utilisateurs. Les scénarii précédemment vus dans le *diagramme de séquence de la connexion d'un client* concernant les connexions des clients dans une salle virtuelle s'appliquent également.

Sur les diagrammes 3.6 et 3.7, des pages 81 et 82, sont représentés trois clients. Pour deux d'entre eux (le client *PC* et le client *Tablet*), le même proxy d'adaptation leur est alloué alors que pour le troisième (le client *Mobile*) sera connecté via un proxy d'adaptation différent. Dès lors qu'il y a au moins deux clients dans une même session de vidéoconférence, ils partagent chacun leurs flux respectifs avec leur proxies d'adaptation alloués. En début de session, ils émettront une requête vers leur proxy d'adaptation pour faire la demande des flux des participants distants.

Comme illustré sur le diagramme de séquence, il est possible qu'un proxy différent soit alloué à un des clients. Comme c'est le cas avec le client *Mobile* où il émet la demande des flux des clients *PC* et *Tablet* à son proxy d'adaptation. Ce dernier n'a aucune connaissance de ces clients et remonte donc ces informations au serveur central. Comme expliqué précédemment, le serveur central contient un registre répertoriant tous les clients connectés et les proxies d'adaptation qui leurs ont été alloués.

Ce serveur central indique au proxy d'adaptation du client *Mobile* à quels proxies d'adaptation (si plusieurs clients sont connectés sur plusieurs proxies d'adaptation différents) les autres participants sont connectés. Le proxy d'adaptation du client *Mobile* lui communique ces informations et le client *Mobile* initie une connexion avec les proxies d'adaptation (s'il y en a plusieurs) et demande les flux manquants. Le diagramme de séquence qui suit sur deux pages (81 et 82) explique ce fonctionnement en détails ainsi que les diiférents échanges de messages nécessaires.

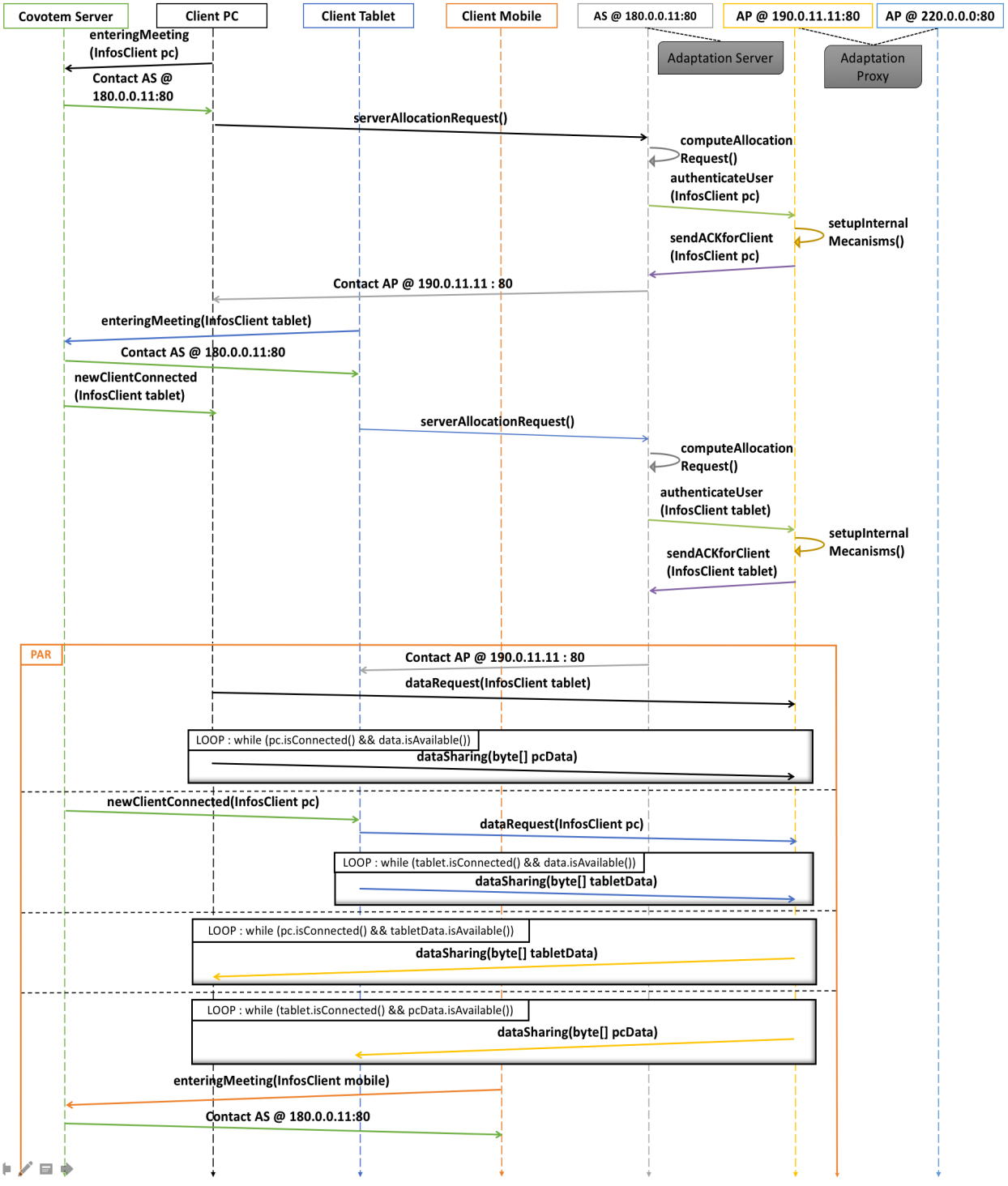


FIGURE 3.6 – Le diagramme de séquence avec plusieurs clients (début)

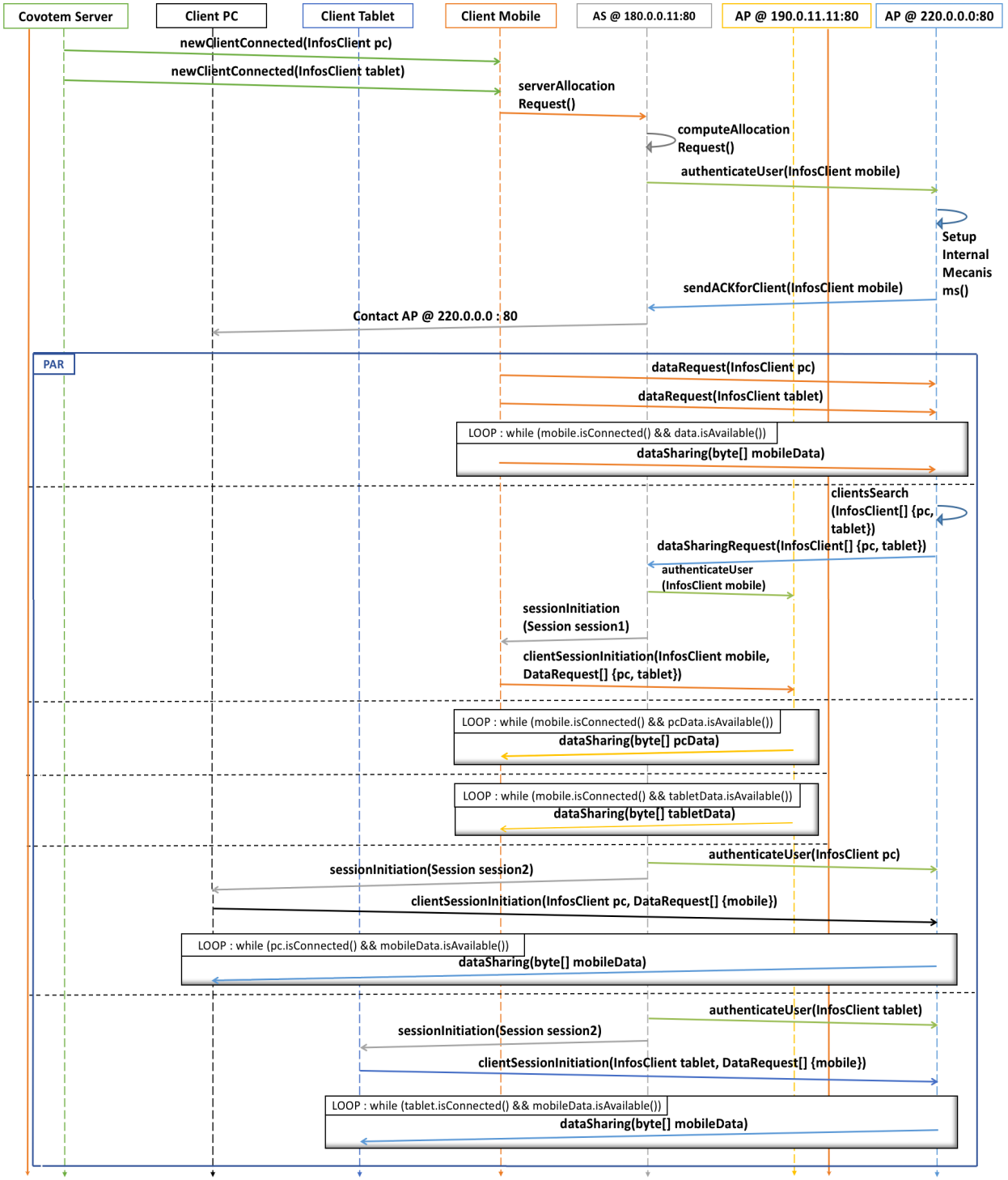


FIGURE 3.7 – Le diagramme de séquence avec plusieurs clients (fin)

3.3/ BASE DE DONNÉES DES SESSIONS

3.3.1/ BASE DE DONNÉES *Versus* ONTOLOGIES DE PROFIL D'UTILISATEUR

Le champ d'investigation des ontologies est devenu indispensable en particulier dans le domaine du web : le profil de l'utilisateur. En particulier dans notre domaine, le profil du

professionnel de santé est important si l'on veut que les données soient adaptées le plus finement possible à la demande de l'utilisateur. Cet aspect peut être critique dans les applications de télémédecine.

Domaine d'utilisation

Les ontologies, qui sont orientées profil de l'utilisateur, sont définies sur la base de l'analyse des informations caractérisant les intérêts et les préférences de l'utilisateur. L'idée au niveau des moteurs de recherche est de mieux connaître l'utilisateur afin d'améliorer la qualité des résultats de recherche en affinant les critères.

Recherche bibliographique dans le domaine des ontologies de profil utilisateur

Le domaine de cette thèse n'est pas dans les ontologies, mais concernant le profil utilisateur il était important de pousser plus avant les investigations. K. Skillen et al., dans [Ski12], ont proposé une ontologie de profil sensible au contexte dans des environnements mobiles. Des ontologies de profil utilisateur pour la perception de situations dans les réseaux sociaux ont été présentées dans [Sta08] : ce système est une aide à l'utilisateur sur les réseaux sociaux pour contrôler l'accès spécifique aux données en fonction des catégories de personnes dans des situations données.

S. Calegari et G. Pasi, dans [Cal11], ont présenté la définition formelle de l'ontologie de profil utilisateur basée sur l'utilisation de YAGO (*Yet Another Great Ontology*) [1]. L'objectif de ce système est d'extraire la partie utile de YAGO pour définir le profil de l'utilisateur et de l'organiser en une représentation ontologique cohérente exprimée par un langage tel que RDFS.

Maria et al., dans [Gol07], ont proposé une modélisation du profil de l'utilisateur permettant de s'adapter aux besoins de chaque application sur la base d'une structure générale commune. Dans [Lil10], une ontologie de profil dans le cadre des *achats mobiles personnalisés* a été créée utilisant OWL-DL et mise en œuvre à l'aide de Protégé. A. Hoppe et al., dans [Hop15], proposent l'affichage des données sur une page dynamique en fonction des *log* de navigation internet des utilisateurs. Sont ainsi appliqués une combinaison d'analyse des données, d'ingénierie de l'ontologie, et de traitement des grandes ressources de données.

Discussion sur la gestion du profil utilisateur au sein de VAGABOND

Cette brève étude du domaine des ontologies utilisateurs montre que ce type d'ontologie permet de trouver pour chaque utilisateur les réponses qui lui sont propres : au bon moment, au bon endroit et adaptées au bon utilisateur.

1. http ://www.mpi-inf.mpg.de/.../research/yago-naga/yago/

Dans le cas de notre module d'adaptation, les données qui seront stockées ne seront pas d'une grande *"dynamicité"* et il n'est pas nécessaire d'ajouter de sémantique aux données. En effet, le module d'adaptation ne nécessite pas autant de *"finesse"* que celle requise aux moteurs de recherche.

Dans cette première version de l'intergiciel, nous avons donc choisi une simple base de données des profils utilisateurs.

3.3.2/ SCHÉMA RELATIONNEL DE LA BASE DE DONNÉES DES SESSIONS

Cette section du document présente le schéma relationnel de la base de données regroupant les profils utilisateurs des professionnels de santé qui est utilisée dans l'architecture de VAGABOND. Ces profils sont téléchargés par le client à l'étape d'établissement d'une connexion. Cette base de données fait partie intégrante de l'architecture et aide à appliquer les différentes stratégies d'adaptation (cf le schéma relationnel donné figure 3.8).

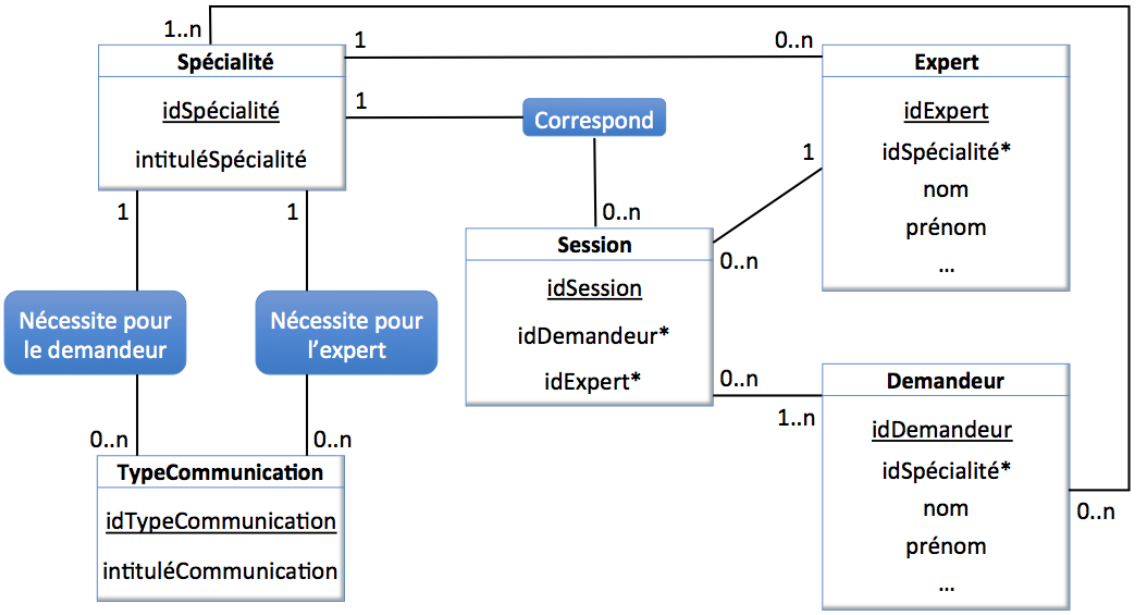


FIGURE 3.8 – Schéma relationnel de la base de données des sessions

Pour résumer

SYNTHÈSE

Dans ce chapitre nous avons présenté l'architecture globale de l'intergiciel VAGABOND qui se compose de plusieurs modules, dont des modules dédiés à l'adaptation des flux et à la stratégie d'une session de vidéoconférence. Nous avons défini un module de surveillance et un module de décision qui sont rattachés à un client et un module d'adaptation qui se compose de plusieurs proxies d'adaptation et d'une base de données pour les différents profils des professionnels de santé.

Afin de bien comprendre à la fois les mécanismes et la finalité de l'intergiciel VAGABOND, il était nécessaire de présenter les diagrammes de séquence et le schéma relationnel de la base de données dont les descriptifs constituent la dernière section de ce chapitre. De plus, ces diagrammes ont permis de mettre en évidence certaines lacunes et défaillances de l'intergiciel et ainsi les corriger lors de la phase de conception.

La collaboration entre les différents modules de l'intergiciel permet aux professionnels de santé d'effectuer des sessions de vidéoconférences dans des conditions optimales et en respectant la bande passante disponible. Un module de surveillance est embarqué sur les clients et surveille l'état du réseau. Les données collectées et issues d'une phase d'observation sont ensuite transmises au module de décision qui est chargé de faire des évaluations quant à l'état du réseau actuel en se basant sur des lois de probabilité telles que l'inférence fréquentiste, la loi binômiale, et l'inférence bayésienne. De ces évaluations résultent des règles d'adaptation.

Ces dernières sont transmises au proxy d'adaptation qui est chargé d'adapter les flux à l'aide de son transcodeur vidéo. En plus de ces règles, le proxy reçoit des données en provenance du module d'adaptation comprenant entre autres des données de profil utilisateur, caractéristiques des terminaux,...

Notons que dès le début de session le transcodeur adapte les flux aux caractéristiques des terminaux récupérées par le protocole SDP, et qu'ensuite les données d'adaptation s'ajoutent dans le cycle de fonctionnement pour adapter dynamiquement les flux.

Dans le chapitre suivant, nous présentons plus en détails les différents modules liés aux techniques d'adaptation. Nous approfondissons les conceptions mises en œuvre dans l'élaboration de l'intergiciel.

Enfin, nous achevons ce chapitre par une section dédiée aux différents profils des professionnels de santé dans le cadre d'une session de vidéoconférence.

4

LES COMPOSANTS DE VAGABOND

Ce chapitre est consacré aux différents composants de l'intergiciel VAGABOND. La première section explique en détails l'établissement de connexions des clients aux serveurs. Le type de partage de flux employé y est également expliqué ainsi que les différents types de pont de vidéoconférence (Selective Forwarding Unit, Multiple Control Unit, ...) qui existent et dont nous nous sommes inspirés. Nous consacrons une section au traitement des paquets, et plus précisément au cheminement des paquets multimédia et quel type de traitement est appliqué par rapport à l'horodatage présent dans chaque paquet reçu dans une session de vidéoconférence par un client.

L'adaptation des flux échangés ainsi que le transcodage qui est présent sur un proxy d'adaptation sont ensuite exposés. Nous avons utilisé la norme d'encodage vidéo H.264 ainsi que le décodage, et la décomposition hiérarchique d'une vidéo au format H.264.

La section suivante est consacrée à la détection de l'état du réseau, les mécanismes de détection et de déclenchement de la stratégie d'adaptation. Nous expliquons également comment nous modélisons l'état du réseau et comment nous appliquons différentes lois de statistiques telles que l'inférence fréquentiste, la loi binomiale, et l'inférence bayésienne afin de prédire la bande passante du réseau. Enfin, nous terminons ce chapitre par le concept de profil utilisateur et la manière dont il est appliqué lorsqu'une nouvelle stratégie d'adaptation est requise dans une session de vidéoconférence.

4.1/ VUE GLOBALE DE L'INTERGICIEL VAGABOND

L'intergiciel VAGABOND s'appuie sur une architecture distribuée de type client-serveur. Rappelons que ce système fonctionne sur le protocole *Transmission Control Protocol (TCP)* et sur les ports 80 et 443 dans les établissements de santé. Les traitements sont répartis sur les clients et les serveurs. Comme nous l'avons identifié dans la partie I de ce document (*état de l'art*), le contexte d'un système distribué est composé de plusieurs éléments. Et son adaptation doit prendre en compte tous les éléments du contexte, ce qui demeure très difficile, voire impossible. Dans cette thèse, nous traitons principalement les données multimédia issues d'une session de vidéoconférence, et leur adaptation à la bande passante disponible.

Le système dispose d'un serveur principal et unique. Ce serveur est utilisé comme un registre et gère la connexion des clients. À chaque nouvelle connexion d'un utilisateur au système, une demande est faite auprès du serveur principal, l'*Adaptation Server* accessible sur le port 80 ou 443 dans les établissements de santé français. Ce principe est

également présent dans le protocole SIP (*Session Initiation Protocol*) [Ali13]. Le protocole SIP est développé et maintenu par le groupe de travail SIP spécifié par l'*Internet Engineering Task Force (IETF)*. C'est un protocole de communication permettant l'établissement de session, la manipulation de la partie de signalisation dans une session, et la terminaison d'une session. Ce protocole s'est largement répandu dans les systèmes de vidéoconférence, les applications de type point-à-point, la messagerie instantanée et vocale, les jeux en ligne, ...

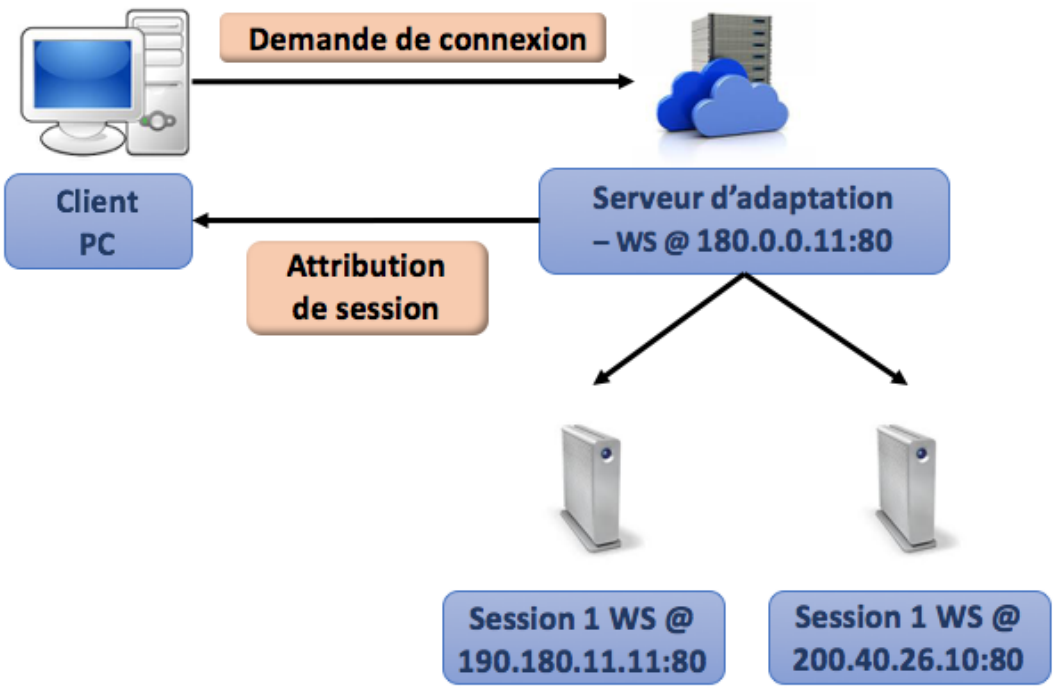


FIGURE 4.1 – Demande de connexion d'un client

SIP est un protocole de signalisation pour l'initiation, la gestion et la terminaison de sessions à travers des réseaux de paquets (les réseaux IP par exemple). Un système SIP dispose de deux composants basic : les agents utilisateur et les agents serveur (*registrars*, proxy, redirection). Les agents utilisateurs participent directement dans le réseau de communication. Ils sont découpés en deux sous-catégorie, les agents utilisateurs clients qui initient des sessions, et les agents utilisateurs serveurs, chargés de réagir aux requêtes. Les *registrars* reçoivent des messages d'enregistrement des agents utilisateurs et enregistre leurs adresses. Il s'agit d'un registre répertoriant tous les agents utilisateur connectés.

Dans l'architectures de VAGABOND, il existe un serveur centralisé appelé le *serveur d'adaptation*. À ce serveur sont associés plusieurs autres serveurs, qui sont en charge des échanges des données issues des sessions de vidéoconférences entre les différents clients. Ces serveurs sont appelés les *proxies d'adaptation*. Le *serveur d'adaptation* connaît tous les *proxies d'adaptation* qui sont déployés car lors de leur déploiement ils s'enregistrent auprès de lui. De cette manière, le *serveur d'adaptation* peut allouer facilement un *proxy d'adaptation* à un nouveau client. Ainsi lorsqu'un client débute une session de vidéoconférence, il contacte en premier lieu le *serveur d'adaptation*. Ce dernier le redirige ensuite vers l'un des *proxies d'adaptation* opérationnels, en choisissant celui qui dispose de la bande passante optimale par rapport à celle du client. Ainsi, un

client n'a besoin de posséder uniquement que de l'adresse et du port d'écoute du *serveur d'adaptation* lors d'une demande de connexion. Ce type de fonctionnement a été mis en place d'une part, afin d'éviter une gestion lourde de la liste exhaustive de tous les *proxies d'adaptation* déployés, et d'autre part afin qu'un *proxy d'adaptation* choisi par le *serveur d'adaptation* offre bien la bande passante optimale. Le mode de fonctionnement de la connexion d'un client dans une session de vidéoconférence est décrit dans la figure 4.1.

4.2/ ÉTABLISSEMENT DE LA CONNEXION D'UN NOUVEAU CLIENT

Lorsque le serveur d'adaptation reçoit une requête d'un nouveau client, il l'analyse en récupérant en particulier les statistiques de la bande passante du client. Ceci étant fait dans le but d'attribuer le serveur optimal : un client avec une bande passante réduite se voit affecter un serveur avec une plus large bande passante dans le but de compenser les latences qui peuvent être induites par l'envoi des paquets du client et, à contrario, lorsqu'un client dispose d'une large bande passante, le serveur le moins chargé lui est attribué.

Ces choix sont faits par le serveur d'adaptation et l'étude de la bande passante disponible par le client est faite par un serveur Iperf ([Dug14]). Iperf est un outil très connu pour mesurer la bande passante d'un réseau. Il crée des flux de données avec les protocoles TCP (également UDP mais que nous n'utiliserons pas) et mesure le débit du réseau par lequel transiteront les paquets. Iperf est une *ré-implémentation* du programme TTCP (*Test TCP*) [USN84] développé par le *National Center for Supercomputing Applications* à l'Université d'Illinois. Cet outil permet de renseigner divers paramètres qui peuvent être utilisés pour tester un réseau, ou alors dans le but de l'optimiser ou de le réguler. Il s'agit d'une application client-serveur qui peut mesurer le débit entre deux hôtes, soit en unidirectionnel, soit en bidirectionnel. Un rapport est établi à chaque mesure répertoriant la taille des données transmises et le débit mesuré. C'est une application multiplateforme qui peut s'exécuter sur n'importe quel réseau. Elle peut donc être utilisée dans les réseaux filaires et sans fil. Elle permet de mesurer la bande passante, la latence, la gigue et la perte de paquets.

Iperf peut être utile dans de nombreux cas. Lors de la résolution d'un problème potentiellement lié au réseau, un test Iperf peut mettre en évidence les caractéristiques réseau, et identifier le problème réseau. Par exemple dans le cas de l'utilisation manuelle par des utilisateurs :

- Si un utilisateur constate les mauvaises performances de sa machine en lien avec une application hébergée sur un serveur, alors un test Iperf TCP entre sa machine et le serveur permettra de déterminer si le problème est lié au réseau ou à la couche applicative.

- Si un utilisateur constate une lenteur généralisée des accès de sa machine, des tests Iperf mettront en évidence les éventuels problèmes réseaux. Si les résultats des tests effectués en ascendant et descendant montrent une forte asymétrie en termes de performances ou de pertes, c'est souvent le signe d'un câble réseau défectueux par exemple.

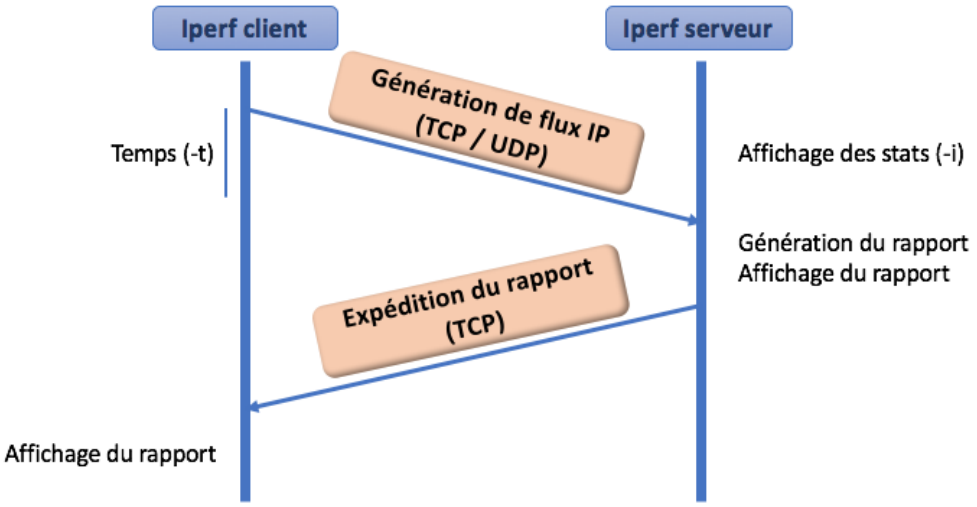


FIGURE 4.2 – Fonctionnement d'Iperf

Iperf fonctionne comme un client/serveur selon le diagramme de la figure 4.2.

Dans l'architecture de VAGABOND, un serveur Iperf est déployé avec le serveur d'adaptation sur le port 80, le port 443 étant réservé au serveur lui-même. Lors de l'établissement d'une session, un client fait une demande auprès du serveur Iperf sur le port 80 et ce dernier retourne alors la bande passante disponible du client. Une autre information qui permet d'optimiser l'envoi de données se trouve également dans ces mesures. Il s'agit du *Maximum Transmission Unit (MTU)*. Le MTU est la taille maximale d'un paquet pouvant être transmis sur la couche réseau sans être segmenté. La découverte de cette valeur peut être utile à l'optimisation du réseau.

Il en résulte que lorsque le serveur d'adaptation reçoit toutes ces données, il affecte au client demandeur un proxy d'adaptation. Le serveur d'adaptation et un client gardent une connexion persistante tant que le client reste dans la session de vidéoconférence. Le serveur d'adaptation partage également pour la session que veut rejoindre le nouveau client la liste des participants à cette dernière. Ce serveur est également responsable de notifier chaque arrivée et départ à tous les participants d'une session de vidéoconférence. Le serveur d'adaptation garde une trace de chaque client actif et du proxy d'adaptation qui lui a été alloué.

Comme indiqué dans la section précédente, ce proxy d'adaptation traite toutes les données venant du client et se charge du routage des paquets entre les différents clients connectés dans la session. La figure 4.3 détaille une connexion établie entre un client et son proxy d'adaptation.

Lors de l'établissement d'une connexion entre un client et un proxy d'adaptation, la première information transmise par le client au proxy d'adaptation concerne le type de périphérique sur lequel le client se trouve : la puissance de calcul, la mémoire vive, la résolution maximale de l'écran du périphérique, la bande passante disponible précédemment calculée avec le serveur d'Iperf sur le serveur d'adaptation, et l'identifiant de la session de vidéoconférence. Ainsi, ces informations sont transmises au moment de l'initiation d'une connexion avec le protocole Session Description Protocol (SDP).

Le protocole SDP a été développé de manière à ce qu'il puisse être utilisé par une large

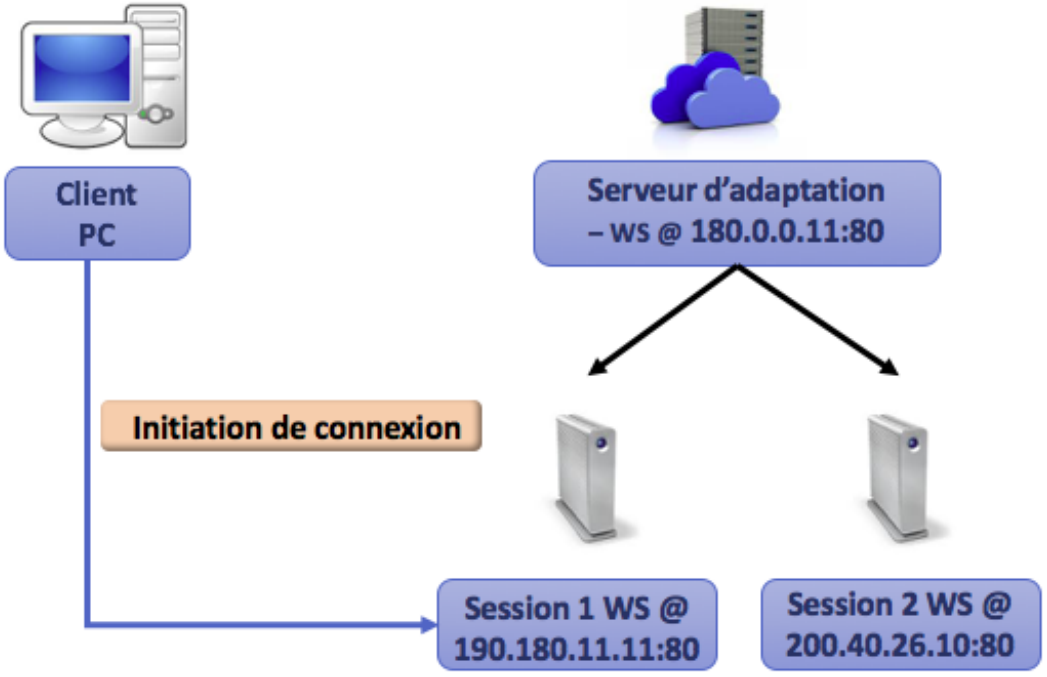


FIGURE 4.3 – Initiation d'une connexion avec le proxy d'adaptation

gamme d'applications réseau. Initialement créé pour résoudre les problèmes inhérents aux communications de type point à point (établissement de connnexion, négociation de session, descriptif des périphériques,...), ce protocole s'est vu s'étendre à d'autres types d'usage. La finalité de SDP est de transporter les informations se rapportant aux flux de données multimédia dans une session multimédia afin de permettre aux récepteurs de cette description de session de participer à cette session. SDP permet d'indiquer l'existence d'une session en fournissant l'ensemble d'informations suivant :

- Le nom et l'objet de la session,
- La durée de la session,
- Le média utilisé pour la session,
- Les caractéristiques de la session (adresses, numéro de port, le format, . . .).

Ce protocole permet préciser des informations complémentaires qui sont très utiles dans notre cas, comme par exemple :

- La bande passante disponible pour la session,
- Les caractéristiques techniques d'un terminal,
- Le type de média utilisé (vidéo, audio, . . .),
- Le protocole de transport utilisé (ce qui permet une évolution vers le protocole Session Initiation Protocol (SIP)),

- Le format du média (H.264 vidéo, H.261 vidéo, MPEG vidéo, . . .).

Un message SDP est composé d'une série de lignes appelées champs dont les noms sont abrégés par une lettre minuscule. Chaque ligne respecte un ordre précis afin d'en simplifier l'analyse lexicale : **v** est un champ obligatoire qui correspond au numéro de version du protocole, **o** est un champ obligatoire qui correspond à l'origine et l'identification de session, **s** est un champ obligatoire qui correspond au nom de la session, **i** est un champ qui correspond aux informations de la session, **e** est un champ qui correspond à une adresse mail, **c** est un champ obligatoire qui correspond aux informations de connexion, **b** est un champ qui correspond aux informations de bande passante, **t** est un champ obligatoire qui correspond à un horodatage quant au début de la session (il peut également inclure la durée de la session), **a** est un attribut de ligne (il peut s'agir ici du type de codec utilisé ou encore des caractéristiques techniques d'un terminal), **m** est une information sur les attributs de ligne (par exemple, type de média).

La figure 4.4 donne un exemple des messages SDP que nous avons définis et qui transitent dans notre intergiciel VAGABOND :

```
v= 0
o= ronnie 960393 IN IP4 51.254.201.50
s= neurologie
i= testEnvoiSDP, session de neurologie
e= ronnie.muthada@ido-in.com
c= IN IP4 51.254.201.50/443
b= 354
t= 2876936337
m= audio 49172 RTP/AVP 0
a= rtpmap :0 PCMU/8000
m= video 23422 RTP/AVP 31
a= rtpmap :31 H264/90000
m= Sony Xperia Z4, 3GB, 1.5 GHz Qualcomm Snapdragon 810 13
a= rtpmap :130 Name, RAM, CPU
m= max 1080*1920 131
a= rtpmap :131 ScreenResolution
```

FIGURE 4.4 – Message SDP dans l'intergiciel VAGABOND

Dans cette thèse, nous ne détaillerons pas le protocole complet SDP mais seulement les attributs utiles à notre l'intergiciel. Le protocole étant lui-même décrit dans la RFC4566 [1].

4.3/ PARTAGE DE DONNÉES

Le proxy d'adaptation fonctionne sur un paradigme appelé un *Selective Forwarding Unit (SFU)*. Ce terme est souvent employé pour décrire un périphérique chargé du routage vidéo dans un système de vidéoconférence. Un SFU est capable de recevoir de multiple flux média et de décider quel flux doit être envoyé et à quel participant. Comme illustré dans la figure 4.5, le principe d'un SFU est qu'un client envoie son flux au SFU et s'il

1. https ://tools.ietf.org/html/rfc4566

existe d'autres participants dans une même session, ce dernier lui enverra les flux de tous les autres participants.

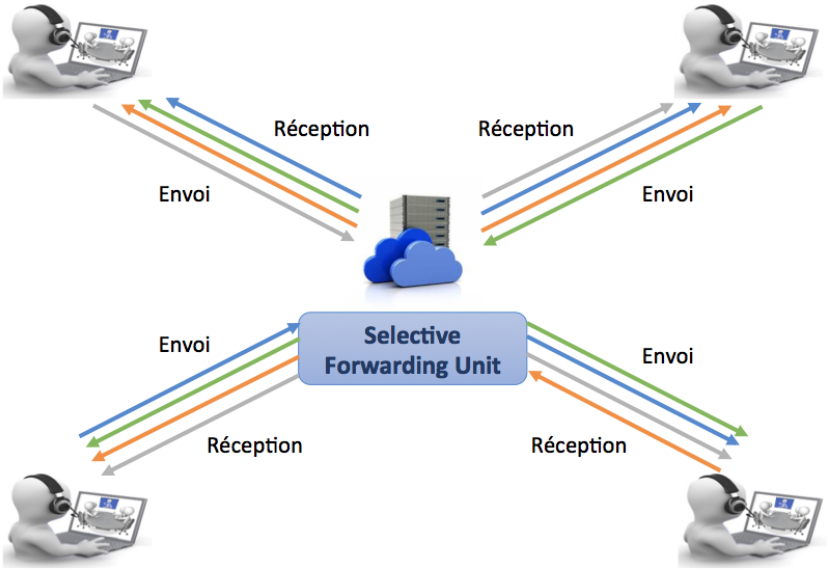


FIGURE 4.5 – Selective Forwarding Unit

Dans le cas de notre intergiciel VAGABOND, un proxy d'adaptation est donc chargé de gérer la réception et l'envoi des paquets multimédia. Dès lors qu'une connexion est établie entre un client et un proxy d'adaptation, s'il existe une conférence en cours avec l'identifiant que le client avait transmis au serveur d'adaptation lors de l'étape d'établissement de la connexion et qu'il existe des participants dans cette session, le client envoie une demande sur les flux des autres participants qu'il partage automatiquement.

Dans le cas contraire, bien entendu aucune demande de partage de flux n'est faite, mais une connexion persistante reste néanmoins active entre le client et son proxy d'adaptation tant que le client reste présent dans la session. Et lorsqu'il existe au moins un autre participant dans une session de vidéoconférence, les clients partagent automatiquement les flux multimédia et font la demande des autres flux multimédia de chacun d'eux. Le partage des flux multimédia se fait directement avec le proxy d'adaptation et le client n'intervient plus une fois que ces flux sont en cours d'envoi. Toutefois, lors de la réception des flux des autres participants se trouvant sur des proxies d'adaptation différents, comme nous avons expliqué dans la section 4.2, un client peut se voir attribuer n'importe quel proxy d'adaptation en fonction de sa bande passante.

Deux cas se présentent alors dans cette démarche :

- Dans le cas le plus simple, admettons que nous ayons deux participants dans une même session, un client *PC* et un client *Tablette*. Dans ce cas, le routage des flux par le proxy d'adaptation ne pose aucun problème. Dès la connexion établie avec le proxy d'adaptation, les clients commenceront à partager leurs flux et demanderont respectivement les flux de l'autre client. Le proxy d'adaptation sera le relai pour le routage des paquets multimédia entre les différents clients. La figure 4.6 illustre ce fonctionnement.

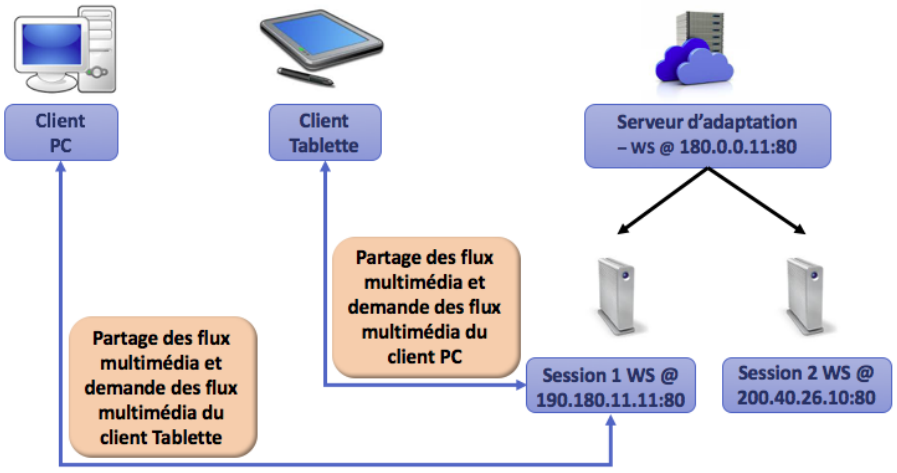


FIGURE 4.6 – Routage des paquets multimédia entre deux clients d'un même proxy d'adaptation

- Le deuxième cas concerne les participants dans une session de vidéoconférence qui ne sont pas tous connectés au même proxy d'adaptation comme illustré sur la figure 4.7. Lorsque ce cas de fonctionnement se produit, chaque client partage ses flux directement avec son proxy d'adaptation.

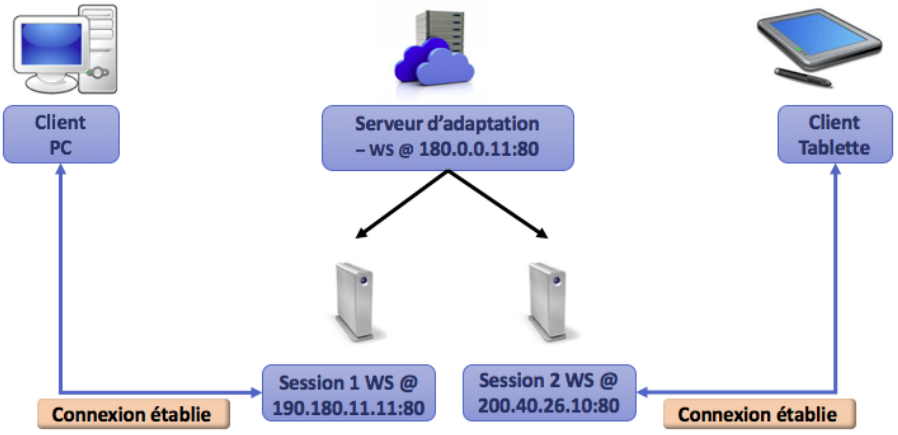


FIGURE 4.7 – Clients connectés sur des proxies différents

Dans un tel cas, le processus complet est décrit dans le schéma 4.8. Ainsi, après avoir lancé le processus de partage de ses propres flux, le client *PC* demande les flux du client *Tablette* à son proxy d'adaptation (étape 1). Comme ce dernier ne possède pas les flux du client *Tablette*, cette demande est remontée au serveur d'adaptation qui détient l'information sur le proxy d'adaptation auquel est connecté le client *Tablette* (étape 2). Le serveur d'adaptation communique cette information au proxy d'adaptation du client *PC* (étape 3), qui lui demande d'initier une connexion avec le proxy d'adaptation du client *Tablette* (étape 4). Le client *PC* initie une connexion avec le proxy d'adaptation du client *Tablette* et transmet une demande pour les flux multimédia de ce dernier (étape 5). Le client *Tablette* procède de la même manière.

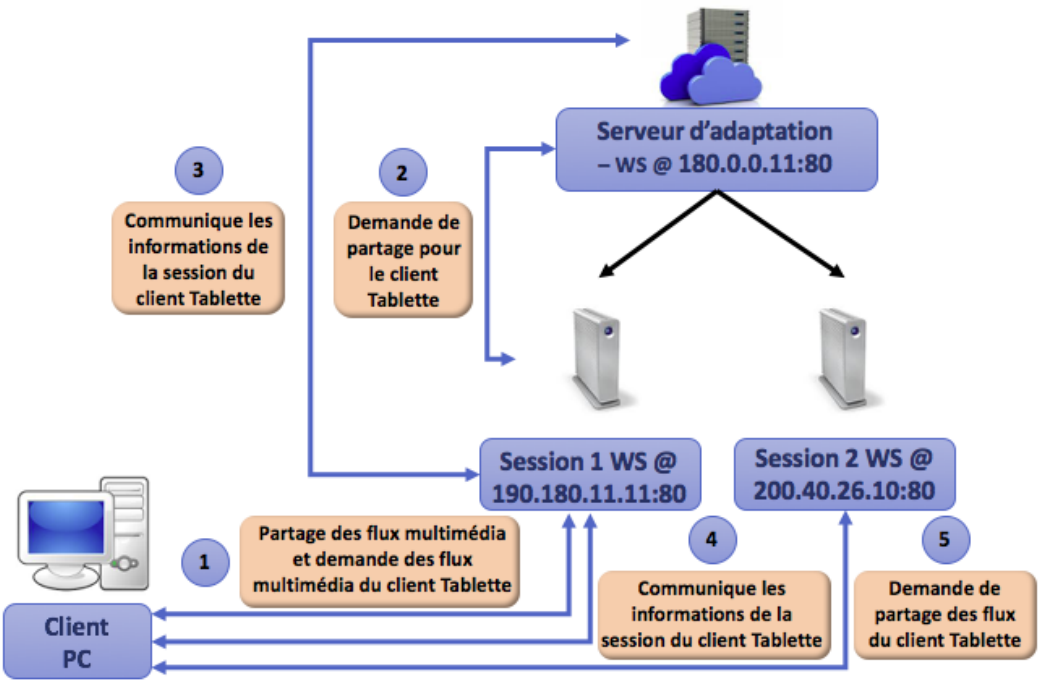


FIGURE 4.8 – Routage des paquets multimédia entre deux clients de différents proxies d'adaptation

4.4/ CHEMINEMENT DES PAQUETS MULTIMÉDIA DANS L'ARCHITECTURE

L'architecture de VAGABOND a été pensée pour surveiller l'état du réseau et réagir proactivement en conséquence. Cette détection se fait tout au long d'une session de vidéoconférence et permet de surveiller la bande passante au cours de la session. Il est important de souligner que tous les paquets échangés dans une session de vidéoconférence sont soumis à des vérifications sur le temps entre l'émission et la réception : ce qui permet indirectement de surveiller la bande passante.

Tous les clients du système ainsi que les serveurs sont synchronisés sur le même serveur Internet de temps, utilisant le *Network Time Protocol (NTP)*. Ce protocole très répandu et connu est utilisé par environ 25 millions de serveurs et d'ordinateurs dans des réseaux privés et publiques pour synchroniser les horloges sur Internet. Il s'agit d'un protocole réseau pour la synchronisation d'horloges entre systèmes informatiques sur des réseaux de données à commutation de paquets et à latence variable. NTP est l'un des plus anciens protocoles d'Internet et a été inventé par le Docteur David L. Mills en 1981 et est opérationnel depuis 1985. Le principe est de synchroniser un serveur avec une horloge atomique qui est ensuite utilisée comme base pour synchroniser un ensemble d'ordinateurs. Le calendrier standard utilisé dans la plupart des pays du monde est le temps universel coordonné (en anglais, *Coordinated Universal Time (UTC)*), qui est basé sur la rotation de la terre autour de son axe, et le calendrier grégorien, qui est basé sur la rotation de la terre autour du soleil. L'heure UTC est diffusée par différents moyens, tels que les systèmes de navigation par radio et par satellite, les liaisons Internet, ... Des récepteurs à usages spéciaux sont disponibles pour de nombreux services de diffusion

de temps, tels que le système de positionnement mondial (en anglais, *Global Positioning System (GPS)*) et d'autres services gérés par divers gouvernements nationaux. Pour des raisons de coût et de commodité, il n'est pas possible d'équiper chaque ordinateur avec l'un de ces récepteurs. Il est plus judicieux d'équiper un certain nombre d'ordinateurs servant de serveurs temporels primaires pour synchroniser un nombre beaucoup plus important de serveurs secondaires et de clients connectés par un réseau commun. Ainsi, un protocole de synchronisation d'horloge réseau distribué est nécessaire pour lire une horloge de serveur, transmettre la lecture à un ou plusieurs clients et régler chaque horloge client au besoin. C'est le but du protocole NTP.

Le protocole de synchronisation détermine le décalage horaire de l'horloge du serveur par rapport à celui du client. Les différents protocoles de synchronisation utilisés aujourd'hui fournissent différents moyens pour le faire, mais ils suivent tous le même modèle général. Sur demande, le serveur envoie un message incluant sa valeur d'horloge actuelle ou son horodatage et le client enregistre son propre horodatage à l'arrivée du message. Pour une meilleure précision, le client doit mesurer le retard de propagation serveur-client pour déterminer son décalage d'horloge par rapport au serveur. Comme il est impossible de déterminer les retards à sens unique, à moins que le décalage de l'horloge réelle ne soit connu, le protocole mesure le délai d'aller-retour total et calcule de délai de propagation en demi-tour. Ceci suppose que les retards de propagation sont statistiquement égaux dans chaque direction. En général, c'est une approximation utile. Cependant, dans l'Internet d'aujourd'hui, les chemins des réseaux et les retards associés peuvent différer considérablement en raison des Fournisseurs d'Accès Internet (FAI) individuels.

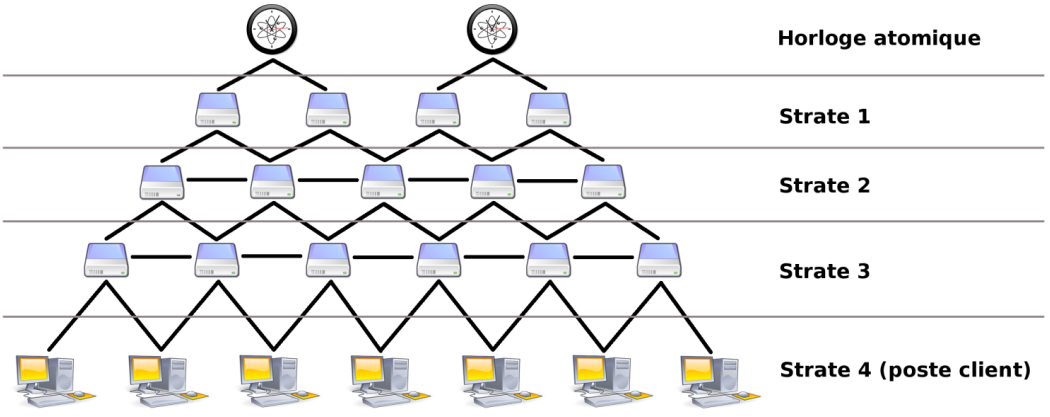


FIGURE 4.9 – Organisation de la communauté NTP - Source : http ://www.ntp.org/

La communauté NTP (cf figure 4.9) est organisée sous la forme d'un graphe ou d'un sous-réseau arborescent, avec les principaux serveurs et les serveurs secondaires, et les clients en bout de chaîne, ou avec un niveau de stratum dans les réseaux d'entreprise. Il est généralement nécessaire, à chaque niveau de stratum, d'utiliser des serveurs redondants et des chemins de réseaux divers afin de se protéger contre les logiciels, les matériels, les réseaux défectueux et les attaques potentiellement hostiles. Les protocoles de synchronisation fonctionnent en un ou plusieurs modes d'association, selon la conception du protocole. Le mode client/serveur, également appelé mode maître/esclave, est pris en charge par le protocole NTP. Dans ce mode, un client se synchronise avec un serveur sans état comme dans le modèle d'appel de méthodes à distance conventionnel

(par exemple, le *Remote Procedure Call (RPC)*).

Les attentes de précision NTP dépendent de l'environnement et des exigences des applications. En pratique, le seul facteur qui affecte la précision pour des mises à jour de longs intervalles sur des grands réseaux est la variation de la température ambiante pour les horloges primaires. Dans des conditions normales et avec les lois de la physique, la fréquence de l'oscillateur d'horloge peut varier de l'ordre d'une partie par million (*Part-Per-Million (PPM)*). Il en résulte une précision de synchronisation de l'ordre de quelques millisecondes avec des intervalles de mise à jour de 15 minutes. Cependant, la précision peut être considérablement améliorée de l'ordre d'une milliseconde avec des intervalles de mise à jour d'une minute, comme c'est le cas avec les horloges primaires.

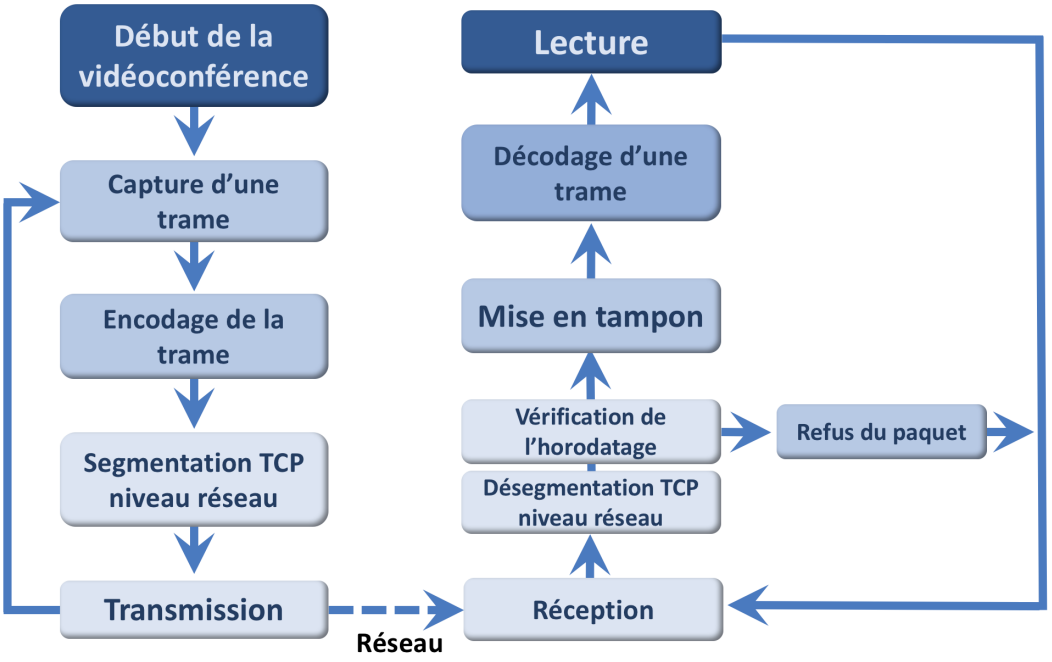


FIGURE 4.10 – Traitement d'un paquet par rapport à l'horodatage

Le facteur qui affecte la précision depuis les serveurs primaires vers les serveurs secondaires est la gigue due aux latences du réseau et du système d'exploitation. À ce niveau, la précision est de l'ordre de quelques microsecondes. Bien entendu, il s'agit d'une propriété du matériel et du système d'exploitation, et non du protocole NTP. En règle générale, la précision sur Internet est proportionnelle au délai de propagation. Pour un Ethernet 100 Mb/s légèrement chargé, la précision est de l'ordre de 100 μs. Pour un chemin Internet intercontinental, la précision peut être de plusieurs dizaines de millisecondes. Sur les réseaux avec de grands retards de propagation asymétrique (transmission satellite et ligne fixe), les erreurs peuvent atteindre **100 ms** au minimum. Il n'existe aucun moyen pour éviter ces erreurs, à condition qu'il existe une connaissance préalable des caractéristiques du chemin d'accès. Ce délai est pris en compte dans l'architecture et à chaque paquet reçu est ajouté ce temps de précision. Le schéma qui suit explique le procédé lors de l'émission d'un paquet jusqu'à sa réception.

Comme le montre la figure 4.10, lorsqu'une session de vidéoconférence démarre, chaque paquet vidéo capturé est tout d'abord encodé au format H.264 avant d'être segmenté et envoyé sur le réseau. L'encodage H.264 sera détaillé dans la section suivante. Les paquets vidéo encodés sont ensuite envoyés au proxy d'adaptation correspondant qui agit

comme un relai entre les différents clients. Un paquet encodé reçu par un client, suite à la phase de désegmentation TCP au niveau du réseau, est soumis à une vérification de l'horodatage. Selon une étude menée par Jack Jansen *et al.* [Jan11], 700 millisecondes serait le temps approprié entre l'émission et la réception d'un paquet dans une session de vidéoconférence de haute qualité. Nous nous basons sur cette étude pour l'acceptation d'un paquet entre son émission et sa réception. À ce temps, s'ajoute également le délai induit par le protocole NTP, soit les 100 millisecondes. Nous avons donc un temps d'acceptation d'un paquet de l'ordre de 800 millisecondes :

Équation 10

Temps d'acceptation d'un paquet TCP

$$T^{Acceptation} = T^{Transmission} + T^{DecalageNTP}$$

$$T^{Acceptation} = 700\ millisecondes + 100\ millisecondes$$

$$T^{Acceptation} = 800\ millisecondes$$

Si le temps entre l'émission et la réception d'un paquet est supérieur à 800 millisecondes, ce paquet est refusé et le prochain est pris en considération. Autrement, il est accepté et mis en mémoire temporaire afin d'être décodé et affiché dans la session de vidéoconférence de l'utilisateur.

Le module de surveillance de VAGABOND se trouve du côté du client et est chargé de fournir des informations concernant le nombre total de paquets vidéo reçus, ceux non retenus et ceux retenus. Il calcule séparément ces informations pour les paquets vidéo et audio. Notons que les paquets audio ne seront pas abordés dans cette thèse car ils sont en général de petite taille et ne constituent pas la principale cause de goulots d'étranglement. Les paquets vidéo sont quant à eux la principale cause de goulots d'étranglement en raison de leur taille. De plus, le fait d'ignorer certains paquets vidéo peut amener une session de vidéoconférence à avoir un taux d'images par seconde (*Frame rate*) plus faible et donc moins fluide, mais elle sera toujours utilisable.

Par exemple, ignorer 50% des paquets audio reçus rend une session de vidéoconférence inintelligible. En revanche, ignorer 50% des paquets vidéo reçus entraîne une dégradation de la fluidité de la vidéo mais la vidéoconférence sera toujours utilisable. Les informations sur la réception des paquets vidéo sont prises en compte toutes les 10 secondes et une probabilité de déclencher une adaptation est calculée pour les 3 prochaines secondes.

Notons : la période de 10 secondes a été choisie car elle nous permet d'avoir suffisamment d'échantillons et la période de 3 secondes nous permet d'être réactive le plus rapidement possible. Mais il s'agit de données empiriques et ces deux périodes seront affinées au fur et à mesure de l'utilisation réelle de VAGABOND lors de son déploiement. Les sections suivantes présentent les stratégies qui sont utilisées dans VAGABOND pour adapter les flux aux bandes passantes disponibles.

4.5/ ADAPTATION DES FLUX ÉCHANGÉS

Les proxies d'adaptation présents dans l'architecture assurent les échanges des flux entre les différents clients présents dans une session. Ils jouent le même rôle que les dispositifs appelés *Selective Forwarding Unit (SFU)* (cf figure 4.5 de la section 4.3). Chaque participant dans une session se connecte à un proxy d'adaptation qui agit comme une SFU. Le proxy d'adaptation reçoit les données multimédia de chaque participant et les transfère aux autres participants de la session.

Comme, il s'agit d'un fonctionnement classique d'une SFU, les proxies d'adaptation détiennent les informations de chaque participant, comme la taille maximale de leur écran, et peuvent ainsi modifier les flux qui transitent en direction de chaque participant.

Dans l'architecture de VAGABOND, nous modifions le format de chaque image encodée afin qu'elle respecte la résolution maximale d'un écran sur lequel elle est destinée à être affichée. Cette translation est effectuée uniquement si la taille de l'écran est inférieure à un facteur d'au moins deux de l'image d'origine : ceci étant fait pour économiser de la bande passante montante au niveau du serveur. Malheureusement, ce fonctionnement présente quelques désavantages au niveau de la consommation CPU et n'est utilisé que lorsqu'il s'agit de dégrader une résolution vidéo. Le flux entrant est décodé puis re-encodé avant d'être envoyé au participant distant. La figure 4.11 illustre ce mode de fontionnement.

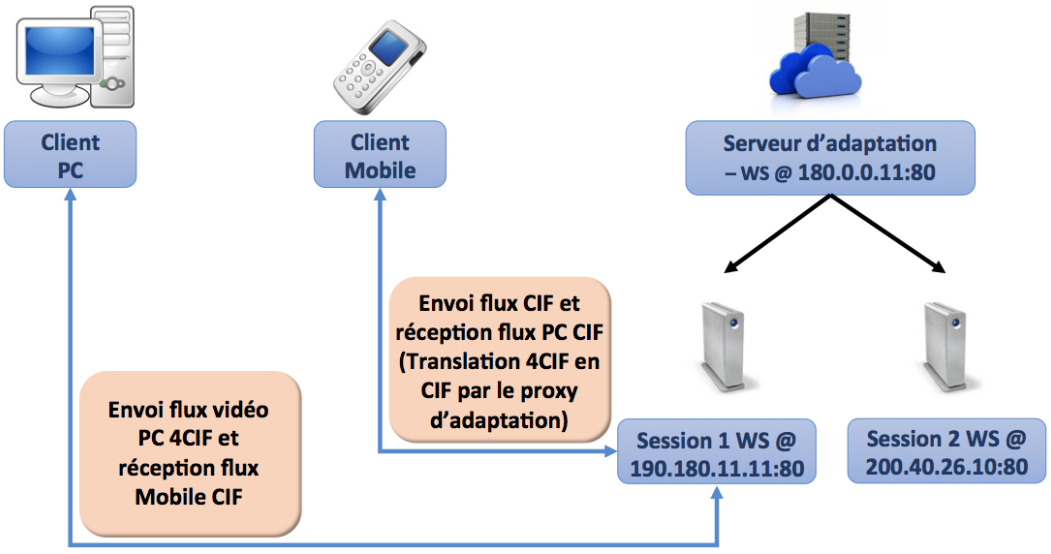


FIGURE 4.11 – Changement de la résolution d'un flux par le proxy d'adaptation

Il existe des dispositifs de ce type qui effectuent le décodage de flux et l'encodage. Ils sont appelés des *Multiple Control Unit (MCU)*. Dans une topologie de contrôle multipoint, chaque participant dans une session initie une connexion avec un serveur qui agit comme un dispositif MCU. Ce dernier reçoit les données multimédia de chaque participant et les décode, mélangeant l'audio et la vidéo de tous les participants dans un flux unique qui est ensuite encodé et envoyé à chaque participant. Les données multimédia peuvent être mélangées avec des spécificités individuelles pour chaque participant, comme par exemple pour un flux audio pour lequel une personne ne souhaite pas entendre son propre flux, ou alors mélangées pour la session entière, comme c'est souvent le cas

pour la vidéo. Une topologie de contrôle multipoint a l'avantage d'être évolutive avec un minimum de charge de traitement pour les clients. Qu'il n'y ait qu'un seul participant ou quelques dizaines de participants, chaque client initiera une seule et unique connexion bidirectionnelle avec le serveur. Les clients n'ont qu'à encoder une fois et décoder une fois, de sorte que la connexion a des exigences de bande passante en amont et en aval relativement constantes. L'inconvénient principal de cette topologie est que les exigences du serveur en matière de consommation CPU est assez importante. Le décodage et l'encodage de chaque flux multimédia est extrêmement gourmand en utilisation CPU. Il est également intéressant de noter que le processus de décodage, de mixage et de codage introduit la latence dans l'envoi et la réception des flux multimédia. La figure 4.12 illustre la topologie d'une MCU.

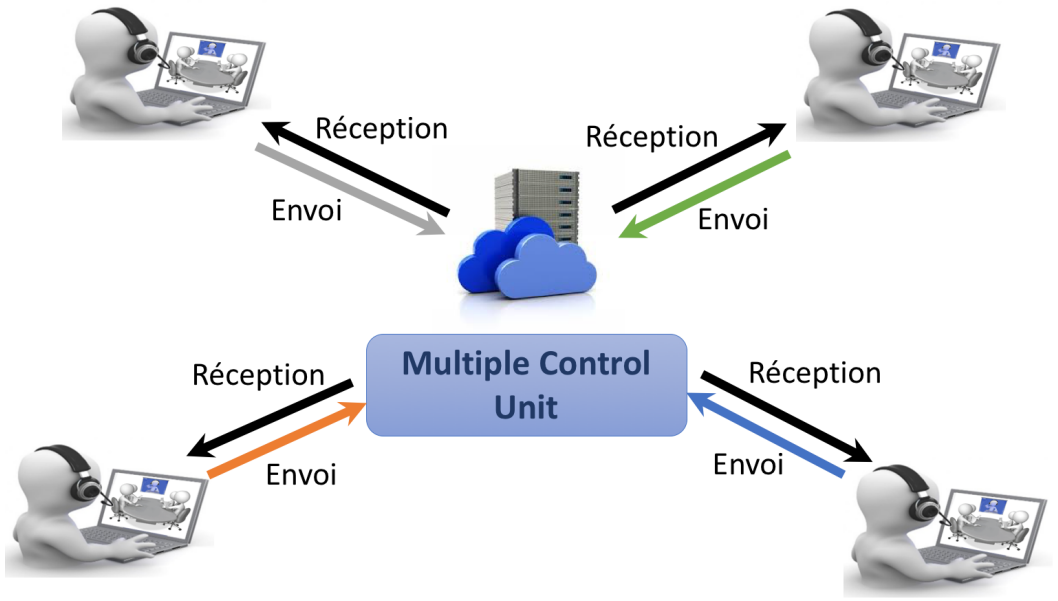


FIGURE 4.12 – Multiple Control Unit

4.5.1/ L'ENCODAGE VIDÉO H.264

L'encodage et le décodage, côté client et côté serveur sont réalisés avec la norme de codage vidéo H.264 ou MPEG-4 AVC (*Advanced Video Coding*). Il s'agit d'un projet développé conjointement par l'*UIT-T Q.6/SG16 Video Coding Experts Group* (VCEG) ainsi que l'*ISO/CEI Moving Picture Experts Group* (MPEG) et est le produit d'un effort de partenariat connu sous le nom de *Joint Video Team* (JVT). La norme H.264/AVC a été développée pour compenser la grande quantité de données à transmettre et à stoker ainsi qu'améliorer le taux de compression en comparaison avec les anciens standards de compression (MPEG1, MPEG2,. . . , H263) tout en gardant la même qualité vidéo. En effet, l'encodeur H.264/AVC permet de réduire la quantité de données de 50% tout en conservant presque la même qualité vidéo.

La compression (encodage) d'une séquence vidéo a pour but de réduire son débit et par conséquence le coût de stockage, ou rendre possible sa diffusion en minimisant la charge du réseau de transport. H.264 fonctionne selon un codage bien spécifique : un codage intra-image pour réduire les redondances spatiales au niveau de chaque image

et un codage inter-image pour éliminer les redondances temporelles entre les images successives. Ceci a pour effet de réduire la taille de la séquence de vidéo. Avec la norme H.264, la première image est obligatoirement codée en codage intra-image. Pour les autres images, les codages inter-image ou intra-image sont utilisés. Le flux compressé (appelé le *coded bitstream*) est le fruit d'une génération d'images traitées (appelées des *frames*). Différents types de profils sont possibles avec l'encodeur. Le plus couramment utilisé est le *baseline profile*.

Avec le *baseline profile*, le format de l'image utilisé est YUV4:2:0. YUV est un espace de couleur prenant en compte la perception de la couleur chez l'humain. Le YUV4:2:0 utilise 4 pixels pour la luminance Y (*Grey scale projection*), un pixel pour la chrominance U (*Blue projection*) et un autre pixel pour la chrominance V (*Red projection*). Puisque le système de vision humaine présente une sensibilité moindre à la couleur qu'à la luminosité, la complexité d'encodage est réduite sans trop affecter la qualité visuelle. Nous ne rentrerons pas davantage dans les détails concernant la disposition des pixels encodés avec le format YUV4:2:0 qui se trouve dans la norme H.264. Les étapes d'encodage consistent à :

- Division en MacroBlocs : Il s'agit de décomposer l'image en blocs appelés MacroBloc. Chaque image d'une séquence vidéo est partitionnée en MacroBlocs de taille 16×16 pixels pour la composante luminance (appelé Y) et de 8×8 pixels pour la chrominance rouge (appelé Cr) et la chrominance bleue (appelé Cb). L'encodage se fait MacroBloc par MacroBloc jusqu'à terminer tous les MacroBlocs de l'image.

- Intra-prédiction : elle est utilisée pour éliminer les redondances spatiales dans une image vidéo. Elle exploite la corrélation spatiale entre les MacroBlocs adjacents dans l'image qui tendent à avoir des propriétés semblables. On peut prévoir le MacroBloc d'intérêt à partir des MacroBlocs voisins (typiquement ceux situés en dessus et à gauche du MacroBloc d'intérêt, puisque ces MacroBlocs auraient été déjà codés). La différence entre le MacroBloc réel et sa prédiction, désignée par le résiduel est alors codée. Il existe trois types d'intra-prédiction :

  - Intra-prédiction pour la luminance 16x16 (appelée Intra16x16) est une prédiction uniforme ; appliquée pour l'ensemble du MacroBloc (16x16). Elle est recommandée dans les cas des zones régulières qui ne contiennent pas beaucoup de détails et qui sont caractérisées par une faible texture.

  - Intra-prédiction pour la luminance 4x4 (appelée Intra4x4) est généralement appliquée dans les zones d'images à haute texture par rapport au 16x16 où il y a beaucoup de détails. Elle est appliquée à la luminance Y afin d'affiner la prédiction et d'obtenir un taux de compression plus élevé avec un bonne qualité vidéo. L'intra 4x4 consiste à décomposer le MacroBloc 16x16 en 16 blocs de taille 4x4 et par la suite faire la prédiction de chaque bloc 4x4 à partir de ces voisins qui ont été précédemment codés suivant un ordre bien déterminé que l'on appelle "ordre conventionnel" présenté dans la figure 4.13.

  - Intra-prédiction pour la chrominance 8x8 (que ce soit rouge ou bleue) est semblable à la prédiction de la luminance intra 16x16. La taille du MacroBloc est 8x8 au lieu de 16x16 et les 4 modes de prédiction intra 16x16 sont appliqués

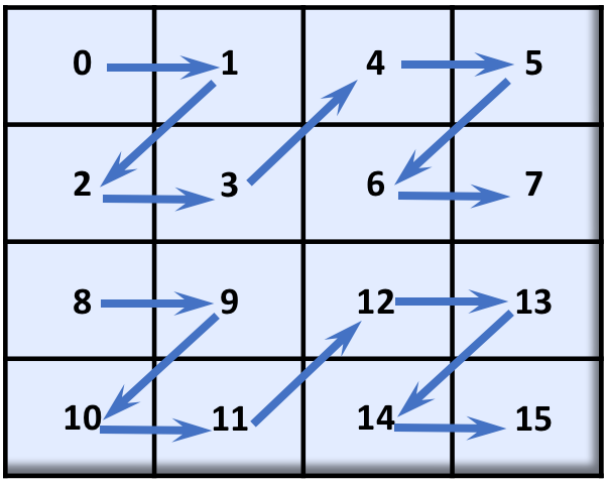


FIGURE 4.13 – Ordre de parcours conventionnel de l'intra 4x4

de la même manière pour les chrominances 8x8. Chaque MacroBloc 8x8 de la chrominance est prédit à partir des pixels voisins hauts et/ou gauches déjà codés et reconstruits. Les deux chrominances rouge et bleue doivent avoir le même mode de prédiction. Le calcul est effectué pour l'une des deux chrominances puis le MacroBloc prédit est déterminé de la deuxième composante selon le mode sélectionné pour la première.

- Inter-prédiction : elle est basée sur l'estimation et la compensation de mouvement afin de réduire les redondances temporelles qui existent entre les images successives. En effet, l'estimation de mouvement consiste à rechercher les parties identiques de l'image courante dans les images précédemment codées et à ne coder que les vecteurs de mouvement de ces parties ainsi que leurs différences. Ceci assure une réduction du débit d'une façon très importante par rapport au codage intra avec une bonne qualité d'image.
  H.264 possède plusieurs algorithmes d'estimation de mouvement dont le plus utilisé est le *Block Matching Algorithm BMA*, c'est-à-dire l'algorithme de correspondance des blocs. Pour chaque MacroBloc de l'image courante, l'algorithme cherche le MacroBloc qui lui correspond le plus dans les images précédemment encodées, dites des "images de références (*keyframes en anglais*)" et plus spécialement dans une zone bien spécifique appelée "fenêtre de recherche (Search window en anglais)" comme présenté dans la figure 4.14.
  La méthode BMA la plus simple est la recherche exhaustive, dite *Full Search*, qui consiste à parcourir la totalité de la fenêtre de recherche pixel par pixel. Le MacroBloc le plus similaire est celui qui fournit une distorsion minimale, dit autrement, le MacroBloc le plus similaire à l'un des MacroBlocs de l'image de référence. En dépit de l'efficacité de cette méthode, le temps d'exécution reste considérable, ce qui explique que l'on trouve dans la littérature différents algorithmes de recherches élaborés comme le *Line Diamond Parallel Search (LDPS)*, le *Three Step Search (TSS)*, le *Nearest Neighbors Search Algorithm (NSS)*, ... [Wer07].
  Un MacroBloc entier peut ne posséder qu'un seul vecteur de mouvement et ceci dans les cas des zones homogènes et uniformes. Mais il est également possible, par exemple pour les zones à hautes textures, que les sous-blocs du MacroBloc puissent posséder chacun un vecteur de mouvement propre. Ceci permet

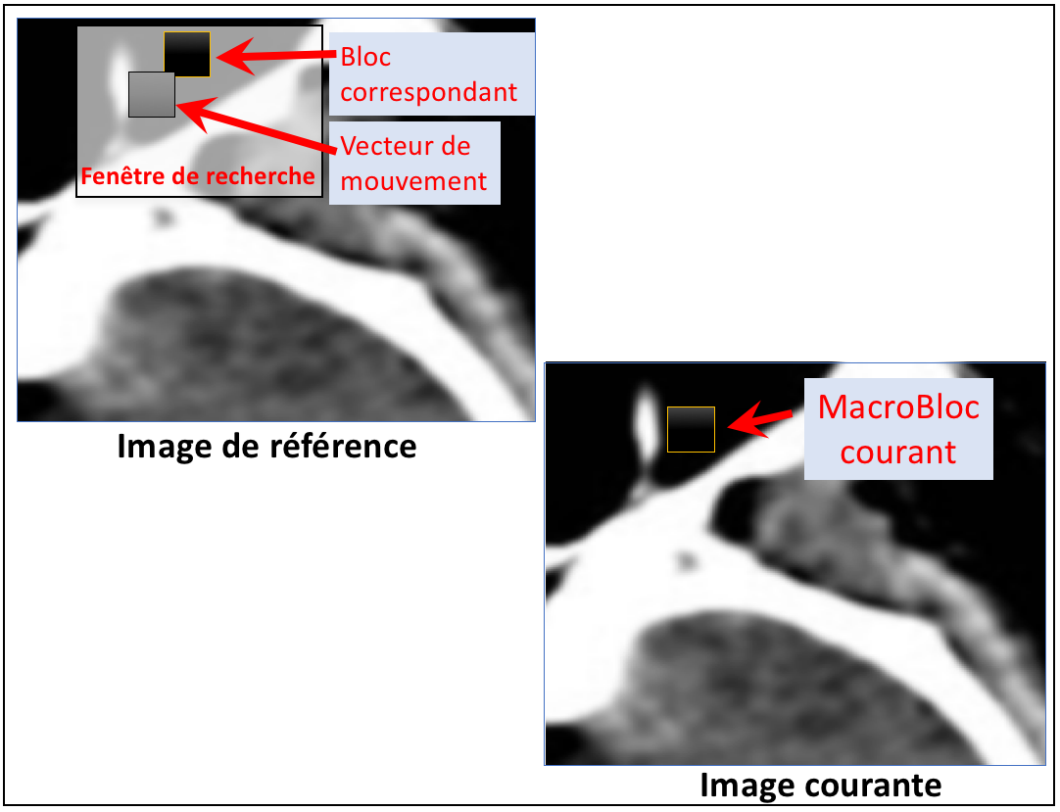


FIGURE 4.14 – Estimation du mouvement avec H.264

de générer plus de précision sur l'origine des blocs et ainsi d'améliorer la qualité vidéo et le taux de compression. Suite à l'estimation de mouvement, l'étape de compensation de mouvement s'établit. Elle consiste à déterminer le MacroBloc prédit selon les meilleurs modes d'inter-prédiction sélectionnés. Ainsi, le MacroBloc prédit est une copie des blocs de l'image de référence selon les vecteurs de mouvement calculés.

- Décision de mode : la détermination du meilleur MacroBloc prédit se fait après avoir effectué les trois types de prédictions (intra 16x16, intra 4x4 et inter) avec tous leurs modes. Le type de prédiction qui donne les résultats les plus performants sera choisi comme la meilleure prédiction et ainsi son mode sera sélectionné comme le meilleur mode de prédiction. Si c'est le mode *"inter"*, alors il faut ensuite effectuer la compensation de mouvement qui consiste à copier les pixels de la fenêtre de recherche, selon le vecteur de mouvement calculé, dans le buffer du MacroBloc prédit.

- Transformée entière : il s'agit d'une des étapes les plus importantes pour l'encodage H.264. Elle permet de passer d'une représentation spatiale (fréquences d'une image) à une représentation fréquentielle (un nouveau graphique qui représente les fréquences de l'image) d'une image. L'outil de base pour ce type de transformation est la Transformée de Fourier. Cette transformée permet de séparer les basses fréquences, qui représentent l'information utile de l'image, des hautes fréquences (moins importantes). L'information utile sera localisée dans un nombre limité de coefficients (appelé coefficient DC qui correspond à un indice

de différence entre l'image de référence, la I-Frame, et l'image courante, la P-Frame). Puisque la transformée s'applique sur le MacroBloc résiduel qui est la différence entre le MacroBloc courant et le meilleur MacroBloc prédit, les coefficients AC (correspond à un indice de différence entre les P-Frames) sont en général des coefficients nuls. Ceci permet de diminuer significativement le nombre de bits nécessaires pour la représentation de données. La nouveauté de la norme H.264/AVC au niveau de la transformée est l'utilisation d'une transformée en *Cosinus Discret TCD* modifiée. C'est une transformée entière *Integer Cosine Transform (ICT)* qui manipule seulement des données entières et elle se base seulement sur des opérations d'addition et de décalage. Par conséquent, elle permet de réduire efficacement le coût d'une implémentation matérielle.

- Quantification : après la transformée entière, la quantification s'applique sur les coefficients du MacroBloc résiduel déjà transformé. Elle consiste à diviser les coefficients transformés par un pas de quantification (*Qstep* en anglais). Ceci permet d'éliminer au maximum les hautes fréquences et d'augmenter le nombre de coefficients nuls. On améliore ainsi le taux de compression mais en contre partie cela provoque une perte de données, et donc une baisse de la qualité.

- Codage entropique : la dernière étape avant la transmission des données sur un canal est le codage des données résiduelles transformées et quantifiées en utilisant le codeur entropique. La norme H.264/AVC présente plusieurs codeurs entropiques :

  - *Context-Adaptive Binary Arithmetic Coding (CABAC)* : il s'agit d'un codage arithmétique. C'est une technique sophistiquée de codage entropique qui produit d'excellents résultats en termes de compression mais possède une grande complexité.

  - *Context-Adaptive Huffman variable-length coding (CAVLC)* : il s'agit d'un codage adaptatif de type Huffman variable, qui est une alternative moins complexe que CABAC pour le codage des tables de coefficients de transformation. Bien que moins complexe que CABAC, CAVLC est plus élaboré et plus efficace que les méthodes habituellement utilisées jusqu'à présent pour coder les coefficients. En effet, en exploitant les probabilités d'occurrence de chaque symbole ou séquence de symboles à émettre, on peut leur associer un mot binaire d'une longueur d'autant plus courte que leur occurrence. Ce qui en résulte d'une compression plus efficace.

- Encapsulation avant l'envoi sur le réseau : après le codage entropique, l'encapsulation est nécessaire avant l'envoi sur un réseau quelconque. La couche *Network Abstraction Layer (NAL)* est chargée d'organiser le flux binaire dans des unités (NALU) afin d'assurer l'intégration et le transport de flux binaire (le *bistream*) sur divers types de réseaux.

4.5.2/ LE DÉCODAGE VIDÉO H.264

Il s'agit de la quantification inverse de la transformée ICT (*Integer Cosine Transform*) inverse. Le décodage a pour but de reconstruire les MacroBlocs codés. Ainsi, après ces

deux étapes, le MacroBloc résiduel est calculé et par la suite additionné avec le meilleur MacroBloc prédit pour obtenir le MacroBloc reconstruit. Ce dernier sera exploité dans la prédiction de MacroBlocs suivant de la même image (les pixels voisins) ainsi que pour les MacroBlocs des images suivantes (fenêtre de recherche dans l'image de référence).

4.5.3/ LES TYPES D'IMAGES DE LA NORME H.264/AVC BASELINE PROFILE

La norme H.264/AVC intègre plusieurs profils de codage qui diffèrent par les outils, les algorithmes ainsi que les options de codage appliquées. Chaque profil est destiné à une application vidéo bien déterminée. On trouve le *Baseline profile* pour les applications mobiles et vidéoconférences, le *Main profile* pour les applications grand public, le *Extended profile* pour la diffusion en streaming et le *High profile* pour les applications TV de haute définition et le stockage. Dans cette thèse, nous nous intéressons uniquement au *Baseline profile*. De ce fait, pour ce type de profil, nous obtiendrons deux types d'images le *I-frame* (Intra-frame) et le *P-frame* (Predicted-frame) comme indiqué par la figure 4.15.

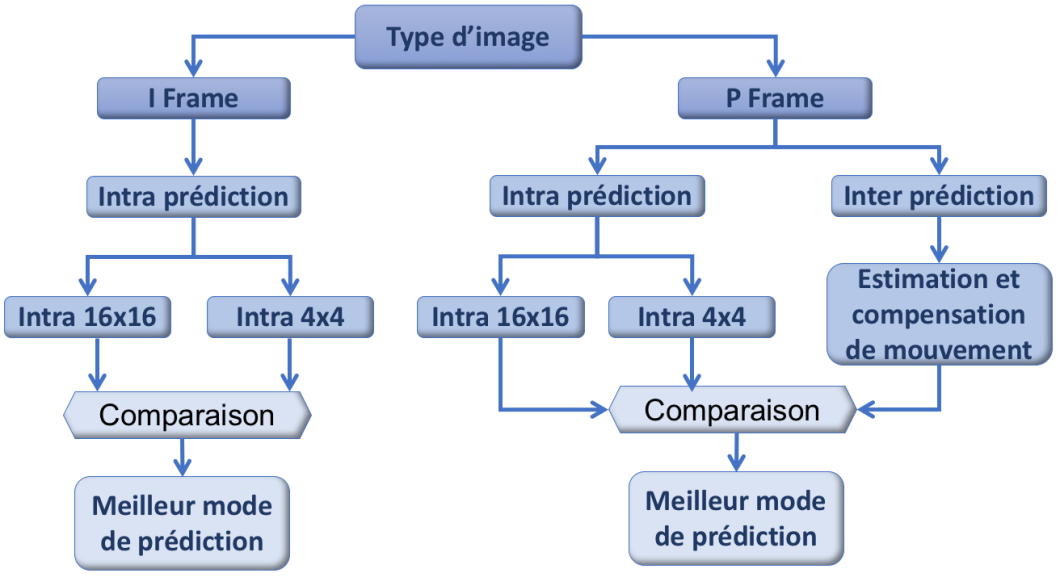


FIGURE 4.15 – Les types d'images pour la norme H.264/AVC Baseline Profile

Au niveau du type d'images *I-frame*, seule l'intra-prédiction est appliquée avec ces deux types de prédiction (intra16x16 et intra4x4). Le meilleur mode qui engendre le minimum de distorsion sera sélectionné. Les images *Intra* (qui sont aussi appelées des *keyframes*) sont généralement utilisées pour rafraichir la scène vidéo quand il y a, par exemple, des changements de plan. Un encodeur peut également les fournir à la demande. Avec les *P-Frame*, les deux types de prédiction sont appliqués : l'intra-prédiction et l'inter-prédiction qui est basée sur l'estimation de mouvement. Une comparaison est effectuée entre les 3 modes de prédiction selon leur calcul de dérivation par rapport à l'image d'origine. Le type de prédiction, qui induit une distorsion minimale, sera sélectionné.

4.5.4/ La décomposition hiérarchique d'une vidéo

Avec la norme H.264/AVC Baseline profile, une séquence vidéo est en réalité une succession de groupes d'images (*Group Of Pictures GOP*, en anglais) comme l'indique la figure 4.16. Chaque *GOP* est un ensemble d'images (les *frames*). La première image de chaque *GOP* est forcément une image Intra, un *I-Frame* (aussi appelée un *keyframe*). Les suivantes sont des images de type Inter, des P-Frames (*Predicted Frames*). La taille du *GOP* est variable et peut prendre par exemple la valeur 8, avec 1 *I-Frame* et 7 *P-Frames* successives.

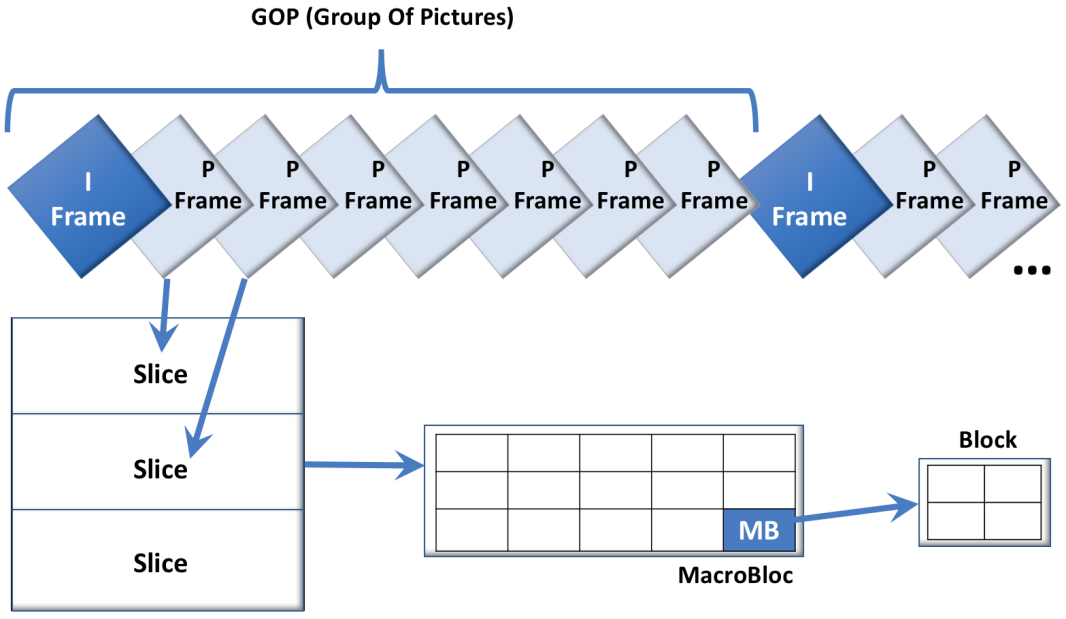


FIGURE 4.16 – La décomposition hiérarchique pour la norme H.264/AVC Baseline Profile

Les images peuvent être également découpées en tranches (*slices* en anglais) indépendamment les unes des autres. Ainsi, les MacroBlocs de la première ligne de chaque slice ne tiennent pas compte des voisinages en haut. La prédiction de ces MacroBlocs se fait seulement en exploitant les pixels voisins du MacroBloc à gauche. Une *slice* peut comporter une ou plusieurs lignes de MacroBlocs. Chaque ligne comporte un nombre N de MacroBlocs de taille 16x16 selon la largeur de l'image. Ces MacroBlocs sont aussi découpés en blocs de taille 4x4 utilisé au niveau de l'intra4x4, transformée ICT, ...

4.6/ Détection de l'état du réseau

L'étude de l'état d'un réseau est un élément essentiel pour toute application collaborative et communicante. Lorsqu'il s'agit de transmettre des données d'une vidéoconférence contrainte au temps réel, le protocole qui est le plus souvent utilisé pour le transport (couche 4 du modèle OSI) est l'UDP (*User Datagram Protocol*) et celui pour la communication informatique (couche 5 du modèle OSI) est le RTP (*Real-Time Transport Protocol*). Nous ne détaillerons pas le protocole RTP dans cette thèse mais plutôt le protocole RTCP (*Real-Time Transport Control Protocol*) qui est étroitement lié au protocole RTP.

Tandis que le RTP livre les données (par exemple audio et vidéo), le RTCP est utilisé pour surveiller les statistiques de transmission et la qualité de service, il contribue aussi à la synchronisation de plusieurs services. Le RTP est généré et reçu sur des numéros de port pairs, et la communication RTCP associée, utilise le numéro de port impair le plus élevé le suivant. RTCP est basé sur des transmissions périodiques de paquets de contrôle par tous les participants dans la session. Il est un protocole de contrôle des flux RTP, permettant de véhiculer des informations basiques sur les participants d'une session, et sur la qualité de service. Le contrôle de flux RTP est réalisé en gardant une évaluation du nombre de participants à une session (sources et récepteurs). À partir de cette évaluation est calculé un intervalle de temps qui sert de période de récurrence à la diffusion des informations SR (*Sender Report*) ou RR (*Receiver Report*) suivant le cas. Globalement, les algorithmes de contrôle limitent le volume des informations de contrôle transmises (les données RTCP donc) à 5% du volume global des échanges de la session. Dans ce volume, 25% sont réservées aux informations des sources (messages SR). On garantit ainsi une possibilité de gérer des groupes de grande taille du point de vue du volume d'informations échangées. Plus le nombre de participants est élevé, moins précise est la vision de chaque participant de l'état du réseau. Les paquets qui sont le plus fréquemment transmis sont SR et RR :

- Paquets RTCP *Sender Report (SR)* : les participants à une session qui à la fois émettent et reçoivent des paquets RTP utilisent les paquets RTCP SR. Un paquet SR contient une en-tête, des informations sur l'émetteur, un certain nombre de blocs de rapports de réception et optionnellement une extension spécifique au profil. Une en-tête contient la version du protocole RTCP, un champ qui indique qu'il y a un bourrage dont la taille est indiquée dans le dernier octet, un champ qui précise le nombre de rapports de réception contenus dans le paquet SR, en considérant un rapport pour chaque source (un maximum de 31 rapports peuvent être inclus dans le paquet SR), le type de paquet (pour un paquet SR, la valeur 200 est utilisée), et la longueur totale du paquet. Les informations sur l'émetteur contiennent une identification de la source spécifique à l'émetteur, un horodatage NTP et RTP, le nombre total de paquets RTP transmis par l'émetteur depuis le début de la session, un champ qui indique le nombre total d'octets RTP (la somme des données brutes).

- Paquets RTCP *Receiver Report (RR)* : ils fournissent aux autres participants de la session l'information concernant le nombre de paquets RTP qu'ils ont émis et qui ont été reçus avec succès par le récepteur. Un paquet RR contient un champ qui précise l'identification de la source dans la session, un champ indiquant la fraction des paquets RTP perdus depuis le dernier rapport émis par ce participant (La fraction représente le rapport entre le nombre de paquets perdus et le nombre de paquets attendus). Le nombre de paquets perdus peut être déduit à partir de l'analyse du numéro de séquence (*Sequence Number*) de chaque paquet RTP reçu), un champ indiquant le nombre total de paquets RTP de la source en question qui ont été perdus depuis le début de la session RTP, un champ précisant le numéro de séquence du dernier paquet RTP reçu depuis cette source, un champ renseignant sur la variation du délai de transmission des paquets RTP, un champ représentant les secondes de l'horodatage NTP utilisé dans le dernier paquet SR reçu depuis la source, et enfin un champ représentant le délai exprimé en unités de 1/65536 secondes entre l'instant de réception du dernier paquet SR de la source

et l'instant d'émission de ce bloc RR.

Les protocoles RTP et RTCP sont adaptés pour la transmission de données temps réel avec le protocole UDP. Les protocoles RTP et RTCP sont principalement utilisés en vidéoconférence, cadre dans lequel les participants sont tour à tour, émetteurs ou récepteurs. Pour le transport de la voix, ils permettent une transmission correcte sur des réseaux bien ciblés. C'est-à-dire, des réseaux qui implémentent une qualité de service adaptée (ATM). Il est aussi possible de s'appuyer sur des réseaux bien dimensionnés (bande passante, déterminisme des couches sous-jacentes, ...), de type LAN d'entreprise. Cependant, ils fonctionnent en stratégie de bout à bout et donc ne peuvent pas contrôler l'élément principal de la communication : le réseau. Pourtant, quels que soient les efforts d'adaptation des émetteurs, ou les moyens mis en œuvre par les récepteurs, c'est au cœur du réseau que les dysfonctionnements critiques sont générés. Le protocole Internet a été volontairement pensé pour reporter l'intelligence vers les systèmes d'extrémité. C'est cette simplicité qui a conduit au succès d'Internet. Il existe d'autres protocoles (que nous ne détaillerons pas) qui ont été définis par l'IETF afin de remédier à ces dysfonctionnements et ainsi d'améliorer les transmissions temps réel, comme par exemple le protocole RSVP (*Resource Reservation Protocol*).

Dans l'architecture de VAGABOND, ces deux protocoles ne sont pas adaptés (ni au protocole UDP, ni à l'utilisation de ports spécifiques, ...) et n'ont donc pas été retenus. En revanche, comme la surveillance de l'état du réseau est un élément déclencheur de la stratégie d'adaptation (cf section 4.4) nous avons mis en place une stratégie de gestion de l'état du réseau. Rappelons que dans l'architecture, si le temps entre l'émission et la réception d'un paquet est supérieur à 800 millisecondes, ce paquet est refusé et le prochain est pris en considération. Dans le cas contraire, ce paquet est accepté et est mis en mémoire temporaire afin d'être décodé et affiché dans la session de vidéoconférence de l'utilisateur. Des données empiriques sont utilisées pour ce qui est de l'intervalle de la prise en charge des paquets retenus (chaque 10 secondes) et du temps de projection pour les calculs probabilistes qui est de 3 secondes. Lorsqu'une anomalie réseau est détectée, une adaptation niveau utilisateur est déclenchée. Ce point sera détaillé dans la section 4.7. Cette adaptation sera déclenchée uniquement si l'étude sur l'état du réseau le permet. L'étude de l'état du réseau est réalisée à l'aide de calculs probabilistes : application de la loi binomiale et application du théorème de Bayes.

4.6.1/ APPLICATION DES LOIS DE PROBABILITÉ

Le but des calculs de probabilité est de pouvoir réduire les retards des paquets. Dans l'intergiciel, tous les paquets émis seront réceptionnés mais certains peuvent être réceptionnés avec du retard. En appliquant des stratégies d'adaptation, nous réduisons ainsi ces retards. Nous considérons le systèmes sans mémoire et avec mémoire :

- Tant qu'aucune adaptation des flux multimédia n'intervient, nous pouvons considérer que la réception des paquets ne dépend pas de la réception des paquets précédents. En l'occurrence, il s'agit d'un processus sans mémoire et donc une loi binomiale peut s'appliquer.

- En revanche, dès que la probabilité de recevoir des paquets retardés atteint un certain seuil, l'adaptation entre en jeu et par conséquent, nous basculons vers un

processus avec mémoire pour lequel une loi binomiale ne peut pas s'appliquer. C'est la raison pour laquelle nous appliquons l'inférence bayésienne.

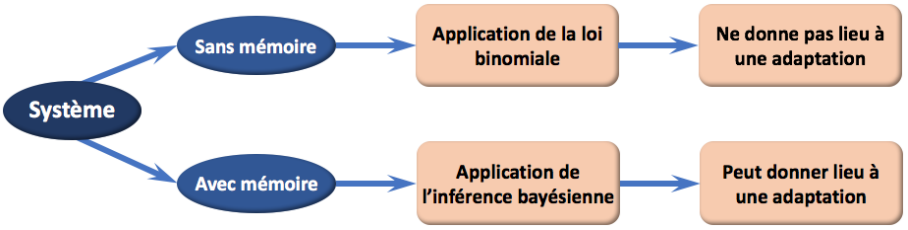


FIGURE 4.17 – Etat du système avec et sans mémoire

La figure 4.17 résume l'application de ces lois de probabilité suivant le type de système que nous étudions et le résultat d'adaptation qui en résulte.

4.6.2/ APPLICATION DE LA LOI BINOMIALE

Dans cette partie, nous proposons de mesurer l'efficience du réseau au travers d'une étude probabiliste. Ainsi, nous mesurons la probabilité qu'un certain nombre de paquets soient acceptés prochainement dans un temps donné. Soit X cet évènement et $P(X = x)$ la probabilité qu'il survienne dans notre système. Compte tenu des caractéristiques de notre système et de son contexte d'utilisation, nous formulons l'hypothèse que :

- les évènements sont indépendants les uns des autres,
- il n'existe que deux issues possibles, succès ou échec,
- les évènements interviennent de manière continue,
- le modèle compte le nombre de succès obtenus à l'issue de n épreuves.

En conséquent, comme présenté dans l'état de l'art, nous pouvons considérer que l'évènement suit une loi binomiale de paramètre $(n; p)$ et se note $B(n; p)$. Une première phase dans l'étude de l'état du réseau consiste donc à appliquer la loi binomiale sur le flux des paquets entrants. Rappelons que la loi binomiale est la suivante :

Équation 11

Loi binômiale

$$p(k) = P(X = k) = \binom{n}{k} p^k q^{n-k}$$

Où :

$$\binom{n}{k} = C_n^k = \frac{n!}{k!(n-k)!}$$

Pour illustrer l'application de cette loi sur le flux des paquets entrants, supposons que dans une séance de vidéoconférence, un expert reçoit 80 paquets vidéo dans un cycle de 10 secondes. Le module de détection d'anomalie réseau détecte que seulement 25 paquets sont retenus sur les 80 reçus. En appliquant l'inférence fréquentiste, le taux de réussite est de 0,3125 $(25/80$; et donc 0,6875 d'échec). Nous sommes en présence d'un

schéma de Bernoulli (car il n'y a que deux issues possibles, échec ou succès, ce qui est notre hypothèse de départ). La loi de Bernoulli de paramètre p associe à l'issue succès (S) la probabilité p et à l'issue échec (E) la probabilité $(1-p)$. L'équation 11 devient donc :

Équation 12

Loi binômiale

$$p(k) = P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

De plus, on appelle schéma de Bernoulli, la répétition n fois, de manière indépendante, à une épreuve de Bernoulli. La variable aléatoire X prenant pour valeurs le nombre de succès suit la loi binomiale avec les paramètres $n = 80$ et $p = 0.3125$.

Nous cherchons à calculer une probabilité sur les 3 prochaines secondes, nous calculons ainsi la moyenne de paquets que nous attendons sur les 3 prochaines secondes. Comme nous avons reçu 80 paquets en 10 secondes, en 3 secondes nous nous attendons à recevoir 24 paquets acceptés ($\frac{80}{10} \times 3 secondes$). La probabilité d'avoir moins de 24 succès dans les 80 essais est donnée par la fonction de distribution cumulée :

Fonction de distribution cumulée :

$$P(X \leq 24) = \sum_{i=0}^{24} P(X = i)$$

$$P(X \leq 24) = \sum_{i=0}^{24} C_{80}^{i} \,.\, 0.3125^{i} \,.\, (1 - 0.3125)^{80-i}$$

$$P(X \leq 24) = 0.4580$$

Nous avons donc, pour ce cas de figure, une probabilité de 0,4580 de déclencher une adaptation au niveau utilisateur (changement de stratégie du profil utilisateur). Le changement de stratégie du profil utilisateur ne se fera donc pas (car la probabilité est inférieure à 0,5) et l'expert médical (l'utilisateur) rencontrera une dégradation de la vidéo mais la vidéoconférence sera toujours utilisable.

Supposons, dans ce deuxième exemple, que sur 104 paquets vidéo, seulement 31 paquets sont retenus dans le prochain cycle de 10 secondes. L'inférence fréquentiste donne un taux de succès de 0,2981 (donc de 0,7019 d'échec). Par conséquent, le calcul de la loi binomiale cumulée sur les 3 prochaines secondes nous donne le résultat suivant de déclencher une adaptation :

$$P(X \leq 31) = \sum_{i=0}^{31} P(X = i)$$

$$P(X \leq 31) = \sum_{i=0}^{31} C_{104}^{i} \,.\, 0.2981^{i} \,.\, (1 - 0,2981)^{104-i}$$

$$P(X \leq 31) = 0.5481$$

Dans l'architecture de VAGABOND, lorsque la distribution binomiale cumulée donne un résultat de plus de 0,5 (signifiant qu'une adaptation est requise), l'inférence bayésienne de la proportion binomiale est utilisée [Mea16, Etz15]. Nous basculons alors vers un système avec mémoire dans lequel ce type de processus peut donner lieu à une adaptation.

4.6.3/ APPLICATION DE L'INFÉRENCE BAYÉSIENNE

L'idée avec le théorème de Bayes est de calculer une probabilité postérieure basée sur une probabilité précédente et une probabilité courante.

La distribution préalable, qui intègre nos croyances subjectives (l'*a priori*), est basée sur le paramètre d'intérêt qui est, dans notre cas, le nombre de paquets retenus. La probabilité courante est la proportion réelle de paquets retenus sur tous les paquets reçus (fonction de vraisemblance de Bernoulli). La base de toutes les statistiques bayésiennes est le théorème de Bayes (voir section 2.2.3).

Comme vu précédemment, dans notre cas, la probabilité courante suit une loi binomiale. Si la distribution antérieure et la distribution postérieure sont dans le même ensemble (ensemble des nombres rationnels, $\mathbb{Q}$), les propriétés antérieure et postérieure sont appelées distributions conjuguées. Nous utilisons la distribution bêta qui est un conjuguée antérieure car le postérieur est également une distribution bêta. Le choix de la distribution bêta vient du fait qu'il s'agit dans notre cas, d'un évènement continu et défini sur l'intervalle $[0, 1]$. La distribution bêta possède deux paramètres connus sous le nom de α et β, ce qui pondère la relation entre le succès et l'échec de la transmission des paquets. Dans notre cas, nous considérons que α est le nombre de paquets retenus et β est le nombre de paquets refusés. Cette pondération nous permet de choisir comment créer notre modèle autour de l'inférence bayésienne. La distribution bêta est de la famille des conjuguées pour la probabilité binomiale. Notre système suit toutes ces hypothèses, ainsi nous pouvons appliquer une inférence bayésienne en utilisant une distribution bêta.

Ainsi, la fonction de densité de probabilité de la distribution bêta que nous retenons est donnée par l'équation 13 :

Équation 13

Distribution bêta :

$$P(\theta|\alpha,\beta) = \frac{\theta^{\alpha-1} \cdot (1-\theta)^{\beta-1}}{B(\alpha,\beta)}$$

Dans cette équation le terme au dénominateur, $B(\alpha,\beta)$, est présent pour agir comme une constante de normalisation et nous donne la distribution de croyance antérieure. En utilisant la distribution bêta, il suffit de spécifier deux paramètres, α et β. Ces deux paramètres correspondent respectivement à la moyenne et à la variance de la distribution bêta. L'application de ces lois dans nos algorithmes de prise de décision nous donne ainsi pour la moyenne des paquets vidéo précédemment retenus (à une étape $T-1$) :

Équation 14

Moyenne antérieure de la distribution bêta :

$$\overline{\pi}_{Anterieur} = \frac{\alpha}{\beta}$$

Où :

- α est le nombre de paquets vidéo antérieurement retenus,
- β est le nombre de paquets vidéo antérieurement refusés.

Et la variance antérieure (*Pre-variance*) est ainsi donnée par :

Équation 15

$$\widehat{\mathrm{Var}}(PrV)_{Anterieur} = \sqrt{\frac{\alpha\beta}{(\alpha+\beta)^2 \ . \ (\alpha+\beta+1)}}$$

Comme notre but est de calculer une probabilité postérieure en prenant en compte une probabilité antérieure et une probabilité courante, la distribution bêta postérieure est :

Équation 16

$$Beta(\gamma + \alpha, N - \gamma + \beta)$$

Où :

- N est le nombre de paquets vidéo actuellement reçus,
- γ est le nombre de paquets vidéo actuellement retenus.

La moyenne postérieure devient donc :

Équation 17

$$\overline{\pi}_{Posterior} = \frac{\gamma + \alpha}{N + \alpha + \beta}$$

Et ainsi la variance postérieure *Post-variance* est :

Équation 18

$$\widehat{\mathrm{Var}}(PoV)_{Posterior} = \sqrt{\frac{(\gamma+\alpha) \ . \ (N-\gamma+\beta)}{(N+\alpha+\beta)^2 \ . \ (N+\alpha+\beta+1)}}$$

Pour illustrer l'application que nous avons faite de l'inférence bayésienne, prenons comme exemple la probabilité binomiale cumulée obtenue dans le deuxième exemple d'application de la loi binomiale dans la section 4.6.2. Rappelons que la loi binomiale avait prédit une probabilité de 0.5481 de déclencher une adaptation à un temps t. Si nous avions pris en compte uniquement ce résultat, nous aurions déclenché une adaptation. Admettons qu'au temps $t-1$, la moyenne était calculée à 0,7901 et l'écart type à 0,045 pour 64 paquets vidéo retenus sur les 81 reçus. Ceci nous donne une distribution de la croyance antérieure de $Beta(\omega|64,17)$. Au temps t, nous observons 31 paquets retenus sur les 104 reçus. Notre distribution bêta postérieure devient :

$$Beta(\omega|\theta+\alpha, N-\theta+\beta) = Beta(\omega|31+64, 104-31+17)$$
$$Beta(\omega|\theta+\alpha, N-\theta+\beta) = Beta(\omega|95,90)$$

Nous pouvons maintenant calculer la moyenne et l'écart type de la probabilité postérieure afin de reproduire des estimations de probabilité (une tendance) sur la réception du prochain ensemble de paquets vidéo dans les 10 prochaines secondes. En particulier, la valeur de $\overline{\pi}_{Posterior}$ est de 0,5135 en utilisant l'équation 17 tandis que l'écart type $\widehat{\mathrm{Var}}(PoV)_{Posterior}$ est de 0,0366 en utilisant l'équation 18. Une moyenne de $\omega = 0.5135$ exprime qu'environ 51,35% des paquets reçus seront retenus dans les prochaines 10 secondes alors que l'écart type de 0,0366 signifie que nous sommes incertains à propos de cette augmentation de 51,35% de 3,66%.

4.7/ ADAPTATION AU PROFIL UTILISATEUR

L'intergiciel VAGABOND a été conçu pour prendre en compte les préférences utilisateur. Il s'agit de préférences dans le cadre d'une vidéoconférence pour les professionnels de santé. Mais ces préférences peuvent être créées pour d'autres domaines d'applications. L'application de ces préférences est une stratégie d'adaptation. Cette stratégie étant définie au préalable, les mécanismes d'aide à la décision définis précédemment nous aident à appliquer la stratégie uniquement lorsque cela s'avère nécessaire. Par exemple, dans une session de vidéoconférence, un utilisateur peut avoir le rôle de demandeur de la session ou bien avoir le rôle d'expert sur site aux côtés du patient.

S'il est le demandeur de la session, cela implique qu'il est nécessaire de lui fournir différents clichés du patient, en revanche sa vidéo n'est pas importante dans cet échange. La communication entre le demandeur et l'expert priorisera le canal audio. S'il est expert il faudra fournir une adaptation permettant de maintenir audio et vidéo.

Dans l'architecture de VAGABOND, une base de données regroupant les besoins des différents professionnels de santé est présente. Nous avons regroupé dans cette base de données les professionnels de santé qui sont les plus à même d'utiliser VAGABOND. Cette base de données pourra être enrichie au fur et à mesure de l'utilisation. Le tableau 4.1 représente quelques uns des besoins en vidéoconférence des professionnels de santé.

Où :

- COA représente une communication orientée audio,
- COV représente une communication orientée vidéo,

Spécialité	Demandeur	Expert
Neurologie	COA	COA, COV, GVS, GAS
Psychiatrie	COA, COV	COA, COV, GVS, GAS
Radiologie	COA	COA, COV
Néphrologie	COA	COA, GAS
Cardiologie	COA	COA, COV
Anatomopathologie	COA	COA
Gériatrie	COA	COA, COV, GVS, GAS
Neuropsychologie	COA, COV	COA, COV, GVS, GAS
Neuropsychiatrie	COA, COV	COA, COV, GAS
Dermatologie	COA	COA, COV, GVS
Infirmerie	COA	COA, COV

TABLE 4.1 – Les exigences des professionnels de santé dans une session de vidéoconférence

- GVS représente des gestes visuels spécifiques. Par exemple, un zoom sur les pupilles d'un patient, demande d'un expert à un patient d'effectuer un geste spécifique (de dessiner un objet, . . .),
- GAS représente des gestes audio spécifiques. Par exemple, d'un patient atteint d'un accident vasculaire cérébral (AVC).

Pour résumer

SYNTHÈSE

Dans ce chapitre, nous avons présenté les fonctionnaliés de l'architecture globale de l'intergiciel VAGABOND puis nous avons détaillé chacun des modules. L'adaptabilité est une notion qui s'appuie sur la capacité des systèmes à pouvoir s'adapter aux besoins des applications qu'ils doivent héberger. Avec tous les acteurs de l'environnement qui peuvent être un élément du système, il devient très difficile voire impossible de prévoir le comportement général, et donc de définir une politique d'adaptation globale.
Dans l'intergiciel de VAGABOND, l'accent est mis principalement sur les données contraintes au temps réel. Des mécanismes d'adaptation sont étudiés pour ces types d'échanges ainsi notre intergiciel a été étudié afin de pouvoir être déployé spécialement dans le milieu hospitalier dans lequel les systèmes de sécurité sont extrêmement exigeants.
Le système dispose d'un serveur principal unique. Ce serveur est utilisé comme un registre et gère la connexion des clients. À chaque nouvelle connexion d'un utilisateur au système, une demande est faite auprès du serveur principal, l'*Adaptation Server*, qui est accessible sur le port 80 ou 443 dans les établissements de santé français. Le serveur d'adaptation dispose de plusieurs sous-serveurs, appelés des *Adaptation Proxies*. Ces proxies d'adaptation sont des serveurs qui agissent directement sur les données et qui sont rattachés à l'*Adaptation Server*. Le choix de l'attribution d'un *Adaptation Proxy* par l'*Adaptation Server* est fait par rapport à l'étude de la bande passante disponible par un client par le biais d'un serveur IPerf. Ce serveur IPerf se trouve sur l'*Adaptation Server*. Le proxy d'adaptation est chargé du routage vidéo dans un système de vidéoconférence. Il est capable de recevoir de multiples flux multimédia et de décider quel flux doit être envoyé et à quel participant. L'architecture de VAGABOND a été pensée pour surveiller l'état du réseau et réagir proactivement en conséquence. Cette détection se fait tout au long d'une session de vidéoconférence et permet de surveiller la bande passante durant toute la durée de cette dernière. Si le temps entre l'émission et la réception d'un paquet est supérieur à 800 millisecondes, ce paquet est refusé et le prochain est pris en considération. Dans le cas contraire, il est accepté et mis en mémoire temporaire afin d'être décodé et affiché dans la session de vidéoconférence de l'utilisateur. Sur ces données est effectuée une étude statistique de l'état de la bande passante descendante. La loi binômiale et l'inférence bayésienne sur une distribution binomiale sont utilisées pour déclencher des adaptations de profils utilisateurs. Ainsi, nous souhaitons être plus tolérants aux fortes variations de la bande passante d'un réseau. Avec une précision plus fine et grâce à ces lois de probabilité, l'adaptation du profil utilisateur n'est déclenchée que lorsque des congestions réseau sévères surviennent. Enfin, TCP étant un protocole de transport fiable et en mode connecté, nous avons eu besoin de concevoir et d'utiliser de nouvelles stratégies d'adaptation intelligentes avec la transmission de données afin de faire face aux problèmes de latence et à la temporisation des sockets.

Le prochain chapitre présente la validation et les résultats obtenus sur les premières expérimentations de l'intergiciel VAGABOND. Ce chapitre présentera les résultats de nos expérimentations pour évaluer les performances de la plateforme.

5

VALIDATION ET RÉSULTATS DES IMPLÉMENTATIONS

Dans ce chapitre, nous exposons les premiers résultats de nos implémentations, ainsi que les résultats obtenus au cours des expérimentations de l'intergiciel VAGABOND.

Tout d'abord, la première section de ce chapitre présente l'application Covotem^{TM} , et plus particulièrement le module de vidéoconférence. Ce module a été complètement modifié afin d'intégrer l'intergiciel VAGABOND. Nous expliquons son fonctionnement et les différentes interfaces existantes et créées.

Puis la section suivante expose les résultats des tests que nous avons effectués dans un espace de production en milieu hospitalier et dans des réseaux locaux restreints (seuls les ports 80 et 443 sont accessibles en TCP) avec un accès à Internet. Nous avons pu déduire de ces tests les valeurs empiriques qui sont utilisées pour le déclenchement automatique des stratégies d'adaptation. Comme nous avons pu le voir dans le chapitre précédent, ces déclenchements se produisent à la suite des études probabilistes conduites sur l'acceptation des paquets vidéo issus d'une session de vidéoconférence.

5.1/ PRÉSENTATION GRAPHIQUE DES IMPLÉMENTATIONS DANS LA PLATEFORME COVOTEM^{TM}

Cette section présente les interfaces graphiques (existantes sur Covotem^{TM} ou nouvellement créées au cours de nos travaux) que les professionnels de santé sont amenés à utiliser lors d'une séance de vidéoconférence.

5.1.1/ CONNEXION D'UN UTILISATEUR ET CONFIGURATION DU PROFILE

L'application est téléchargée automatiquement depuis son serveur et se lance. L'authentification la plus courante est le couple $< identifiant, motdepasse >$. Sur cette fenêtre, un utilisateur doit renseigner un identifiant, un mot de passe, et son espace de collaboration. Ce dernier correspond à la spécialité du professionnel de santé (dermatologie, gériatrie, AVC, ...). L'application possède plusieurs types d'authentification (authentification avec une carte de professionnel de santé (CPS), un mot de passe à usage unique, un code SMS à usage unique, ...) qui sont disponibles ou non en fonction des options

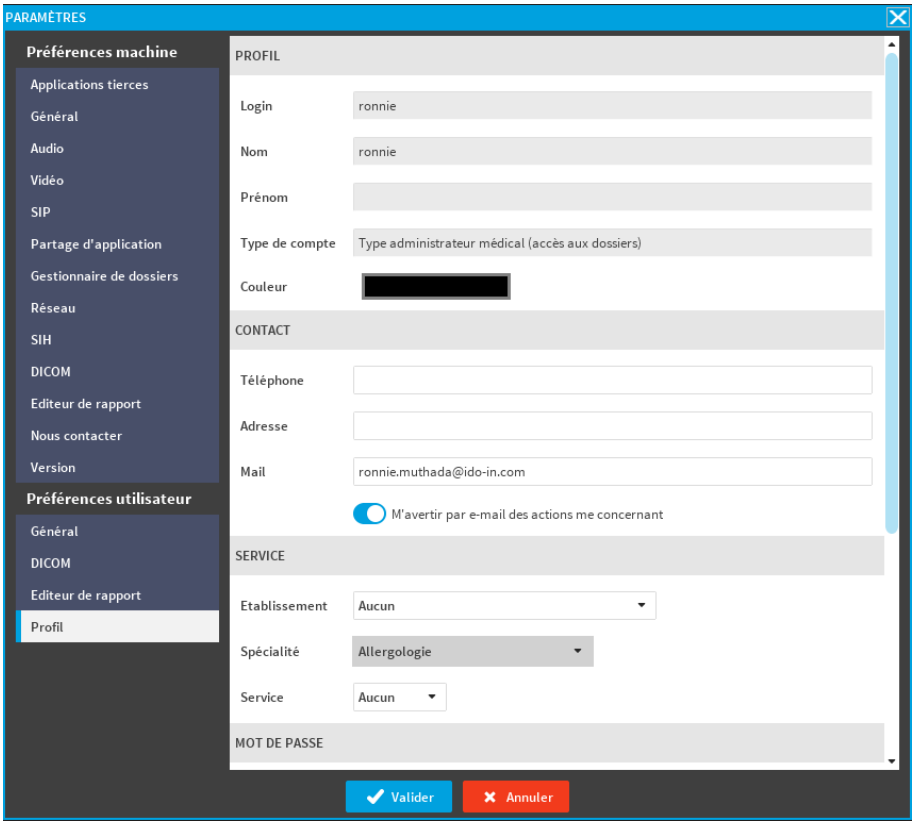


FIGURE 5.1 – Profil d'un utilisateur (nouvelle interface)

souscrites. Dans le cadre de nos travaux, nous utilisons uniquement le mode d'authentification $< identifiant, motdepasse >$.

La spécialité d'un professionnel de santé est définie dans son profil (cf figure 5.1). lors de la création de son compte qui est configuré par un administrateur du système. Cette spécialité est utilisée lors de la mise en œuvre des différentes techniques d'adaptation proactives dans une session de vidéoconférence.

5.1.2/ ENTRÉE DANS UNE SESSION DE VIDÉOCONFÉRENCE

Après une authentification réussie dans l'application, la fenêtre centrale est visible. Comme présenté dans la section 3.1.1, Covotem^{TM} possède plusieurs modules. La figure 5.2 présente la fenêtre principale de l'application et l'emplacement des différents modules accessibles depuis cette dernière.

La liste des salles de vidéoconférence est directement disponible. Avant d'entrer dans une salle de vidéoconférence, il est nécessaire de configurer ses paramètres audio et vidéo. Nous n'entrons pas dans les détails de ces configurations. À l'entrée en salle de vidéoconférence, toutes les ressources audio et vidéo initialement configurées sont disponibles. La figure 5.3 présente un utilisateur dans une salle de réunion virtuelle.

Pour les besoins de validation de notre module d'adaptation, toutes les évaluations que nous menons possèdent un flux audio et un flux vidéo. Les changements de stratégies

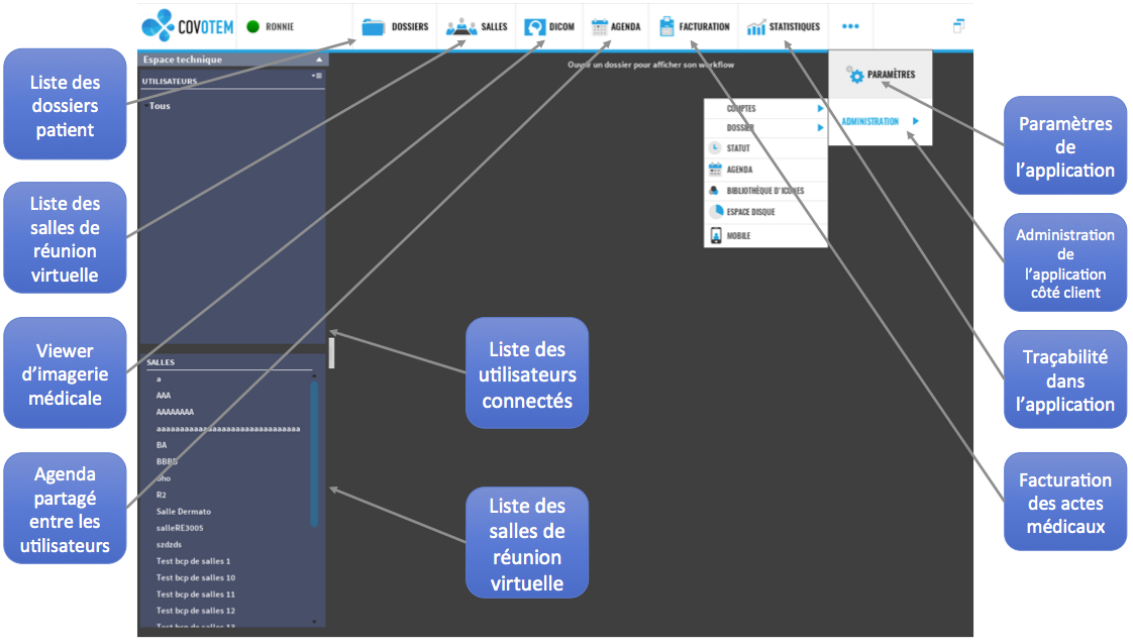


FIGURE 5.2 – Fenêtre principale de l'application

(dégradation de la qualité vidéo, désactivation de la vidéo, ...) interviennent lorsque le module d'adaptation détecte ou prédit des variations importantes de la bande passante disponible d'un client.

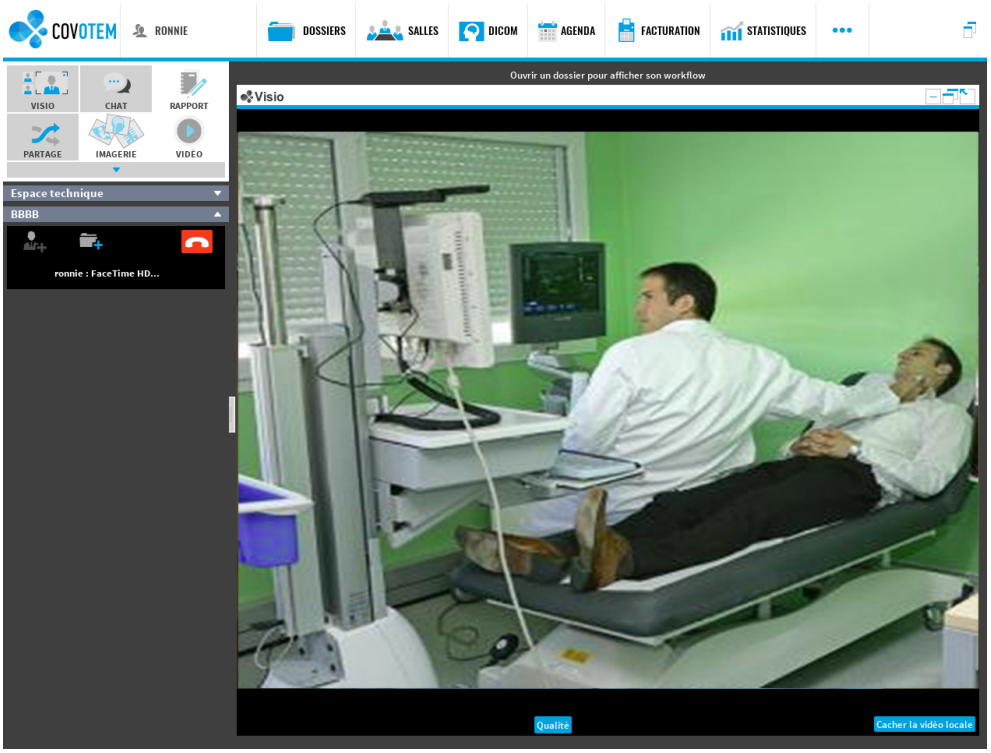


FIGURE 5.3 – Utilisateur en réunion

5.1.3/ APPLICATION D'UNE STRATÉGIE D'ADAPTATION

Le cœur de métier de l'entreprise dans laquelle j'effectue cette thèse CIFRE est le télédiagnostic médical avec comme application phare la neurologie. Dans ce domaine, les médias qui sont importants dans une séance de vidéoconférence sont l'imagerie cérébrale, la vue du patient et le canal sonor. Dans une séance de vidéoconférence de ce type, il y a d'un côté le site demandeur (côté patient) et de l'autre côté le site expert (neurologue sénior). Le site demandeur n'a pas de neurologue expert et fait donc appel au site expert. La vidéo de l'expert n'est pas primordiale et peut ne pas être transmise au site demandeur. L'audio est un élément important car l'expert doit pouvoir donner des directives au patient directement ou via les personnels de santé présents. Sur le site demandeur, l'audio et la vidéo sont deux éléments importants car l'expert doit pouvoir entendre et voir le patient. En début d'une session, l'audio et la vidéo sont disponibles pour tous les participants. La stratégie changera lorsque des anomalies sur les canaux de transmission seront détectées par le module de surveillance présent sur les clients. Le module de décision déclenchera ou non, en fonction de ses résultats, une adaptation.

Pour nos tests, nous avons, à l'aide de *NetLimiter*, restreint la bande passante du réseau. Dans le cas de la neurologie, il en résulte que sur le site expert la vidéo est automatiquement désactivée, et sur le site demandeur les flux audio et vidéo sont toujours activées.

5.2/ ALGORITHMES ET APPLICATIONS DES STRATÉGIES D'ADAPTATION

L'intergiciel VAGABOND possède des mécanismes de perception et de décision qui aident aux divers changements de stratégies possibles. Le module de décision utilise des lois de probabilités pour déclencher des décisions d'adaptation des différentes sessions de vidéoconférence. L'objet de cette section est de présenter les algorithmes que nous avons développés.

5.2.1/ ALGORITHME DE LA LOI BINOMIALE

Lorsqu'une session de vidéoconférence est en cours, le module de surveillance est chargé de surveiller l'état du réseau en vue d'une éventuelle adaptation. De plus, suivant les types de terminaux une adaptation automatique est faite pour la résolution des tailles des écrans.

L'état de l'art nous a permis de mettre en évidence qu'une adaptation proactive est plus intéressante car elle permet à une application d'être la plus réactive possible, ainsi nos différents algorithmes ont été développés dans ce sens. L'algorithme (figure 5.4) qui suit est utilisé toutes les 10 secondes et permet une première étude de l'état du réseau. Il prend en entrée le nombre total de paquets reçus dans un intervalle de 10 secondes et le nombre de paquets retenus. Rappelons que les paquets retenus sont ceux qui respectent le délai de 800 millisecondes entre l'envoi d'un paquet vidéo encodé avec le standard H.264 et sa réception. Cet algorithme est représenté sous forme de pseudo code et a été implémenté dans l'intergiciel VAGABOND.

L'algorithme fonctionne de la sorte :

```
début
    /* paquetsRecus : nombre total de paquets vidéo reçus en 10 secondes; */
    /* paquetsAcceptes : nombre total de paquets vidéo acceptés ayant un
       horodatage inférieur à 800 millisecondes; */
    /* Retourne : un booléen pour indiquer si une action supplémentaire est
       nécessaire ou non */
    paquetsPerdus ← paquetsRecus - paquetsAcceptes ;
    /* Calcul du taux de succès sur les échantillons (inférence
       fréquentiste) */
    tauxSucces ← paquetsAcceptes / paquetsRecus ;
    /* Calcul du taux d'échec sur les échantillons */
    tauxEchec ← 1 − tauxSucces ;
    si (tauxEchec > 0.5) alors
        /* Sur les 3 prochaines secondes, quel est le nombre de paquets
           attendus */
        nbPaquetsAttendus ← 3 . paquetsRecus / 10 ;
        /* Les nouveaux paramètres pour le calcul de la distribution
           binomiale est donc : nbPaquetsAttendus; tauxSucces; paquetsRecus */
        prob ← calculBinomial(nbPaquetsAttendus, tauxSucces, paquetsRecus) ;
        si (prob > 0.5) alors
            retourner vrai ;
        sinon
            retourner faux ;
        fin si
    sinon
        retourner faux ;
    fin si
fin
```

Algorithme de calcul de la distribution binomiale

distributionBinomiale

```
début
    /* nbPaquetsAttendus : nombre de paquets attendus dans les 3 prochaines
       secondes; */
    /* tauxSucces : taux de succès sur les échantillons actuels; */
    /* paquetsRecus : nombre d'échantillons reçus; */
    tauxEchec ← 1 - tauxSucces ;
    paquetsIgnores ← paquetsRecus − nbPaquetsAttendus ;
    borneSup ← paquetsIgnores + nbPaquetsAttendus ;
    prob ← 0.0 ;
    pour chaque i de paquetsIgnores à borneSup incluse faire
        prob += borneSup! / (i! . (borneSup−i)!) . tauxSucces^i . tauxEchec^(borneSup−i) ;
    fin pour chaque
    retourner prob ;
fin
```

FIGURE 5.4 – Algorithme de calcul du taux de succès sur l'arrivage des paquets : verificationBinomiale($paquetsRecus$, $paquetsAcceptes$)

- Toutes les 10 secondes, le nombre de paquets entrants est analysé,
- Les nombres de paquets perdus et retenus sont calculés,
- Le taux de succès est ainsi déduit de ces deux nombres (il s'agit d'une probabilité calculée à l'aide de l'inférence fréquentiste),
- Si le taux d'échec déduit à partir du taux de succès est supérieur à 0.5, alors un calcul binomial est effectué,
- Le nombre de paquets attendus sur les prochaines 3 secondes est calculé en fonction des paquets reçus dans 10 dernières secondes,
- Les nouveaux paramètres utilisés pour le calcul de la distribution cumulée binomiale deviennent :
  - nbPaquetsAttendus : il s'agit du nombre de paquets attendus dans les 3 prochaines secondes qui est calculé en fonction du nombre total de paquets reçus des 10 dernières secondes,
  - tauxSucces : le taux de succès des paquets reçus dans les 10 dernières secondes,
  - paquetsRecus : le nombre total de paquets reçus dans les 10 dernières secondes,

- Si le retour du calcul de la fonction de distribution cumulée binomiale retourne un résultat supérieur à 0.5, alors l'algorithme retourne vrai
- Dans tous les autres cas, l'algorithme retourne faux.

L'algorithme qui suit est utilisé lorsque l'algorithme de calcul de la distribution cumulée binomiale donne un résultat positif : c'est-à-dire une probabilité supérieure à 0.5 de déclencher une adaptation.

5.2.2/ ALGORITHME DE L'INFÉRENCE BAYÉSIENNE

Une deuxième vérification consiste à utiliser l'inférence bayésienne. Comme expliqué dans la partie de l'état de l'art, l'inférence bayésienne est une méthode d'inférence permettant de déduire la probabilité d'un événement à partir de celles d'autres événements déjà évalués (nous avons précédemment appliqué la distribution cumulée binomial). Elle s'appuie principalement sur le théorème de Bayes. Cet algorithme, utilisé dans l'intergiciel VAGABOND, est présenté dans la figure 5.5.

Cet algorithme utilise une étape $N-1$ avant de pouvoir donner un résultat. Chaque résultat est dépendant de l'étape précédente. Ainsi, le maillage qui se forme entre toutes les étapes permet de prendre en compte une étape précédente (un passé), et une étape courante (le présent), pour en déduire une probabilité caractéristique du futur dite postérieure (soit supérieure à 0.5 qui signifie une prédiction d'amélioration, soit dans le cas contraire qui signifie une prédiction de dégradation de la bande passante). Le fonctionnement de cet algorithme est le suivant :

- À l'étape d'initialisation, les valeurs de α et de β sont mémorisées. Ces valeurs correspondent respectivement au nombre de paquets reçus et de paquets perdus. On obtient ainsi les deux paramètres de la distribution bêta. Cette étape se passe au début de chaque session de vidéoconférence. L'étape d'initialisation est alors terminée et l'algorithme retourne faux comme résultat,
- Passée l'étape d'initialisation, le taux de succès est calculé en fonction du nombre de paquets acceptés et de paquets reçus. De ce taux de succès est ensuite calculé, le taux d'échec,
- Une moyenne des paquets reçus est ainsi obtenue et cette moyenne correspond au taux de succès,
- Un coefficient de variation des paquets reçus est ensuite calculé. Ce coefficient indique le degré d'exactitude du résultat de notre moyenne,
- La moyenne et le coefficient précédemment calculés correspondent à un instant t. Comme nous souhaitons prédire l'instant $t+1$ par le biais d'un calcul de probabilité, nous devons donc calculer une moyenne et un coefficient de variation postérieurs,
- Si la moyenne actuelle est strictement inférieure à 0.5 et le coefficient de variation strictement inférieur à 10, alors le calcul de la moyenne postérieure et du coefficient de variation postérieur peuvent s'effectuer. Dans le cas contraire, l'algorithme se termine en mettant à jour les paramètres α et β et en retournant faux comme résultat,
- Les calculs de la moyenne postérieure et du coefficient de variation postérieur s'effectuent en calculant tout d'abord les deux paramètres α et β de la distribution bêta,
- Le paramètre α_{post} correspond au nombre de paquets acceptés à l'instant t ajouté à la valeur actuelle de α,

début

/* $paquetsRecus$: nombre total de paquets vidéo reçus en 10 secondes; */
/* $paquetsAcceptes$: nombre total de paquets vidéo acceptés ayant un horodatage inférieur à 800 millisecondes; */
/* Retourne : un booléen pour indiquer si une action supplémentaire est nécessaire ou non */
$paquetsPerdus \leftarrow paquetsRecus - paquetsAcceptes$;
/* Début de l'algorithme, $initialisation$ est une variable globale */
si ($initialisation$) **alors**
  /* α représente le nombre de paquets acceptés */
  $\alpha \leftarrow paquetsAcceptes$;
  /* β représente le nombre de paquets perdus */
  $\beta \leftarrow paquetsPerdus$;
  $initialisation \leftarrow$ faux ;
  retourner *faux* ;
fin si
/* Calcul du taux de succès sur les échantillons (inférence fréquentiste) */
$tauxSucces \leftarrow \frac{paquetsAcceptes}{paquetsRecus}$;
/* Calcul du taux d'échec sur les échantillons */
$tauxEchec \leftarrow 1 - tauxSucces$;
$moyenneReception \leftarrow tauxSucces$;
$coeffVariationReception \leftarrow \sqrt{\frac{paquetsAcceptes \,.\, paquetsPerdus}{(paquetsAcceptes+paquetsPerdus)^2.(paquetsAcceptes+paquetsPerdus+1)}}$;
/* Calcul du pourcentage */
$coeffVariationReception \leftarrow coeffVariationReception \,.\, 100$;
si ($moyenneReception < 0.5$) && ($coeffVariationReception < 10$) **alors**
  /* Calcul des nouvelles variables de la distribution β */
  $\alpha_{post} \leftarrow paquetsAcceptes + \alpha$;
  $\beta_{post} \leftarrow paquetsPerdus + \beta$;
  /* Mise à jour des variables α et β */
  $\alpha \leftarrow paquetsAcceptes$;
  $\beta \leftarrow paquetsPerdus$;
  /* Calcul des nouvelles probabilités, application du théorème de Bayes */
  $moyenneReception_{post} \leftarrow \frac{\alpha_{post}}{\alpha_{post} + \beta_{post}}$;
  $coeffVariationReception_{post} \leftarrow \sqrt{\frac{\alpha_{post} \,.\, \beta_{post}}{(\alpha_{post}+\beta_{post})^2 \,.\, (\alpha_{post}+\beta_{post}+1)}}$;
  /* Calcul du pourcentage */
  $coeffVariationReception_{post} \leftarrow coeffVariationReception_{post} \,.\, 100$;
  si ($moyenneReception_{post} > 0.5$) && ($coeffVariationReception_{post} < 10$) **alors**
    retourner *vrai* ;
  sinon
    retourner *faux* ;
  fin si
sinon
  /* Mise à jour des variables α et β */
  $\alpha \leftarrow paquetsAcceptes$;
  $\beta \leftarrow paquetsPerdus$;
  retourner *faux* ;
fin si

fin

FIGURE 5.5 – Algorithme de calcul du taux de succès sur l'arrivage des paquets (2^{ieme} vérification) : verificationBayesienne($paquetsRecus$, $paquetsAcceptes$)

- Le paramètre β_{post} correspond au nombre de paquets perdus à l'instant t ajouté à la valeur actuelle de β,
- Les paramètres α et β sont ensuite mis à jour et la moyenne postérieure et le coefficient de variation postérieur sont ensuite déduits. Le coefficient de variation postérieur indique le degré d'exactitude du résultat de notre moyenne postérieure,
- L'algorithme retourne vrai si la moyenne de paquets acceptés postérieure est supérieure à 0.5 avec un coefficient de variation postérieur qui est inférieur à 10. Dans le cas contraire, le résultat de l'algorithme est faux.

5.2.3/ APPLICATION DES STRATÉGIES D'ADAPTATION

Les deux algorithmes (utilisant la loi binomiale figure 5.4, ou la théorème de Bayes figure 5.5) sont au centre du moteur de raisonnement de VAGABOND. Le fonctionnement global du système de raisonnement dans l'intergiciel VAGABOND qui utilise ces deux algorithmes est donné dans la figure 5.6 :

```
début
  /* Session de vidéoconférence en cours */
  répéter (toutes les 10 secondes)
    /* paquetsRecus : nbre tot. de paquets vidéo reçus en 10 secondes */
    /* paquetsAcceptes : nbre tot. de paquets vidéo acceptés ayant un horodatage < 800ms */
    paquetsRecus ⟵ ModuleSurveillance.recupererPaquetsRecus();
    paquetsAcceptes ⟵ ModuleSurveillance.recupererPaquetsAcceptes();

    /* Première vérification avec l'inférence fréquentiste et la distribution cumulée binomiale */
    /* algorithme figure 5.4 */
    resultatVerificationBinomiale ⟵ verificationBinomiale (paquetsRecus, paquetsAcceptes);

    /* Deuxième vérification avec l'inférence Bayésienne : algorithme figure 5.5 */
    resultatVerificationBayesienne ⟵ verificationBayesienne (paquetsRecus, paquetsAcceptes);

    si (resultatVerificationBinomiale) && (resultatVerificationBayesienne) alors
      /* Applications des règles d'adaptation pour le session de vidéoconférence en cours */
      /* si les vérifications en montre la nécessité */
      appliquerStrategieAdaptation(specialiteVideoConference);
    fin si
  jusqu'à (la fin de la session de vidéoconférence);
fin
```

FIGURE 5.6 – Algorithme d'application des stratégies d'adaptation : monitorerEtatReseau()

Le fonctionnement de cet algorithme est le suivant :

- Tant que la session de vidéoconférence est en cours, toutes les 10 secondes, le nombre de paquets vidéo reçus et acceptés est pris en compte depuis le module de surveillance,

- Une première vérification consiste à consulter les résultats des *deux* algorithmes de *verificationBinomiale* et *verificationBayesienne* (cf figures 5.4 et 5.5). La vérification bayésienne étant dans tous les cas effectuée afin de mettre à jour

l'algorithme qui, rappelons-le, est un algorithme avec état et donc qui utilise toujours une étape $N - 1$,

- Si l'algorithme *verificationBinomiale*, qui considère le système sans état (cf. section 4.6.1) retourne vrai, et l'algorithme *verificationBayesienne*, qui considère le système avec état (cf. section 4.6.1) retourne vrai, cela donnera lieu à une adaptation. Autrement le nombre de paquets reçus et acceptés sur les 10 prochaines secondes est pris en compte et l'algorithme continue ainsi jusqu'à ce qu'une règle d'adaptation soit appliquée (si l'état de la bande passante devient trop instable).

5.3/ ÉVALUATION DU MODULE DE DÉCISION DE L'INTERGICIEL VAGABOND

Dans cette section, nous présentons les résultats de nos évaluations des modules de l'intergiciel VAGABOND : en particulier le module d'évaluation. Les courbes les plus pertinentes qui démontrent la bonne réactivité du module face à des variations dynamiques de la bande passante d'un réseau sont étudiées. Cependant, nous n'avons pas inclus, dans ce document, toutes les courbes qui ont été produites lors notre travail de tests. L'ensemble est accessible à l'adresse : https://maincare.sharefile.com/d-s943ccb3ab7e4041b.

5.3.1/ ÉVALUATION DES DÉLAIS DE TRANSMISSION DE PAQUETS VIDÉO CLIENT/CLIENT AU SEIN DE L'INTERGICIEL

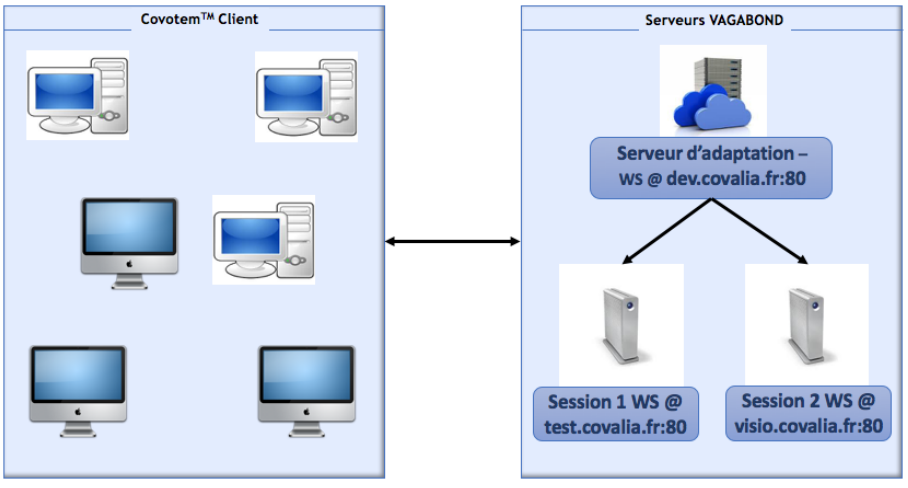


FIGURE 5.7 – Configuration de nos tests

Afin de déterminer , le délai de transmission maximal client/client au sein de l'intergiciel, nous avons mené des évaluations sur le réseau d'une box ADSL (cf figure 5.7). Nous avons configuré ce réseau pour qu'il soit le plus proche possible des réseaux que nous pouvons rencontrer dans le milieu hospitalier, c'est-à-dire, qu'il est restreint et seuls les ports 80 et 443 en TCP sont ouverts. Tous les autres ports sont fermés sur le pare-feu.

Les mesures ont été faites sur des postes ayant tous un processeur Intel Core i7, 8 Go de mémoire vive et une connexion Internet avec un débit montant d'une moyenne de 1 Mbps et un débit descendant d'une moyenne de 7 Mbps. Ils sont tous équipés d'un dispositif audio ainsi que d'une webcam. Ce sont des postes de développement qui sont utilisés chez Ido-In. Une version de l'application de test a été déployée sur des serveurs distants. Le serveur central d'adaptation a été déployé sur le serveur de test d'Ido-In. Deux proxies d'adaptation ont été déployés sur deux autres serveurs de test d'Ido-In également. Cela nous permet d'être dans un environnement de déploiement réel et ainsi avoir des mesures réelles et non théorique. Nous cherchons ici à valider que la plupart des paquets vidéo prennent moins de 800 millisecondes entre le moment de la capture, de l'encodage, de la segmentation chez le client émetteur et le moment où il est reçu et désegmenté chez le client receveur (cf. section 4.4). Ce test a été effectué plusieurs fois entre deux clients émetteur et récepteur. Chaque 20 secondes l'horodatage d'un paquet désegmenté est pris en compte pour une durée de 2000 secondes : 20 secondes permettent un temps suffisant pour observer des variations, et 2000 secondes permettent d'obtenir suffisamment de mesures.

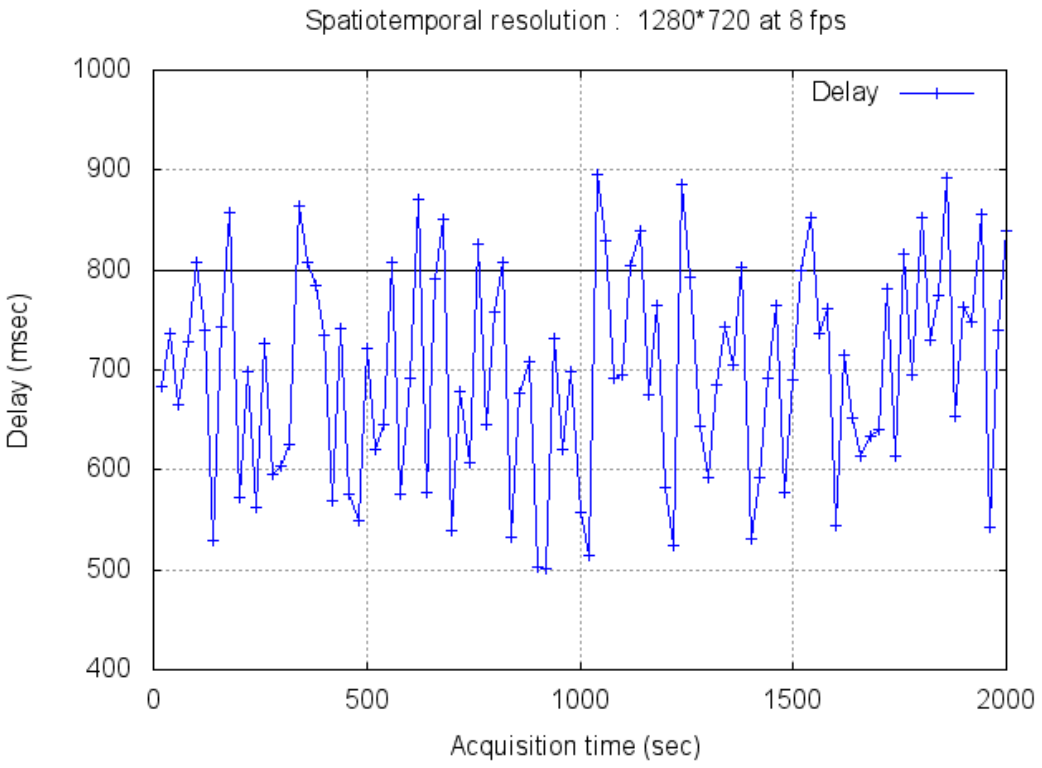


FIGURE 5.8 – Délai de transmission vidéo pour une résolution de 1280 × 720 @ 8 fps

Dans la figure 5.8, la moyenne des délais de transmission est de 697 ms. Il s'agit d'un des résultats obtenus à la suite de plusieurs tests. La résolution de cette vidéo est de 1280 × 720 pixels. Il s'agit d'une qualité qui est supportée par l'application Covotem^{TM} et est gourmande en ressources (CPU, mémoire, bande passante). De par ce graphique, nous pouvons constater que la plupart (plus de trois quarts) des paquets reçus ont un horodage acceptable et seront traités par le client récepteur. Une adaptation n'a pas eu lieu dans cette première phase d'évaluation.

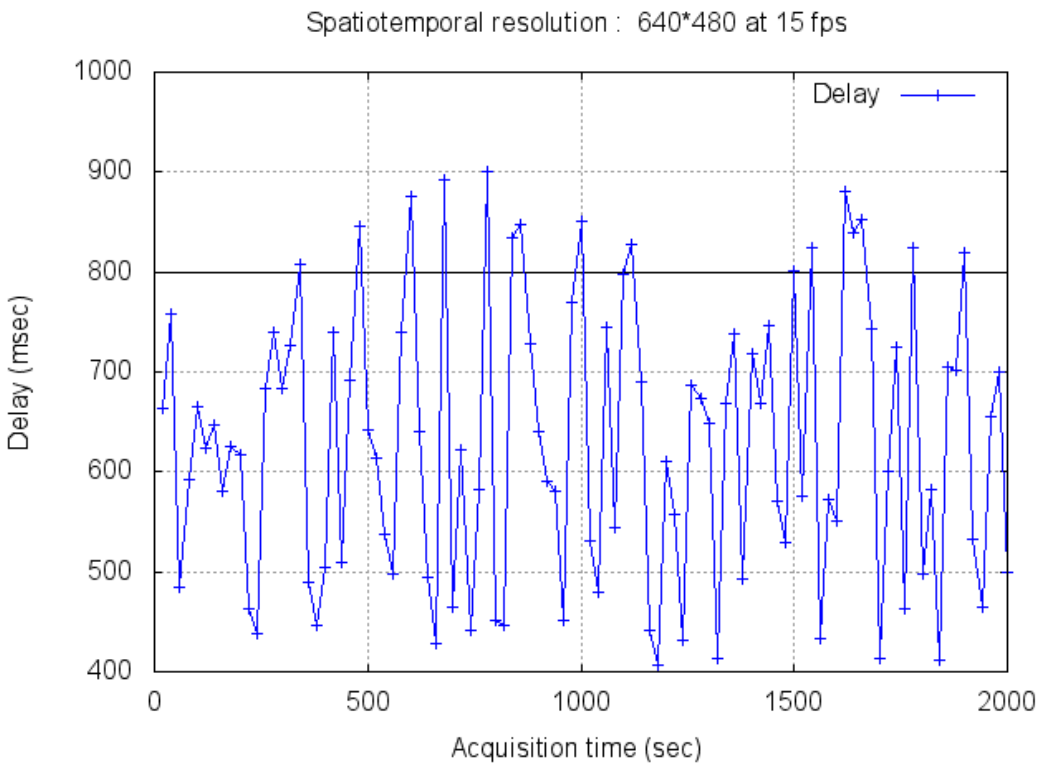


FIGURE 5.9 – Délai de transmission vidéo pour une résolution de 640 × 480 @ 15 fps

Dans la figure 5.9, la moyenne est de 628 ms. La résolution de cette vidéo est de 640 × 480 pixels. Il s'agit d'une qualité qui est supportée par l'application Covotem™ et est moyennement gourmande en ressources (CPU, mémoire, bande passante). De par ce graphique, nous pouvons constater que la plupart (plus de trois quarts) des paquets reçus ont un horodage acceptable et seront traités par le client récepteur. Une adaptation n'a pas eu lieu dans cette évaluation.

Nous avons effectué nos évaluations sur ces deux qualités car il s'agit des qualités les plus gourmandes en ressources.

5.3.2/ ÉVALUATION DU MODULE DE DÉCISION DANS UN RÉSEAU RESTREINT

Cette section présente les évaluations sur le mécanisme de décision de VAGABOND. De plus, nous comparons et expliquons les lois de probabilités qui sont utilisées et sont complémentaires. Pour rappel, l'inférence fréquentiste est utilisée comme première vérification de l'état de la bande passante. Une vérification binomiale est ensuite effectuée si l'inférence fréquentiste a produit un résultat négatif. Si l'inférence fréquentiste et la loi binomiale ont produit un résultat négatif, alors l'inférence bayésienne est utilisée afin de vérifier l'hypothèse en cours (amélioration ou dégradation de la bande passante). Les exemples qui suivent correspondent à des mesures qui ont été effectuées dans un réseau restreint en bande passante. Ces mesures ont été faites sur un poste avec un processeur Core i7, 8 Go de mémoire vive et une connexion Internet avec un débit montant d'un peu moins de 1 Mbps et un débit descendant d'un peu moins de 7 Mbps. L'applica-

tion NetLimiter [1] a été installée sur les postes et configurée pour laisser passer le trafic en émission et réception au maximum 300 kbits par seconde. Nous cherchons ici à valider que la décision d'adaptation est correctement prise au moment opportun et lorsque les lois de probabilité prédisent que la bande passante ne s'améliora pas dans les prochaines secondes.

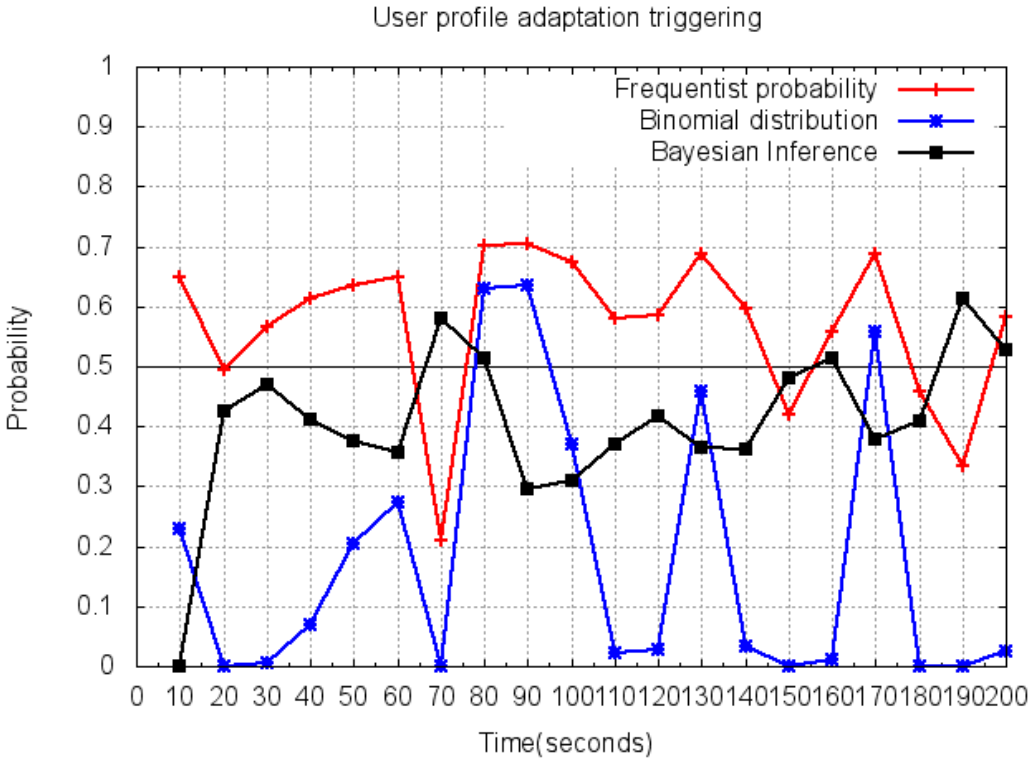


FIGURE 5.10 – Tests sur une session de vidéoconférence dans un réseau restreint

Dans cet exemple (cf figure 5.10), une adaptation aurait dû avoir lieu à $t = 10$ avec une probabilité fréquentiste (courbe de couleur rouge) à 65%. La loi binomiale ne prévoit pas de déclencher une adaptation car elle donne un résultat de moins de 30%. À $t = 80$, la probabilité fréquentiste et la loi binomiale annoncent une adaptation mais l'inférence bayésienne prévoit que plus de 50% de paquets seront reçus. L'adaptation n'a donc pas été prévue à $t = 80$ mais à $t = 90$ l'adaptation a été identifiée comme nécessaire car l'inférence bayésienne prévoit que moins de 40% de paquets seront reçus. À ce même instant, l'inférence fréquentiste et la loi binomiale prévoient l'étude d'une adaptation avec plus de 60% à l'aide de l'inférence bayésienne. Le déclenchement a eu lieu seulement à cet instant $t = 90$. Dans cette première phase de test, il n'y a pas d'adaptation car nous ne testons ici que la prise de décision. Sur ce même graphique, nous pouvons constater qu'à $t = 170$, le déclenchement d'une adaptation aurait également eu lieu. En effet, à $t = 170$, l'inférence fréquentiste et la loi binomiale prédisent avec une probabilité de plus de 50%, le déclenchement d'une adaptation et l'inférence bayésienne prédit que moins de 50% de paquets seront reçus.

1. NetLimiter est une application capable de limiter la bande passante en réception et en émission sur un poste Windows (https ://www.netlimiter.com/)

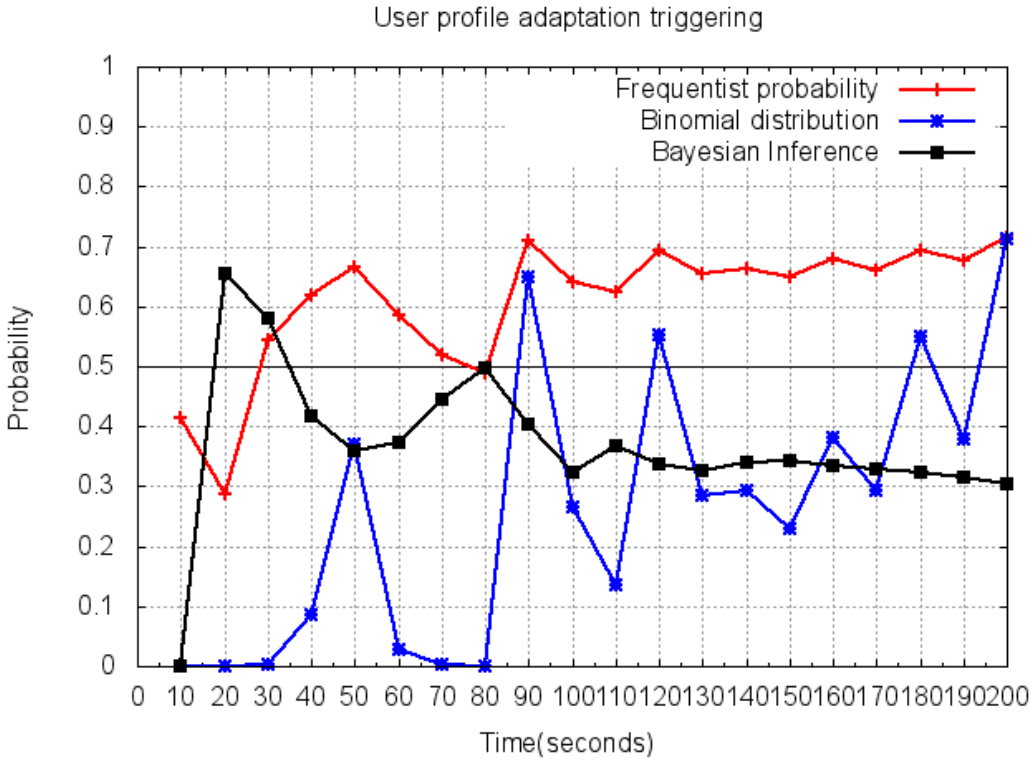


FIGURE 5.11 – Résultats d'une session de vidéoconférence dans un réseau restreint

Un deuxième exemple est donné dans la figure 5.11. À $t = 20$, l'inférence bayésienne prédit une réception de plus de 60% des paquets. Une adaptation a été décidée à $t = 90$ car l'inférence bayésienne prévoit que moins de 50% de paquets seront acceptés. À ce même instant, l'inférence fréquentiste et la loi binomiale prédisent une adaptation avec plus de 60%.

5.3.3/ ÉVALUATION DU MODULE DE DÉCISION DANS UN RÉSEAU MOBILE (TYPE 3G, 3G+, 4G)

Les réseaux mobiles sont des réseaux sans fil qui ont souvent une large bande passante (meilleure que l'ADSL) mais qui ont la particularité d'être instables. Une session de vidéoconférence étant gourmande en bande passante, nous avons réalisé nos évaluations uniquement sur des réseaux de type 3G et 4G. Le réseau mobile 3G est la troisième génération de normes de téléphonie mobile (UMTS) et propose un débit théorique à 1,9 mégabits par seconde. Le réseau 3G+ aussi appelé HSDPA permet de monter le débit d'échange de données théorique à 14,4 mégabits par seconde. Il existe également le réseau H+ qui se situe entre la 3G et la 4G, également appelé *Dual Carrier* ou HSPA+. Ce type de réseau offre un débit théorique de 42 mégabits par seconde. Avec la quatrième génération (4G LTE), l'échange de données peut dépasser les 100 mégabits par seconde. En réalité, la bande passante est partagée entre les utilisateurs d'une même borne. Ainsi, moins il y a d'utilisateurs utilisant le réseau et plus le débit est élevé. Il existe également d'autres types de réseaux mobiles, encore plus puissants

comme la 4G+ et la 4G+ UHD. Les deux offrant un débit théorique de 200 à 300 mégabits par seconde.

De par les évaluations que nous avons menées, nous avons rédigé un tableau récapitulatif de ces types de réseaux mobiles et des débits moyens montants et descendants que nous avons rencontrés. La table 5.1 résume ces évaluations.

Type de réseau	Caractéristiques
Réseau 3G / Classique	Débit descendant moyen : 200 kbps Débit montant moyen : 100 kbps Une communication orientée audio uniquement devrait être possible
Réseau 3G+ / HSDPA	Débit descendant moyen : 1500 kbps Débit montant moyen : 750 kbps Une communication orientée audio et vidéo de qualité faible devrait être possible
Réseau H+	Débit descendant moyen : 4000 kbps Débit montant moyen : 1500 kbps Une communication orientée audio et vidéo devrait être possible
Réseau 4G	Débit descendant moyen : 8000 kbps Débit montant moyen : 3000 kbps Une communication orientée audio et vidéo de haute qualité devrait être possible
Réseau 4G LTE, 4G+, 4G+ UHD	Débit descendant moyen : 15000 kbps Débit montant moyen : 7500 kbps Une communication orientée audio et vidéo de haute qualité devrait être possible

TABLE 5.1 – Tableau récapitulatif des débits minimums requis pour une session de vidéoconférence mobile

Nous avons également mené plusieurs évaluations sur notre moteur de raisonnement dans ces types de réseaux. Les mobiles utilisés étaient essentiellement de type Android, récents et donc compatibles avec les réseaux mobiles 3G et 4G. Le résultat d'une évaluation d'une session de vidéoconférence sur un mobile dans un réseau 3G est présenté sur la figure 5.12. Comme nous pouvons le constater, ce type de réseau est très instable. Ce qui est dû à plusieurs facteurs comme notamment les interférences avec d'autres réseaux cellulaires.

Sur nos évaluation (cf figure 5.12), une adaptation est prédite et devrait avoir lieu à $t = 30$ avec une probabilité fréquentiste supérieure à 60% (courbe de couleur rouge). La loi binomiale ne prévoit pas de déclencher une adaptation car elle donne un résultat de moins de 20%. À $t = 80$, la probabilité fréquentiste et la loi binomiale annoncent une adaptation avec plus de 50% et l'inférence bayésienne prévoit que moins de 40% de paquets seront reçus. Le déclenchement a donc lieu à cet instant. Dans cette première phase de test, il n'y a pas d'adaptation car nous ne testons ici que la prise de décision. Sur ce même graphique, nous pouvons constater qu'à $t = 90, t = 100, t = 170$ et $t = 180$, une adaptation doit également être déclenchée. En effet, à $t = 90, t = 100, t = 170$ et $t = 180$, l'inférence fréquentiste et la loi binomiale prédisent avec une probabilité de plus de 50%, le déclenchement d'une adaptation et l'inférence bayésienne prédit que moins de 40% de paquets seront reçus.

Une deuxième évaluation est donnée dans la figure 5.13. Les réseaux de type 4G ayant une plus large bande passante, un plus grand nombre de paquets est accepté. Toute-

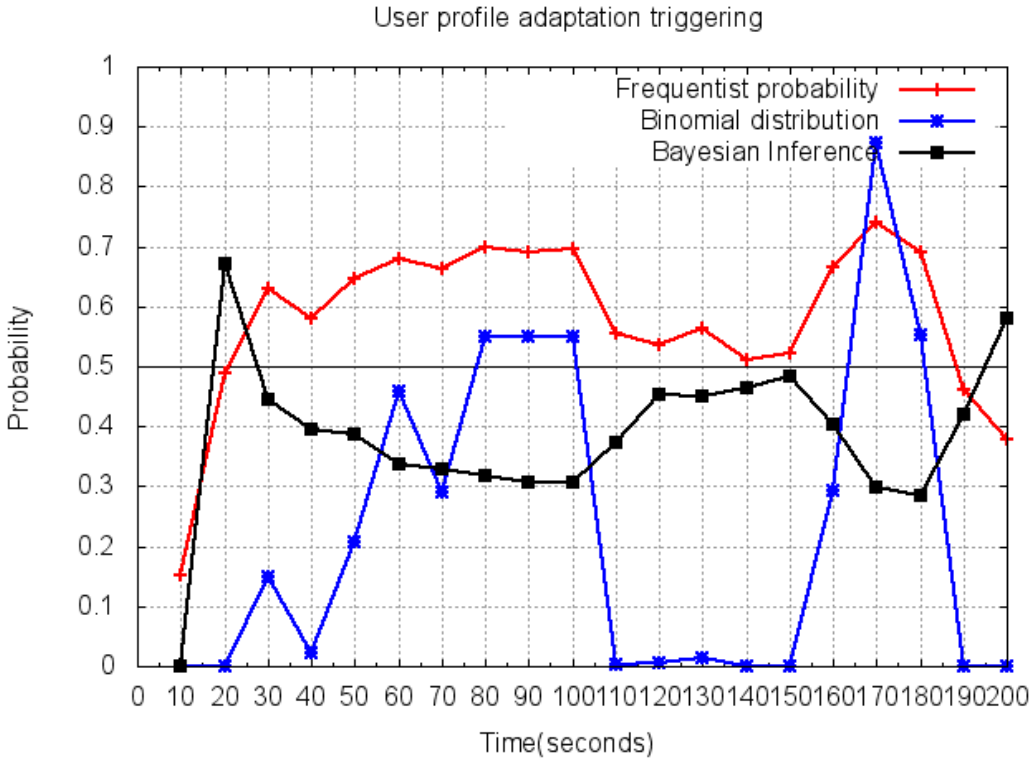


FIGURE 5.12 – Résultats d'une session de vidéoconférence dans un réseau mobile, type 3G

fois, les perturbations peuvent se produire et lorsque c'est le cas, nous voulons savoir si notre moteur de décision est réceptif à ce genre d'évènement et si un déclenchement d'adaptation aura bien lieu. Différentes évaluations ont été effectuées (disponibles sur notre site https://maincare.sharefile.com/d-s943ccb3ab7e4041b), dont une présentée à la figure 5.13. En utilisant uniquement une inférence fréquentiste, une adaptation aurait eu lieu à $t = 90$. À cet instant, la loi binomiale n'est pas favorable à cette décision malgré le fait que l'inférence bayésienne le soit. Aucune adaptation n'est déclenchée. À $t = 140$, l'inférence bayésienne prédit une réception de paquets d'environ 30%. L'inférence fréquentiste ainsi que le loi binomiale sont favorables à une adaptation. Une adaptation a donc lieu à $t = 140$. Par la suite, le réseau est stable et l'utilisateur peut choisir un retour à l'état initial, s'il le souhaite.

5.3.4/ ÉVALUATION DU SYSTÈME AVEC ET SANS LE MODULE DE DÉCISION ET L'APPLICATION D'UNE RÈGLE D'ADAPTATION

L'adaptation que nous appliquons dans l'intergiciel permet de limiter le nombre de paquets vidéo qui sont ignorés et donc non traités. Les flux vidéo étant gourmands en bande passante, cela a pour effet de rendre une session de vidéoconférence inutilisable si les paquets audio ont du retard également. En effet, beaucoup de paquets audio ignorés rendent le son inaudible et donc inutilisable.

Nous avons également souhaité évaluer l'impact des adaptations sur le nombre de pa-

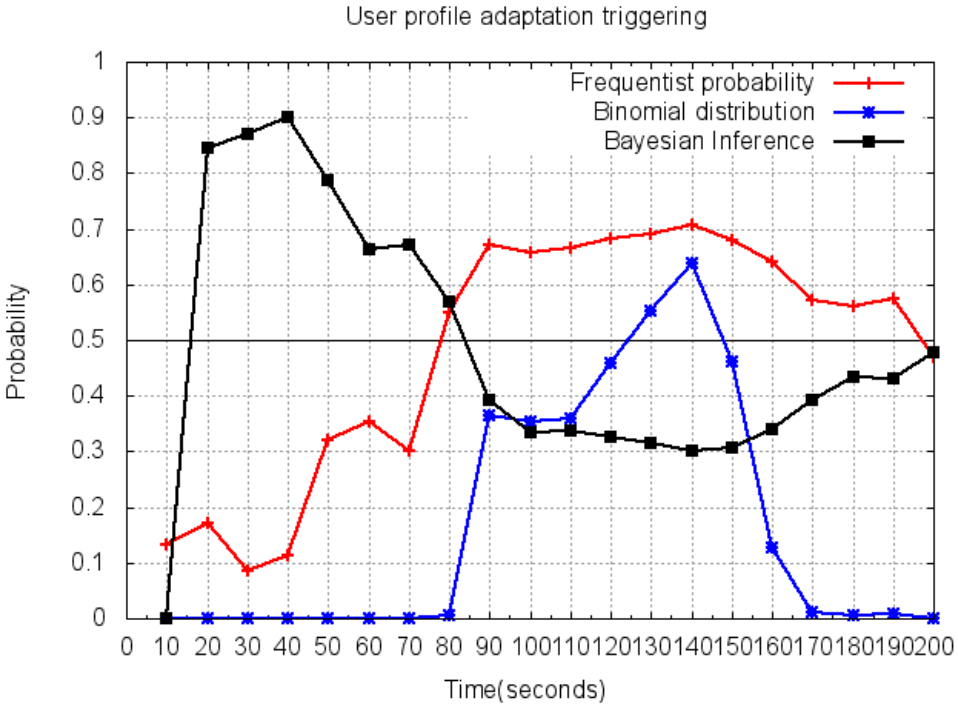


FIGURE 5.13 – Résultats d'une session de vidéoconférence dans un réseau mobile, type 4G

quets vidéo qui sont ignorés. Pour cela, nous avons évalué un processus avec une règle d'adaptation appliquée et un autre sans règle d'adaptation appliquée. Ces deux processus se trouvent sur le même poste et donc recoivent en même temps les paquets vidéo. La règle d'adaptation qui a été appliquée dans le cadre d'un contexte de télédiagnostic en dermatologie. Après l'application de la règle d'adaptation, la vidéo de l'émetteur a une fréquence (*frames per second*) plus basse mais avec des images de meilleure résolution. La figure 5.14 montre les résultats de ces deux processus (un processus appliquant une règle d'adaptation et un autre processus ne l'appliquant pas). La courbe bleue démontre clairement que lorsqu'une adaptation a été appliquée, le nombre de paquets vidéo ignorés est nettement réduit. À l'inverse, la courbe rouge démontre que le nombre de paquets vidéo ignorés a considérablement augmenté.

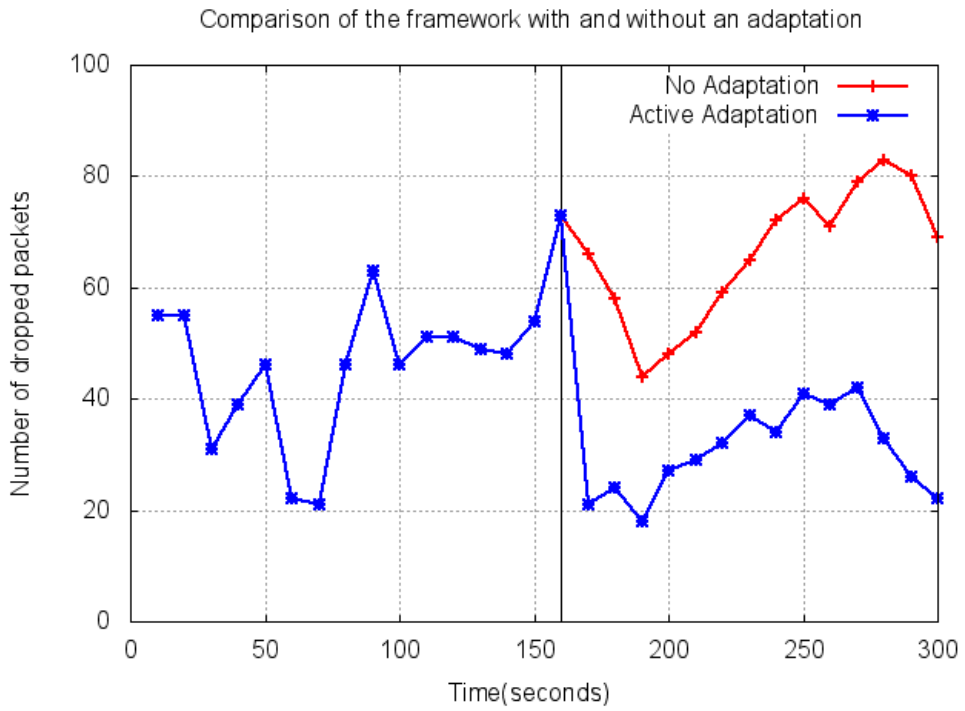


FIGURE 5.14 – Comparaison entre des sessions de vidéoconférence adaptée et non adaptée

Pour résumer

SYNTHÈSE

Ce chapitre a permis d'exposer les différents algorithmes qui sont utilisés dans l'intergiciel VAGABOND. L'intergiciel VAGABOND a été pensé afin de proposer des mécanismes d'adaptation au niveau des clients et au niveau du serveur. Le module de surveillance se trouve sur le client et est en charge de surveiller l'état du réseau par le biais des nombres de paquets reçus et de paquets acceptés. Cette information est ensuite exploitée par le module de décision qui utilise différentes lois de probabilités afin de déclencher une adaptation niveau utilisateur. Des lois de probabilités telles que l'inférence fréquentiste, la loi binomiale ou encore l'inférence bayésienne sont utilisées.

Au cours de la phase de test, nous montrons dans ce chapitre que le module de décision est bien réactif aux changements de l'état du réseau et ainsi les premiers résultats montrent que les décisions prisent sont conformes à nos attentes en termes de réactivité et déclenchement d'adaptation.

CONCLUSION ET PERSPECTIVES

CONCLUSION GÉNÉRALE

Les travaux présentés dans ce mémoire ont exposé l'étude et la conception d'une nouvelle plateforme conçue pour répondre spécifiquement aux demandes des professionnels de santé dans des séances de vidéoconférences pour le télédiagnostic médical. Cette plateforme est capable d'observer son environnement et d'appliquer proactivement des décisions d'adaptation.

Le premier chapitre de l'état de l'art a permis de montrer que lorsque nous cherchons à adapter un système, nous devons poser 4 questions : 1)Pourquoi devons nous adapter ? 2)Qu'est ce que nous adaptons ? 3)Quand est-ce que nous adaptons ? 4)et enfin comment va se dérouler l'adaptation ? Face à ces questions primordiales, nous répondons avec notre intergiciel en appliquant une stratégie d'adaptation, dans le contexte d'une application distribuée, lorsque le réseau qui permet aux différents terminaux et serveurs de communiquer commence à saturer. Quand nous nous retrouvons dans de telles situations, nous choisissons d'appliquer une stratégie d'adaptation en modifiant les caractéristiques d'une session de vidéoconférence pour un utilisateur. Ces caractéristiques concernent les préférences des professionnels de santé pour une session de vidéoconférence. Elles ont été élaborées avec et pour les professionnels de santé en prenant en compte les éléments les plus pertinents dans une telle session de vidéoconférence. Enfin, l'intergiciel VAGABOND est conçu dans le but d'assister les professionnels de santé dans leurs diagnostics lors des Réunions de Concertations Pluridisciplinaires(RCP) par exemple qui se dérouleront ainsi dans de bonnes conditions.

Nous avons structuré ce document en 2 parties :

- La première est consacrée aux différents états de l'art. Son premier chapitre étudie la notion d'adaptabilité. Les définitions et les enjeux de l'adaptabilité sont exposés. Nous apportons une réflexion sur le concept des systèmes collaboratifs, en particulier leurs modes de fonctionnement, à savoir les modes asynchrone et synchrone. Nous mettons en évidence également les raisons pour lesquelles la troisième ère de l'informatique ubiquitaire repose sur les mécanismes d'adaptation. Enfin, nous définissons les types d'adaptation existants : l'adaptation active et l'adaptation proactive. Ainsi, nous montrons qu'une adaptation proactive est plus intéressante et permet de préparer de nouvelles actions pour les appels à venir lors de la détection d'un contexte pertinent. La détection d'un contexte pertinent se fait grâce à un historique des contextes observés ou par un mécanisme d'apprentissage.
  Le deuxième chapitre de cette partie est orienté sur les thématiques autour de la notion de contexte et des mécanismes de prise de décision. Lorsque nous parlons d'adaptation, la notion de contexte dans les environnements distribués est

incontournable. Nous apportons donc une attention très particulière sur cette notion et son utilisation dans les applications distribuées. L'informatique ubiquitaire impose de considérer de multiples utilisateurs et de multiples dispositifs. Ainsi, avec de multiples utilisateurs répartis géographiquement, la mobilité devient un élément essentiel à prendre en compte avec des techniques d'adaptation. Nous étudions plusieurs intergiciels conçus pour la prise en compte du contexte et analysons leurs techniques d'observation du contexte et les différents types de raisonnements logiques utilisés. Nous rappelons les logiques utilisées : la logique des propositions, la logique du premier ordre, l'inférence bayésienne, l'inférence fréquentiste, la loi binomiale, le raisonnement non-monotone, le raisonnement flou, . . . Pour la prise en compte d'un élément du contexte, les raisonnements logiques cités dans la dernière partie de cet état de l'art peuvent être utilisés. La logique des propositions et la logique du premier ordre sont utiles afin de formaliser les observables issus du contexte en langage mathématique. L'inférence bayésienne permet de réviser ou de calculer la probabilité d'une hypothèse et est utile lorsque la loi binomiale n'est pas suffisante pour vérifier la véracité de l'hypothèse sur des évènements stochastiques. L'épreuve de Bernoulli (qui se solde uniquement par succès ou échec), donne lieu au schéma de Bernoulli (répétition n fois des épreuves de manière indépendante), et ainsi la loi binomiale qui correspond au nombre de succès à l'issue du schéma de Bernoulli. Le raisonnement non-monotone permet de situer la classe de raisonnement (raisonnement probabiliste, meilleur raisonnement, raisonnement par défaut, . . .).

- La deuxième partie de ce manuscrit de thèse est consacrée à la contribution de ces travaux. Tout d'abord, le chapitre 3 illustre la conception globale de l'intergiciel VAGABOND et les éléments le composant, dont des modules dédiés à l'adaptation des flux et à la stratégie d'une session de vidéoconférence. Nous avons défini un module de surveillance et un module de décision qui sont rattachés à un client et un module d'adaptation qui se compose de plusieurs proxies d'adaptation et d'une base de données pour les différents profils des professionnels de santé. Afin de bien comprendre à la fois les mécanismes et la finalité de l'intergiciel VAGABOND, il était nécessaire de présenter les diagrammes de séquence et le schéma relationnel de la base de données dont les descriptifs constituent la dernière section de ce chapitre. De plus, ces diagrammes ont permis de mettre en évidence certaines lacunes et défaillances de l'intergiciel et ainsi les corriger lors de la phase de conception.
  Puis, le chapitre 4 présente les aspects fonctionnels de l'intergiciel VAGABOND ainsi que ses composants. Dans l'intergiciel de VAGABOND, l'accent est mis principalement sur les données contraintes au temps réel. Sur ces données est effectuée une étude statistique de l'état de la bande passante descendante. La loi binomiale et l'inférence bayésienne sur une distribution binomiale sont utilisées pour déclencher des adaptations. Ainsi, nous souhaitons être plus tolérants aux fortes variations de la bande passante d'un réseau. Avec une précision plus fine et grâce à ces lois de probabilité, l'adaptation n'est déclenchée que lorsque des congestions réseau surviennent. Enfin, TCP étant un protocole de transport fiable et en mode connecté, nous avons eu besoin de concevoir et d'utiliser de nouvelles stratégies d'adaptation intelligentes avec la transmission de données afin de faire face aux problèmes de latence et à la temporisation des sockets. Des mécanismes d'adaptation sont étudiés pour ces types d'échanges et ainsi notre intergiciel a été

étudié afin de pouvoir être déployé spécialement dans le milieu hospitalier dans lequel les systèmes de sécurité sont extrêmement restrictifs.

Le cœur de ce travail a permis de proposer l'intergiciel VAGABOND (cf. figure 5.15) qui s'articule autour de 5 composants : les clients, un module de surveillance, les proxies d'adaptation, le module d'adaptation, et le module de décision.

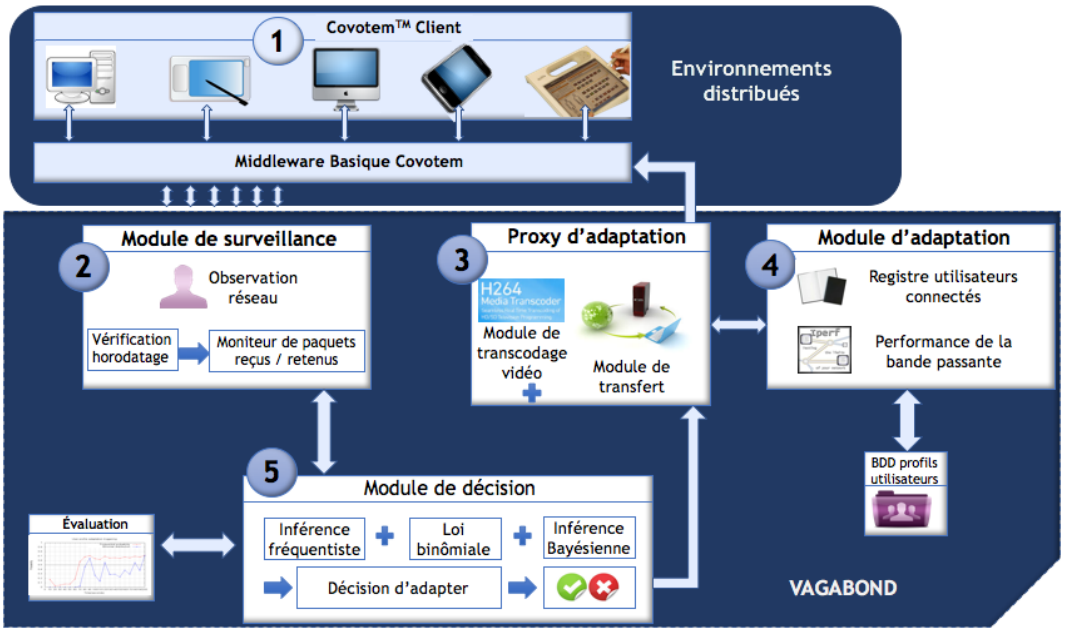


FIGURE 5.15 – L'architecture globale de la plateforme VAGABOND

- Le repère 1 représente les clients de l'application : il s'agit ici des clients Covotem™ (cf section3.1.1).

- Le repère 2 correspond au module de surveillance qui fait partie de l'architecture de VAGABOND. Ce module est embarqué sur les clients et surveille l'état du réseau. Concrètement, il est chargé de vérifier l'horodatage de chaque paquet vidéo entrant. Un moniteur de paquets est ensuite chargé de comptabiliser les nombres de paquets retenus et rejetés.

- Le proxy d'adaptation 3 est chargé de transcoder (décodage et encodage dans un nouveau format) les trames de vidéo : soit dans un format adapté au terminal du client (en particulier au début du cycle), soit en fonction des choix du module d'adaptation. À noter que le transcodage se fera dans une résolution inférieure ou égale à l'originale. Tous les flux échangés se font avec le protocole TCP et sont chiffrés en AES. Chaque proxy d'adaptation déployé est rattaché à un serveur d'adaptation unique.

- Sur le schéma 5.15, ce serveur est noté module d'adaptation 4. Ce module possède un registre unique répertoriant tous les proxies d'adaptation lui étant rattachés, un registre répertoriant tous les utilisateurs connectés à l'intergiciel. Il possède également un serveur IPerf (voir la section 4.2) qui est utilisé à l'étape

d'établissement d'une connexion entre un client et le module d'adaptation de VAGABOND. Tous les échanges entre ce serveur et les clients se font par le biais de Webservices de type REST.

- Les données collectées et issues d'une phase d'observation sont ensuite transmises au module de décision 5 qui est chargé de faire des évaluations quant à l'état du réseau actuel en se basant sur des études statistiques de ces données difficilement prédictibles. En outre, des lois de probabilité telles que l'inférence fréquentiste, la loi binômiale, et l'inférence bayésienne seront utilisées. De ces évaluations sont définies des règles d'adaptation. Ces dernières sont transmises au proxy d'adaptation qui est chargé (en tenant compte des paramètres des terminaux, des profils utilisateurs, ...) de diffuser les nouveaux paramètres de la vidéoconférence en cours.

De plus, nous avons élaboré les différents diagrammes d'activités que nous avons conçus afin de permettre la compréhension des actions de chaque élément du système.

Nous avons implémenté cet intergiciel afin d'effectuer des tests de performances. Il en résulte un système possédant des mécanismes d'adaptation au niveau des clients et au niveau des proxies d'adaptation. Les adaptations sont déclenchées uniquement si les lois de probabilités que nous utilisons dans le module de décisions donnent un verdict positif. Les différentes lois que nous utilisons sont : l'inférence fréquentiste, la loi binomiale (basée sur le schéma de Bernoulli), et l'inférence bayésienne. Toutes ces lois sont utilisées et travaillent en collaboration afin de donner un verdict final quant à la nécessité d'appliquer une stratégie d'adaptation. Nous avons également démontré, à l'aide de différent tests, que le temps moyen, entre la capture d'un paquet vidéo chez un émetteur et le moment où il est décodé chez le récepteur, est inférieur à 800 millisecondes. Ce temps correspond à un délai durant lequel une session de vidéoconférence est utilisable sans prendre trop de temps sur la transmission des paquets dans le but de limiter le nombre de paquets rejetés. Nous avons montré, au travers de différents résultats, que le module de décision est réactif aux variations de la bande passante dans un réseau de communication et ainsi nous avons montré que l'intergiciel prend des décisions qui sont conformes à nos attentes en termes de réactivité et de déclenchement d'adaptation.

En résumé, notre nouvel intergiciel permet d'appliquer des règles d'adaptation qui sont complètement transparentes pour un utilisateur. Il prend des décisions proactivement en prenant en compte des éléments pertinents de son contexte, tels que la variation dynamique de la bande passante d'un réseau et la spécialité d'un professionnel de santé. Il a été spécialement conçu pour les professionnels de santé afin de leur permettre de se focaliser sur leur métier plutôt que sur le paramétrage de leur session de vidéoconférence. Les résultats des premières utilisations sont encourageants, en particulier pour les utilisateurs qui manquent de compétences dans l'utilisation de l'application Covotem^{TM}ou qui simplement n'ont pas le temps d'effectuer des paramétrages lorsqu'il y a une réelle urgence à traiter.

PERSPECTIVES

Ces travaux offrent différentes perpectives qui nous paraissent très intéressantes, tant sur la partie de la prise de décision, sur certaines stratégies d'adaptation que sur la

partie d'encodage vidéo avec la norme H.264. Les valeurs empiriques que nous utilisons actuellement dans l'intergiciel doivent également être affinées. Enfin, il est important dans le cadre de cette thèse, effectuée sur un financement CIFRE, de valider l'intégration de nos travaux au sein des produits de l'entreprise.

A, Prise de décisions : vers une approche MCMC

L'intergiciel VAGABOND possède un module capable de déclencher une adaptation. Nous avons donc besoin de mesurer la tolérance aux différentes variations de la bande passante d'un réseau. Pour ce faire, nous avons besoin d'estimer la fréquence d'adaptation. Suite à l'étude de la partie 4.6.1, nous avons démontré que ce module suit différentes lois de probabilité telles que l'inférence fréquentiste, la loi binomiale, et l'inférence bayésienne en se basant sur un schéma de Bernoulli pour les différents expériences. L'inférence bayésienne est la seule à pouvoir décider finalement si une adaptation aura lieu ou non. Nous pouvons très probablement gagner en précision sur nos calculs probabilistes si nous utilisons la méthode de Monte-Carlo par chaînes de Markov (en anglais, *Markov Chain Monte Carlo, MCMC*).

Les méthodes de Monte-Carlo par Chaînes de Markov (MCMC) sont une classe de méthodes d'échantillonnage à partir de distributions de probabilité. Parfois, calculer de telles probabilités peut être mathématiquement très complexe, voire inexacte ou tout simplement impossible. Toutefois, nous pouvons toujours exécuter des implémentations informatiques pour simuler l'ensemble des jeux plusieurs fois et calculer la probabilité du nombre de victoires divisé par le nombre de parties joué. À cela s'ajoutent les chaînes de Markov. Avant de comprendre ces chaînes, il est intéressant de connaitre la propriété de Markov.

Supposons que nous ayons une système à M états possibles, et que nous passons d'un état à l'autre. La propriété de Markov dit, qu'étant donné un processus qui est dans un état X_n à un moment donné, la probabilité de $X_{n+1} = k$ où k est l'un des M états que le processus peut passer, dépendra de l'état du processus à un instant donné et non sur la façon dont il a atteint l'état actuel.

En formalisant mathématiquement, cela donne l'équation 19 :

Équation 19

$$P(X_{n+1} = k | X_{n=k_n}, X_{n-1} = k_{n-1}, ..., X_1 = k_1) = P(X_{n+1} = k | X_n = k_n)$$

Dans la perspective d'une évolution du protocole aux variations dynamiques de la bande passante d'un réseau, il est important d'évaluer la fréquence des adaptations. Les caractéristiques sont les suivantes :

- les évènements sont indépendants les uns des autres,
- il n'existe que deux issues possibles, succès ou échec,
- les évènements interviennent de manière continue,
- le modèle compte le nombre de succès obtenus à l'issue de n épreuves.

Les caractéristiques du système satisfont les hypothèses à l'utilisation des méthodes de Monte-Carlo par Chaînes de Markov. Le travail que nous avons effectué dans la partie 4.6.1, et notamment avec la distribution bêta, peut donc servir de base pour ces calculs par MCMC. Les Chaînes de Markov nécessitent une distribution stationnaire. Il s'agit d'une distribution de probabilité qui reste inchangée dans le temps dans la Chaîne de Markov et donc nous pouvons définir une probabilité pour chaque état du système. Intuitivement, nous pouvons penser à une marche aléatoire sur une chaîne. Nous pouvons visiter des nœuds plus souvent que d'autres en fonction des probabilités des nœuds.

Les méthodes MCMC nous fournissent des algorithmes pour créer des Chaînes de Markov dont la distribution bêta est sa distribution stationnaire, étant donné que nous pouvons échantillonner à partir d'une distribution uniforme. Si nous commençons à partir d'un état aléatoire et que nous passons à l'état suivant, et cela plusieurs fois, nous finirons par créer une chaine de Markov. La distribution stationnaire de cette chaîne et les états dans lesquels nous sommes après plusieurs passages pourraient être utilisés comme échantillons pour la distribution bêta. Un tel algorithme MCMC est l'algorithme *Metropolis Hastings*. Pour échantillonner depuis une distribution bêta, prenons une Chaîne de Markov arbitraire P avec des états infinis sur l'intervalle $[0, 1]$. Respectant l'algorithme *Metropolis Hastings*, nous avons défini les étapes suivantes afin de construire notre Chaîne de Markov :

- Début avec un état initial, dans notre cas, le résultat sur une inférence fréquentiste i,
- Au cours d'une session de vidéoconférence, prise en compte d'un autre résultat de l'inférence fréquentiste, appelé *état proposé* j,
- Calcul de la *probabilité d'acceptation* :

Équation 20

$$a_{ij} = min(s_j/s_i, 1)$$

Où :

- $s_i = Ci^{\alpha-1}(1-i)^{\beta-1}$,
- $s_j = Cj^{\alpha-1}(1-j)^{\beta-1}$.

Où :

- C est la constante de normalisation et dans notre cas, $\frac{1}{B(\alpha,\beta)}$ (cf. équation 13),

- Tirer un nombre au hasard dans l'intervalle $[0, 1]$. Si ce nombre est supérieur à 0.5, accepter la probabilité d'acceptation. Sinon prendre un autre *état proposé*,
- Répéter les étapes précédentes plusieurs fois de suite.

Cette perspective sur la suite de nos travaux de recherche fera l'objet très prochainement d'une publication. Les Chaînes de Markov seront intégrées à l'intergiciel, le rendant ainsi plus intelligent et plus tolérant aux variations dynamiques de la bande passante.

B, Stratégies d'adaptation en session de videoconférence mixte

Il s'agira d'approfondir nos recherches sur les stratégies d'adaptation mises en œuvre actuellement dans l'intergiciel. Cette articulation est indispensable dans le sens où actuellement des sessions de vidéoconférence mixtes entre professionnels de santé ne peuvent pas avoir lieu. En effet, actuellement une session de vidéoconférence dans l'intergiciel correspond à une spécialité bien précise (dermatologie, neurologie, . . .). Il s'agira de faire en sorte que les sessions soient mixtes et permettent un mélange de spécialités (neurologie couplée à de la radiologie par exemple).

Afin de pouvoir réaliser ce type de fonctionnement, il nous faudra résoudre les conflits entre les stratégies d'adaptation actuellement définies dans la base de données de l'intergiciel. Pour cela, nous pensons que définir des ontologies sur ces stratégies d'adaptation pourrait être une solution à notre problématique et ainsi résoudre les différents conflits qui peuvent surgir au cours d'une session de vidéoconférence mixte. Par exemple les mots clés des stratégies seraient définis dans une ontologie.

Notons que les ontologies pour affiner l'utilisation que nous faisons des profils utlisateurs pourrait également faire l'objet d'une piste intéressante.

C, Encodage H.264 SVC

Dans l'architecture de VAGABOND (cf section 4.5), nous modifions la résolution de chaque image encodée afin qu'elle puisse respecter la résolution maximale de l'écran sur lequel elle est destinée à être affichée. Cette transformation n'est effectuée que si la taille de l'écran est inférieure à un facteur d'au moins deux de l'image d'origine : ceci ayant pour but d'économiser de la bande passante montante au niveau des proxies d'adaptation. Malheureusement, ce fonctionnement présente quelques désavantages au niveau de la consommation CPU.

Une solution, afin de palier ces désavantages, serait l'utilisation de la norme *H.264 Scalable Video Coding (SVC)* au lieu de la norme *H.264 Advanced Video Coding (AVC)* qui est actuellement utilisée dans l'intergiciel. Nous parlons ici du concept de multiplexage video, c'est-à-dire, une technique qui nous permet de faire passer plusieurs flux vidéo encodés à travers un seul support de transmission [Gro17]. La norme H.264 SVC le permet et procure une évolutivité compatible avec les réseaux de communication avec une augmentation modérée de la complexité du décodeur par rapport à la norme *H.264 AVC* qui fonctionne en mono-couche (un seul flux de données avec une unique résolution).

H.264 SVC prend en charge divers facteurs tels que le débit d'un réseau, le format accepté par un terminal, et peut s'adapter à la puissance de calcul de ce dernier. Il prend également en charge la conversion sans perte au format H.264 AVC. Ces fonctionnalités fournissent des améliorations aux applications de transmission et de stockage. H.264 SVC a réalisé des améliorations significatives dans l'efficacité du codage avec un degré accru d'évolutivité par rapport aux profils évolutifs des normes de codage vidéo antérieures. La figure 5.16 *(source : http ://blog.schertz.name/2012/07/video-interoperability-in-lync-2013/)* résume le codage qui est produit avec la norme H.264 SVC. Nous remarquons qu'un émetteur peut émettre plusieurs résolutions d'une même vidéo. En comparaison avec la norme H.264 AVC,

SVC augmente la bande passante nécessaire pour la transmission d'une vidéo de seulement 10% *(source : https ://www.hhi.fraunhofer.de/en/departments/vca/research-groups/image-video-coding/research-topics/svc-extension-of-h264avc.html)*. Cette comparaison se trouve également sur la page web de la source citée.

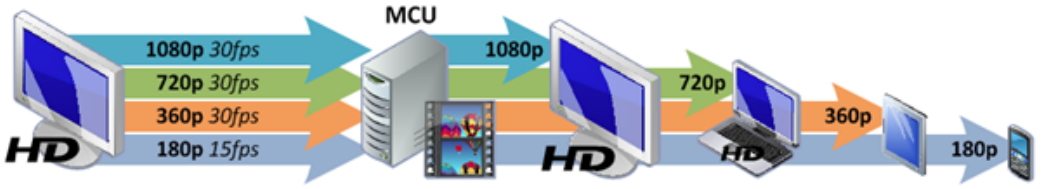


FIGURE 5.16 – Exemple d'un encodage H.264 SVC

La figure 5.17 montre le cas d'utilisation dans l'intergiciel VAGABOND. L'idée serait de remplacer l'encodeur / décodeur H.264 AVC actuellement présent sur les clients et les proxies d'adaptation par un encodeur / décodeur H.264 SVC présent uniquement sur les clients. Nous allégerons ainsi les proxies d'adaptation de la tâche de devoir décoder un flux et le re-encoder dans un autre format.

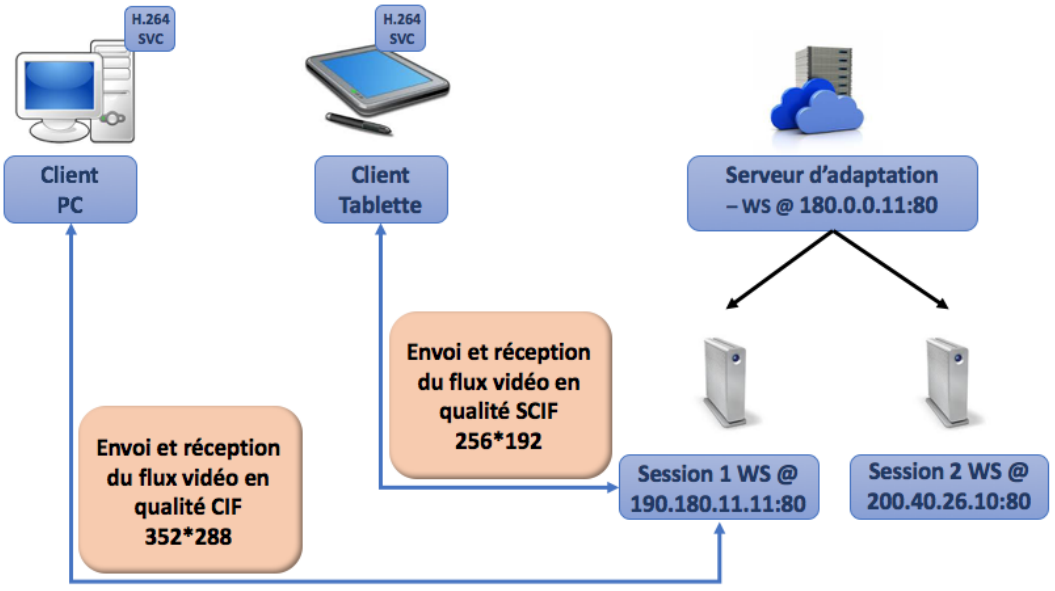


FIGURE 5.17 – L'utilisation de l'encodage H.264 SVC dans la plateforme VAGABOND

Le besoin d'un codage vidéo évolutif, qui permet l'adaptation à la volée de certaines exigences telles que la capacité d'affichage et de traitement des dispositifs ciblés et les conditions de transmission variables, provient de l'évolution continue des dispositifs récepteurs. Comme déjà mentionné dans cette thèse, le codage vidéo est aujourd'hui utilisé dans une large gamme d'applications telles que la messagerie multimédia, la vidéoconférence, la télévision numérique terrestre, . . . La transmission vidéo dans de tels systèmes est exposée à des conditions de transmission variables, qui peuvent être traitées en utilisant des techniques d'adaptation et des fonctionnalités de flexibilité. Il devient alors souhaitable dans des environnements hétérogènes, qu'un codage / décodage vidéo soit réalisé une seule fois, tout en permettant une interopérabilité entre les différents éditeurs / concepteurs de systèmes de vidéoconférence.

D, Valeurs empiriques actuellement utilisées dans le module de décision

Une des perspectives les plus importantes de notre travail consiste à affiner les valeurs empiriques qui sont utilisées lors de la phase d'évaluation de l'état de la bande passante d'un réseau. Comme mentionné dans la partie 4.4, nous utilisons des valeurs empiriques qui sont utilisées dans les algorithmes d'aide à la décision. Ces valeurs sont :

- Une période de 10 secondes qui correspond à un temps durant lequel le nombre de paquets vidéo total réceptionnés et acceptés est pris en compte en vue des différents calculs de probabilités qui en découleront. Cette période est paramétrable dans le module de surveillance et son résultat est pris en compte par le module de décision. Cette période a été choisie car elle nous permet d'avoir suffisamment d'échantillons.
- Une période de 3 secondes qui correspond à un temps durant lequel nous nous attendons à recevoir X paquets : nombre ramené au nombre de paquets réceptionnés sur les dix dernières secondes.
- La limite de probabilité avant de déclencher une adaptation est actuellement fixée à 50% dans les algorithmes vus dans le chapitre 5. Ceci est valable pour les paquets vidéo. En effet, ignorer 50% des paquets vidéo résultera en une dégradation de la fluidité de la vidéo mais qui rend la session de vidéoconférence toujours utilisable. En revanche pour des paquets audio, 50% de ces paquets perdus rendront définitivement une session de vidéoconférence inintelligible.

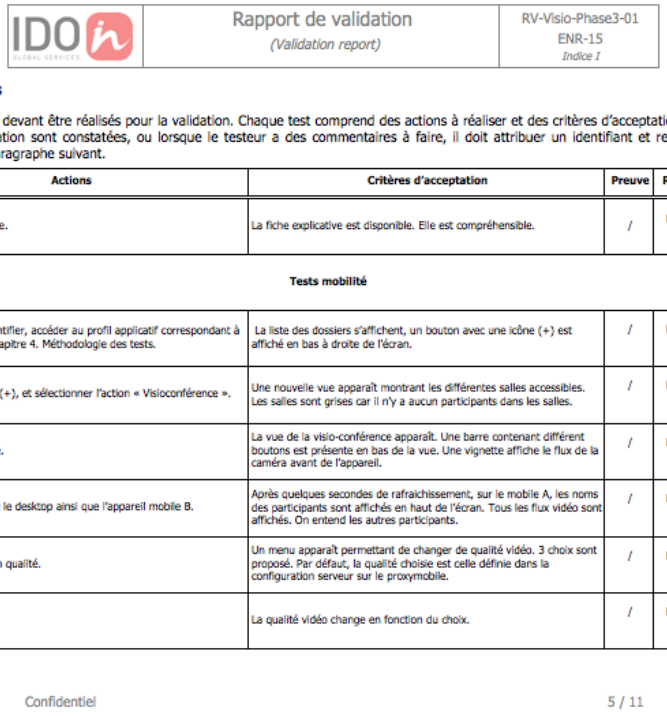
Notre travail très consistera prochainement à affiner ces valeurs. Notons que nous ne traitons actuellement que les paquets vidéo car d'une part, ce sont les paquets les plus volumineux, et d'autre part, il s'agit du type de données qui est la cause même des goulots d'étranglements dans les réseaux. Nous devons par la suite prendre en compte les paquets audio et la limite de probabilité actuellement retenue à 50% ne sera pas envisageable comme pour les paquets vidéo. De plus, la période de 10 secondes étant adaptée pour l'instant à nos calculs devra être plus précise. Il s'agira de trouver un compromis entre la réactivité au sein de l'intergiciel et le nombre suffisant d'échantillons. Enfin, la période de 3 secondes qui nous aide à nous projeter dans un futur devra être affinée en restant toujours le plus réactif que possible. Nous pourrons affiner ces valeurs après une utilisation prolongée de l'intergiciel et une enquête sur le ressenti des utilisateurs en conditions réelles.

E, Intégration aux produits de la suite Covotem^{TM}

Cette thèse nous a permis de valider la partie de faisabilité. En effet, chez Ido-In, avant qu'une version de l'application soit mise en production chez les clients, une phase de faisabilité est primordiale et est le début de chaque projet (recherche et développement). Nous nous situons donc à l'étape de fin de faisabilité. Après cette phase, il s'agira de finir les développements, de rédiger et de faire passer un rapport de validation, d'intégrer ce développement dans le tronc commun de l'application, et de réaliser des tests d'intégrations. Et c'est seulement après toutes ces étapes que l'application pourra être déployée chez les clients si aucune anomalie n'est détectée en amont.

Topic	IOS	Android
Development Tools	Xcode	1. Java tools (Eclipse, ..) + ADT (Android Developer Tool) 2. Android Studio
Programming Language	Objective – C with Swift	JAVA
File to publishing	IPA (IOS Application Archive)	APK (Android Application Package)
Publishing	App Store	Google Play Store
Reviewing	Long App Review (7 days)	No App Review (take time only one hours)

TABLE 5.2 – Comparaison des développements iOS et Android



IDO in | Rapport de validation *(Validation report)* | RV-Visio-Phase3-01 ENR-15 *Indice I*

5. Tableau de tests

Ce paragraphe décrit les tests devant être réalisés pour la validation. Chaque test comprend des actions à réaliser et des critères d'acceptation. Lorsque des déviations par rapport aux critères d'acceptation sont constatées, ou lorsque le testeur a des commentaires à faire, il doit attribuer un identifiant et reporter sur commentaire ou la déviation constatée dans le paragraphe suivant.

# SF	# Test	Actions	Critères d'acceptation	Preuve	Résultat	Id	Initiales testeur/date
NA	1.	Lire la fiche explicative.	La fiche explicative est disponible. Elle est compréhensible.	/	Réussi / Echoué		
Tests mobilité							
SF003	2.	Sur le mobile A, s'identifier, accéder au profil applicatif correspondant à l'EC définit dans le chapitre 4. Méthodologie des tests.	La liste des dossiers s'affichent, un bouton avec une icône (+) est affiché en bas à droite de l'écran.	/	Réussi / Echoué		
SF003	3.	Cliquer sur le bouton (+), et sélectionner l'action « Visioconférence ».	Une nouvelle vue apparaît montrant les différentes salles accessibles. Les salles sont grises car il n'y a aucun participants dans les salles.	/	Réussi / Echoué		
SF003	4.	Sélectionner une salle.	La vue de la visio-conférence apparaît. Une barre contenant différent boutons est présente en bas de la vue. Une vignette affiche le flux de la caméra avant de l'appareil.	/	Réussi / Echoué		
SF003	5.	Faites rentrer en salle le desktop ainsi que l'appareil mobile B.	Après quelques secondes de rafraichissement, sur le mobile A, les noms des participants sont affichés en haut de l'écran. Tous les flux vidéo sont affichés. On entend les autres participants.	/	Réussi / Echoué		
SF003	6.	Appuyer sur le bouton qualité.	Un menu apparaît permettant de changer de qualité vidéo. 3 choix sont proposé. Par défaut, la qualité choisie est celle définie dans la configuration serveur sur le proxymobile.	/	Réussi / Echoué		
SF003	7.	Changer de qualité.	La qualité vidéo change en fonction du choix.	/	Réussi / Echoué		

Confidentiel 5 / 11

TABLE 5.3 – Rapport de validation chez Ido-In

Les développements qui ont été effectués jusqu'à présent, l'ont été en langage Java et ciblent donc les terminaux de type ordinateurs. La suite des développements concernent les mobiles Android et iOS. Il s'agit des systèmes d'exploitation les plus utilisés et le tableau 5.2 donne quelques éléments comparatifs des capacités de développement sur ces deux types de plateforme.

Sur iOS, le développement nécessite l'utilisation de Xcode IDE avec le SDK iOS très performant et un simulateur efficace. Mais publier une application Apple est difficile, l'application mobile est testée pendant 7 jours en moyenne et les développeurs doivent payer une redevance annuelle récurrente de 99 $. Sur Android, le développement s'effectue

principalement en Java et en utilisant simultanément le SDK Android. La publication d'une application Android est simple : il suffit de signer l'application via un assistant Eclipse pra-

tique et de télécharger le fichier APK sur Google Play. Cela prend seulement quelques heures et coûte uniquement 25 $.

Après nos développements, il s'agira de rédiger et de transmettre des rapports de validation. Les rapports de validation se présentent comme illustrés dans la figure 5.3. Il s'agit d'utiliser des pas de tests avec un oracle. La personne, effectuant les tests, lit et réalise les pas de tests et valide si le résultat est conforme aux critères d'acceptation.

Si ce rapport est validé, alors les développements sont intégrés dans le tronc commun de l'application (version *release* décidée au préalable). Suivent ensuite des tests d'intégrations. Il s'agit d'effectuer des tests fonctionnels et au besoin, d'effectuer quelques débogages. Lorsque toutes ces étapes sont réalisées et sont passées avec succès, la version embarquant tous les nouveaux développements validés peut être déployée chez les clients dans un environnement de pré-production. Cet environnement est mis à disposition pour les clients afin qu'ils puissent effectuer davantage de tests et remonter les anomalies, s'il y en a. Lorsque la version dans l'environnement de pré-production est validée, cette dernière peut enfin être mise sur l'environnement de production à destination des clients finaux.

MA BIBLIOGRAPHIE PERSONNELLE

ARTICLES EN REVUE RÉFÉRENCÉE

[Hen16] Henriet, Julien and Lang, Christophe and Muthada Pottayya, Ronnie and Breschi Jimenez Ramirez, Karla. A self-adaptable distributed CBR version of the EquiVox system. Biomedical Engineering : Applications, Basis and Communications (SJR : 0.165). Vol. 28(4), September 2016, Pages :1650028 (16 pages).

PUBLICATION EN CONFÉRENCE INTERNATIONALE AVEC COMITÉ DE LECTURE D'AUDIENCE INTERNATIONALE, DONT LES ACTES SONT PUBLIÉS

[Mut17a] Muthada Pottayya, Ronnie and Lapayre, Jean-Christophe and Garcia, Eric. An Adaptive Videoconferencing Framework for Collaborative Telemedicine. Advanced Information Networking and Applications (AINA), 2017 IEEE 31st International Conference on (classée A2 dans Conference Ranks, Qualis). Pages : 197–204, Taipei - Taiwan.

[Mut17b] Muthada Pottayya, Ronnie and Lapayre, Jean-Christophe and Garcia, Eric Development of an Adaptive Videoconferencing Framework for Collaborative Telemedicine. The 21st IEEE International Conference On Computer Supported Cooperative Work in Design (CSCWD 2017) (classée B dans Core). Pages : 263–268, Wellington - Nouvelle Zélande.

AUTRES PUBLICATIONS

[Mut17c] Muthada Pottayya, Ronnie and Lapayre, Jean-Christophe and Garcia, Eric. Adaptability of multimedia streams applied to medical telediagnosis : mobility and data exchange. École d'été Rescom - la 5G et l'Internet des Objets. 13 au 17 Juin 2016, Guidel-Plage - France.

[Mut17d] Muthada Pottayya, Ronnie and Lapayre, Jean-Christophe and Garcia, Eric. Towards an adaptive videoconferencing system in secured telemedicine applications. Collegium international franco-suisse SMYLE (SMart sYstems for a better LifE). 22 au 23 Septembre 2016, EPFL, Microcity Neuchâtel - Suisse.

BIBLIOGRAPHIE

[Abo99] Gregory D Abowd, Anind K Dey, Peter J Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *International Symposium on Handheld and Ubiquitous Computing*, pages 304–307. Springer, 1999.

[Abo00] Gregory D Abowd and Elizabeth D Mynatt. Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 7(1) :29–58, 2000.

[Abo12] Saeid Abolfazli, Zohreh Sanaei, and Abdullah Gani. Mobile cloud computing : A review on smartphone augmentation approaches. *CoRR*, abs/1205.0451, 2012.

[Abr04] Chadia Abras, Diane Maloney-Krichmar, and Jenny Preece. User-centered design. *Bainbridge, W. Encyclopedia of Human-Computer Interaction. Thousand Oaks : Sage Publications*, 37(4) :445–456, 2004.

[Ahn06] Sungjin Ahn and Daeyoung Kim. Proactive context-aware sensor networks. In *European Workshop on Wireless Sensor Networks*, pages 38–53. Springer, 2006.

[Akk07] Faisal Akkawi, Atef Bader, Daryl Fletcher, Kayed Akkawi, Moussa Ayyash, and Khaled Alzoubi. Software adaptation : A conscious design for oblivious programmers. In *2007 IEEE Aerospace Conference*, pages 1–12. IEEE, 2007.

[Alf11] Germán H Alferez and Vicente Pelechano. Context-aware autonomous web services in software product lines. In *Software Product Line Conference (SPLC), 2011 15th International*, pages 100–109. Ieee, 2011.

[Ali13] Akbar Ali, Nehal Ahmad, Mohd Sharique Akhtar, and Aditya Srivastava. Session initiation protocol. *International Journal of Scientific and Engineering Research*, 4(1) :1–6, 2013.

[And14] Robert A Anderson, Paul L Bleisch, Shawn L Hargreaves, Michael T Klucher, Josefa MG Nalewabau, and Eli J Tayrien. Transporting and processing foreign data, September 23 2014. US Patent 8,843,881.

[Apa15] Apache. Apache jena framework. https://jena.apache.org. Accessed : 2015-12-18.

[Bah12] Paramvir Bahl, Richard Y Han, Li Erran Li, and Mahadev Satyanarayanan. Advancing the state of mobile cloud computing. In *Proceedings of the third ACM workshop on Mobile cloud computing and services*, pages 21–28. ACM, 2012.

[Bat10] Amal Battou, Ali El Mezouary, Chihab Cherkaoui, and Driss Mammass. The granularity approach of learning objects to support adaptability in adaptive learning systems. *Journal of Theoretical and Applied Information Technology*, 18(1) :24–34, 2010.

[Ber06] Antonia Bertolino, Wolfgang Emmerich, Paola Inverardi, and Valérie Issarny. Softure : Adaptable, reliable and performing software for the future. *Future Research Challenges for Software and Services (FRCSS)*, 2006.

[Bet10] Claudio Bettini, Oliver Brdiczka, Karen Henricksen, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, and Daniele Riboni. A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6(2) :161–180, 2010.

[Bla09] Gordon S Blair, Geoff Coulson, Phillippe Robin, and Michael Papathomas. An architecture for next generation middleware. In *Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing*, pages 191–206. Springer-Verlag, 2009.

[Bre02] Patrick Brézillon, C Kintzig, G Poulain, G Privat, and PN Favennec. Expliciter le contexte dans les objets communicants. *Les Objets Communicants*, 21 :295–303, 2002.

[Bro95] Peter J Brown. The stick-e document : a framework for creating context-aware applications. *ELECTRONIC PUBLISHING-CHICHESTER-*, 8 :259–272, 1995.

[Bro97] Peter J Brown, John D Bovey, and Xian Chen. Context-aware applications : from the laboratory to the marketplace. *IEEE personal communications*, 4(5) :58–64, 1997.

[Cal11] Silvia Calegari and Gabriella Pasi. Definition of user profiles based on the YAGO ontology. In *Proceedings of the 2nd Italian Information Retrieval (IIR) Workshop, Milan, Italy*, pages 1–4, 2011.

[Cam94] Andrew Campbell, Geoff Coulson, and David Hutchison. A quality of service architecture. *SIGCOMM Comput. Commun. Rev.*, 24(2) :6–27, April 1994.

[Cap03] Licia Capra, Wolfgang Emmerich, and Cecilia Mascolo. Carisma : Context-aware reflective middleware system for mobile applications. *IEEE Transactions on software engineering*, 29(10) :929–945, 2003.

[Cet09] Carlos Cetina, Pau Giner, Joan Fons, and Vicente Pelechano. Autonomic computing through reuse of variability models at runtime : The case of smart homes. *Computer*, 42(10) :37–43, 2009.

[Che03] Guaning Chen and David Kotz. Context-sensitive resource discovery. Technical report, DTIC Document, 2003.

[Che05] Harry Chen, Tim Finin, and Amupam Joshi. Semantic web in the context broker architecture. Technical report, DTIC Document, 2005.

[Chu14a] Kyung-Yong Chung. Recent trends on convergence and ubiquitous computing. *Personal and Ubiquitous Computing*, 18(6) :1291–1293, 2014.

[Chu14b] Kyung-Yong Chung, Junseok Yoo, and Kuinam J Kim. Recent trends on mobile computing and future networks. *Personal and Ubiquitous Computing*, 18(3) :489–491, 2014.

[DeL13] Rogério De Lemos, Holger Giese, Hausi A Müller, Mary Shaw, Jesper Andersson, Marin Litoiu, Bradley Schmerl, Gabriel Tamura, Norha M Villegas, Thomas Vogel, et al. Software engineering for self-adaptive systems : A second research roadmap. In *Software Engineering for Self-Adaptive Systems II*, pages 1–32. Springer, 2013.

[Dey99] Anind K Dey, Daniel Salber, Masayasu Futakawa, and Gregory D Abowd. An architecture to support context-aware applications. *Georgia Institute of Technology*, 1999.

[Dey00] Anind K Dey and Gregory D Abowd. Cybreminder : A context-aware system for supporting reminders. In *International Symposium on Handheld and Ubiquitous Computing*, pages 172–186. Springer, 2000.

[Dey01b] Anind K Dey. Understanding and using context. *Personal and ubiquitous computing*, 5(1) :4–7, 2001.

[Dey01a] Anind K Dey, Gregory D Abowd, and Daniel Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-computer interaction*, 16(2) :97–166, 2001.

[Dou04] Paul Dourish. What we talk about when we talk about context. *Personal and ubiquitous computing*, 8(1) :19–30, 2004.

[Dug14] J Dugan, S Elliott, B Mah, J Poskanzer, and K Prabhu. Iperf—the network bandwidth measurement tool. *URL : https :///iperf.fr/(visité le 13/01/2016)*, 2014.

[Erg15] Ergolab. La conception centrée utilisateur. http://www.ergolab.net/articles/conception-centree-utilisateur.php. Accessed : 2015-04-11.

[Etz15] Alex Etz. Understanding bayes : Updating priors via the likelihood, 2015.

[Faj15] Jose Oscar Fajardo, Ianire Taboada, and Fidel Liberal. Improving content delivery efficiency through multi-layer mobile edge adaptation. *IEEE Network*, 29(6) :40–46, 2015.

[Fer15] Denzil Ferreira, Vassilis Kostakos, and Anind K Dey. Aware : mobile context instrumentation framework. *Frontiers in ICT*, 2 :6, 2015.

[Fox98] Armando Fox, Steven D Gribble, Yatin Chawathe, and Eric A Brewer. Adapting to network and client variation using infrastructural proxies : Lessons and perspectives. *IEEE Personal Communications*, 5(4) :10–19, 1998.

[Fra98] David Franklin and Joshua Flaschbart. All gadget and no representation makes jack a dull environment. In *Proceedings of the AAAI 1998 Spring Symposium on Intelligent Environments*, pages 155–160, 1998.

[Fug14] Alfonso Fuggetta and Elisabetta Di Nitto. Software process. In *Proceedings of the on Future of Software Engineering*, pages 1–12. ACM, 2014.

[Gar02] David Garlan, Daniel P Siewiorek, Asim Smailagic, and Peter Steenkiste. Project aura : Toward distraction-free pervasive computing. *IEEE Pervasive computing*, 1(2) :22–31, 2002.

[Gar11] Gartner Company. Gartner lowers pc forecast as consumers diversify computing needs across devices. http://www.gartner.com/newsroom/id/1570714, March 2011. Consulted 2015.

[Gka08] Stavroula Gkatzidou and EJ Pearson. A vision for truly adaptable and accessible learning objects. *ASCILITE*, pages 340–342, 2008.

[Gka10] Voula Gkatzidou and Elaine Pearson. Exploring the development of adaptable learning objects. a practical approach. In *2010 10th IEEE International Conference on Advanced Learning Technologies*, pages 307–309. IEEE, 2010.

[Gol07] Maria Golemati, Akrivi Katifori, Costas Vassilakis, George Lepouras, and Constantin Halatsis. Creating an ontology for the user profile : Method and applications. In *In Proceedings of the First International Conference on Research Challenges in Information Science (RCIS*, pages 407–412, 2007.

[Goo74] D. B. Osteyee and I. J. Good. Information, weight of evidence, the singularity between probability measures and signal detection. In *Lecture Notes in Mathematics, Springer-Verlag*. Springer-Verlag, 1974.

[Gro17] B. Grozev, G. Politis, E. Ivov, T. Noel, and V. Singh. Experimental evaluation of simulcast for webrtc. *IEEE Communications Standards Magazine*, 1(2) :52–59, 2017.

[Hea13] Eric W Healy, Sarah E Yoho, Yuxuan Wang, and DeLiang Wang. An algorithm to improve speech recognition in noise for hearing-impaired listeners. *The Journal of the Acoustical Society of America*, 134(4) :3029–3038, 2013.

[Hop15] Hoppe Anett, Ana Roxin, and Christophe Nicolle. Ontology-based integration of web navigation for dynamic user profiling. *Informatica Economica*, 19(1/2015) :15, 2015.

[Hul97] Richard Hull, Philip Neaves, and James Bedford-Roberts. Towards situated computing. In *Wearable Computers, 1997. Digest of Papers., First International Symposium on*, pages 146–153. IEEE, 1997.

[Inv06] Paola Inverardi. Software of the future is the future of software ? In *Trustworthy Global Computing*, pages 69–85. Springer, 2006.

[Inv09] Paola Inverardi and Massimo Tivoli. The future of software : Adaptation and dependability. In *Software Engineering*, pages 1–31. Springer, 2009.

[Jac11] Joab Jackson. Aes proved vulnerable by microsoft researchers, 2011.

[Jan11] Jack Jansen, Pablo Cesar, Dick CA Bulterman, Tim Stevens, Ian Kegel, and Jochen Issing. Enabling composition-based video-conferencing for the home. *IEEE Transactions on Multimedia*, 13(5) :869–881, 2011.

[Jar11] Dmitri Jarnikov and Tanır Özçelebi. Client intelligence for adaptive streaming solutions. *Signal Processing : Image Communication*, 26(7) :378–389, 2011.

[Joh01] Mathias Johanson. A rtp to http video gateway. In *Proceedings of the 10th international conference on World Wide Web*, pages 499–503. ACM, 2001.

[Jun13] Eun-Young Jung, Jong-Hun Kim, Kyung-Yong Chung, and Dong Kyun Park. Home health gateway based healthcare services through u-health platform. *Wireless personal communications*, 73(2) :207–218, 2013.

[Jur04] Matjaz B Juric, Bostjan Kezmah, Marjan Hericko, Ivan Rozman, and Ivan Vezocnik. Java rmi, rmi tunneling and web services comparison and performance analysis. *ACM Sigplan Notices*, 39(5) :58–65, 2004.

[Kha13] Mohammed Fethi Khalfi and Sidi Mohamed Benslimane. Toward a generic infrastructure for ubiquitous computing. *International Journal of Advanced Pervasive and Ubiquitous Computing (IJAPUC)*, 5(1) :66–85, 2013.

[Kic97] Gregor Kiczales, John Lamping, Anurag Mendhekar, Chris Maeda, Cristina Lopes, Jean-Marc Loingtier, and John Irwin. Aspect-oriented programming. In *European conference on object-oriented programming*, pages 220–242. Springer, 1997.

[Kin02] Tim Kindberg and Armando Fox. System software for ubiquitous computing. *IEEE pervasive computing*, 1(1) :70–81, 2002.

[Kob04] Alfred Kobsa. A component architecture for dynamically managing privacy constraints in personalized web-based systems. In *International Workshop on Privacy Enhancing Technologies*, pages 177–188. Springer, 2004.

[Kru16] John Krumm. *Ubiquitous computing fundamentals*. CRC Press, 2016.

[Lil10] Lillian Hella and John Krogstie. A profile ontology for personalised mobile shopping support. *1st International Workshop on Adaptation, Personalization and REcommendation in the Social-semantic Web (APRESW 2010)*, pages 1–12, 2010.

[Lim14] Christopher Lima, Mário Antunes, Diogo Gomes, Rui Aguiar, and Telma Mota. A context-aware framework for collaborative activities in pervasive communities. *International Journal of Distributed Systems and Technologies*, pages 31–43, 2014.

[Mar14] Clarissa Cassales Marquezan, Florian Wessling, Andreas Metzger, Klaus Pohl, Chris Woods, and Karl Wallbom. Towards exploiting the full adaptation potential of cloud applications. In *Proceedings of the 6th International Workshop on Principles of Engineering Service-Oriented and Cloud Systems*, pages 48–57. ACM, 2014.

[Mck04] Philip K McKinley, Seyed Masoud Sadjadi, Eric P Kasten, and Betty HC Cheng. A taxonomy of compositional adaptation. *Technical report number MSU-CSE-04-17*, 2004.

[Mea16] Robert E. Mealey. Oscure analytics, 2016.

[Meh13] Abid Mehmood and Dayang NA Jawawi. Aspect-oriented model-driven code generation : A systematic mapping study. *Information and Software Technology*, 55(2) :395–411, 2013.

[Mit15] Karan Mitra, Arkady Zaslavsky, and Christer hlund. Context-aware qoe modelling, measurement, and prediction in mobile computing systems. *IEEE Transactions on Mobile Computing*, 14(5) :920–936, 2015.

[Mod06] Martin Modahl, Bikash Agarwalla, T Scott Saponas, Gregory Abowd, and Umakishore Ramachandran. Ubiqstack : a taxonomy for a ubiquitous computing software stack. *Personal and Ubiquitous Computing*, 10(1) :21–27, 2006.

[Moo85] Robert C Moore. Semantical considerations on nonmonotonic logic. *Artificial intelligence*, 25(1) :75–94, 1985.

[Moo12] Hyungsik Roger Moon and Frank Schorfheide. Bayesian and frequentist inference in partially identified models. *Econometrica*, 80(2) :755–782, 2012.

[Nar14] Ram Lakshmi Narayanan, Yinghua Ye, Anuj Kaul, and Mili Shah. Mobile video streaming. *Advanced Content Delivery, Streaming, and Cloud Services*, pages 141–158, 2014.

[New03] Alan Newberger and Anind Dey. Designer support for context monitoring and control. *IRB-TR-03-017, Intel Research Berkeley*, 2003.

[Opp97] Reinhard Oppermann and R Rasher. Adaptability and adaptivity in learning systems. *Knowledge transfer*, 2 :173–179, 1997.

[Pas98] Jason Pascoe. Adding generic contextual capabilities to wearable computers. In *Wearable Computers, 1998. Digest of Papers. Second International Symposium on*, pages 92–99. IEEE, 1998.

[Per14] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. Context aware computing for the internet of things : A survey. *IEEE Communications Surveys & Tutorials*, 16(1) :414–454, 2014.

[Rah14] M Reza Rahimi, Jian Ren, Chi Harold Liu, Athanasios V Vasilakos, and Nalini Venkatasubramanian. Mobile cloud computing : A survey, state of art and future directions. *Mobile Networks and Applications*, 19(2) :133–143, 2014.

[Rah15] Ahmad Rahmati, Clayton Shepard, Chad Tossell, Lin Zhong, and Philip Kortum. Practical context awareness : measuring and utilizing the context dependency of mobile usage. *IEEE Transactions on Mobile Computing*, 14(9) :1932–1946, 2015.

[RFC3170] Bob Quinn and Kevin Almeroth. Rfc 3170 : Ip multicast applications : Challenges and solutions, september 2001. *Status : Informational*, 2001.

[Riv00] Michel Riveill, Marie-Claude Pellegrini, Olivier Potonniée, and Raphaël Marvie. Adaptabilité des applications pour des usagers mobiles. *OCM 2000*, 2000.

[Rod98] Tom Rodden, Keith Cheverst, K Davies, and Alan Dix. Exploiting context in hci design for mobile systems. In *Workshop on human computer interaction with mobile devices*, pages 21–22. Glasgow, 1998.

[Roh16] G Rohini and A Srinivasan. Multi server based cloud-assisted real-time transrating for http live streaming. *Indian Journal of Science and Technology*, 9(3), 2016.

[Rus08] Aaron Russ, Wolfgang Hesse, and Dirk Müller. Ambient information systems-do they open a new quality of is ? In *SIGSAND-EUROPE*, pages 93–108, 2008.

[Rya99] Nick Ryan, Jason Pascoe, and David Morse. Enhanced reality fieldwork : the context aware archaeological assistant. *Bar International Series*, 750 :269–274, 1999.

[Sac05] Daniele Sacchetti, Y Bromberg, Nikolaos Georgantas, Valérie Issarny, Jorge Parra, and Remco Poortinga. The amigo interoperable middleware for the networked home environment. In *6th International Middleware Conference, Workshops Proceedings, Grenoble, France*, 2005.

[Sah03] Debashis Saha and Amitava Mukherjee. Pervasive computing : a paradigm for the 21st century. *Computer*, 36(3) :25–31, 2003.

[Sca02] Walt Scacchi. *Process Models in Software Engineering*, pages 993–1005. John Wiley & Sons, Inc., 2002.

[Sch94] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on*, pages 85–90. IEEE, 1994.

[Sha86] Marc Shapiro. Structure and encapsulation in distributed systems : the proxy principle. In *Int. Conf. on Distr. Comp. Sys.(ICDCS)*, pages 198–204, 1986.

[Ski12] Kerry-Louise Skillen, Liming Chen, ChrisD. Nugent, MarkP. Donnelly, William Burns, and Ivar Solheim. Ontological user profile modeling for context-aware application personalization. In *Ubiquitous Computing and Ambient Intelligence*, volume 7656 of *Lecture Notes in Computer Science*, pages 261–268. Springer Berlin Heidelberg, 2012.

[Smi82] Brian Cantwell Smith. *Procedural reflection in programming languages*. PhD thesis, Massachusetts Institute of Technology, 1982.

[Sou02] João Pedro Sousa and David Garlan. Aura : an architectural framework for user mobility in ubiquitous computing environments. In *Software Architecture*, pages 29–43. Springer, 2002.

[Sta08] Johann Stan, Elod Egyed-Zsigmond, Adrien Joly, and Pierre Maret. A user profile ontology for situation-aware social networking. In *3rd Workshop on Artificial Intelligence Techniques for Ambient Intelligence (AITAmI2008)*, 2008.

[Ste87] Mark Stefik, Gregg Foster, Daniel G Bobrow, Kenneth Kahn, Stan Lanning, and Lucy Suchman. Beyond the chalkboard : computer support for collaboration and problem solving in meetings. *Communications of the ACM*, 30(1) :32–47, 1987.

[Str08] John Strassner, Yan Liu, Michael Jiang, Jing Zhang, Sven van der Meer, Mícheál Ó Foghlú, Claire Fahy, and Willie Donnelly. Modelling context for autonomic networking. In *Network Operations and Management Symposium Workshops, 2008. NOMS Workshops 2008. IEEE*, pages 299–308. IEEE, 2008.

[The99] David Thevenin and Joëlle Coutaz. Plasticity of user interfaces : Framework and research agenda. In *Proceedings of INTERACT*, volume 99, pages 110–117, 1999.

[USN84] TTCP USNA. a test of tcp and udp performance, 1984.

[Van08] Lex Van Velsen, Thea Van Der Geest, Rob Klaassen, and Michael Steehouder. User-centered evaluation of adaptive and adaptable systems : a literature review. *The knowledge engineering review*, 23(03) :261–281, 2008.

[Wan05] Roy Want and Trevor Pering. System challenges for ubiquitous & pervasive computing. In *Proceedings of the 27th international conference on Software engineering*, pages 9–14. ACM, 2005.

[Wan07] Wang Yanping and Wang Yiding. Based on genetic algorithm, wap image self-adaptive transfer technology. *School of Computer Science and Engineering, Wuhan University, Wuhan 430072, China*, 33(11) :196–198, 2007.

[War97] Andy Ward, Alan Jones, and Andy Hopper. A new location technique for the active office. *IEEE Personal Communications*, 4(5) :42–47, 1997.

[Wei93] Mark Weiser. Ubiquitous computing. *Computer*, 26(10) :71–72, 1993.

[Wei99] Mark Weiser. The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.*, 3(3) :3–11, July 1999.

[Wer07] Imen Werda, Haithem Chaouch, Amine Samet, Mohamed Ali Ben Ayed, Nouri Masmoudi, E Akbal, B Ergen, H Muljadi, H Takeda, K Ando, et al. Optimal dsp-based motion estimation tools implementation for h. 264/avc baseline encoder. *IJCSNS*, 7(5) :141, 2007.

[Win01] Terry Winograd. Architectures for context. *Human-Computer Interaction*, 16(2) :401–419, 2001.

[Yeo01] Alvin W Yeo. Global-software development lifecycle : An exploratory study. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 104–111. ACM, 2001.

[Zad65] Lotfi A Zadeh. Fuzzy logic and its applications. *New York, NY, USA*, 1965.

TABLE DES FIGURES

LISTE DES TABLES

Résumé :

Dans le domaine médical, la plupart des établissements (hôpitaux, cliniques, ...) utilisent des applications distribuées dans le cadre de la télémédecine.
Comme la sécurité de l'information est primordiale dans ces établissements, ces applications doivent pouvoir traverser les barrières de sécurité (passerelles sécurisées comme les proxies Web, les pare-feu, ...). Le protocole UDP (User Datagram Protocol en anglais), qui est classiquement recommandé pour les applications de vidéoconférence ou toutes autres données soumises à la contrainte temps-réel, n'est pas utilisable par ces dispositifs de sécurité (sauf si des ports fixes sont explicitement configurés : ce qui est considéré comme une violation de sécurité au sein de ces établissements).
Dans cette thèse, nous proposons une nouvelle plateforme appelée VAGABOND (Video Adaptation framework, crossing security GAteways, Based ON transcoDing) qui fonctionne de manière très efficace et originale sur la base du protocole TCP (Transmission Control Protocol). VAGABOND est composée de proxies d'adaptation, appelés des AP (pour Adaptation Proxy), qui ont été conçus pour prendre en considération les préférences utilisateurs des professionnels de santé, les hétérogénéités des périphériques, et les variations dynamiques de la bande passante dans un réseau. VAGABOND est capable de s'adapter tout aussi bien au niveau utilisateur qu'au niveau réseau.
La loi binômiale et l'inférence bayésienne sur une proportion binômiale sont utilisées pour déclencher des adaptations de profils utilisateurs. Ainsi, nous souhaitons être plus tolérants aux fortes variations de la bande passante d'un réseau. Avec une précision plus fine et grâce à ces lois de probabilité, l'adaptation n'est déclenchée que lorsque des congestions réseau sévères surviennent. Enfin, TCP étant un protocole de transport fiable et en mode connecté, nous avons eu besoin de concevoir et d'utiliser de nouvelles stratégies d'adaptation intelligentes avec la transmission de données afin de faire face aux problèmes de latence et à la temporisation des sockets.

Mots-clés : Adaptation distribuée, Proxies, Vidéo sur TCP, loi Binômiale, inférence Bayésienne, Télémédecine, et la Vidéoconférence

Abstract:

In the medical area, most of medical facilities (hospitals, clinics, ...) use distributed applications in the context of telemedecine.
As information security is mandatory, these applications must be able to cross the security protocols (secured gateways like proxies, firewalls, ...). User Datagram Protocol (UDP), which is classically recommended for videoconferencing applications, does not cross firewalls or proxies unless explicitly configured fixed ports are declared. These fixed ports are considered as a security breach.
In this thesis, we propose a novel platform called VAGABOND (Video Adaptation framework, crossing security GAteways, Based ON transcoDing) which works, in a very efficient and original way; on TCP (Transmission Control Protocol). VAGABOND is composed of Adaptation Proxies (APs), which have been designed to take into consideration medical experts videoconferencing preferences, device heterogeneities, and network dynamic bandwidth variations. VAGABOND is able to adapt itself at the user and network levels.
The cumulative binomial probability law and the Bayesian inference on a binomial proportion are used to trigger adaptations. In fact, we aim at being more tolerant to severe network bandwidth variations. With a finer precision and following these probability laws, user profile adaptation is only triggered when severe network congestions arise. However, as TCP is a reliable transport protocol, we needed to design and to employ new intelligent adaptation strategies together with data transmission in order to cope with latency issues and sockets timeout.

Keywords: Distributed adaptation, Proxies, Video on TCP, Binomial law, Bayesian inference, Telemedicine, and Videoconferencing

■ École doctorale SPIM 16 route de Gray F - 25030 Besançon cedex
■ tél. +33 (0)3 81 66 66 02 ■ ed-spim@univ-fcomte.fr ■ www.ed-spim.univ-fcomte.fr

UNIVERSITÉ DE FRANCHE-COMTÉ