G. Leroy
A. Lefebvre
A. Blanc
C. Couturier
R. Ludinard
IMT Atlantique
October 11, 2018

Groupe 13: Proposition de specification de protocole c2w
Protocole de specification

Abstract

Specification pour la version binaire du protocole de c2w. L'enjeu de ce protocole est de permettre la communication entre le serveur et les clients de l'application Chat While Watching. Il s'agit d'une application constituee d'une "salle principale" ou les utilisateur peuvent chater ou choisir un film ainsi que des salles correspondantes au films. Les utilisateurs regardant un meme film accede a un chat dedie.

Table of Contents

1. Introduction 2
2. Format des paquets 2
2.1. Format general 3
2.2. Format du paquet de type 0 : acquittement 4
2.3. Format du paquet de type 1 : envoi du pseudo 4
2.4. Format du paquet de type 2 : selection du film 5
2.5. Format du paquet de type 3 : quitter la salle de film . . 6
2.6. Format du paquet de type 4 : quitter l'application . . . 6
2.7. Format du paquet de type 5 : envoi de la liste des films 7
2.8. Format du paquet de type 6 : liste des utilisateurs actuels 8
2.9. Format du paquet de type 7 : message du chat 9
2.10. Format du paquet de type 8 : Acceptation de connexion . . 10
2.11. Format du paquet de type 9 : Refus de connexion 10
3. architecture globale 10
4. Consequences des messages 11
4.1. Dans le cas du serveur : 11
4.2. Dans le cas du client : 13
5. Fiabilisation 13
5.1. Introduction 13
5.2. Architecture 14
5.3. Protocole Send & Wait 15
5.3.1. Emetteur: 15
5.3.2. Rupture de connexion: 16

5.3.3. Recepteur: 17
6. Exemple : 17
7. Exemples de scenarios possibles 19
7.1. Scenario 1: Connexion et choix d'un film 19
7.2. Scenario 2: Communication 20
7.3. Scenario 3: Changement de salle 21
8. Conclusion 22
Authors' Addresses 22

1. Introduction

Le protocole doit permettre la communication entre le serveur de l'application Chat While Watching et les utilisateurs. Le protocole supporte TCP et UDP pour l'envoi des paquets c2w. Le serveur peut envoyer trois types de paquets vers un client: la liste de films, la liste des utilisateurs connectes et des messages de chat. Le client peut envoyer quatre types de paquets vers le serveur : connexion sur l'application, entree dans une salle de film, sortie d'une salle de film, messages de chat et sortie de l'application. On suppose que tous les clients connaissent l'adresse et le numero de port du serveur c2w. Il n'y a aucune communication directe client-client.

Hypothese : Deux clients ont necessairement des couples (adresse IP, port) distincts.

Hypothese : On fixe le nombre maximal de reemissions a 7 et l'intervalle de reemission a 1 seconde.

Hypothese : Lors d'une session (duree delimitee par l'instant de connexion et l'instant ou l'utilisateur quitte l'application), un client envoie moins de 65535 paquets (acquittements non inclus). De meme, le serveur envoie moins de 65535 paquets (acquittements non inclus) a un meme client lors d'une session.

Hypothese : Les champs contenant du texte sont encodes en UTF-8.

Hypothese : La longueur des pseudos et des noms de film ne doit pas exceder 251 octets une fois encode en UTF-8.

Hypothese : Les octets sont organises dans l'ordre standard des communications reseau (big endian).

2. Format des paquets

Il existe differents types de paquets. Tous les paquets commencent par un meme entete.

2.1. Format general

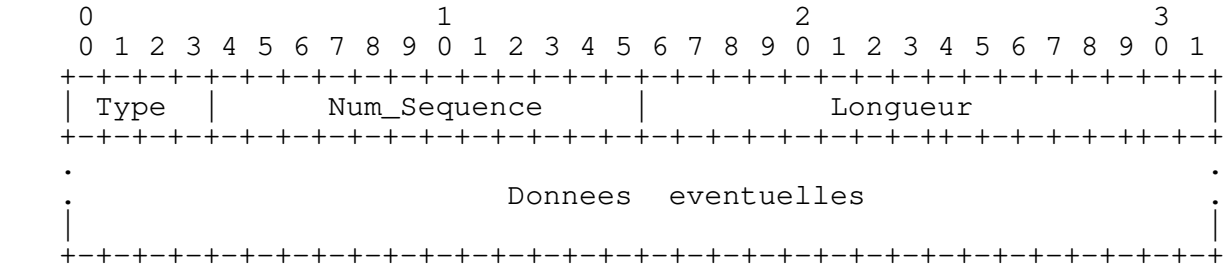


Figure 1

Type (4 bits): Valeur de 0 a 15 indiquant le type de paquet comme indique ci-dessous (Table 1).

Num_Sequence (12 bits): Numerotation des paquets pour controler les envois et receptions, gerer l'ordre et les pertes de paquets. Chaque entite, serveur ou client, a son propre compteur. Ce compteur est initialise a 0 a chaque connexion et on l'incremente a chaque envoi de paquet. Num_Sequence prend la valeur du compteur de l'emetteur avant incrementation. Son utilite est expliquee plus en detail dans la partie "Fiabilisation" (Section 5).

Longueur (16 bits): Entier non signe donnant la longueur totale du paquet (entete + donnees) en octets.

Donnees (longueur variable): Ce champ varie selon le type de paquet et peut etre eventuellement vide ou divise en plusieurs champs. comme specifie plus bas pour chaque type de paquet.

Type	Codage binaire	Description	Emetteur
0	0000	Acquittement	client+serveur
1	0001	Envoi du pseudo	client
2	0010	Selection du film	client
3	0011	Quitter la salle de film	client
4	0100	Quitter l'application	client
5	0101	Liste des films	serveur
6	0110	liste des utilisateurs	serveur
7	0111	Message du chat	client+serveur
8	1000	Acceptation de connexion	serveur
9	1001	Refus de connexion	serveur

Table 1: Types de paquets

2.2. Format du paquet de type 0 : acquittement

Il s'agit d'un paquet du client vers le serveur ou du serveur vers un client qui represente un acquittement. Il est envoye pour confirmer au destinataire qu'un paquet provenant de lui et ayant pour numero de sequence Num_Sequence a ete recu.

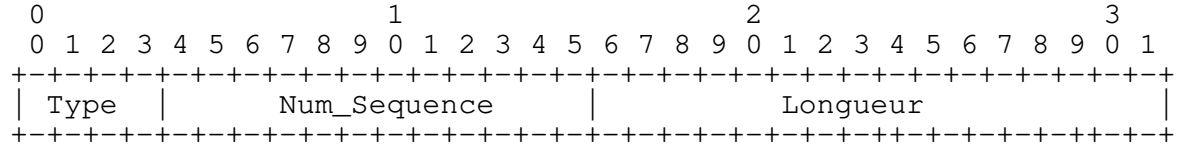


Figure 2

Type (4 bits): 0000 pour annoncer que le paquet est un acquittement (cf. Table 1).

Num_Sequence (12 bits): No de sequence. Sa valeur est celle du Num_Sequence du dernier paquet (de type autre que acquittement) recu de cet emetteur (cf. Section 2.1).

2.3. Format du paquet de type 1 : envoi du pseudo

Il s'agit d'un paquet du client vers le serveur specifiant le pseudo choisi par l'utilisateur au moment de sa connexion.

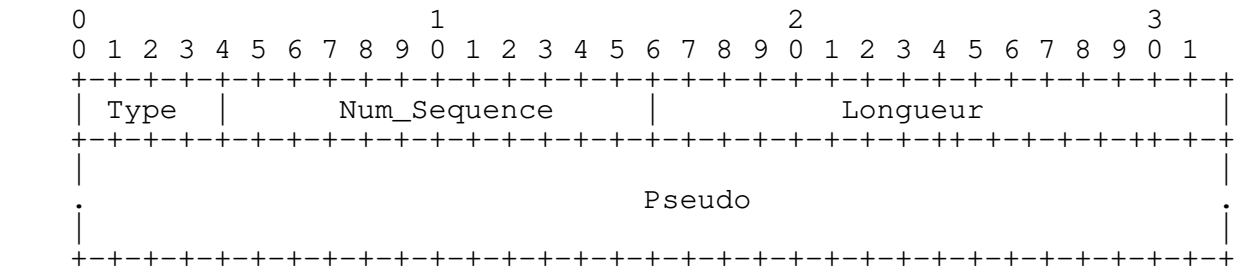


Figure 3

Type (4 bits): 0001 pour annoncer que le paquet correspond a l'envoi du pseudo (cf. Table 1).

Num_Sequence (12 bits): No de sequence (cf. Section 2.1)

Pseudo (longueur variable): Ce champ est une chaine de caracteres correspondant au pseudo choisi par l'utilisateur. Tout caractere UTF-8 est accepte. La longueur du pseudo est deduite de la longueur totale du message.

2.4. Format du paquet de type 2 : selection du film

Il s'agit d'un paquet du client vers le serveur qui envoie le nom du film choisi par l'utilisateur. Lorsque le serveur recoit ce paquet, il met a jour le status de l'utilisateur. Le status passe de 'A' a 'M'. Le client garde aussi en memoire son status.

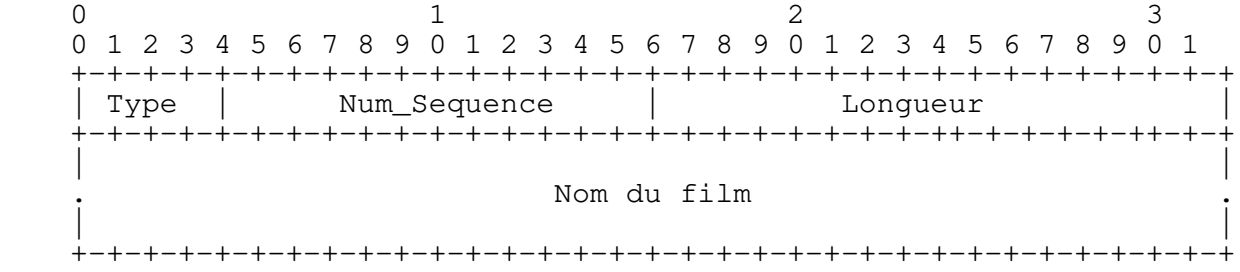


Figure 4

Type (4 bits): 0010 pour annoncer que le paquet correspond a la selection d'un film (cf. Table 1).

Num_Sequence (12 bits): No de sequence (cf. Section 2.1)

Nom du film (longueur variable): Ce champ est une chaine de caracteres correspondant au nom du film choisi par

l'utilisateur. Tout caractere UTF-8 est accepte. La longueur du nom du film est deduite de la longueur totale du message.

2.5. Format du paquet de type 3 : quitter la salle de film

Il s'agit d'un paquet du client vers le serveur pour signaler que l'utilisateur quitte la salle de film et revient dans la salle principale. Lorsque le serveur recoit ce paquet, il met a jour le status de l'utilisateur. Le status passe de 'M' a 'A'.

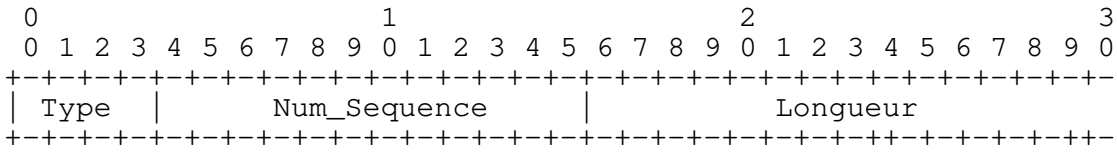


Figure 5

Type (4 bits): 0011 pour annoncer que l'utilisateur quitte la salle de film (cf. Table 1).

Num_Sequence (12 bits): No de sequence (cf. Section 2.1)

2.6. Format du paquet de type 4 : quitter l'application

Il s'agit d'un paquet du client vers le serveur pour signaler que l'utilisateur quitte l'application c2w. Lorsque le serveur recoit ce paquet, l'utilisateur disparait du systeme (donnees supprimees du serveur) et la liste d'utilisateur est mise a jour. Le client emetteur se ferme juste apres l'emission de ce paquet. Remarque: ce paquet sera tout de meme acquitte; si le paquet n'atteint pas le serveur, la gestion des erreurs obligera le serveur a supprimer les donnees de l'utilisateur et a mettre a jour la liste d'utilisateur.

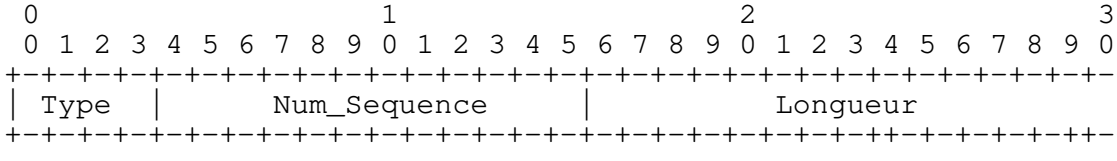


Figure 6

Type (4 bits): 0100 pour annoncer que l'utilisateur quitte l'application (cf. Table 1).

Num_Sequence (12 bits): No de sequence (cf. Section 2.1)

2.7. Format du paquet de type 5 : envoi de la liste des films

Il s'agit d'un paquet du serveur vers le client pour specifier les films disponibles et l'adresse et le port du serveur ou ils sont disponibles. La liste des films ne varie pas lors du fonctionnement de l'application, ce paquet n'est donc envoye qu'une fois a chaque utilisateur (au moment de leur connexion, juste apres la liste des utilisateurs)

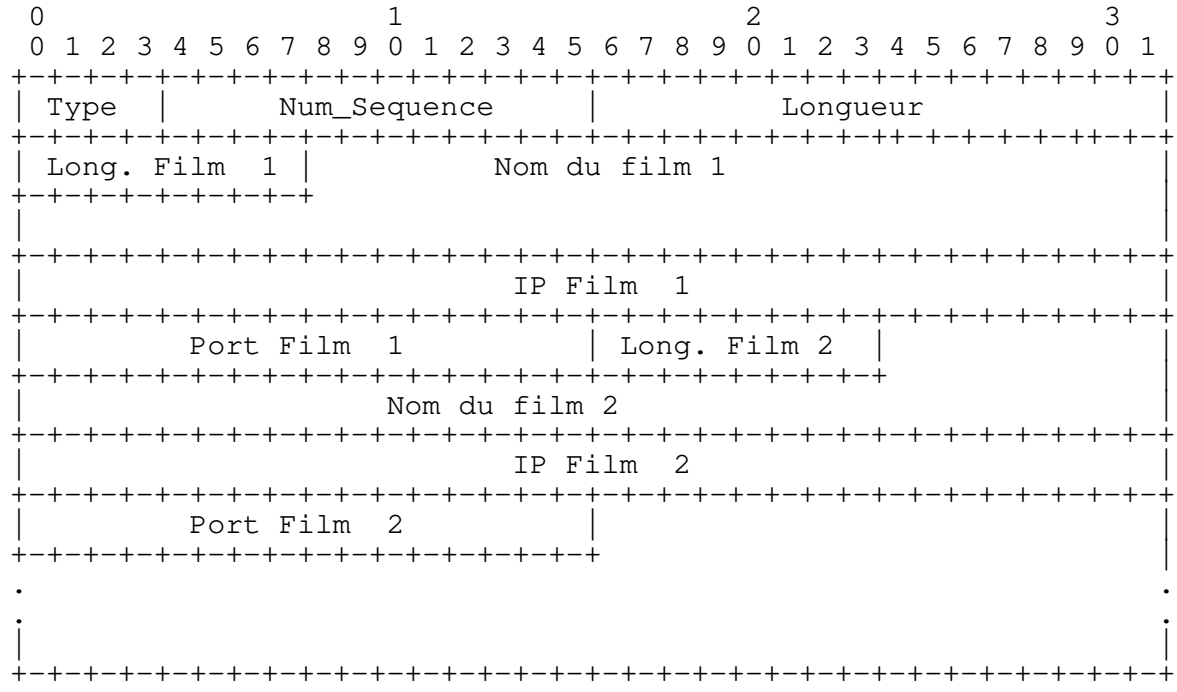


Figure 7

Type (4 bits): 0101 pour annoncer que le paquet correspond a la liste des films disponibles et de leur coordonnees (cf. Table 1).

Num_Sequence (12 bits): No de sequence (cf. Section 2.1)

Long. Film i (8 bits): Entier non signe indiquant la longueur du nom du film i en octets.

Nom du film i (longueur variable): Ce champ est une chaine de caracteres correspondant au nom du film i. La longueur est limitee a 256 octets une fois encode en UTF-8.

2.8. Format du paquet de type 6 : liste des utilisateurs actuels

Il s'agit d'un paquet du serveur vers le client pour specifier les utilisateurs presents. Si le paquet est a destination d'un utilisateur dans la salle principale, tous les utilisateurs presents dans le systeme sont listes. S'il est dans une salle de film, seuls les utilisateurs presents dans cette salle seront listes. Dans les deux cas, on precise aussi leur statuts, c'est a dire A pour un utilisateur dans la salle principale et M pour un utilisateur dans une salle de film. Lorsqu'un utilisateur change de salle, ce paquet doit etre emis pour chaque utilisateur present dans la salle de depart et la salle d'arrivee (ie ce paquet est recu par un utilisateur aussitot qu'une mise a jour de la liste des utilisateurs present dans sa salle de chat est necessaire).

Note: lorsqu'un utilisateur vient d'etre accepte, ses coordonnees figurent bien dans la liste qui lui est envoyee par le serveur.

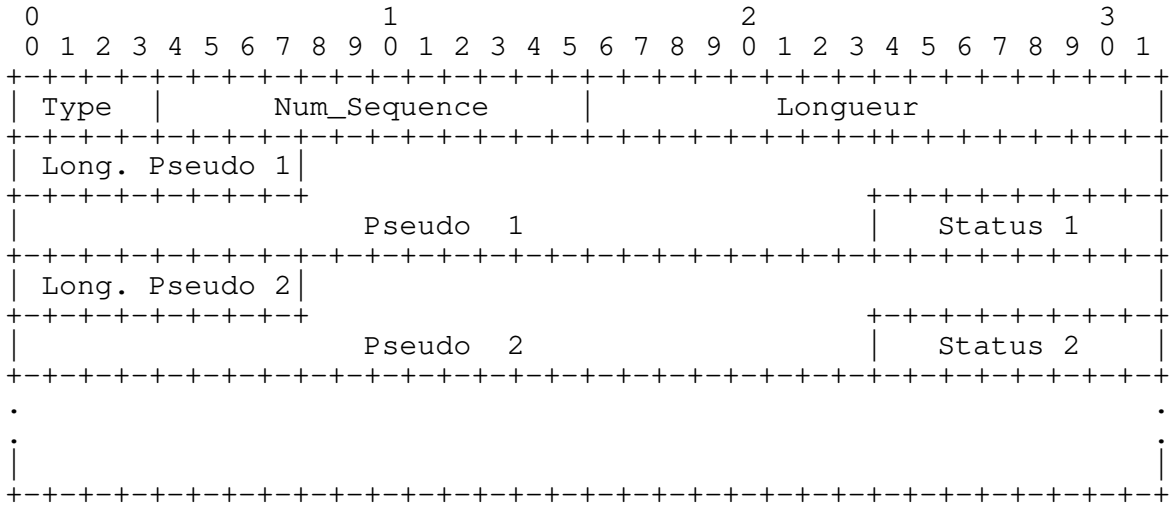


Figure 8

Type (4 bits): 0110 pour annoncer que le paquet correspond a une liste d'utilisateurs (cf. Table 1). L'utilisateur destinaire doit connaitre son propre status en permanence. Il peut donc determiner s'il s'agit de la liste de tous les utilisateurs du systeme ou seulement de ceux present dans la meme salle de film.

Num_Sequence (12 bits): No de sequence (cf. Section 2.1)

Long. Pseudo i (8 bits): Entier non signe indiquant la longueur du pseudo de l'utilisateur i en octets.

Pseudo i (longueur variable): Ce champ est une chaine de caracteres correspondant au pseudo de l'utilisateur i. La longueur est limitee a 256 octets une fois encode en UTF-8.

Status i (8 bits): Entier non signe valant 0 si l'utilisateur associe est dans la salle principale ('A') ou 1 s'il est dans une salle de film ('M').le status (A ou M) de l'utilisateur i.

2.9. Format du paquet de type 7 : message du chat

Il s'agit d'un paquet du client vers le serveur ou du serveur vers un client qui contient un message de chat. S'il est a destination d'un client, son contenu est affiche dans le chat. S'il est a destination du serveur, le message est retransmis a chaque utilisateur present dans la salle de l'emetteur (emetteur inclu).

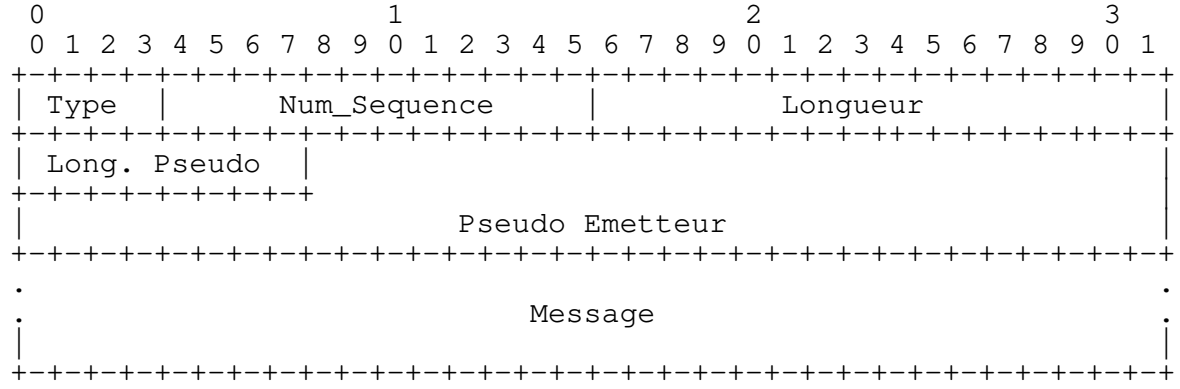


Figure 9

Type (4 bits): 0111 pour annoncer que le paquet correspond a un message de chat (cf. Table 1).

Num_Sequence (12 bits): No de sequence (cf. Section 2.1)

Long. Pseudo (8 bits): Entier non signe indiquant la longueur du pseudo de l'emetteur du message.

Pseudo (longueur variable): Ce champ est une chaine de caracteres correspondant au pseudo de l'emetteur du message. La longueur est limitee a 256 octets une fois encode en UTF-8.

Message (longueur variable): Ce champ est une chaine de caracteres correspondant au message transmis. Tout caractere est accepte. La longueur du message est deduite de la longueur totale du message.

2.10. Format du paquet de type 8 : Acceptation de connexion

Il s'agit d'un paquet du serveur vers le client pour lui confirmer que sa demande de connexion a ete acceptee.

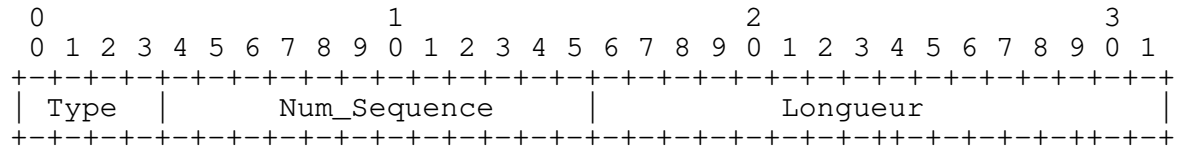


Figure 10

Type (4 bits): 1000 pour annoncer que le paquet est une acceptation de connexion. (cf. Table 1).

Num_Sequence (12 bits): No de sequence (cf. Section 2.1)

2.11. Format du paquet de type 9 : Refus de connexion

Il s'agit d'un paquet du serveur vers le client pour lui indiquer que sa demande de connexion a ete refusee. Cela peut du a differentes raisons (pseudo deja pris, pseudo non conforme aux specifications...).

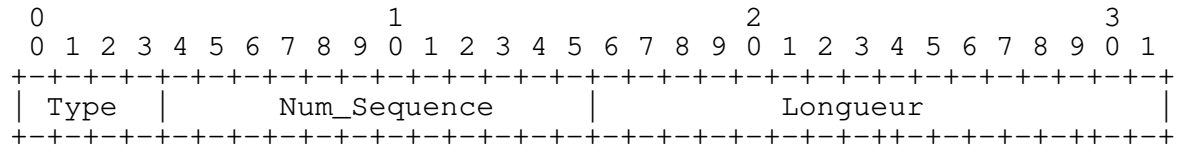
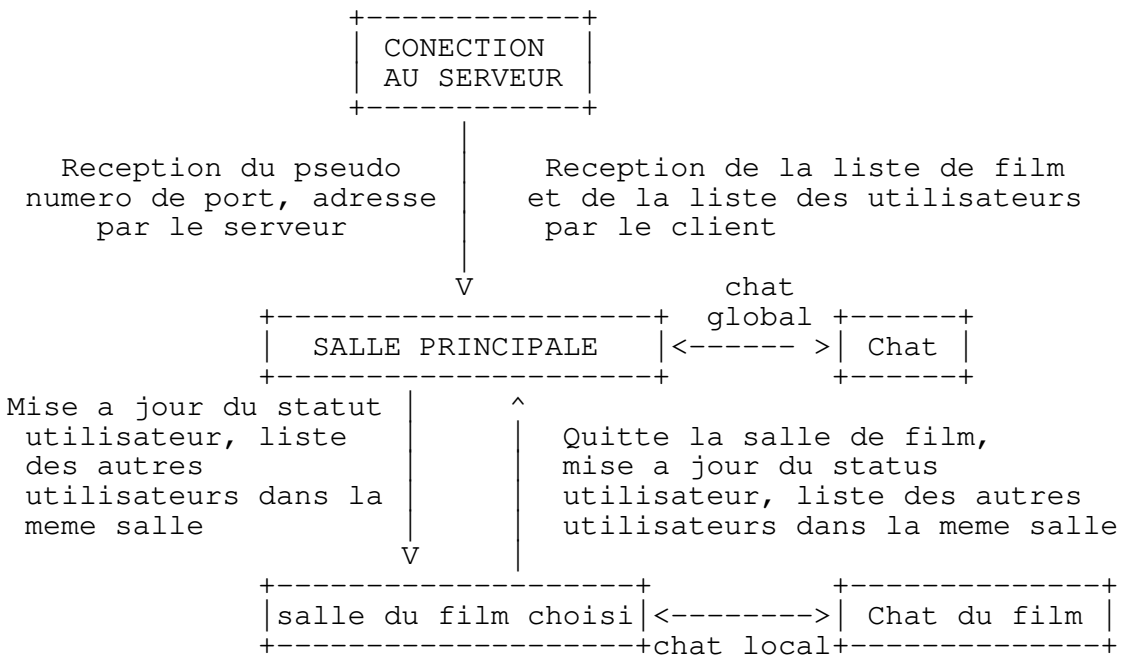


Figure 11

Type (4 bits): 1001 pour annoncer que le paquet est un refus de connexion. (cf. Table 1).

Num_Sequence (12 bits): No de sequence (cf. Section 2.1)

3. architecture globale



Le systeme ne comprend que des liaisons serveur-utilisateurs

4. Consequences des messages

4.1. Dans le cas du serveur :

A la reception d'un paquet, le serveur effectue les actions suivantes:

Type message recu	Reaction du serveur
Type 1 (envoi du pseudo)	- si nouveau client, initialisation des variables pour la gestion des erreurs a l'emission et a la reception pour ce client - si le pseudo et deja pris: envoyer un refus de connexion (type 9) et s'arreter la (i.e. supprimer les donnees liees a l'utilisateur une fois l'acquittement recu). Sinon, envoyer une acceptation de connexion (type 8) et enchainer les actions suivantes: - stockage en memoire du pseudo, de l'IP et du port du client. - le client est place dans la salle principale. - mise a jour et envoi de la liste des utilisateurs aux utilisateurs dans la salle principale (type 6). - envoi de la liste des films (type 5)
Type 2 (selection du film)	- changement de la salle en memoire. - mise a jour et envoi de la liste des utilisateurs aux utilisateurs dans la salle principale et dans la salle du film choisi (type 6).
Type 3 (quitter la salle de film)	- changement de la salle en memoire. - mise a jour et envoi de la liste des utilisateurs aux utilisateurs dans la salle principale et dans la salle quittee.
Type 4 (quitter l'application)	- supprime les donnees liees a l'utilisateur - mise a jour et envoi de la liste des utilisateurs aux utilisateurs dans la salle principale (type 6).
Type 7 (message du chat)	- renvoi le message a tous les clients (emetteur inclus) presents dans la meme salle que le client emetteur.
Type 0 (Acquittement)	- stoppe les reemissions pour le paquet portant le no de sequence acquitte. - supprime le paquet de la liste des paquets en attente d'acquittement.

Figure 12

4.2. Dans le cas du client :

A la reception d'un paquet, le client effectue les actions suivantes:

Type message recu	Reaction du client
Type 5 (envoi de la liste des films)	- stockage de la liste des films. - affichage dans l'interface de la liste des films.
Type 6 (liste des utilisateurs actuels)	- met a jour l'affichage de la liste des utilisateurs present dans la salle de film (si le client regarde un film) ou dans la salle principale (si le client est dans la salle principale).
Type 7 (message du chat)	- affiche le message dans le chat sauf s'il en est l'emetteur.
type 8 (Acceptation de connexion)	- ne fait rien.
type 9 (Refus de connexion)	- affiche un message d'erreur a l'utilisateur. - quitte l'application (ou relance la fenetre de demande de connexion.
type 0 (Acquittement)	- stoppe les reemissions pour le paquet portant le no de sequence acquitte. - supprime le paquet de la liste des paquets en attente d'acquittement.

Figure 13

5. Fiabilisation

5.1. Introduction

Le protocole c2w doit pouvoir fonctionner au dessus d'UDP comme de TCP. Il doit donc notamment pallier les imperfections d'UDP. En particulier, avec UDP on peut:

o perdre des paquets

- recevoir certains paquets en retard
- ne pas recevoir les paquets dans l'ordre ou ils ont ete emis
- recevoir certains paquets en doublon (duplication de paquet)

Pour cela le protocole c2w prevoit une couche de fiabilisation. Cette couche est mise en oeuvre quel que soit le protocole de transport sous-jacent (i.e. elle sera utilise avec UDP ET avec TCP).

On notera que, telle qu'elle est specifiee ci-dessous, cette couche peut etre vue comme une sous couche independante des echanges applicatifs decrits jusqu'ici. Son implementation peut donc en etre dissociee.

5.2. Architecture

La fiabilisation du protocole c2w repose sur les choix suivants:

- Independance des noeuds: Tous les equipements (aka noeuds) sont symetriques. On ne parle donc pas de client ou de serveur mais plutot d'emetteur et de recepteur. Un meme noeud joue alternativement les deux roles: emetteur lorsqu'il envoie des paquets et recepteur lorsqu'il en recoit.

- Independance communications: Les differents moyens mis en place (compteurs, timers...) sont dedies a une connexion emetteur-recepteur donnee. Si un noeud entretient des connexions avec plusieurs autres noeuds (eg. un serveur ayant plusieurs clients), il doit instancier un systeme de controle pour chacune de ces connexions.

- Protocole Send & Wait: Le principe retenu repose sur une transmission de type Send & Wait telle que decrite au paragraphe suivant (Table 1).

- Acquittement explicite: Chaque message (sauf les acquittements) fait l'objet d'un acquittement par un message specifique (de type 0).

- Numeros de sequence: Un numero de sequence permet d'identifier chaque message (ses eventuelles repetitions portant le meme numero) et d'y associer l'acquittement correxspondant.

5.3. Protocole Send & Wait

On s'interesse a une liaison unidirectionnelle emetteur-recepteur:

- l'emetteur envoie des paquets de type 1 a 7 et recoit des acquittements.
- le recepteur envoie des acquittements et recoit des paquets.

Tous les messages recus (sauf les acquittements) sont systematiquement acquites par un message de type 0 (Acquittement) (Section 2.2) dont le champ Num_Sequence porte la meme valeur que celui du message acquitte.

Pour une connexion donnee, l'emetteur doit attendre d'avoir recu l'acquittementne du paquet en cours avant d'envoyer le paquet suivant.

Chaque noeud possede un compteur "Emission" et un compteur "Reception". Ces compteurs sont initialises a 0 a la connexion d'un client ie au moment ou deux liaisons emetteur-recepteur sont crees. Il y a donc 2x2=4 compteurs a initialiser lors d'une connexion d'un client au serveur. Ces compteurs permettent d'attibuer les "Num_Sequences" necessaires pour detecter les doublons, les pertes et les receptions dans le desordre.

/!\ Le serveur possede donc autant de paires de compteurs qu'il y a de clients dans le systeme. Un client connecte possede deux compteurs, un "Emission" et un "Reception".

La gestion de l'ordre et des pertes de paquets implique d'effectuer des comparaisons entre les compteurs et des valeurs du champ Num_Sequence. Pour permettre une continuite lorsque le compteur revient a 0 lorsqu'il depasse 4095 (2^12-1)

5.3.1. Emetteur:

L'emetteur garde en memoire la liste de (Num_Sequence, Paquet, Nbr de tentatives d'emission) pour chaque paquet emis non acquitte.

Lorsqu'un acquittement est recu, on effectue dans cet ordre:

- Controle d'IP et port, s'ils ne correspondent pas a un couple en memoire (au couple du serveur pour un client et a un couple client pour le serveur), un nouveau contexte est cree pour cette connexion.

- Si le numero de sequence qui est acquitte (la valeur "Num_Sequence") n'est pas celui attendu pour cette connexion, l'acquittement est ignore (c'est un doublon d'acquittement, il a deja ete recu).

- Si c'est celui qui etait attendu:

  * On stoppe la reemission de ce paquet.

  * Le compteur d'emission de cette connexion est incremente de 1 (modulo 2^12).

  * On peut proceder a l'emission du paquet (type 1 a 7) suivant (s'il y en a dans la fille d'attente).

Lorsqu'un paquet est emis, on effectue dans cet ordre:

- Si le paquet a emettre est un acquittement (type 0), le champ Num_Sequence reprend la valeur du champ le champ Num_Sequence du paquet acquitte et l'acquittement est envoye immediatement.

- Si le paquet a emettre est un paquet de donnees (type 1 a 7):

  * La valeur du compteur d'emission est recopiee dans le champ Num_Sequence du paquet.

  * Le paquet est emis.

  * Une reemission a intervalle de temps reguliers est effectuee jusqu'a la reception d'un acquittement. La longueur de l'intervalle est 1s.

  * Si le nombre de retransmissions depasse 7, l'emetteur rompt la connexion.

5.3.2. Rupture de connexion:

Si un client ou le serveur stoppe une liaison, l'autre extremite de la liaison s'arretera aussitot qu'elle tentera d'emettre ou reemettre un paquet (le nombre max de reemission finira par etre atteint). Cote serveur, toutes les donnees du client sont supprimees. Cote client, plus aucun paquet n'est emis, un message indiquant que la connexion a ete perdue est affiche et on demande au client de se reconnecter.

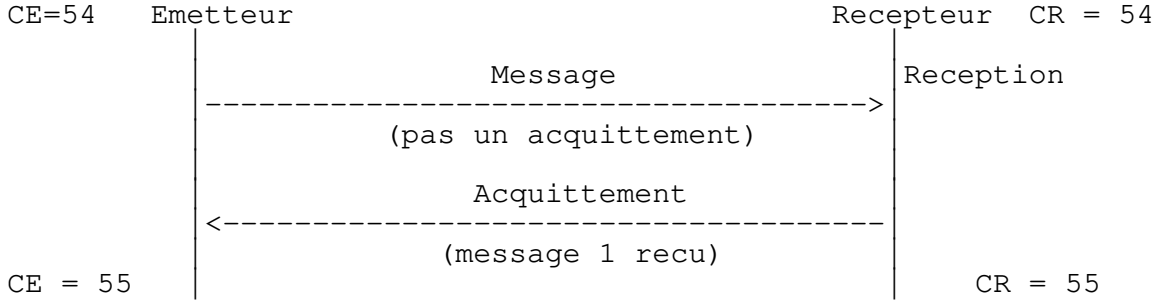
5.3.3. Recepteur:

Le recepteur garde en memoire le numero de sequence du dernier paquet recu de chaque correspondant.

Lorsqu'un paquet (autre qu'un acquittement, de type 0) est recu on effectue dans cet ordre:

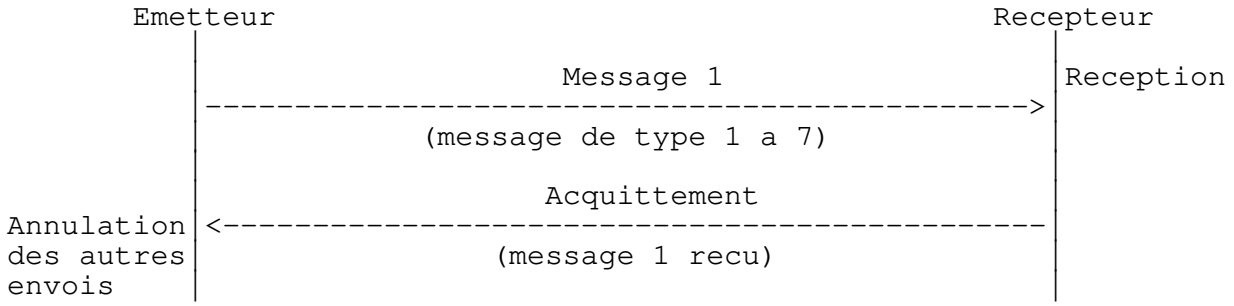
- Controle d'IP et port, s'ils ne correspondent pas a un couple en memoire (au couple du serveur pour un client et a un couple client pour le serveur), un nouveau contexte est cree pour cette connexion.
- Un acquittement est envoye (quelque soit le type (1-7) de paquet) avec pour valeur "Num_Sequence" la valeur "Num_Sequence" du paquet recu.
- Suivant la valeur de Num_Sequence:
  - Si Num_Sequence != compteur "Reception", c'est un doublon et le paquet est ignore.
  - si Num_Sequence = compteur "Reception", c'est un nouveau paquet:
    - Le message est remonte a la couche de niveau superieur.
    - Le compteur "Reception" est incremente de 1 (modulo 2^12).

6. Exemple :

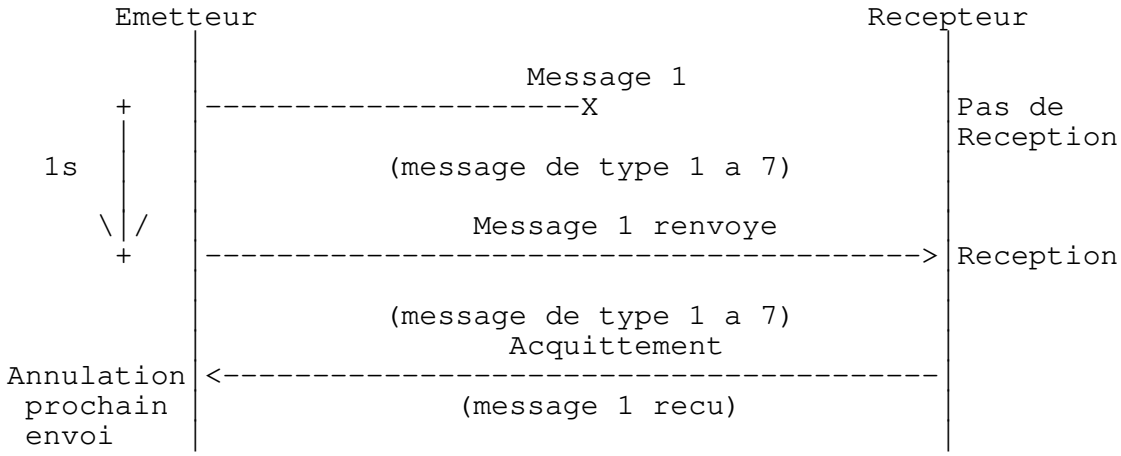


Voici la presentation schematique de quelques cas possibles:

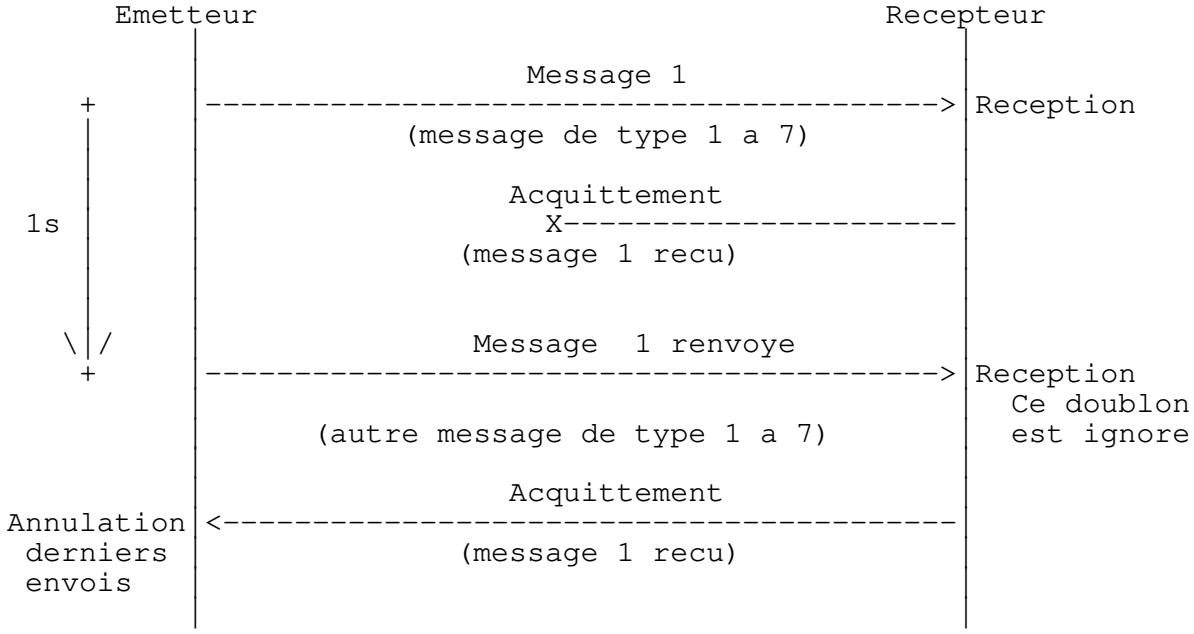
Si tout se passe bien :



S'il y a un probleme de transmission, apres un certain temps il y a reemission du paquet par l'emetteur.



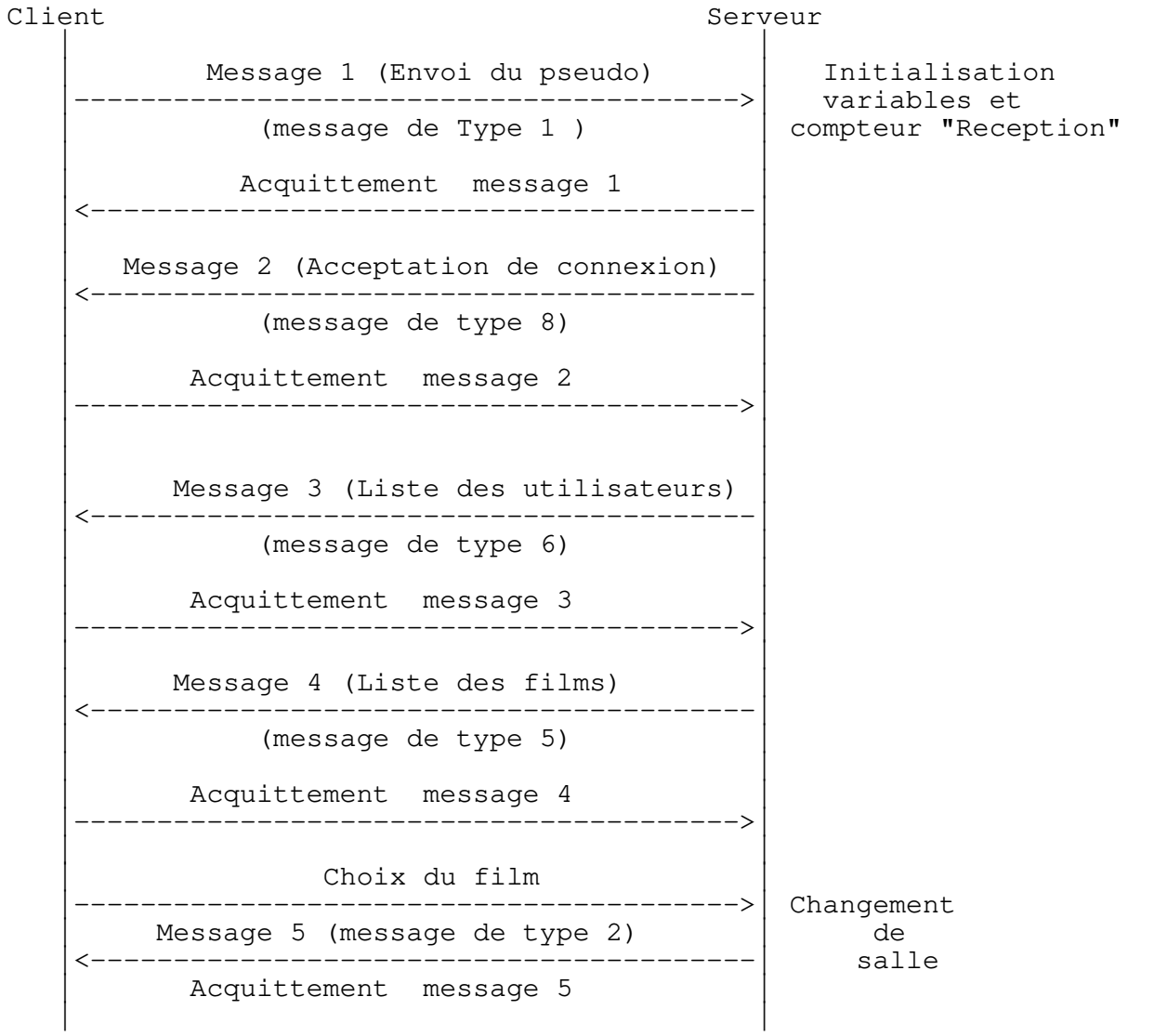
S'il y a un probleme de transmission de l'acquittement, il y a reemission du paquet par l'emetteur automatiquement car il n'y a pas annulation.



7. Exemples de scenarios possibles

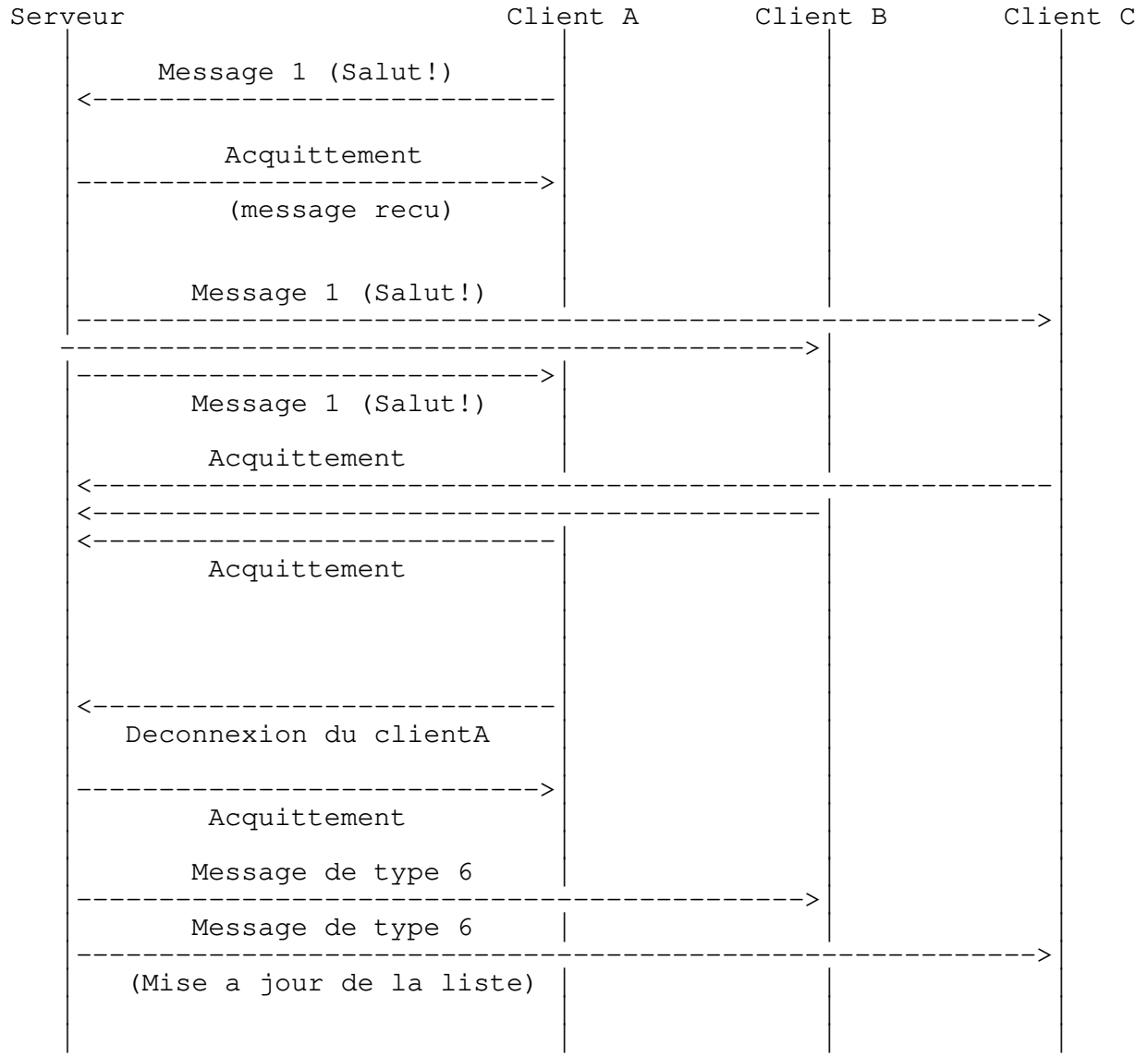
7.1. Scenario 1: Connexion et choix d'un film

Dans ce cas tres simple d'utilisation, le client va se connecter sur le serveur. Le serveur recoit alors son identifiant et lui attribue toutes les variables initilialisees par defaut et correspondant a l'arrivee d'un nouvel individu.



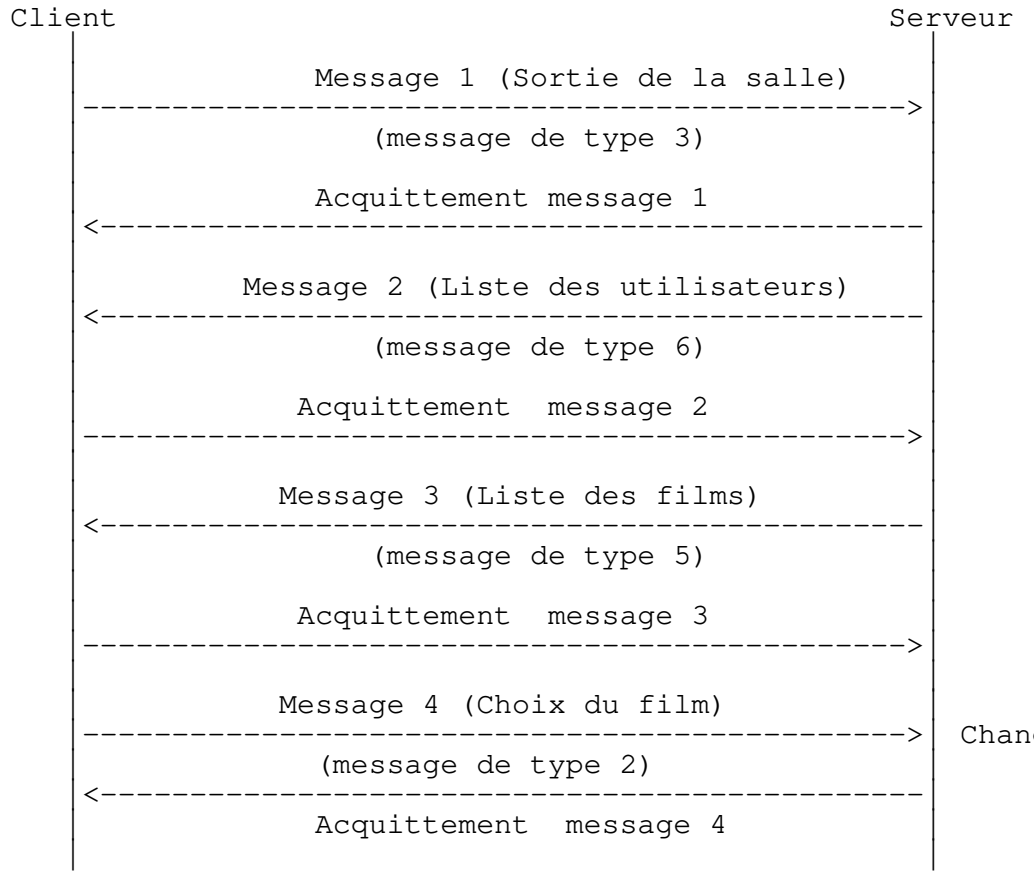
7.2. Scenario 2: Communication

Nous considerons que les clients A,B et C sont deja connectes et se trouvent dans la meme salle de chat afin de communiquer. Client A va envoyer un message puis se deconnecter.



7.3. Scenario 3: Changement de salle

Le client cherche a changer de salle, soit il est dans une salle de film, soit il est dans la salle principale. Dans le second cas, il doit juste choisir un film comme le montre le scenario numero 1. Mais dans le premier cas, il doit d'abord revenir a la salle principale pour revenir a la liste des films:



8. Conclusion

En conclusion, la specification proposee ici vise a etre autant efficace que fiable. Les differents types de messages permettent une differenciation des messages recus directement dans la mesure ou chaque message a un type particulier et donc un role bien defini. Le protocole dispose aussi d'un processus d'acquittement pour controler la bonne circulation des paquets au sein des differentes liaisons client-serveur.

Authors' Addresses

Gavin Leroy
IMT Atlantique
Brest
France

Email: gavin.leroy@imt-atlantique.net

Alexandre Lefebvre
IMT Atlantique
Brest
France

Email: alexandre.lefebvre@imt-atlantique.net

Alberto Blanc
IMT Atlantique
Rennes
France

Email: alberto.blanc@imt-atlantique.fr

Christophe Couturier
IMT Atlantique
Rennes
France

Email: christophe.couturier@imt-atlantique.fr

Romaric Ludinard
IMT Atlantique
Rennes
France

Email: romaric.ludinard@imt-atlantique.fr