TP1 NE324 - Le protocole à fenêtre glissante

Owen Rougé et Adrian Bonnet 15/02/19

Préparation : les questions effectuées en préparation du TP sont les questions 3.a et 3.c.

1. **Prise en main**
    a. Première exécution

Nous avons le schéma suivant : A envoie 4 trame `data` à B, B attend d'avoir un data à envoyer. Après `acktimeout`, B envoie un `ack` 4 qui ne sera pas reçu par A. Tout ceci recommence après `Timeout(0)`, A recommence à envoyer des `data`, B les acquitte après `acktimeout` après réception du `data` 4. A réception du `ack` 4, A envoie les `data` 5 à 8, que B reçoit.

Pour chaque trame, le simulateur utilise trois nombres : le premier est le numéro de séquence, le deuxième est le numéro d'acquittement, et le troisième est le champs de données, qui contient le numéro de la trame.

2. **Fenêtres et tampons**
    a. 1er échange

voir annexes 1 et 2

    b. SWS = 5 et RWS = 4

Dans cette configuration et en simplex, une erreur de protocole se produit. On a une erreur sur le glissement de la fenêtre, quand on revoit la donné "15" le rws croit qu'il est dans sa fenêtre (croit que la donnée reçue est la 23) alors que c'est l'envoie d'une donné déjà reçue. Le numéro de séquence n'est pas 2 fois supérieur à la taille de la fenêtre d'envoi (on a 2*5 > 8)

voir annexes 3 et 4

    c. SWS = 3 et RWS =2

La question ne se pose plus car NSEQ > 2 SWS. (on a NSEQ = 8 et 2 SWS = 6).

3. **Performances**
    a. SWS et RWS (Send Window Size et Receive Window Size

Si on a SWS > RWS, alors il faudra émettre de nouveau plusieurs trames, car toutes les trames de la fenêtre d'émission ne pourront être acquittées avec la fenêtre de réception si il y a un taux d'erreur non nul (. Il faut donc au minimum SWS = RWS.

Avoir RWS > SWS n'est pas plus utile que RWS = SWS car, même en cas d'erreur, l'émetteur ne pourra pas envoyer plus de trame que SWS, donc il sera bloqué si une trame n'est pas reçue. Cette méthode utilise cependant plus de mémoire.

b. SWS = 4, performances en fonction de RWS

Dans le fichier `sim.sh`, il faut ajouter l'argument `duplex` lors de l'écriture de fichier (ligne 6) pour exécuter `swp`.

voir annexe 5

c. Cas d'erreurs

Dans le cas d'une corruption de trame (`cksum_err`), aucune instruction n'est réalisée (`break;`), alors que dans le cas d'une perte de trame (`timeout`), l'instruction `send_data_frame` est réalisée (on retransmet). Le cas d'une perte de trame sera plus efficace que la corruption, car il faudra alors attendre que l'émetteur de trame corrompue réalise qu'il n'a rien reçu pour transmettre de nouveau pour que, enfin, la trame arrive.

Expérimentalement : 50 secondes entre les deux émissions pour une corruption de trame, et 50 entre les deux émissions pour une non réception de trame.

4. Un bug...

Dans le code source mis à disposition sur Chamilo, ligne 99, la fonction `start_ack_timer` est appelée. Cet appel se fait avant de vérifier si cette même trame a déjà était reçue. Dans le code du cours, l'appel se fait après vérification. Dans le cadre du code du cours, en cas de non réception d'un `ack` par l'émetteur des données, le récepteur n'enverra jamais de nouveaux `ack`, même si la même trame se présente plusieurs fois, la communication est donc bloquée.

5. Modification

a. Fonction NAK

Pour implémenter l'envoi de `nak`, nous avons créé un void sur le modèle de `send_ack_frame`, nommé send_nak_frame, qui est appelé dans le cas d'un `checksum error` (seulement si le message erroné est un data). Nous avons aussi ajouté dans la réception d'une trame, le fait que celle-ci puisse être un `nak`, et qu'il faut donc renvoyer la trame correspondante.

b. Comparaison graphique

Nous avons fait la comparaison graphique du simulateur avec et sans `nak`, mais nous n'obtenons aucune différence entre les deux exécutions. Nous pensons que le renvoi systématique d'un `data` correspondant lors de la réception d'un `nak` peut augmenter le temps de transmission.

voir annexes 6 et 7 et code de `swp.c` dans l'archive (lignes 57, 108 et 144)

TP1 NE324 - Le protocole à fenêtre glissante

Owen Rougé et Adrian Bonnet 15/02/19

Annexes :

annexe 1 :

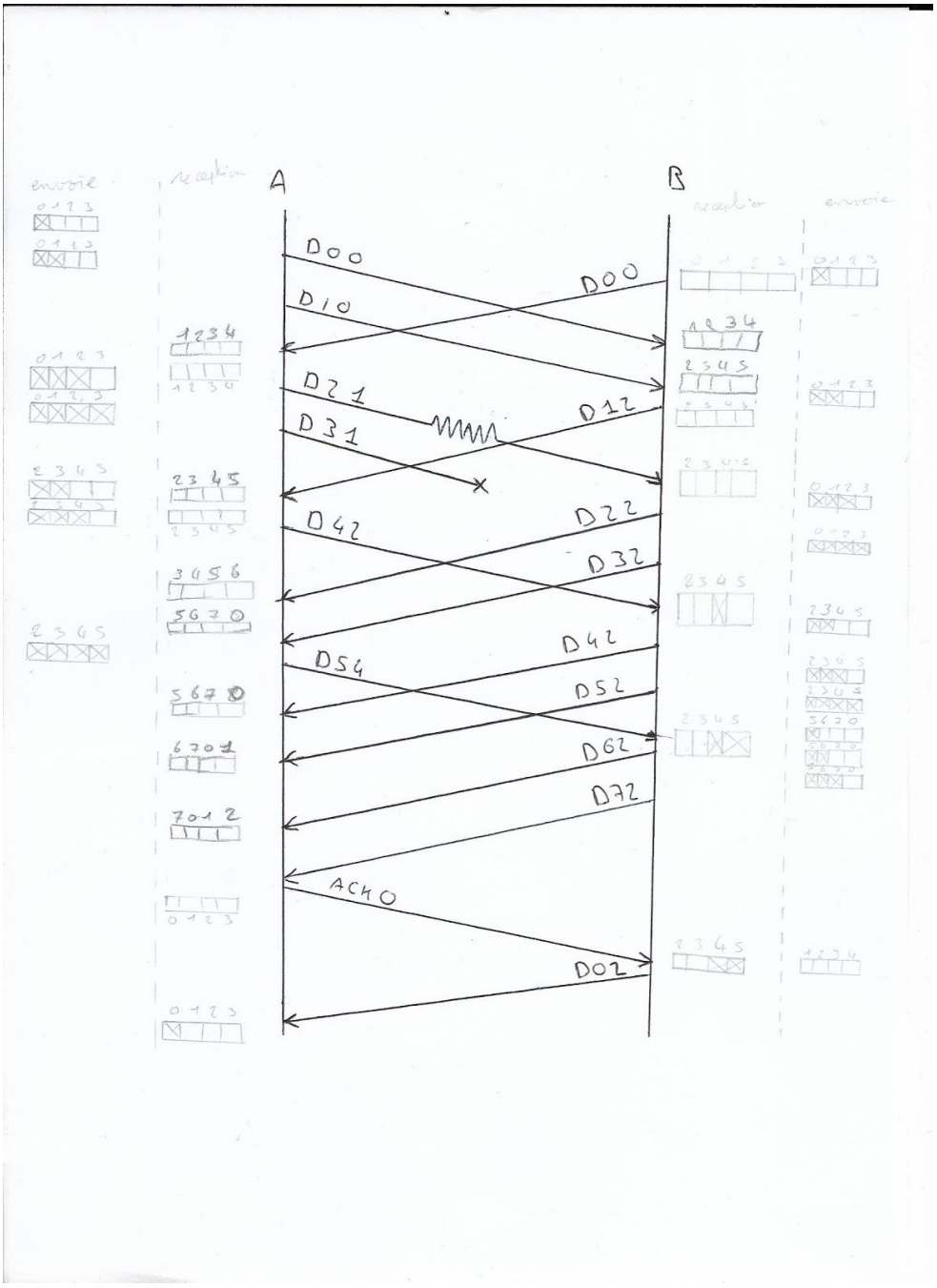


Diagramme du premier échange obtenu avec le simulateur

TP1 NE324 - Le protocole à fenêtre glissante

Owen Rougé et Adrian Bonnet 15/02/19

annexe 2 :

```
                    A                    B
                ---------            ----------

 1          Data 0 0 "0"  -->
 1                                <--  Data 0 0 "0"
 2          Data 1 0 "1"  -->
 2                                Data 0 0 "0"  -->
 3          <--  Data 0 0 "0"
 3                                Data 1 0 "1"  -->
 4          Data 2 1 "2"  -->
 4                                <--  Data 1 2 "1"
 5          Data 3 1 "3"  -->*
 5                                Data 2 1 "2"  -->*
 6          <--  Data 1 2 "1"
 6                                <--  Data 2 2 "2"
 7          Data 4 2 "4"  -->
 7                                <--  Data 3 2 "3"
 8          <--  Data 2 2 "2"
 8                                Data 4 2 "4"  -->
 9          <--  Data 3 2 "3"
 9                                <--  Data 4 2 "4"
10          Data 5 4 "5"  -->
10                                <--  Data 5 2 "5"
11          <--  Data 4 2 "4"
11                                Data 5 4 "5"  -->
12          <--  Data 5 2 "5"
12                                <--  Data 6 2 "6"
13                                <--  Data 7 2 "7"
14          <--  Data 6 2 "6"
15          <--  Data 7 2 "7"
40  Ack timeout
40          Ack  0        -->
41                                Ack  0        -->
42                                <--  Data 0 2 "8"
43                                <--  Data 1 2 "9"
44          <--  Data 0 2 "8"
44                                <--  Data 2 2 "10"
45          <--  Data 1 2 "9"
45                                <--  Data 3 2 "11"
46          <--  Data 2 2 "10"
47          <--  Data 3 2 "11"
```

Capture d’écran de la première simulation

TP1 NE324 - Le protocole à fenêtre glissante

Owen Rougé et Adrian Bonnet 15/02/19

annexe 3 :

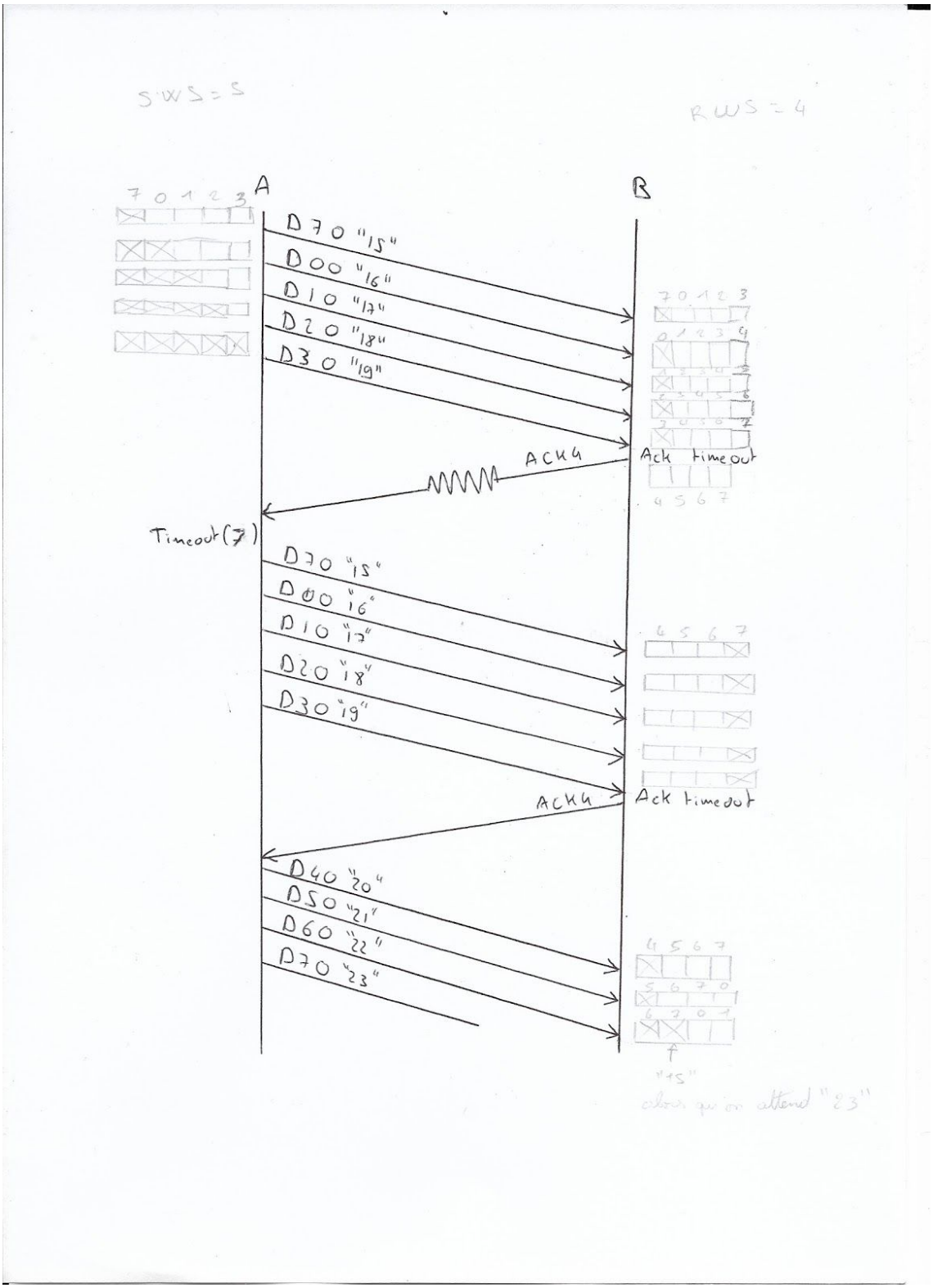
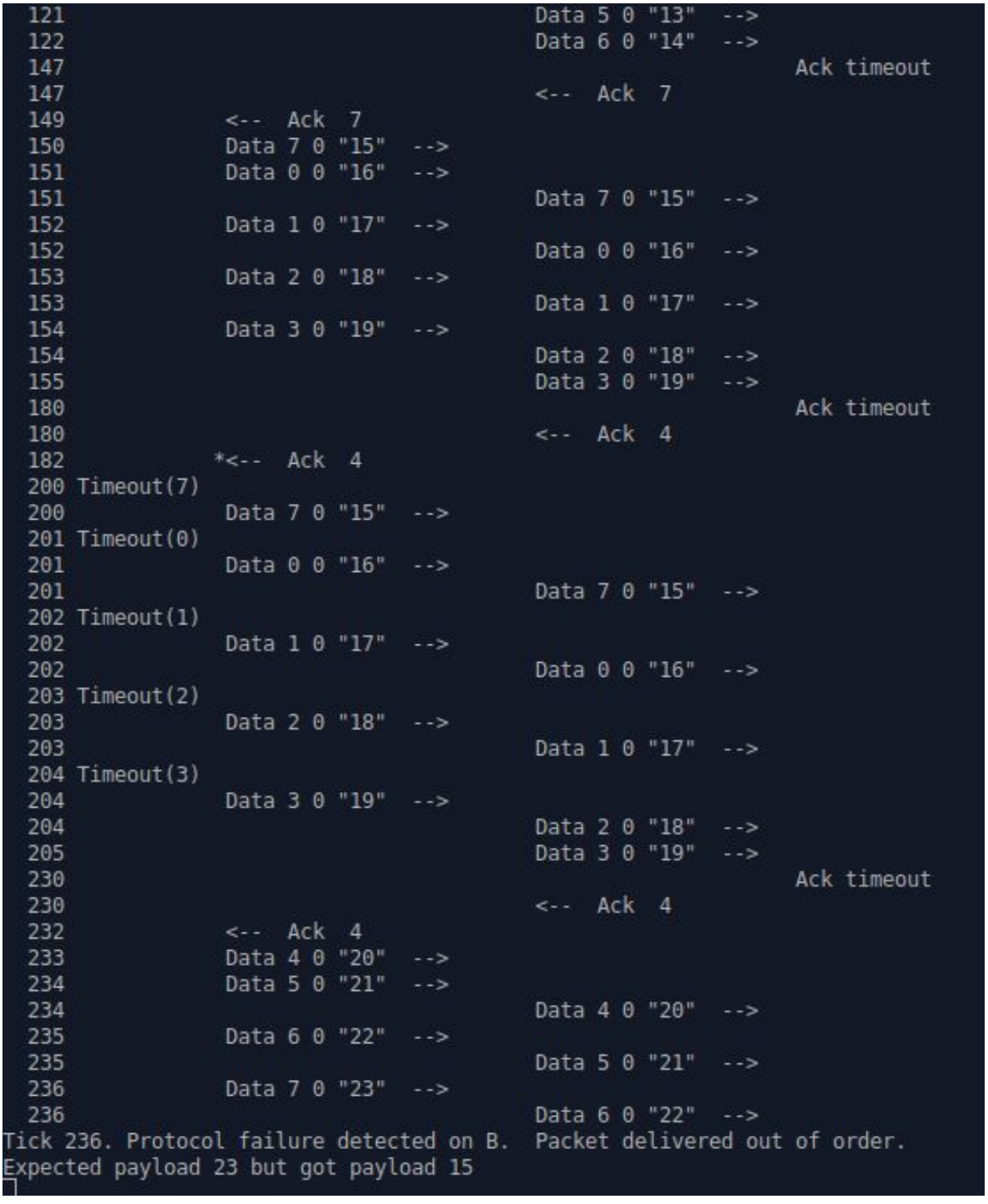


Diagramme de l'échange avec erreur : lors de la dernière réception, le "15" est gardé en cache de réception, alors que le "23" est attendu

annexe 4 :

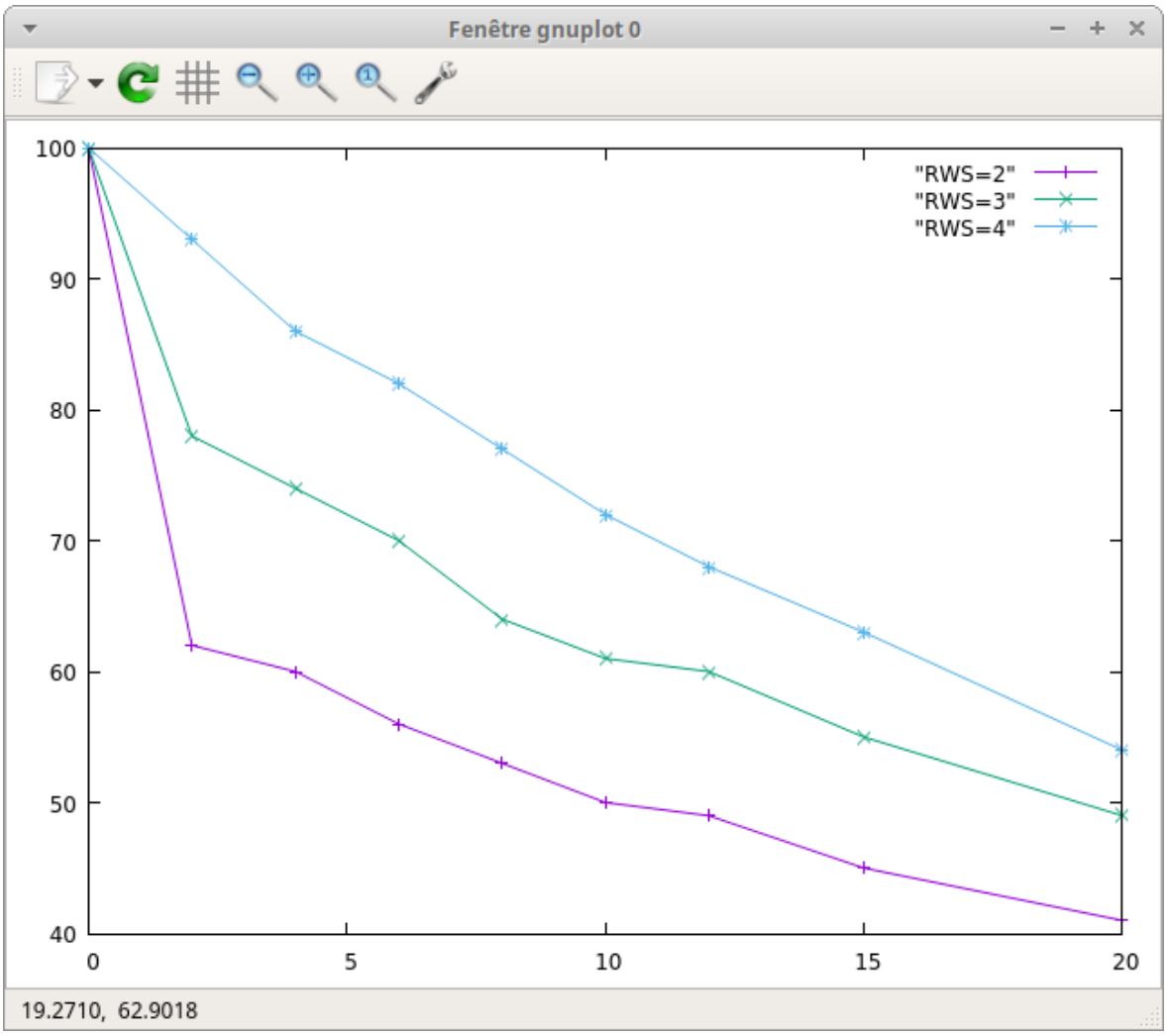


Capture d'écran de l'exécution avec erreur de protocole

TP1 NE324 - Le protocole à fenêtre glissante

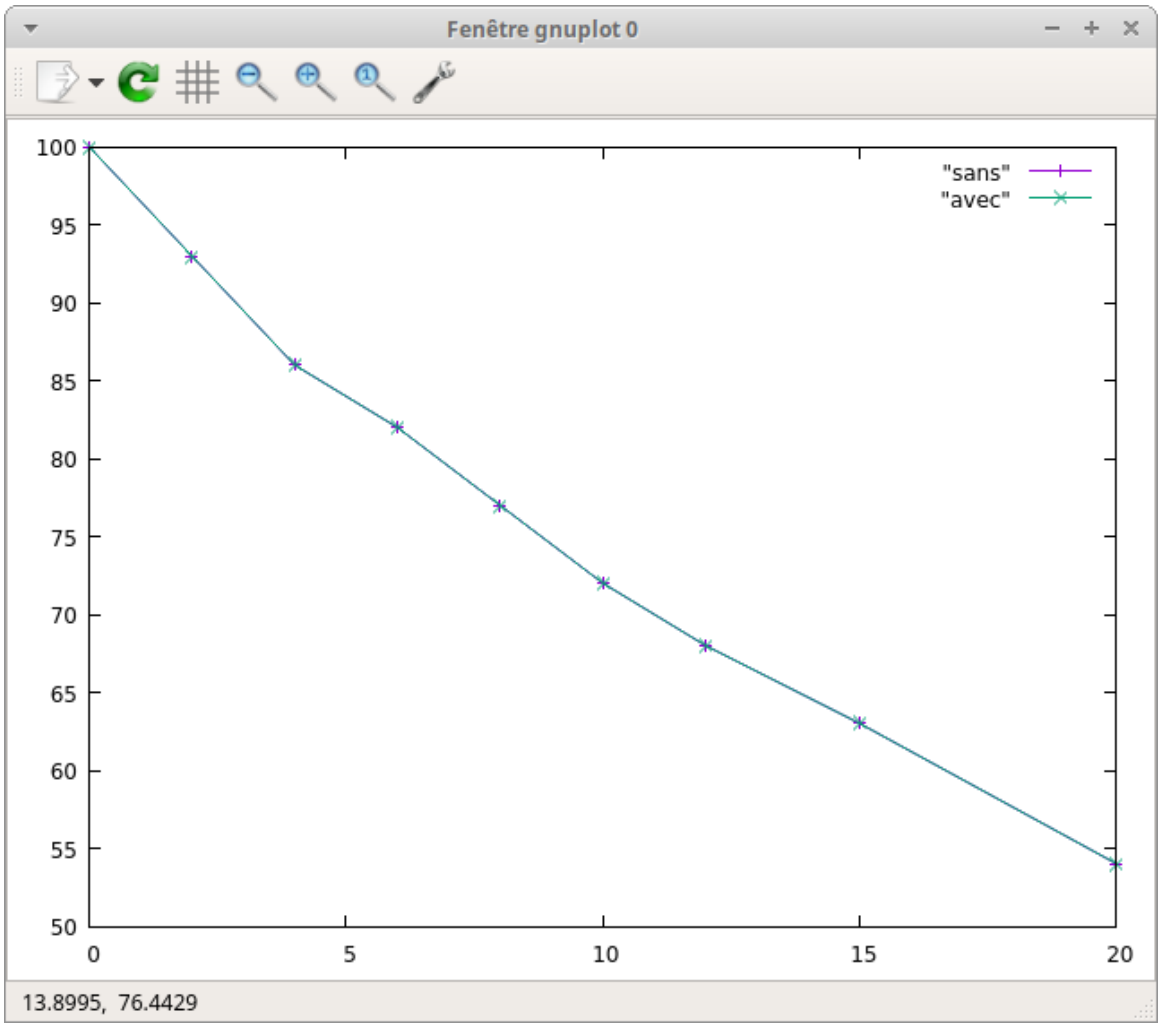
Owen Rougé et Adrian Bonnet 15/02/19

annexe 5 :



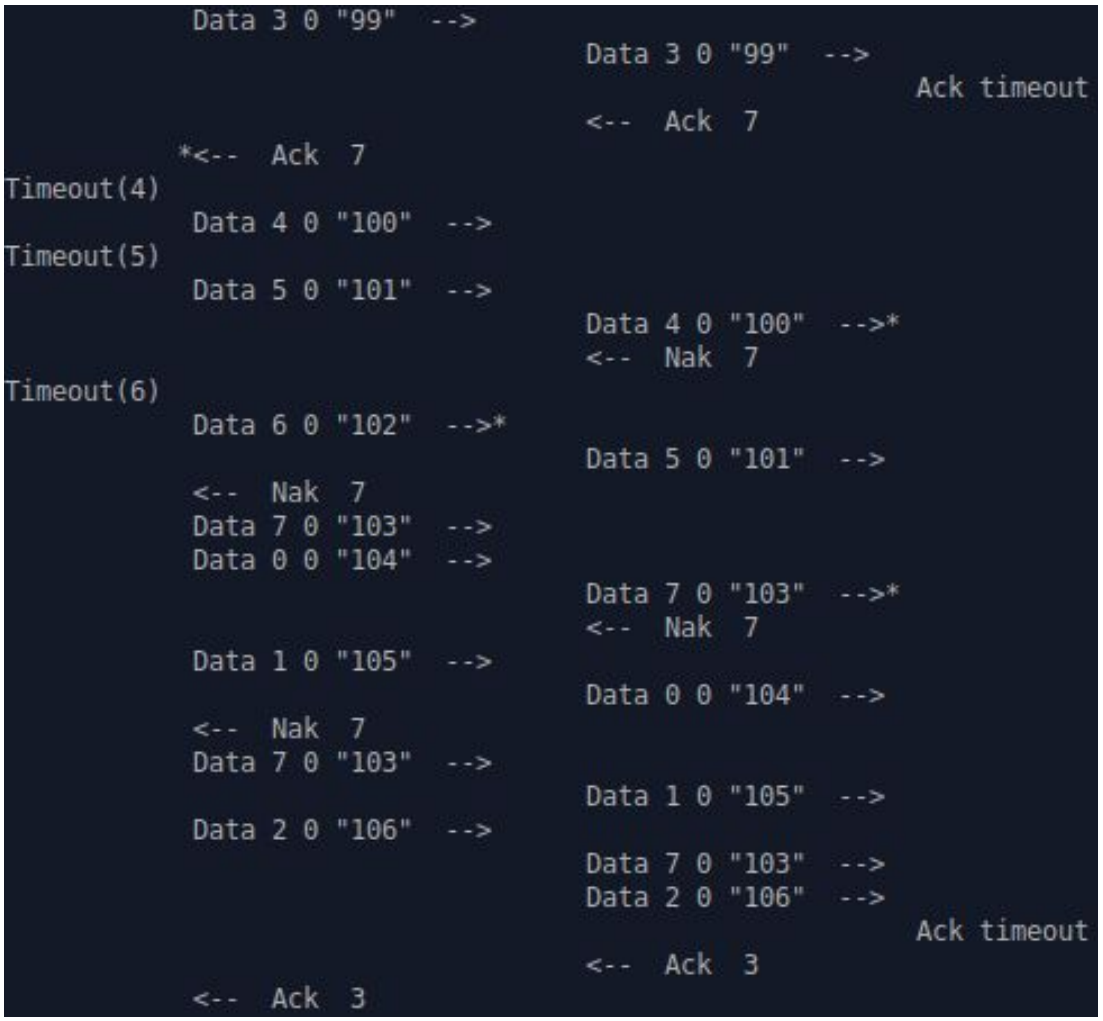
Efficacité du protocole en fonction du taux de pertes ou d'erreurs, selon RWS

annexe 6 :



Efficacité du protocole en fonction du taux de pertes ou d'erreurs,
avec et sans implémentation de `nak`

annexe 7 :



Exemple d'échange où le nak 7 n'est pas utile,
parce que le message 7 n'a pas été envoyé en premier lieu
puis parce que le message 6 (perdu) n'est pas redemandé