TELECOM Nancy 2A - RS -Réseaux
TD n°3 Protocoles niveau Transport

1 Protocole de transport simple : STOP-AND-WAIT

Exercice 1 Soit un protocole de transport de données simple de type STOP-AND-WAIT qui est implanté selon l'architecture suivante (cf Figure 1) :

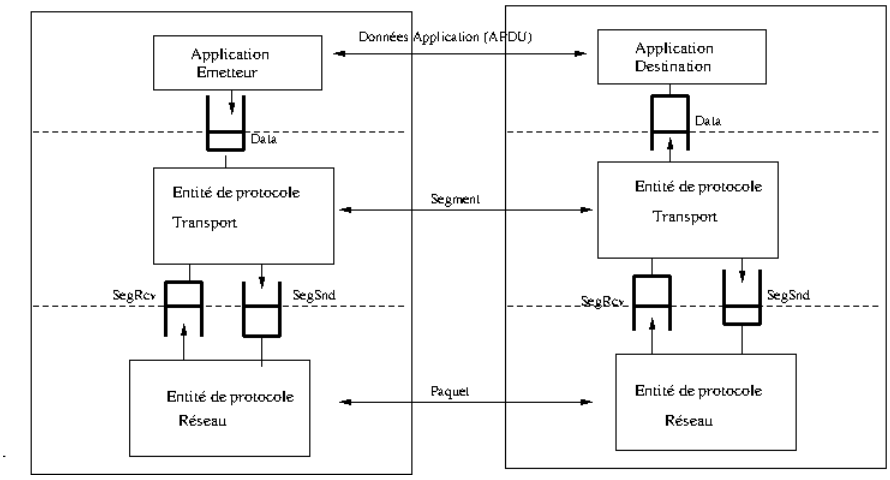


FIGURE 1 – Architecture

- Le protocole transport est considéré comme unidirectionnel du point de vue des données applicatives et le protocole réseau est de type *best effort*.
- Les différentes PDU sont les suivantes :

  Data : donnée de l'application

  SegRcv : segment reçu par la couche transport

  SegSnd : segment émis par la couche transport
- Les événements reçus par la couche transport :

  transmit(Data) : demande de transmission d'une donnée par l'application

  recv(SegRcv) : réception d'un segment par la couche réseau
- Les actions réalisées par la couche transport :

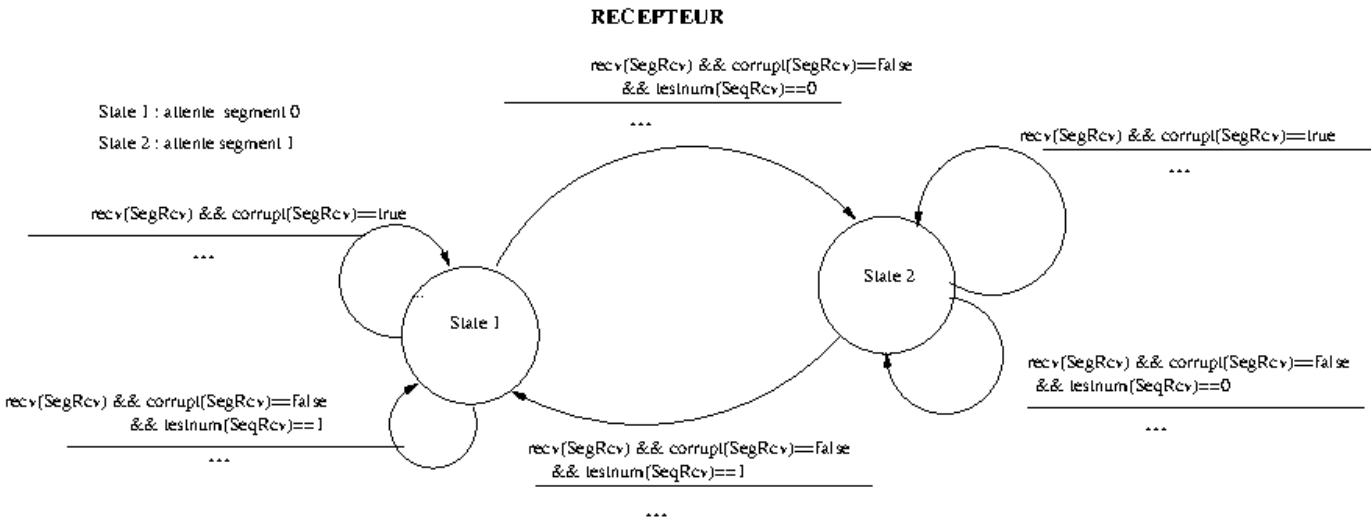
  deliver(Data) : remise de la donnée à l'application

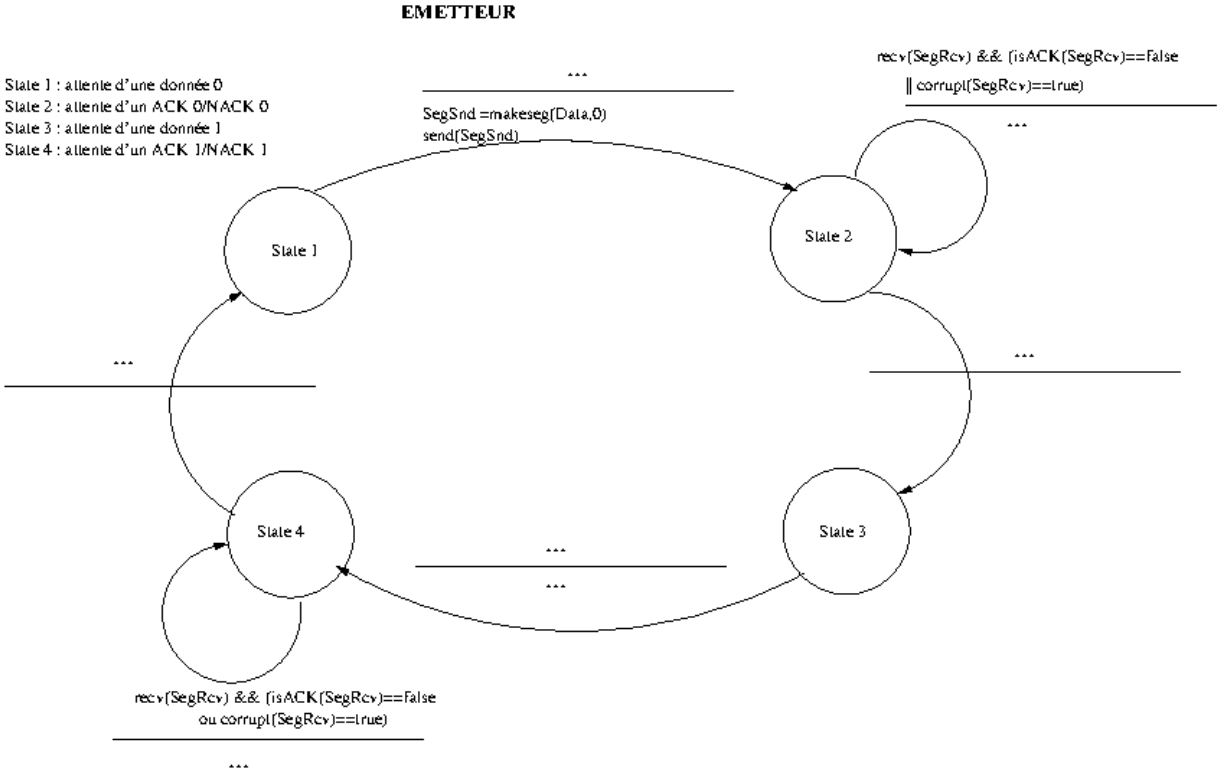
  send(SegSnd) : émission d'un segment vers la couche réseau

Ce protocole stop-and-wait utilise deux types d'acquittements :
- acquittement positif qui permet d'acquitter un segment non corrompu ;
- acquittement négatif qui permet de signaler que le segment reçu est corrumpu et donc de demander sa retransmission.

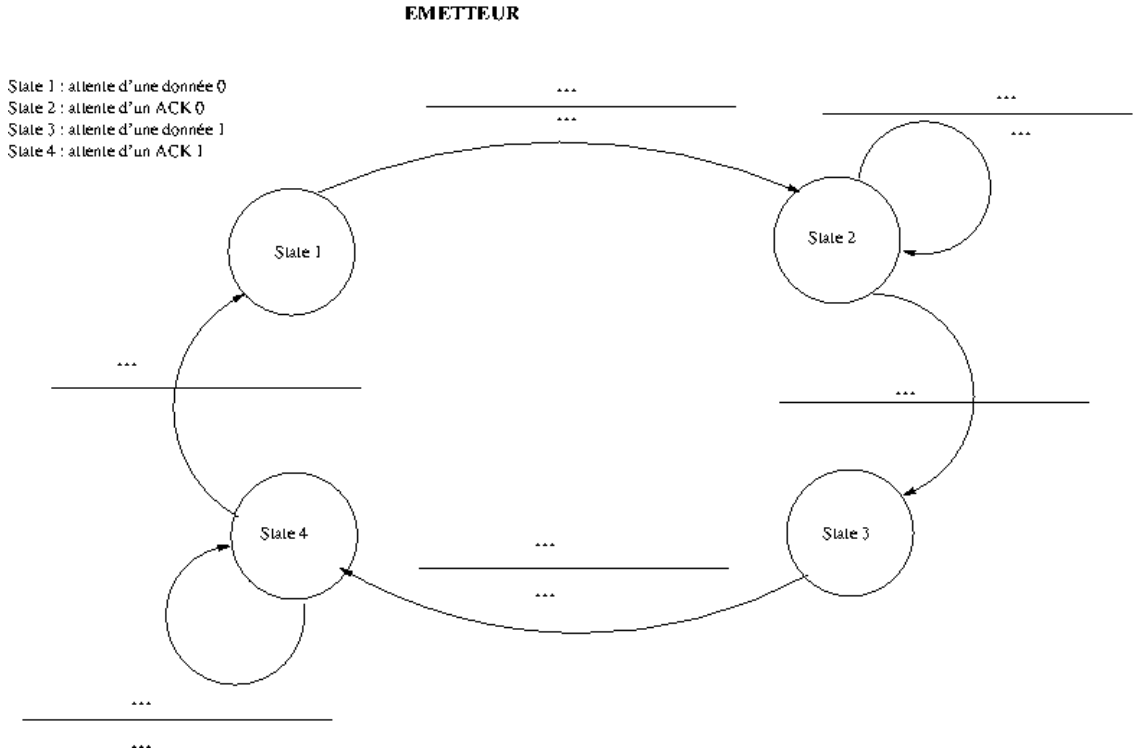
Vous disposez des fonctions suivantes :
- **makeseg(Buffer, [NumSeg]) :** retourne le segment **SegSnd** à transmettre. Buffer peut être de type NACK, ACK ou Data
- **isACK(SegRcv) :** retourne un booléen et détermine si le segment reçu est un ACK
- **corrupt(SegRcv) ;** retourne un booléen et détermine si le segment reçu est corrompu
- **extract (SegRcv) :** retourne la donnée applicative du segment
- **testnum(SegRcv) :** retourne le numéro du segment reçu

1. On considère dans un premier temps que le canal de transmission peut corrompre les données transmises et les acquittements, mais qu'il ne peut pas perdre ou détruire des données.

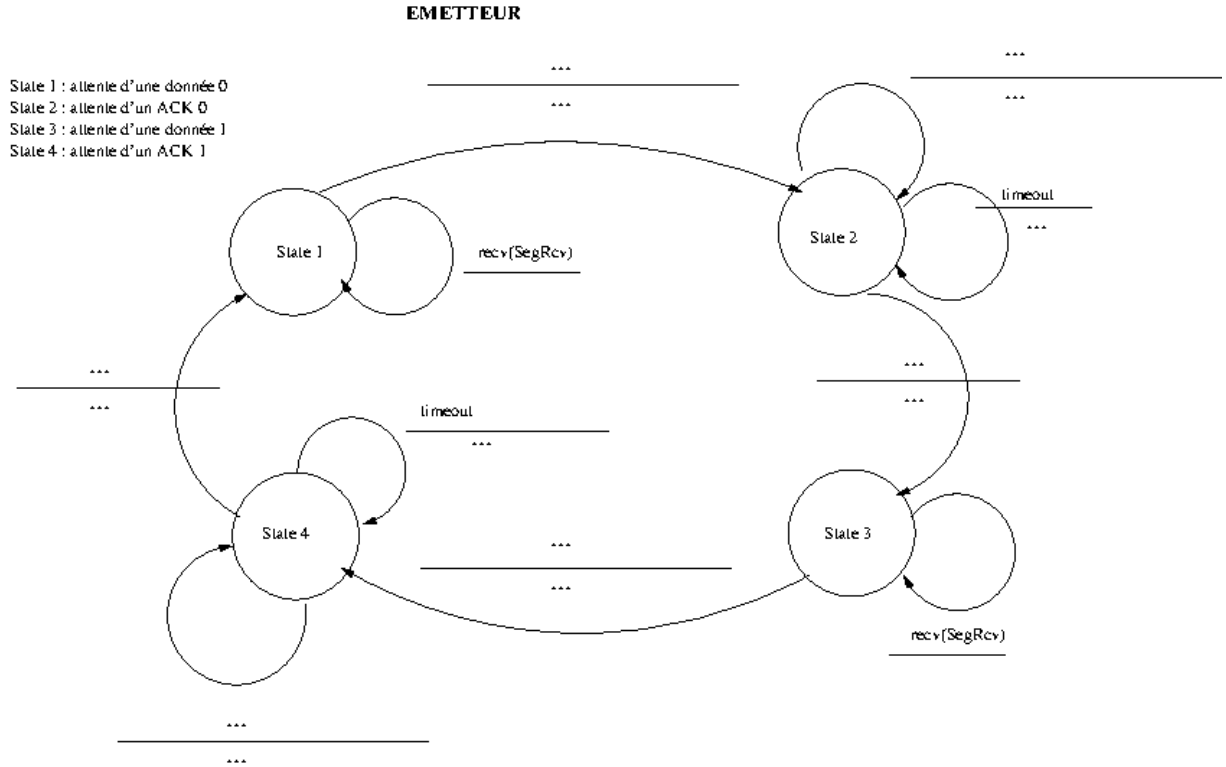




2. Modifier l'automate précédent de l'émetteur en supprimant le NACK. Comment dans ce cas, l'émetteur peut savoir que la donnée est corrompue ?



3. Modifier l'automate précédent de l'émetteur en prenant en compte le fait que les données et les acquittements peuvent se perdre.



Exercice 2 Dans le cas du protocole STOP-AND-WAIT, déterminer le taux d'utilisation de la liaison. On définit ce taux comme étant le ratio du temps de transmission sur le temps total pour émettre la donnée et obtenir un acquittement.

1. Donner une formule générique du taux d'utilisation U, en fonction de a ; $a = Tprop/Ttrans$. Soit, $Ttrans$, le temps de transmission de la donnée et $Tprop$, le temps de propagation du signal. On négligera le temps d'attente dans les buffers et les files d'attente des routeurs. La taille de l'acquittement est supposée faible.
2. Soit une liaison satellite géostationnaire, une taille de données de 4000 bits et un débit de 1 Mbit/s. On considère une vitesse de propagation des signaux d'environ 300 000 km/s. L'émetteur et le récepteur sont terrestres et situés dans la zone de couverture du satellite. Rappeler l'altitude d'un satellite géostationnaire, puis déterminer le taux d'utilisation de la liaison.
3. Soit une liaison câble coaxial de 10km et un taux de transmission de 1Mbit/s. On considère que la vitesse de propagation des signaux est d'environ 200 000 km/s dans les câbles. La taille de la donnée est toujours de 4000 bits. Déterminer le taux d'utilisation de la liaison.
4. Qu'en déduisez-vous ?

2 Protocoles UDP/TCP

Exercice 3 Quelles caractéristiques sont vraies pour UDP ?

a) Contrôle de flux

b) Sans connexion

c) Démultiplexage

d) Ordre des paquets

Exercice 4 Parmi les éléments suivants, quels sont ceux qui sont inclus dans un en-tête TCP mais pas dans un en-tête UDP ?

a) port d'origine

b) numéro de séquence

c) taille des fenêtres

d) port de destination

e) numéro d'accusé de réception

Exercice 5 Quelle est la spécificité des numéros de séquence en TCP ?

CORRECTIONS

Correction 1 Il faut amener les étudiants à construire les automates qui sont donnés à la fin de ce document. A la suite de l'exercice, les étudiants doivent être à même de comprendre ce qu'est un protocole et une machine d'états associée à un protocole.

1. Si les segments ACK et NACK peuvent être corrompus, l'émetteur de la donnée ne sait pas comment les interpréter. Une façon simple de faire est donc de retransmettre alors le segment. De ce fait ; il faut que le récepteur soit capable de savoir si la donnée a déjà été reçue ou non (s'il s'agit d'un nouveau segment ou du précédent) Donc il faut numéroter les segments. Une numérotation modulo 2 suffit. Il faut alors considérer les fonctions suivantes :

   makeseg(buffer, numseq) : retourne le segment **SegSnd** à transmettre à la couche réseau. Le buffer peut être de type : NACK, ACK, DataReq. Ajouter le numéro de séquence du segment pour DataReq

   testnum(SegRcv) : retourne 1 ou 0
2. Utiliser uniquement des ACKs. En cas de duplication, on ré-émet la donnée. Pour l'instant pas besoin du timeout. car pas de perte de données sur le canal. Pour détecter les acquittements dupliqués, il faut les donc numéroter.
3. Il faut ajouter le temporisateur. Juste à faire remarquer aux étudiants : Il existe des cas où le fait de n'avoir que la numérotation 0 et 1 pose problème, notamment quand les délais peuvent être grands. Exemple : A envoie segment 0 à B. L'acquittement 0 arrive après le déclenchement du timer. A a donc émis deux fois le segment 0 . A envoie le segment 1 (suite au ACK 0) à B qui reçoit le 2ème segment 0 après le segment 1 (suite à des délais).

Correction 2

1. On envoie une donnée et avant d'envoyer la suite on attend l'acquittement
2. Les amener à déterminer de manière plus précise, le taux d'utilisation U. Soit, $Ttrans$, le temps de transmission de la donnée, $Tprop$, le temps de propagation du signal, $Tproc$ le temps de traitement de traitement de la donnée et $Tack$ le temps de transmission de l'acquittement. On ne prend pas en compte le temps d'attente dans les éléments intermédiaires comme les routeurs

   Le temps pour transmettre la donnée et recevoir l'acquittement est donc

   `T = Ttrans + Tprop + Tproc + Tack + Tprop + Tproc.`

   Si Tack et Tproc sont supposés faibles, alors

   `T = Ttrans + 2⋆Tprop = Ttrans + RTT.`

   Le débit réel est de 1/T paquets par seconde.

   Le taux d'utilisation U est le rapport du temps pour transmettre la donnée $Ttrans$ sur le temps total pour envoyer la donnée et obtenir la réponse.

   `Donc U =Ttrans / (Ttrans + 2⋆Tprop). Soit a=Tprop/Ttrans, alors U=1/(1+2⋆a)`
3. Liaison satellite $=>$ Tprop = Distance / Vitesse de propagation = 2*36000/300 000 =240ms = 0,24s
   Ttrans = 4000bits/1 000 000 = 4/1 000 = 1/250 s = 0,004 s = 4ms
   a = 0,240 * 250 = 60
   U = 1/(1 + 120) = 1/121 = 0,00826 = 0,826%
4. Tprop = Distance /Vitesse de propagation = 10 / 200 000 = 1/20 000 = 0,00005s = 0,05 ms = 50μs
   Ttrans = 4000bits / 1 000 000 = 4/ 1000 = 1/250 s =4ms = 4000μs
   a = 0,00005 * 250 = 0,0125
   U = 1 /(1+ 0,025) = 1/1,025 = 0,975 = 97,5
5. Le protocole stop and wait est inefficace quand les temps de propagation sont élevés. Il faut un moyen de continuer à envoyer des paquets avant réception d'un acquittement. D'où les protocoles à fenêtres (cf exo ci-dessous).

Correction 3 Contrôle de flux, Avec connexion, Ordre des paquets

Correction 4 numéro de séquence, numéro d'accusé de réception et taille des fenêtres

Correction 5 Numérotation séquentielle des octets

Solution des automates de la Section 1

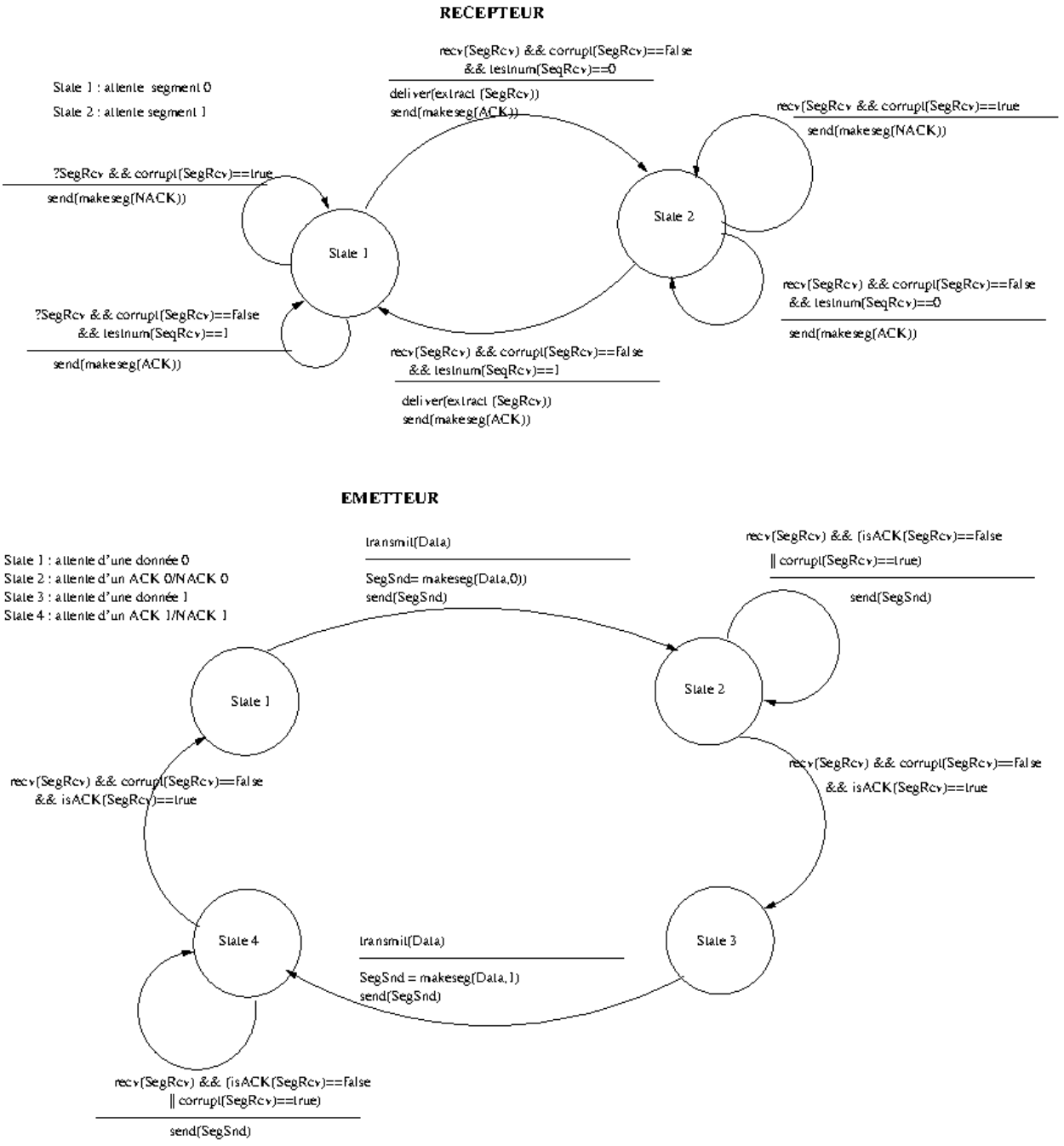


FIGURE 2 – Automate 2

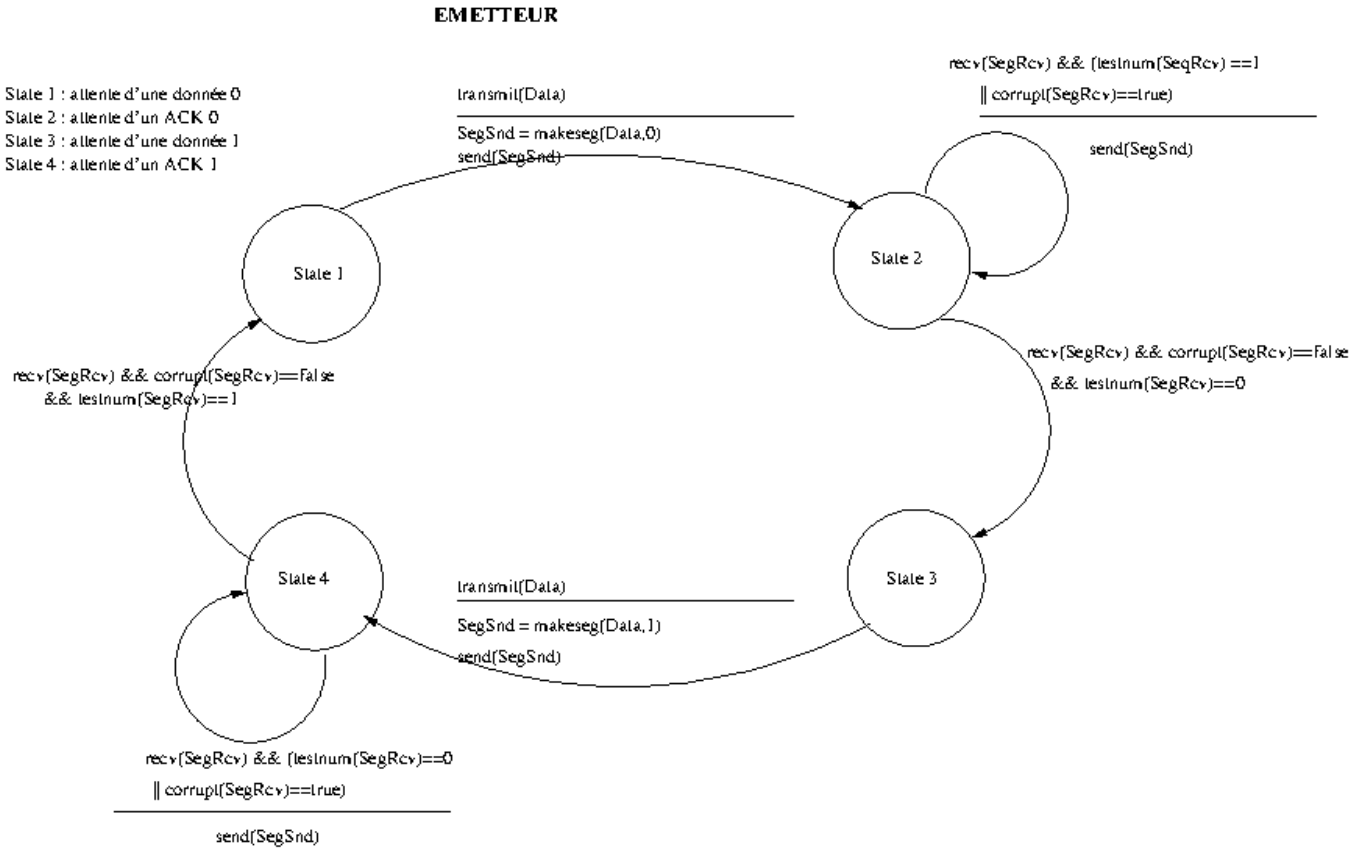


FIGURE 3 – Automate 3

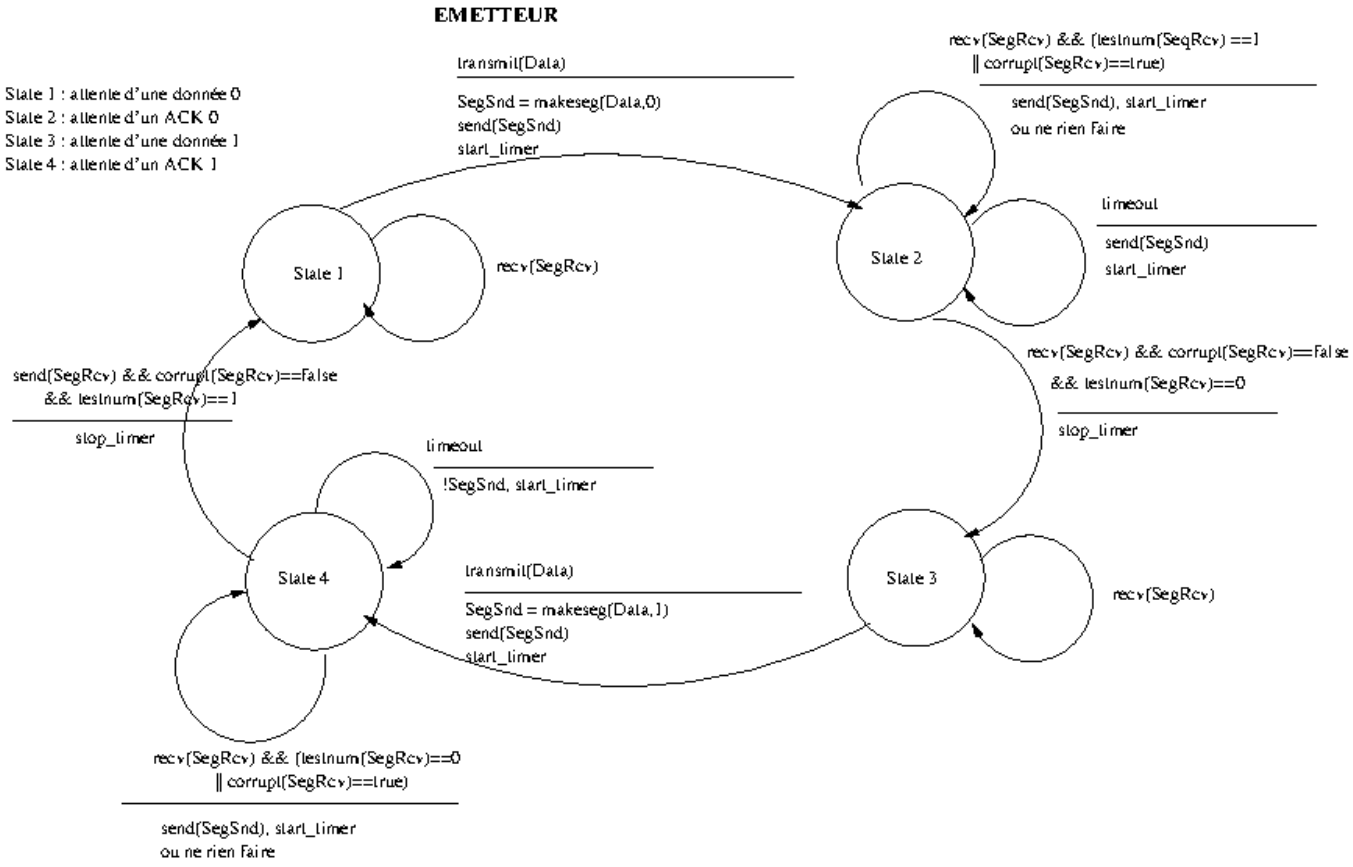


FIGURE 4 – Automate 4