¿Por qué el transmisor de stop-and-wait y Go-back-N hacen nada cuando llega un ACK dañado o duplicado?

- Caso Stop-and-wait

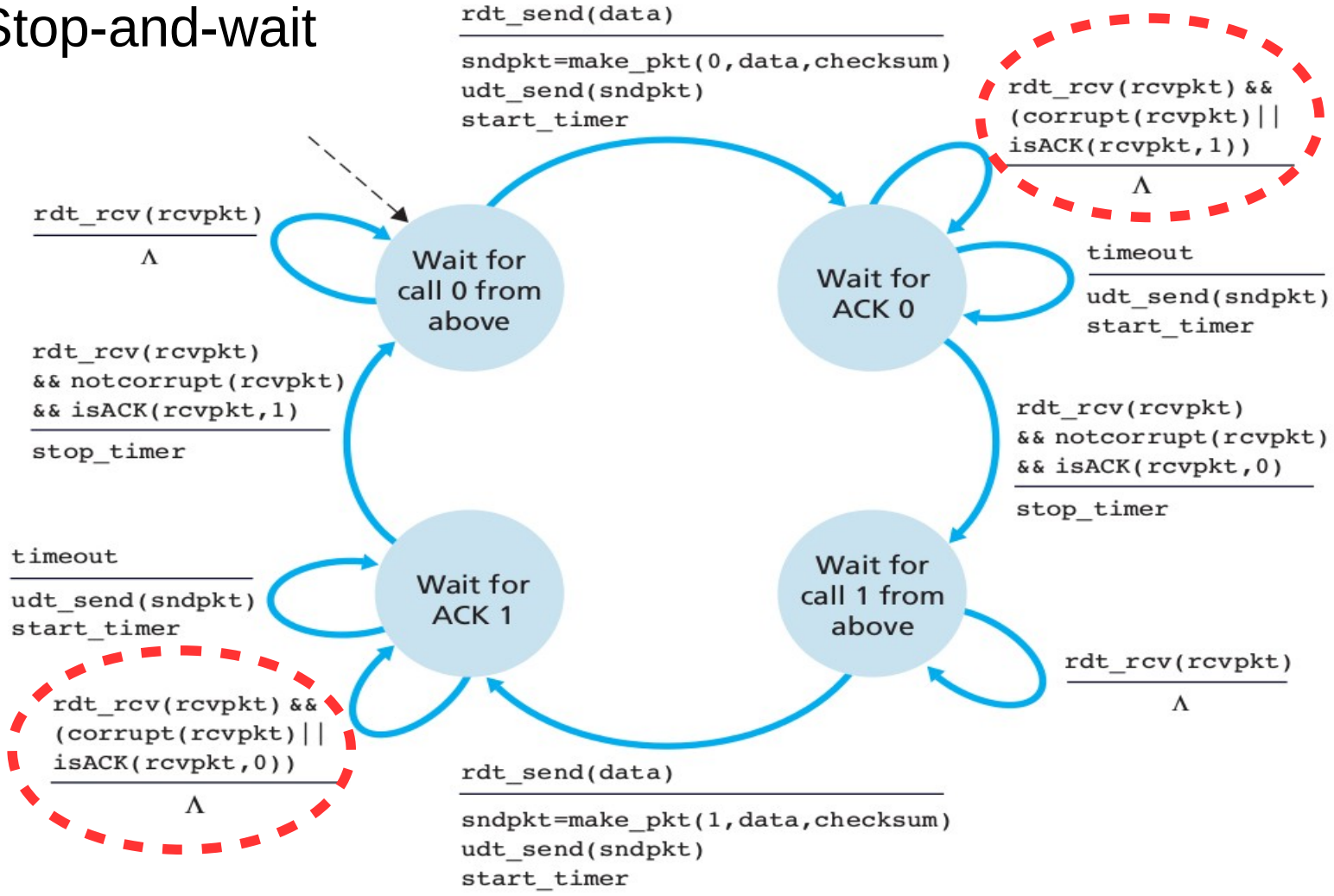
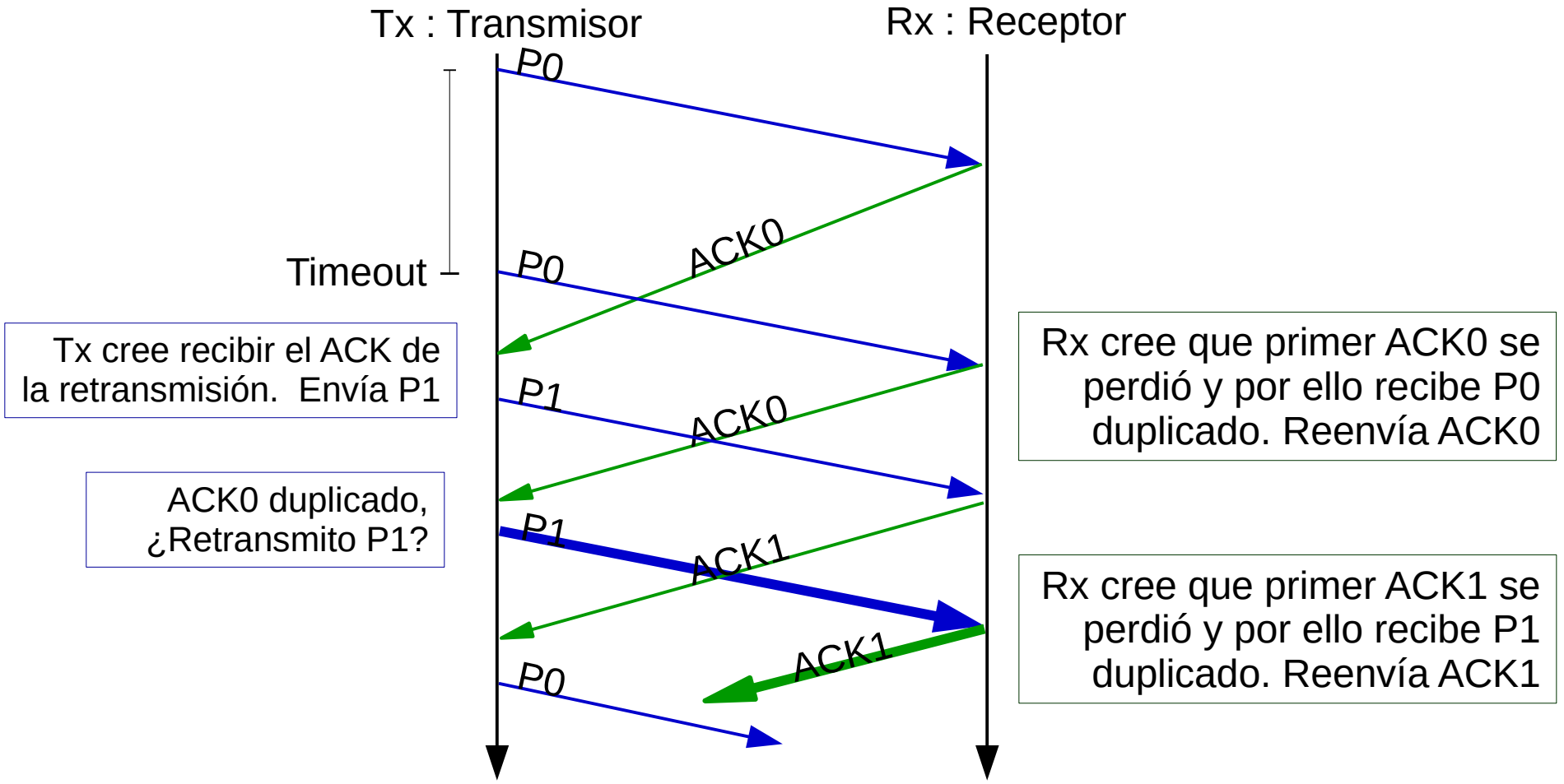


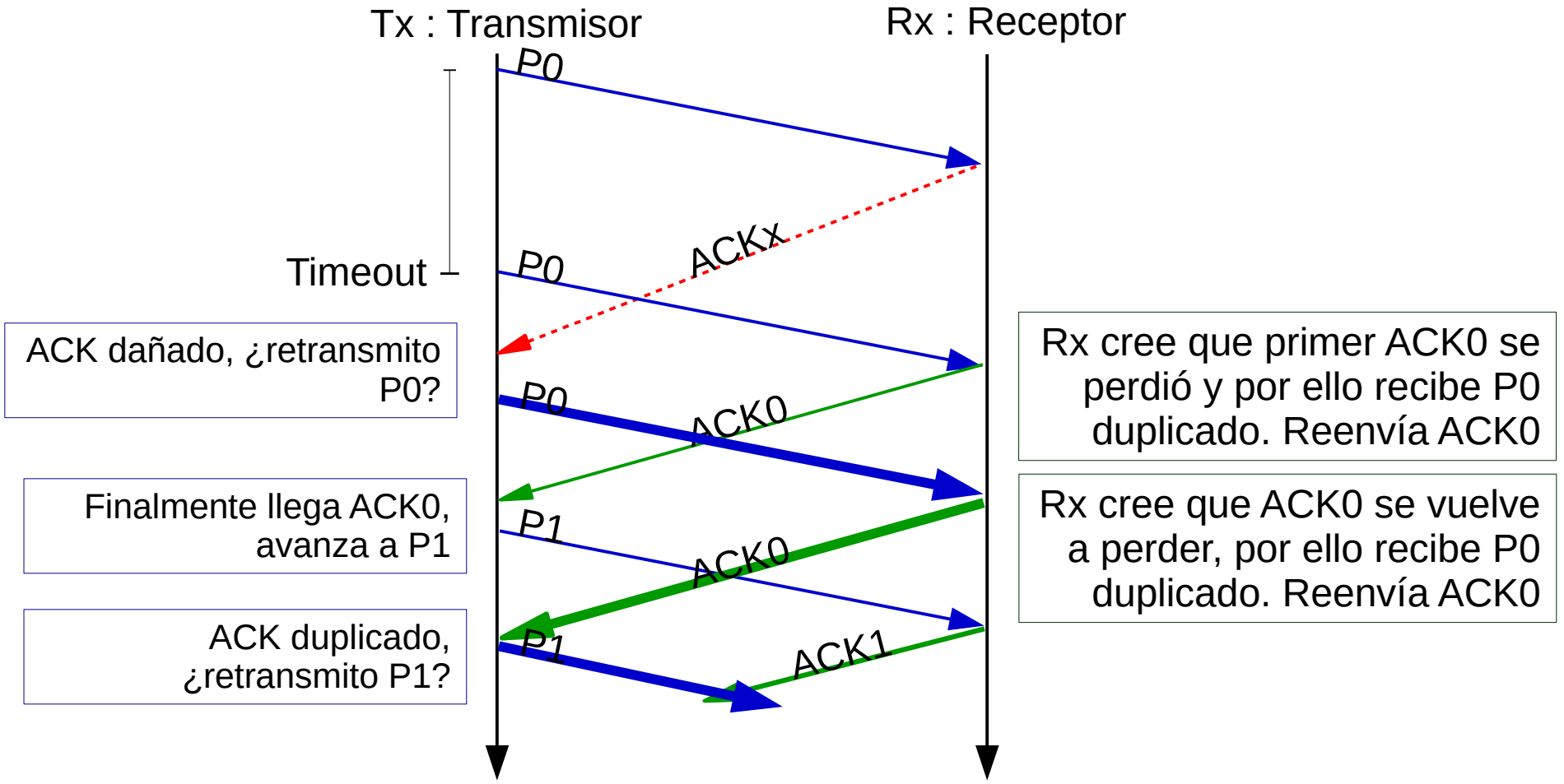
Figure 3.15 ♦ rdt3.0 sender

Caso 1: Timer prematuro



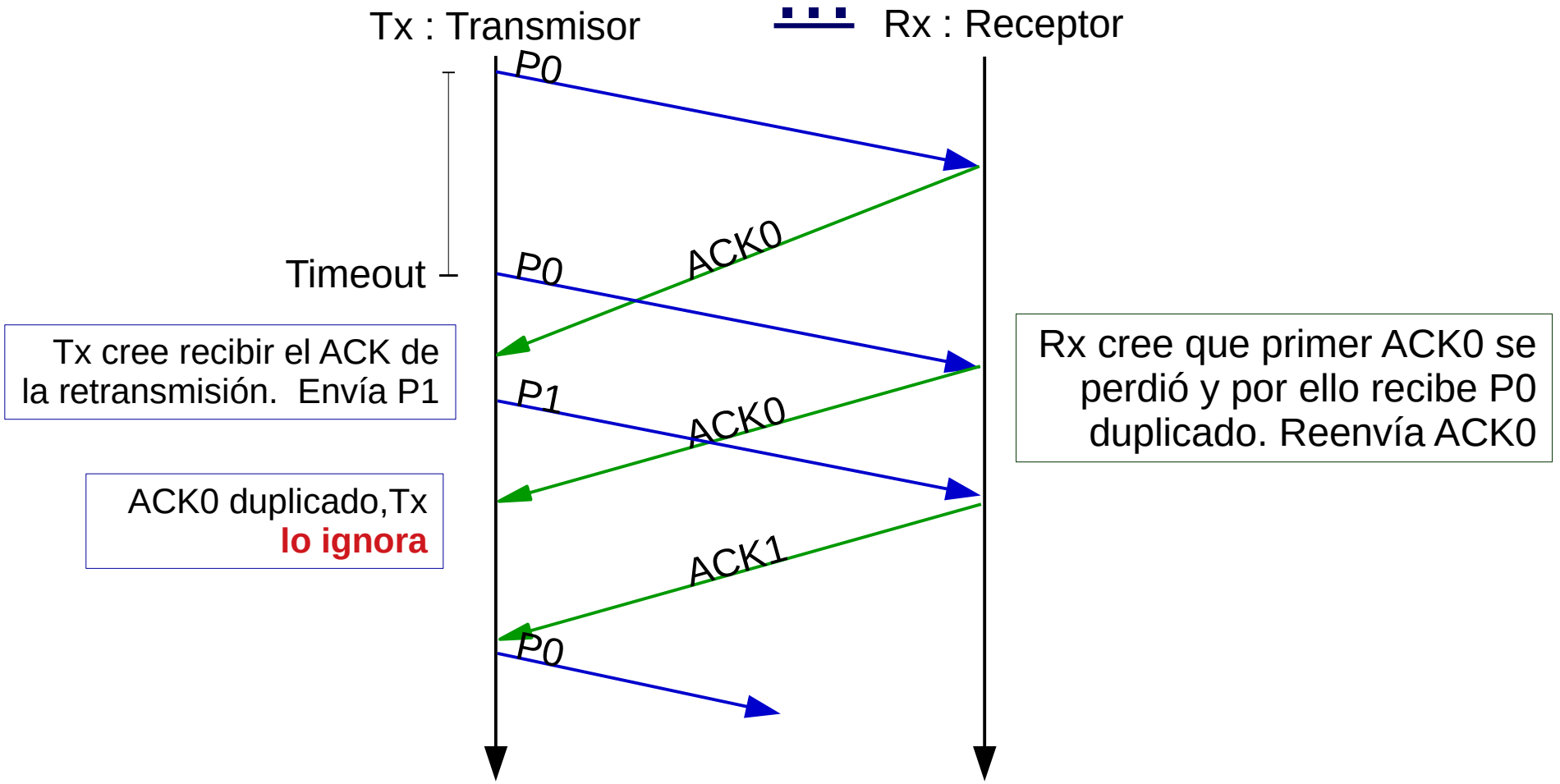
- Como los ACK se pueden perder, cuando llega un duplicado al Rx, éste debe reenviar el ACK. No tiene otra opción.
- Si Tx reenvía el paquete cuando llega un ACK duplicado, **terminará enviando dos veces cada paquete. Mala idea.**

Caso 2: Además ACK dañado



- Si Tx reenvía el paquete cuando llega un ACK dañado en este escenario, también terminará enviando dos veces cada paquete. Mala idea.

Caso 3: Supongamos Tx, hace nada



- Si Tx ignora el ACK duplicado, todo se comporta como se desea. Buena idea.

Caso Go-Back-N

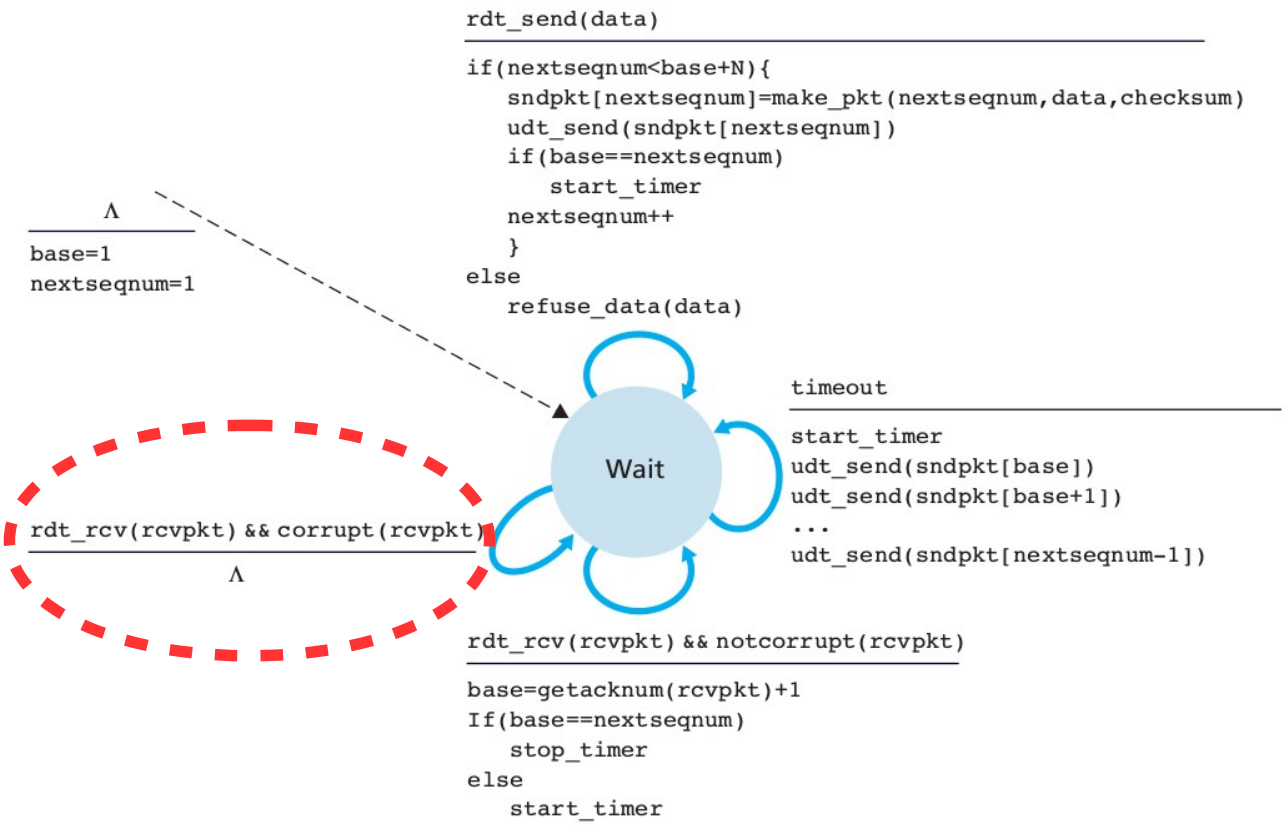
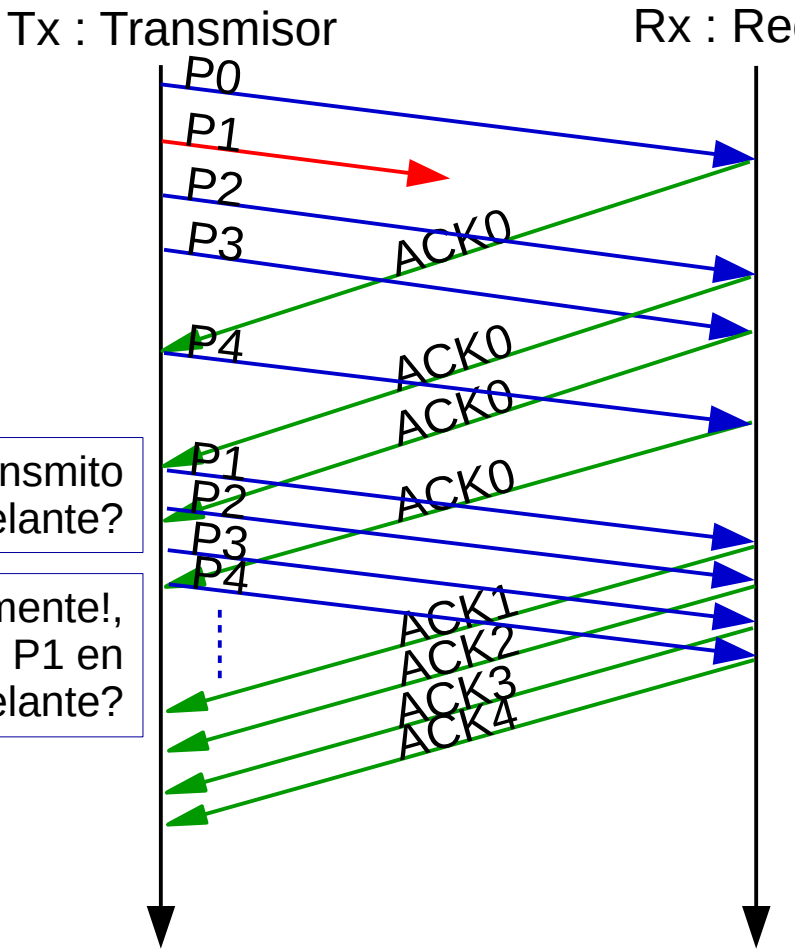


Figure 3.20 ♦ Extended FSM description of GBN sender

Caso 1: ACK duplicado



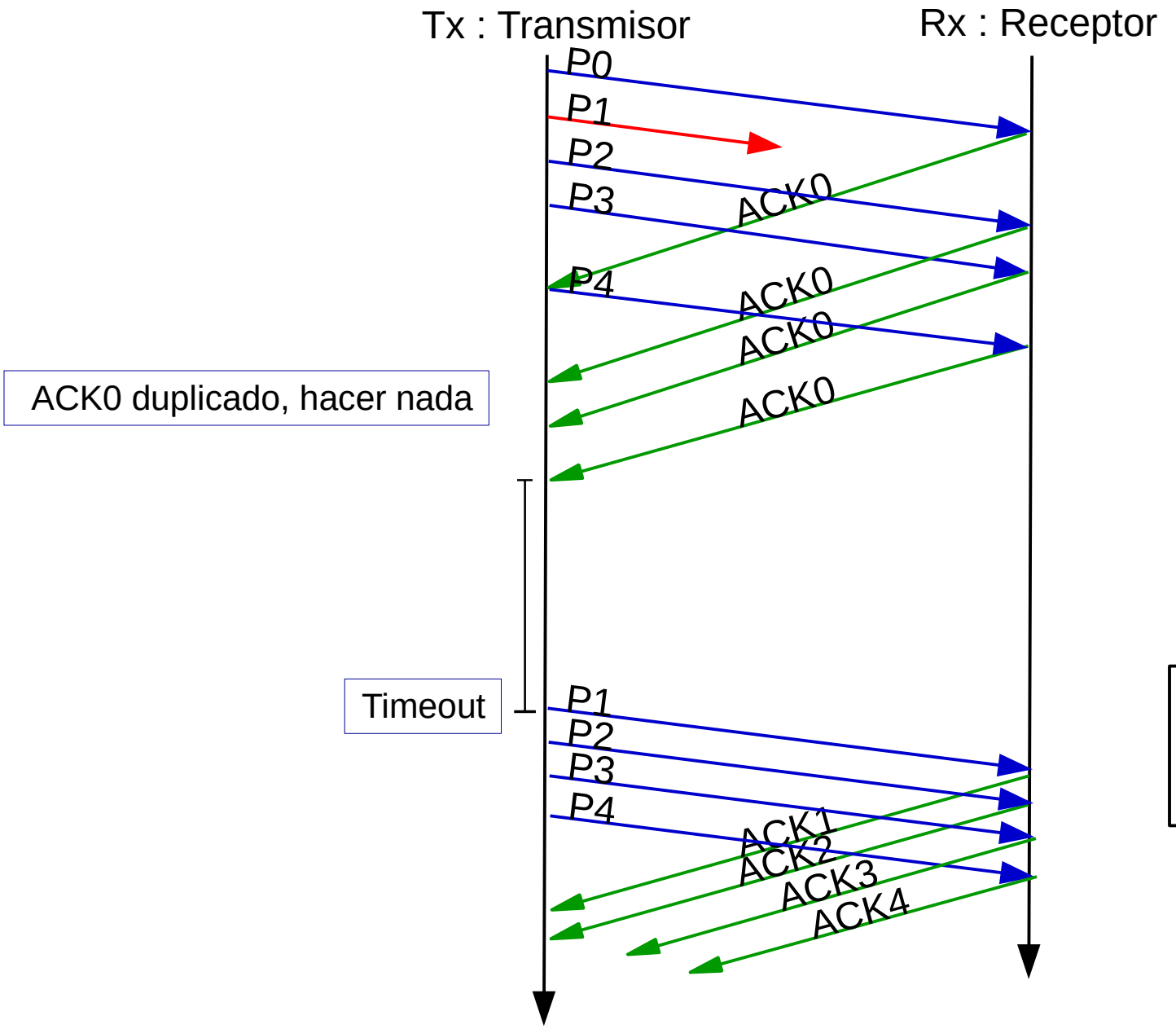
ACK0 duplicado ¿Retransmito P1 en adelante?

ACK0 duplicado nuevamente!, ¿Retransmito P1 en adelante?

Es mala idea retransmitir ante un ACK repetido

- Como los ACK se pueden perder, cuando llega un duplicado al Rx, éste debe reenviar el ACK. No tiene otra opción.
- Si Tx reenvía el paquete cuando llega un ACK duplicado, terminaría enviando varias veces varios paquetes. **Mala idea**

Supongamos Tx hace nada....



¿Por qué reiniciar el timer ante la llegada de cada ACK?

Propuesta de modificación de Go-Back-N, me apoyan?

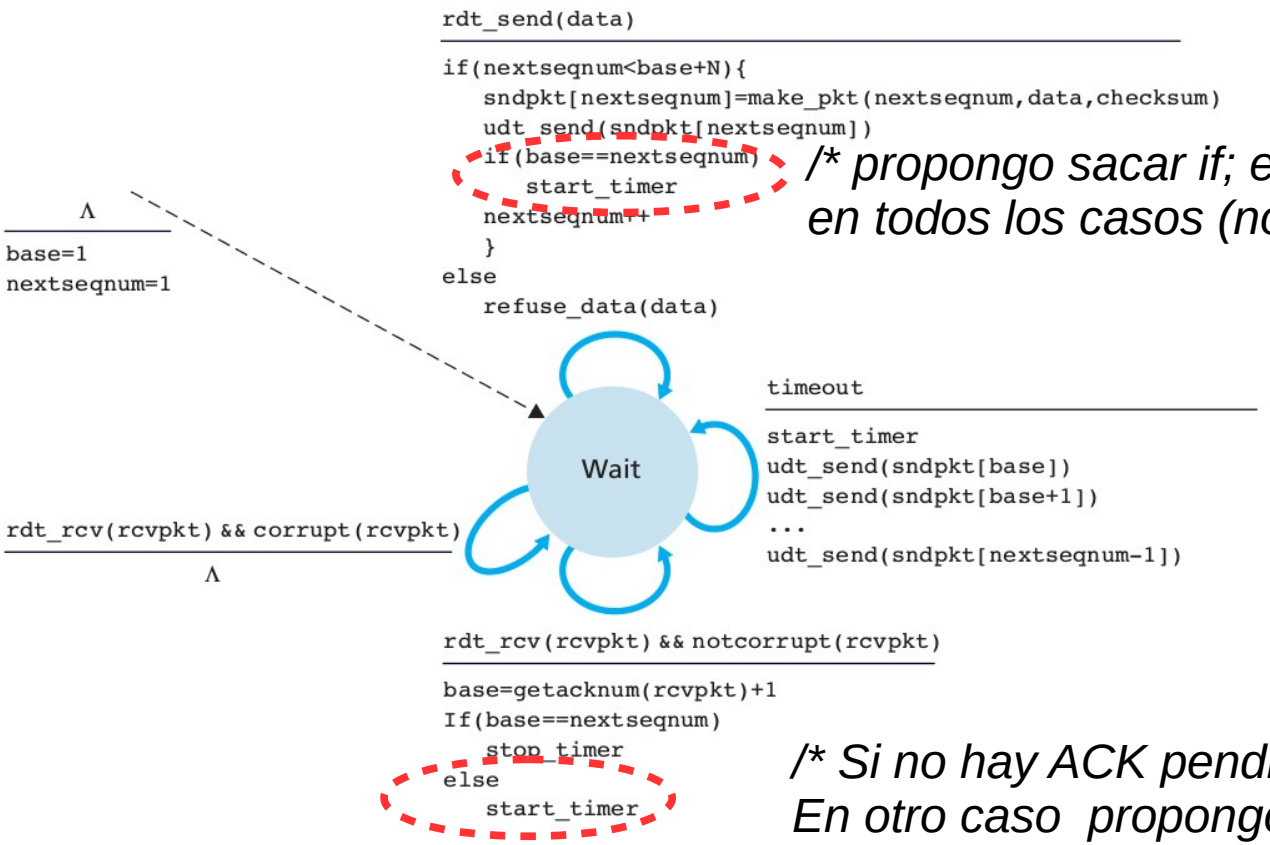


Figure 3.20 ♦ Extended FSM description of GBN sender

¿Estaría usted de acuerdo con reiniciar el timer cada vez que se envíe un paquete nuevo y eliminar la reiniciación del timer cuando llega un ACK?

Sin reiniciar timer al llegar ack: Éste sería el diagrama....