# LABORATORIO 4

TRANSFERENCIA DE DATOS CONFIABLE RDT2.0

**Propósito**

Implementar el protocolo de Transferencia de Datos Confiables `rdt2.0` explicado en clases.

**Generalidades**

Escriba y testee un programa `emisor_rdt2.0` y un programa `receptor_rdt2.0` en una misma computadora. Luego ejecute ambos programas en computadoras separadas, y verifique que se pueden comunicar. Para ello, consideramos el protocolo analizado en clases `rdt2.0`.

Para este proyecto, deberá modificar el Laboratorio 3, de manera que ahora cumpla las especificaciones de la máquina de estado presentada en la figura 3.10. Además deberá seguir haciendo uso de la **red** provista por la cátedra. A diferencia de los laboratorios anteriores, todo el proyecto deberá ser implementado desde cero. Sólo podrá hacer uso de los archivos `paquete.py`, `constantes.py` y `network.py`.

El archivo `constantes.py` presenta valores que vamos a usar tanto en el emisor como en el receptor. La idea de este archivo, es que nos permita definir dichos valores en un sólo lugar y luego poder usarlos en otros lugares mediante sus variables.

**Procedimientos y Detalles**

1. Estudie las máquinas de estado enseñadas en clase, presentes en la figura 3.10 del teórico. Si presta atención, no difiere mucho de la lógica presentada en los laboratorios ya realizados.

2. Estudie la API `network` provista por la cátedra. Le será de utilidad para poder hacerla funcionar correctamente.

2. Desarrolle un programa `emisor_rdt2.0.py` que cumpla las especificaciones de la máquina de estado de la figura 3.10 `a`. Para ello, escribas las funciones que se llaman durante el normal funcionamiento del mismo. El receptor envía datos a la `network`. La misma espera recibie pares de la forma `(receptor, paquete)`, donde `receptor` es una tupla de la forma `(dir_ip, dir_port)`, con las direcciones del receptor. `Paquete` es un paquete creado con la api de la clase.

3. Desarrolle un programa `receptor_rdt2.0.py` que cumpla las especificaciones de la máquina de estado de la figura 3.10 `b`. Para ello, escribas las funciones que se llaman durante el normal funcionamiento del mismo. Tenga en cuenta que el receptor, espera mensajes de la red, de la forma `(emisor, paquete)`, donde `emisor` es una tupla de la forma `(dir_ip, dir_port)` con las direcciones del emisor del `paquete`.

4. Realizar un test de loopback con el `emisor_rdt2.0.py` y el `receptor_rdt2.0.py` ejecutándose en una misma computadora como se explicó en clases.

5. Testear el cliente y el servidor ejecutando el primero en una computadora y el segundo en otra.

6. Modificar el servidor para que no sólo imprima el mensaje en pantalla, sino que además guarde un registro de todas las palabras recibidas y que haya impreso.

7. Como los puertos de conexión son de carácter efímeros, vamos a requerir pasar por parámetro dicho valor. Como también el servidor se va a ejecutar en diferentes máquinas, el cliente debe de poder recibir no sólo el puerto como parámetro, sino también la dirección ip del servidor con quien intenta establecer comunicación.