1. Como estamos trabajando con XML-RPC, usando la trasferencia por HTTP y TCP tendríamos un retraso de la señal de audio y video, es decir se quedaría pausada hasta que se reenvíe el paquete que se perdió y sufriría un retraso el envío de audio y video con respecto a la grabación del lado del cliente. No se permitiría la pérdida de paquetes.
2. Al desempeño de la red no directamente, al desempeño de la aplicación si ya que por cada paquete perdido se aculularia mas retraso de las señales y si la pérdida sigue se podrían perder más paquetes ya que como no se está sacando de la cola esta se llenaría y en ese caso la red estaría mandando el paquete que sigue en un loop.
3. Usando el protocolo UDP ya que si perdemos un frame o un segundo de audio es mejor que la conexión no tenga retrasos y la comunicación podría seguir bastante aceptable, se resuelve en la capa de Transporte al igual que en el modelo TCP/IP
4. Reporte del código
    a. El flujo es el siguiente: inicia la interfaz de login para autentificar al usuario y las direcciones IP de los servidores local y remoto, posteriormente se inicia la ventana de chat y los hilos correspondientes al cliente y al servidor, el servidor local recibe mensajes de los dos clientes, con la diferencia de que el cliente remoto hace que los guarde en un buffer y el cliente local obtiene los mensajes de ese buffer. Para la parte de la audio se definen hilos nuevos que llevan a cabo ya sea la grabación de audio y su transmisión o bien la recepción del mismo y su reproducción; estos hilos se ejecutan cuando se selecciona el botón definido en la interfaz.

    Para la parte del video se definen hilos nuevos que llevan a cabo la grabación del video y la transmisión, antes de mandar se convierten los “frames” en una cadena de caracteres para pasar a ser convertida en binario y del lado donde se recibe se reconstruye el video.

    b. En el directorio GUI se localizan los archivos *.ui, que contiene la especificación gráfica de los elementos que se dibujan en la interfaz del usuario, además de los archivos:
        i. **Chat.py**. Contiene la clase Chat, que se encarga de definir la ventana de chat, dibujar y obtener texto en y desde los campos de texto.
        ii. **LlamadaCurso.py**. Que define una ventana que se muestra cuando se lleva a cabo una llamada de audio y video
        iii. **Login.py**. Contiene dos clases: RequestHandler, que inicializa la ventana de Login para la conexión del chat y todos los servicios; Ventana, que toma de la interfaz las ip's, y el nombre de usuario e inicializa los servidores y la instancia del usuario; y App, que es la clase principal que encapsula la clase Ventana dentro de una aplicación propia.
    c. En el directorio Channel se encuentran los siguientes archivos:
        i. **ApiClient.py**. Define la clase que implementa el comportamiento del lado del cliente del chat: inicia un hilo para tener el servidor siempre activo y si hay un mensaje en el buffer lo pasa a la interfaz; envía un mensaje al servidor destino; inicia los hilos de audio y llama a los métodos para la llamada; reproduce el audio que va llegando del otro

usuario; envía el audio al servidor destino; termina la llamada parando los hilos ejecución.

ii. **ApiServer.py**. Define la clase que implementa el comportamiento del lado del cliente del servidor: define los procedimientos que ofrece nuestro servidor, que serán llamados por el cliente con el que estamos hablando, debe de hacer lo necesario para mostrar el texto en nuestra pantalla o bien reproducir el audio recibido; indica si el servidor se encuentra en servicio; deja vacío el buffer de mensajes o de audio del servidor y los regresa a la interfaz.

iii. **Channel.py**. Debería contener métodos independientes que implementen la comunicación mediante diferentes técnicas que no afectan al resto de la aplicación.

iv. **ThreadEx.py**. Contiene una clase que extiende a la clase threading.Thread, de manera que hace posible detener la ejecución del hilo.

d. Principales problemas:
   i. Modularizar los métodos que llevan a cabo la comunicación directa entre las instancias de la aplicación, es decir, independizar la conexión directa de manera que deje de estar en las clases Api y más bien se implemente en la clase Channel.
   ii. Enviar el video correctamente y mostrarlo al ser recibido
e. Si el usuario presiona múltiples veces el botón llamar, la aplicación produce una excepción, que no afecta el funcionamiento del chat, ni el de la primera llamada en curso.
f. El problema no solucionado fue el que se explica en el inciso **d**.