Department of
Computer Science and Engineering
University of Dhaka

Lets Meet: Real-Time Video Conferencing Application with Advanced Networking Implementation and Performance Optimization

---

CSE 3111: Computer Networking Lab
Batch: 28 (3rd Year 1st Semester 2025)

Submitted To
Dr. Ismat Rahman (IsR)
Mr. Jargis Ahmed (JA)
Mr. Palash Roy (PR)

Submitted By
[Your Name] (Roll: [Your Roll])
[Partner Name] (Roll: [Partner Roll])

Date of Submission: June 26, 2025

Contents

1 TITLE OF THE PROJECT PROPOSAL

Project Title: Lets Meet - Real-Time Video Conferencing Application

Summary: This project aims to develop a comprehensive video conferencing application that demonstrates fundamental computer networking concepts through practical implementation. The application will enable real-time audio and video communication between multiple participants over a network, incorporating advanced networking protocols, socket programming, and multimedia streaming technologies. The system will address the growing need for reliable, low-latency communication platforms while showcasing core networking principles such as client-server architecture, peer-to-peer communication, quality of service management, and network security.

2 PROBLEM DOMAIN & MOTIVATIONS

The rapid digitization of communication and the increasing demand for remote collaboration tools have highlighted the critical importance of efficient video conferencing systems. Our motivation for developing "Lets Meet" stems from several key factors:

- **Mo1 - Educational Application of Networking Concepts:** The need to practically implement and understand complex networking concepts such as socket programming, TCP/UDP protocols, real-time data transmission, and network optimization through a real-world application.
- **Mo2 - Performance and Quality Optimization:** Current video conferencing solutions often suffer from latency issues, poor video quality, and network congestion problems. There is a need to explore and implement advanced networking techniques to minimize these issues.
- **Mo3 - Cost-Effective Communication Solution:** Many commercial video conferencing platforms require expensive licenses or have limitations on meeting duration and participant count. A custom solution can provide unrestricted access while maintaining quality.
- **Mo4 - Network Infrastructure Understanding:** The project provides an opportunity to deeply understand how real-time multimedia applications handle network challenges such as packet loss, jitter, bandwidth limitations, and network congestion.
- **Mo5 - Security and Privacy Concerns:** With increasing awareness about data privacy, there is a growing demand for secure communication platforms where users have control over their data transmission and storage.

3 OBJECTIVES/AIMS

The primary objectives and goals of the "Lets Meet" project are:

- **Ob1 - Real-Time Communication Implementation:** Develop a robust real-time audio and video communication system that can handle multiple concurrent users with minimal latency and high-quality transmission.

- **Ob2 - Advanced Networking Concepts Integration:** Implement and demonstrate key networking concepts including socket programming, client-server architecture, peer-to-peer communication, quality of service (QoS) management, and network protocol optimization.
- **Ob3 - Scalable Architecture Design:** Create a scalable system architecture that can efficiently handle varying numbers of participants while maintaining performance and ensuring optimal resource utilization.
- **Ob4 - Network Performance Optimization:** Implement advanced techniques for handling network challenges such as adaptive bitrate streaming, error correction, packet loss recovery, and bandwidth management.
- **Ob5 - Security and Authentication:** Integrate comprehensive security measures including user authentication, encrypted data transmission, and secure connection establishment.
- **Ob6 - Cross-Platform Compatibility:** Ensure the application works seamlessly across different operating systems and network configurations.

4 Project Features

The "Lets Meet" video conferencing application incorporates the following major features based on our multi-socket client-server architecture:

4.1 Core Communication Features:

- **Real-Time Video Streaming:** Continuous video frame capture and transmission using UDP sockets for low-latency communication between multiple participants.
- **Real-Time Audio Communication:** High-quality audio streaming with microphone input capture and playback using dedicated UDP audio sockets.
- **Multi-Party Conferencing:** Support for multiple simultaneous participants with efficient client management through centralized server architecture.

4.2 Advanced Networking Features:

- **Multi-Socket Architecture:** Implementation of separate sockets for different data types - TCP for text messages and file transfers, UDP for video and audio streaming.
- **Protocol Optimization:** Strategic use of TCP for reliable data transmission (messages, files) and UDP for real-time media streaming to minimize latency.
- **Message Serialization:** Custom message handling using Python's pickle serialization for efficient data packaging and transmission.
- **Threaded Communication:** Multi-threaded client-server communication using QThread for non-blocking GUI operations.

4.3 User Interface and Experience:

- **PyQt6 GUI Interface:** Modern, responsive desktop application interface with real-time video display for all participants.
- **Camera and Microphone Controls:** Toggle functionality for enabling/disabling video and audio streams with immediate visual feedback.
- **Real-Time Chat System:** Integrated text messaging system with file sharing capabilities alongside video communication.
- **Dynamic Participant Display:** Automatic UI updates when participants join or leave the conference.

4.4 File Sharing and Media Management:

- **File Transfer System:** Chunk-based file sharing through TCP connections with progress tracking and error handling.
- **Media Device Integration:** Direct camera and microphone access through custom Camera and Microphone classes.
- **Client State Management:** Comprehensive tracking of all connected clients with their video and audio status.
- **Signal-Based Updates:** Qt signal-slot mechanism for real-time GUI updates without blocking the main thread.

5 Block Diagram/Work Flow Diagram

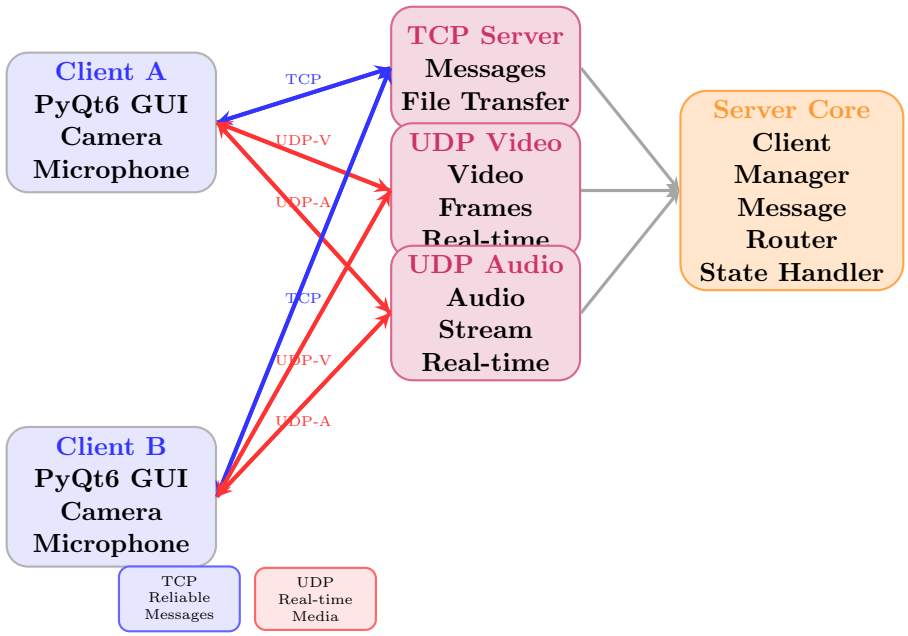

Figure 1: Modern Multi-Socket Video Conferencing Architecture

System Workflow & Data Flow:

1. **Client Initialization:** PyQt6 GUI launches with integrated Camera and Microphone objects for real-time media access

2. **Multi-Protocol Connection:** Each client establishes three specialized socket connections:

   - TCP Socket: Reliable transmission for chat messages and file transfers
   - UDP Video Socket: Low-latency video frame streaming
   - UDP Audio Socket: Real-time audio data transmission

3. **Threading Architecture:** ServerConnection class orchestrates multiple QThreads for concurrent operations without GUI blocking

4. **Media Broadcasting:** Continuous capture and transmission loop (`media_broadcast_loop()`) handles real-time media streaming

5. **Message Processing:** Central message handler (`handle_msg()`) manages participant events (ADD/REMOVE/POST/DISCONNECT)

6. **Real-Time Updates:** Qt signal-slot mechanism ensures instant GUI updates for dynamic participant management

6 TOOLS & TECHNOLOGIES

The following tools and technologies will be utilized throughout the project development:

7.1 Programming Languages & Frameworks:

- **Python:** Primary programming language for both client and server-side development, providing extensive networking libraries and multimedia processing capabilities through built-in socket programming and threading support.
- **PyQt6:** Modern cross-platform GUI framework for creating the desktop application interface, providing native look and feel with powerful signal-slot mechanism for real-time updates.
- **Threading (QThread):** For managing multiple concurrent operations including media capture, data transmission, and GUI updates without blocking the main application thread.

7.2 Networking & Communication:

- **Python Socket Library:** Core networking implementation using both TCP and UDP sockets for different data types - TCP for reliable message/file transfer and UDP for real-time media streaming.
- **Pickle Serialization:** Python's built-in serialization library for converting Message objects and data structures into transmittable byte streams.
- **Multi-Socket Architecture:** Custom implementation of separate socket connections for main communication, video streaming, and audio streaming to optimize network performance.

7.3 Media Processing & Hardware Integration:

- **Camera Class:** Custom Python class for video capture and frame processing, handling camera initialization, frame grabbing, and video stream management.
- **Microphone Class:** Custom audio capture implementation for real-time microphone input processing and audio stream generation.
- **Media Device Management:** Direct integration with system cameras and microphones through Python's multimedia libraries and Qt's media handling capabilities.

7.4 Application Architecture:

- **Client Class:** Core client representation managing individual participant state, media devices, and connection status.
- **ServerConnection Class:** QThread-based server communication handler managing all client-server interactions and data flow.
- **Message Class:** Custom message structure for standardized communication protocol between clients and server.
- **Constants Configuration:** Centralized configuration management for server addresses, ports, and application settings.

7.5 GUI & User Experience:

- **MainWindow Class:** Primary GUI controller managing video display, chat interface, and user controls with real-time updates.
- **Qt Signals & Slots:** Event-driven programming model for handling user interactions, network events, and GUI updates without thread blocking.
- **Dynamic UI Updates:** Real-time participant management with automatic GUI adjustments when clients join or leave conferences.

7.6 Development & Testing:

- **Python Standard Library:** Extensive use of built-in libraries for networking, threading, data serialization, and system integration.
- **Cross-Platform Compatibility:** PyQt6 and Python ensure the application runs seamlessly across Windows, macOS, and Linux systems.
- **Modular Design:** Separation of concerns with distinct modules for GUI (qt_gui.py), networking, media handling, and configuration management.

7 CONCLUSION

The "Lets Meet" video conferencing application represents a comprehensive implementation of advanced computer networking concepts through a practical desktop application built with PyQt6 and Python socket programming. This project demonstrates our deep understanding of multi-socket architecture, real-time data transmission, and efficient client-server communication protocols.

Through the development of this application, we have gained hands-on experience with complex networking protocols including strategic use of TCP for reliable data transmission and UDP for low-latency media streaming. The project showcases various networking concepts including multi-threaded socket programming, message serialization using pickle, dynamic client management, and real-time multimedia communication.

The multi-socket architecture with separate connections for different data types (TCP for messages/files, UDP for video/audio) demonstrates optimal protocol selection based on data requirements. The implementation of threaded communication using QThread ensures non-blocking operations while maintaining real-time performance for video conferencing functionality.

The PyQt6-based desktop interface provides a modern, responsive user experience with real-time video display, chat functionality, and seamless file sharing capabilities. The modular design with separate classes for Client, ServerConnection, Camera, and Microphone management showcases proper software engineering principles alongside networking implementation.

By completing this project, we have created a robust, efficient video conferencing platform that not only meets the academic requirements of CSE 3111 but also demonstrates practical applications of computer networking principles including socket programming, protocol optimization, and real-time multimedia streaming. The project serves as a comprehensive example of how theoretical networking concepts can be applied to solve real-world communication challenges.

This project perfectly aligns with the objectives of the Computer Networking Lab course, combining theoretical knowledge with practical implementation to create a functional application that addresses modern communication needs while demonstrating mastery of core networking concepts.