

# Descrizione preliminare del Progetto

## WebRTC P2P Remote Controller

### Introduzione e Obiettivi

Il progetto ha l'obiettivo di realizzare un sistema di **controllo remoto in tempo reale** che permetta di utilizzare uno smartphone come dispositivo di input (gamepad/controller) per interagire con un'applicazione in esecuzione su un computer desktop (Host).

L'architettura deve essere **Peer-to-Peer (P2P)** per garantire la minima latenza possibile, eliminando la necessità di un server centrale per il transito dei dati di input durante la sessione attiva. Il sistema deve dimostrare l'utilizzo avanzato dei protocolli **WebRTC**, in particolare dei **Data Channels**, e la gestione delle problematiche di attraversamento NAT (NAT Traversal).

### Specifiche Informali

#### Descrizione dello Scenario

L'utente apre una pagina web sul proprio PC Desktop ("Host"). Il sistema genera un codice univoco (o QR Code) per una sessione. L'utente scansiona il codice con il proprio smartphone ("Controller"). I due dispositivi instaurano una connessione diretta. Da quel momento, ogni pressione di tasto o movimento sullo smartphone viene replicato istantaneamente sul PC.

Il sistema è composto da due moduli principali:

1. **Modulo Host (PC):** L'applicazione desktop o web che riceve gli input e l'audio, e visualizza l'interfaccia di connessione.
2. **Modulo Client (Mobile):** L'interfaccia utente sul telefono che funge da controller e sorgente audio.

La comunicazione avviene tramite protocollo WebRTC (o equivalente) per garantire una connessione diretta P2P a bassa latenza sulla rete locale o via internet.

### Requisiti Funzionali

#### Flusso di Connessione (Pairing)

Il processo di connessione deve essere rapido e senza configurazioni IP manuali.

- **RF-01 Generazione Sessione (Host):** All'avvio, il Modulo Host deve generare un ID di sessione univoco e istanziare un canale di segnalazione.
- **RF-02 Visualizzazione QR Code:** Il Modulo Host deve mostrare a schermo un codice QR contenente le informazioni necessarie alla connessione (URL del client mobile + Session ID + Token di autenticazione).
- **RF-03 Scansione e Handshake (Client):** Il dispositivo mobile, tramite la fotocamera, scansiona il QR code. L'applicazione deve decodificare i dati e avviare immediatamente la negoziazione della connessione P2P con l'Host.
- **RF-04 Conferma Connessione:** Una volta stabilito il canale P2P, entrambi i dispositivi devono mostrare un feedback visivo (es. "Connesso", cambio schermata da QR a Controller).

## Interfaccia Controller (Modulo Client)

L'interfaccia mobile deve replicare un gamepad fisico con controlli touch reattivi.

- **RF-05 D-Pad Virtuale:**
  - Deve essere presente una croce direzionale (D-Pad) sulla parte sinistra dello schermo.
  - Deve rilevare 4 direzioni (Su, Giù, Sinistra, Destra) e combinazioni diagonali se supportate.
  - Deve inviare l'evento "pressione" e "rilascio" immediato all'Host.
- **RF-06 Pulsanti Azione:**
  - Devono essere presenti due pulsanti principali, denominati **A** e **B**, posizionati sulla parte destra per l'uso con il pollice.
  - Il layout deve essere ergonomico per l'uso in modalità landscape (orizzontale).
- **RF-07 Toggle Microfono:**
  - Deve essere presente un pulsante dedicato (icona microfono) per attivare/disattivare la cattura audio.
  - **Stato Mute:** L'invio dell'audio è sospeso. Icona sbarrata o rossa.
  - **Stato Active:** L'invio dell'audio è attivo. Icona accesa o verde.
- **RF-08 Feedback Visivo/Aptico:** Alla pressione di qualsiasi tasto virtuale, l'interfaccia deve fornire un feedback visivo (illuminazione tasto) e, se possibile, aptico (vibrazione del telefono).

## Gestione Audio (Audio Bridge)

Il telefono funge da microfono remoto per il PC.

- **RF-09 Cattura Audio:** Quando il Toggle Microfono è attivo (vedi RF-07), il Modulo Client deve acquisire l'audio dal microfono integrato dello smartphone.
- **RF-10 Streaming Audio P2P:** Il flusso audio deve essere codificato (es. Opus codec) e trasmesso in tempo reale tramite il canale P2P all'Host.
- **RF-11 Ricezione e Output (Host):** Il Modulo Host deve ricevere il flusso audio e renderlo disponibile al sistema operativo o all'applicazione target (es. riproducendolo su un dispositivo audio virtuale o passandolo direttamente al software di gioco).

- *Nota:* L'utilizzo finale dell'audio (es. chat vocale in gioco) è demandato al software di terze parti; questo sistema si limita al trasporto del segnale.

## Gestione Input (Lato Host)

- **RF-12 Mapping Input:** L'Host deve tradurre i pacchetti dati ricevuti dal Client in eventi di input comprensibili per il PC (es. pressione tasti tastiera 'W', 'A', 'S', 'D' per il D-pad, o emulazione XInput per controller Xbox).

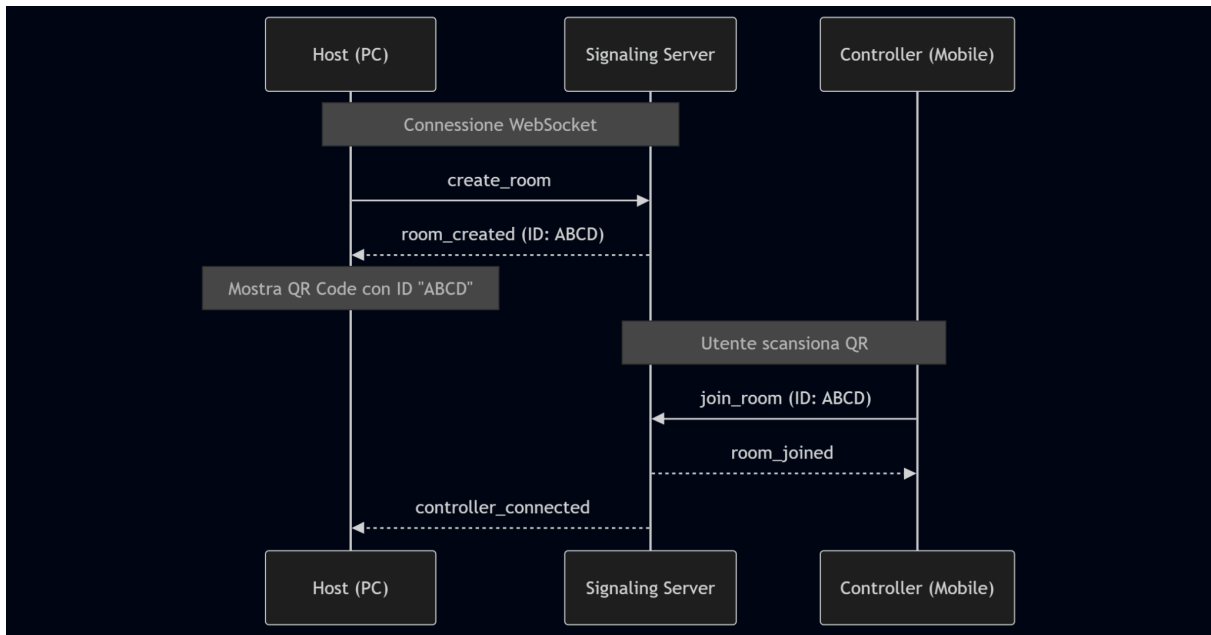
## Requisiti Non Funzionali

- **RNF-01 Latenza:** La latenza tra la pressione del tasto sul Client e la ricezione sull'Host deve essere inferiore a 100ms (ideale < 50ms) per garantire la giocabilità.
- **RNF-02 Sincronizzazione Audio:** L'audio trasmesso non deve avere un ritardo percepibile superiore a 200ms rispetto alla voce reale per evitare l'effetto "fuori sincrono" nelle comunicazioni.
- **RNF-03 Responsività UI:** L'interfaccia mobile deve adattarsi automaticamente alle dimensioni dello schermo dello smartphone (Responsive Design) e forzare l'orientamento orizzontale.
- **RNF-04 Resilienza:** Se la connessione cade, l'Host deve mettere in pausa l'input e mostrare un avviso, tentando la riconnessione automatica se possibile.
- **RNF-05 Riconoscimento del dispositivo:** La web app riconosce se è un cellulare a contattarla per primo e nel caso, invece di mostrare il QR-code (vedi RF-02), avvisa di leggere il QR-code dal pc con la fotocamera.

## Diagrammi di Sequenza (Flussi Operativi)

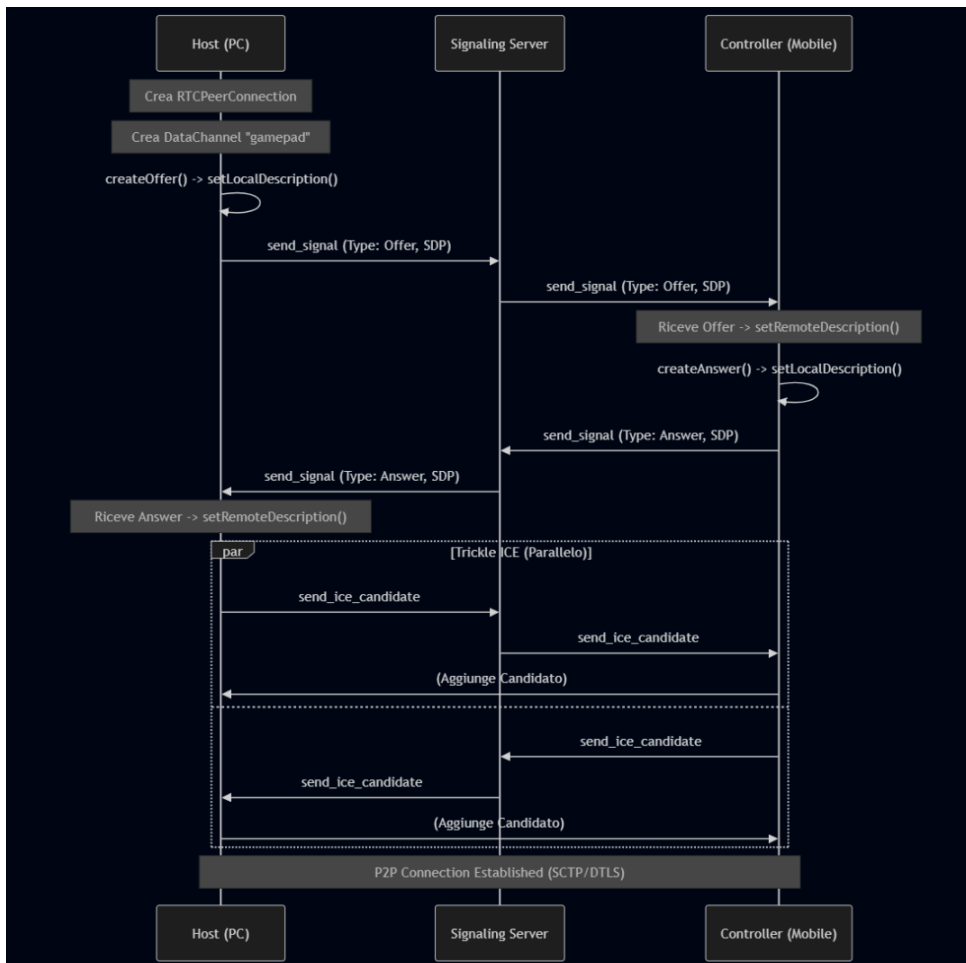
### Fase 1: Discovery & Room Creation (Signaling via WebSocket)

L'obiettivo è far entrare i due client nella stessa "stanza" logica sul server per potersi scambiare i messaggi WebRTC.



## Fase 2: WebRTC Handshake (JSEP)

Una volta nella stessa stanza, avviene la negoziazione P2P. Il server agisce solo da relay trasparente.



### Fase 3: Data Transmission (Runtime)

Il server di signaling non è più coinvolto. La comunicazione è diretta.

