# HiFiReM: un approccio unificato, web e nativo, per la didattica musicale remota

**Matteo Sacchetto**[1], **Cristina Rottondi**[1]
**Antonio Servetti**[2]
[1]DET - [2]DAUIN, Politecnico di Torino
name.surname@polito.it

**Louena Shtrepi, Marco Masoero**
DENERG, Politecnico di Torino
name.surname@polito.it

**Andrea Valle**
StudiUm/CIRMA - Università di Torino
name.surname@unito.it

## ABSTRACT

Le applicazioni professionali di trasmissione audio a bassa latenza finalizzate a supportare sessioni musicali in rete sono state tradizionalmente sviluppate come software nativi, specifici per ciascun sistema operativo, con conseguenti difficoltà di installazione, configurazione e utilizzo. Ciò ne ha ristretto l'uso ad una nicchia di musicisti esperti di informatica. Questo studio propone una implementazione fruibile via browser, realizzata sia come applicazione nativa sia come applicazione web, rivolta in particolar modo alla didattica musicale, al fine di facilitare lo svolgimento di sessioni musicali in rete con basso ritardo di trasmissione ed elevata qualità audio.

## 1. INTRODUZIONE

L'interesse per la realizzazione di performance musicali a distanza (Networked Music Performance, in breve NMP), spinto dalle necessità derivate dal confinamento imposto durante la recente emergenza pandemica, ha messo in evidenza l'esistenza di numerose soluzioni software per suonare in rete sviluppate in ambito accademico, ma il cui uso è ancora limitato all'interno di nicchie di utilizzatori altamente specializzati, soprattutto per quanto riguarda le conoscenze tecnico-informatiche.

Per contro, le soluzioni più adottate nella pratica, soprattutto per la didattica musicale remota, si sono avvalse di strumenti per videoconferenza già largamente diffusi e conosciuti (ad es. Skype, Google Meet, Zoom) i quali presentano evidenti problemi di prestazioni e qualità.

Quanto accaduto, preferire soluzioni di minore qualità, ma dall'utilizzo più immediato, ha evidenziato la necessità e l'urgenza di migliorare la facilità d'uso delle soluzioni per NMP già esistenti. In tale ottica, il progetto HiFiReM descritto in questo articolo si pone due obiettivi fondamentali: $i$) realizzare un'applicazione per la didattica musciale remota utilizzabile via web, senza necessità di installazione e fruibile direttamente via browser; $ii$) sviluppare una soluzione nativa (basata su una box hardware con l'applicazione integrata) che soddisfi esigenze professionali di alta fedeltà audio, ma sempre facilmente utilizzabile tramite la stessa interfaccia web.



Dopo una breve panoramica sullo stato dell'arte nella Sezione 2, segue nella Sezione 3 la descrizione dell'architettura della soluzione proposta, mentre i risultati ottenuti durante una fase di validazione preliminare sono riportati in Sezione 4. La Sezione 5 conclude l'articolo.

## 2. STATO DELL'ARTE

La quasi totalità delle applicazioni per NMP (ad es. SonoBus [1], JackTrip [2], SoundJack [3]) sono applicazioni native, ossia software sviluppati per un dispositivo specifico utilizzando un linguaggio di programmazione anche esso specifico per la piattaforma ove vengono eseguiti (macOS, Windows, Android). Un'applicazione web, invece, è scritta in un linguaggio più "generico" che viene interpretato ed eseguito da un browser, pertanto può essere utilizzata su ogni piattaforma senza necessità di adattamento o installazione.

L'applicazione web paga il vantaggio della semplicità di utilizzo in termini di maggiori limitazioni nell'ottimizzazione delle proprie prestazioni, in particolare per quanto riguarda la riduzione della latenza di acquisizione, elaborazione e trasmissione dei dati audio, perché è forzata ad utilizzare soltanto le funzionalità che il browser le mette a disposizione. Tali limitazioni hanno spinto, fino ad oggi, a preferire la modalità nativa per applicazioni "esigenti" quali quelle di NMP [4].

Tuttavia l'implementazione degli standard Web Audio e Web RTC nei maggiori browser [1] ha permesso di migrare sul web anche applicazioni per l'elaborazione e la trasmissione audio: per primi i programmi di videoconferenza, in seguito anche software per intere digital audio workstation (es. Audiotool [2]).

Allo stato attuale, tuttavia, non sono ancora presenti soluzioni web in grado di trasmettere audio a basso ritardo e senza compressione, dal momento che l'implementazione base di WebRTC non lo permette. Soluzioni web come Google Meet, Jitsi, LiveLab e altre, soffrono di ritardi bocca-orecchio dell'ordine del centinaio di millisecondi e della riduzione di qualità dovuta alla compressione audio, soprattutto nelle alte frequenze [5]. In aggiunta, gli algoritmi di elaborazione audio pensati per le comunicazioni vocali possono introdurre alterazioni del suono a causa dei meccanismi di aggiustamento automatico del guadagno

[1] Il livello di supporto delle funzionalità nei vari browser può essere verificato online: *https://caniuse.com/audio-api*, *https://caniuse.com/rtcpeerconnection*

[2] *https://www.audiotool.com/*

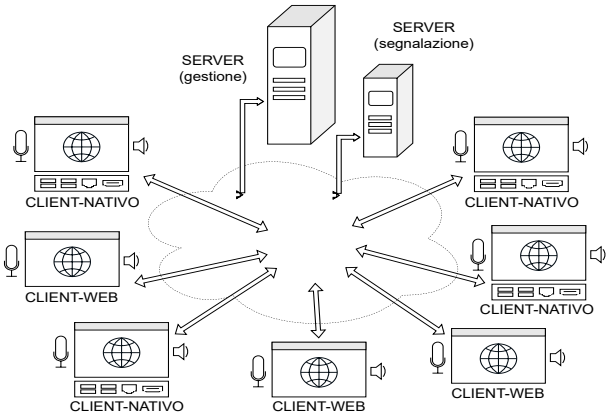


**Figura 1**. Architettura di HiFiReM con le componenti server e le due tipologie di client (web e nativo) unificate dall'utilizzo della stessa interfaccia nel browser.

che modificano i livelli sonori. Inoltre, gli algoritmi di cancellazione del rumore tendono a smorzare suoni stazionari sostenuti, mentre gli algoritmi di adattamento della velocità di riproduzione alle fluttuazioni di banda del canale possono modificare a tratti altezza e timbro dei suoni.

HiFiReM ambisce a superare le limitazioni sopra citate e costituisce il primo approccio unificato, web e nativo, per la trasmissione di audio stereo ad alta qualità (non compresso) per NMP e, nello specifico, finalizzata a supportare conservatori e scuole di musica nell'erogazione della didattica musicale remota.

## 3. ARCHITETTURA

Il sistema HiFiReM è strutturato in due componenti fondamentali, una server e una client. La componente *server* comprende due funzionalità: di *segnalazione*, per gestire la comunicazione tra i diversi client; di *segreteria*: per gestire le necessità di organizzazione didattica. La componente *client* è anche essa suddivisa in due parti/implementazioni, una web ed una nativa. Indipendentemente da quale implementazione venga utilizzata, in entrambe l'interazione utente avviene tramite il browser per mezzo della stessa GUI (Graphical User Interface), come illustrato in Fig.1.

### 3.1 Componente server

La componente server per la gestione della didattica (*segreteria*) permette di integrare nell'interfaccia web un ambiente in cui, a partire dalla definizione dei ruoli di amministratore, segreteria, maestri/docenti e allievi, gli utenti possono essere organizzati in classi. Qui, ciascuna classe può riunirsi in una "stanza" per suonare insieme da remoto. Le stanze rappresentano il concetto di aule didattiche e sono configurate in modo da poter gestire scenari differenti come lezioni individuali, lezioni a gruppi, prove d'insieme (soltanto tra allievi senza il docente) e lezioni con docenti esterni (masterclass).

La componente server per la *segnalazione* si occupa invece di mettere in comunicazione i vari client. Questa componente tiene traccia delle varie stanze attive e si occupa di selezionare il protocollo di segnalazione corretto in base al tipo di stanza, astraendo dunque il fatto che si stiano utilizzando due stack tecnologici completamente diversi tra la componente client–nativo e quella client–web [3] .

### 3.2 Componente client–web

La componente client–web rappresenta al momento l'unica soluzione web per la trasmissione a bassisimo ritardo con audio originale, senza le alterazioni causate dalla compressione digitale e dall'elaborazione audio intrinseca del browser.

L'implementazione è resa possibile dall'utilizzo innovativo degli standard Web Audio e WebRTC Invece di affidare la gestione del flusso audio ai componenti convenzionali *MediaStream* ed *RTCPeerConnection* si è provveduto a "dirottare" lo stream audio sull'*RTCDataChannel* (pensato convenzionalmente per la trasmissione di dati, non di flussi audio) prelevando i campioni dal MediaStream tramite un *AudioWorklet*. Per i dettagli tecnici si rimanda a [6].

Nell'implementare questa soluzione, che permette di poter controllare a più basso livello il flusso di trasmissione secondo le proprie esigenze, è stato inoltre possibile ridurre l'overhead di gestione tramite l'utilizzo degli *SharedArrayBuffers* così da non introdurre ritardi addizionali e ottenere una latenza molto ridotta [5].

### 3.3 Componente client–nativo

La componente client–nativo è stata realizzata per gli scenari più esigenti, non limitati all'interazione del docente con il singolo allievo, ma in cui più soggetti suonano contemporaneamente. Questa soluzione, realizzata su *box* Raspberry Pi 4B, è visibile in Fig. 2: l'audio è acquisito, trasmesso e riprodotto direttamente tramite la box, con latenza compatibile con altri ambienti per NMP [7], mentre l'interazione dell'utente (configurazione, connessione, controllo) avviene tramite browser riutilizzando, in modo integrato, la stessa interfaccia dell'implementazione del client web.

[3] Si noti però che, a causa di sostanziali differenze in termini di ritardi, riportati in Sez. 4.2, non è al momento ragionevole connettere a una stessa stanza client web e nativi perché gli utenti web sarebbero temporalmente disallineati rispetto agli utenti nativi

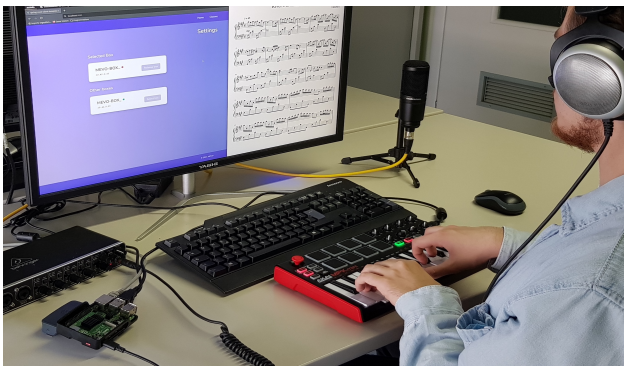

**Figura 2**. Utilizzo del sistema HiFiReM tramite applicazione web (finestra lilla sul monitor) o box Raspberry Pi 4B (in basso a sinistra) sempre per mezzo dell'interfaccia unificata nel browser.

## 4. VALIDAZIONE E RISULTATI

Dal punto di vista delle funzionalità del sistema il tratto caratterizzante della soluzione proposta, specifico per la didattica musicale remota, è rappresentato dalla concezione delle *stanze* per l'esecuzione musicale come ambienti didattici. A differenza delle classiche soluzioni per NMP dove i musicisti interagiscono tra di loro da pari, qui viene data al docente (o ad un tecnico audio terzo) la possibilità di controllare le impostazioni dei dispositivi degli allievi e provvedere al mixaggio audio personalizzato per docente e studenti.

Dal punto di vista, invece, delle prestazioni del sistema l'implementazione software è stata confrontata con le prestazioni di altri software per NMP al fine di validare i due fattori di merito principali in questo contesto: a) la qualità del segnale audio ricevuto, b) il ritardo globale bocca-orecchio. Per entrambe queste verifiche, al fine di poter valutare le prestazioni ottime ottenibili dai differenti software al netto delle caratteristiche della rete, si è scelto di lavorare in uno scenario di rete locale cablata a 1 Gb/s dove l'influenza di banda di trasmissione, ritardo e perdite fosse trascurabile. Per la stessa ragione il buffer di dejitter è stato dimensionato al valore minimo possibile in modo che i) compensasse il jitter della rete locale rendendo il numero di pacchetti scartati al ricevitore trascurabile, e ii) fosse (grossomodo) identico tra le diverse applicazioni confrontate: dell'ordine di 6–9 ms per le soluzioni native e dell'ordine di 20 ms per le soluzioni web [4] .

### 4.1 Qualità audio

La scelta di trasmettere audio non compresso adottata dalla soluzione proposta e da molti software nativi per NMP garantisce in sè (salvo perdite) la massima qualità audio e non necessita di confronti tra gli applicativi.

Nel caso della soluzione web si è provveduto comunque a svolgere un test di ascolto "indicativo" per confrontare la soluzione proposta (con audio non compresso) con una soluzione WebRTC tradizionale (con audio compresso tramite Opus, controllo del guadagno, riduzione del rumore e cancellazione dell'eco) tra quelle spesso utilizzate per la didattica musicale remota durante l'emergenza COVID-19.

Il test è stato svolto in ambiente controllato con livello di rumore di fondo in accordo con le raccomandazioni della ITU-R BS.1116-1 da parte di sette soggetti che si sono autodichiarati normoudenti e che hanno valutato quattro stimoli corrispondenti a differenti tipologie di audio selezionati tra il materiale già utilizzato in un precedente studio [8]: violino (6,7s), cembalo (6,9s), soprano (2,8s), soprano vibrato (8,1s). Per ogni stimolo, si è confrontato l'audio non compresso trasmesso dalla soluzione proposta con quello compresso trasmesso da una applicazione WebRTC tradizionale tramite un test di tipo ABX [9] presentato in cuffia (Sennheiser 600HD) ad ogni partecipante.

| | Violino | | Cembalo | | Soprano | | Soprano v. | |
|---|---|---|---|---|---|---|---|---|
| Soggetto | P | T | P | T | P | T | P | T |
| #1 | × | ✓ | × | × | × | × | ✓ | ✓ |
| #2 | × | ✓ | × | ✓ | × | ✓ | × | ✓ |
| #3 | × | ✓ | ✓ | × | ✓ | ✓ | × | × |
| #4 | × | ✓ | × | ✓ | × | × | × | ✓ |
| #5 | × | × | ✓ | ✓ | × | ✓ | ✓ | × |
| #6 | × | ✓ | × | ✓ | × | ✓ | × | ✓ |
| #7 | ✓ | ✓ | × | ✓ | × | ✓ | × | ✓ |
| Totale | 1 | 6 | 2 | 5 | 1 | 5 | 2 | 5 |

**Tabella 1**. Risultati del test d'ascolto di tipo ABX, con indicazione, per ciascun soggetto, dei riconoscimenti corretti (✓) e di quelli errati (×) per ciascuna delle due soluzioni, quella proposta (P) e quella terza (T).

In questa procedura vengono presentati all'ascoltatore tre stimoli: lo stimolo $A$ e lo stimolo $B$, che hanno una differenza nota, e lo stimolo $X$. Il compito dell'ascoltatore è d'identificare se $X$ è uguale ad $A$ o a $B$. Se non vi è alcuna differenza udibile tra i due segnali, le risposte dell'ascoltatore dovrebbero essere distribuite in modo binomiale in modo tale che la probabilità di rispondere $X = A$ sia uguale alla probabilità di rispondere $X = B$, ovvero il 50%. Questo punteggio viene interpretato come un'indicazione dell'assenza di differenze percettive tra $A$ e $B$.

Il numero minimo di risposte corrette necessarie per indicare una differenza percettiva può essere dato dalla probabilità cumulativa inversa di una distribuzione binomiale, basata sul numero di prove, livello di confidenza e probabilità di risposta corretta. Per le condizioni del test effettuato il numero minimo di risposte corrette necessarie per indicare una differenza percettiva è 6.

I risultati sono presentati in Tab. 1. Nel caso dell'audio trasmesso dal sistema HiFiReM , in media solo 1,5 soggetti su 7 hanno individuato correttamente l'audio non originale nelle varie prove d'ascolto, mentre nel caso dell'audio trasmesso da una applicazione web terza in media ben 5,25 persone su 7 hanno individuato correttamente l'audio non originale. [5]

### 4.2 Ritardo bocca-orecchio

L'analisi del ritardo bocca-orecchio, i cui risultati sono riportati in Fig.3, ha fornito risultati significativamente diversi nel caso della componente nativa e della componente web. La componente nativa si è dimostrata essere al pari, se non migliore, delle altre implementazioni con cui ci si è confrontati: SonoBus e JackTrip. La componente web ha evidenziato un ritardo bocca-orecchio molto più significativo, tra i 100 e i 150 ms, quando eseguita su sistemi operativi Windows o Ubuntu. Ha ottenuto invece prestazioni decisamente migliori su macOS con browser Firefox, dove la latenza misurata è stata di circa 30-40 ms (ritardi tra 50-60 ms si sono misurati utilizzando invece il browser Chrome su macOS). Seppur non confrontabili con il ritardo delle soluzioni native, questi risultati preliminari indi-

[4] Non è possibile impostare un valore preciso al millisecondo di questo parametro perché spesso è calcolato come multiplo della dimensione di un frame audio, che dipende dalle specifiche scelte implementative dell'applicazione in uso.

[5] Sebbene il test non possa reputarsi statisticamente significativo rappresenta un riscontro promettente sulla qualità del sistema, volendo svolgere successivamente prove più sistematiche su un più ampio numero di soggetti e contenuti audio.

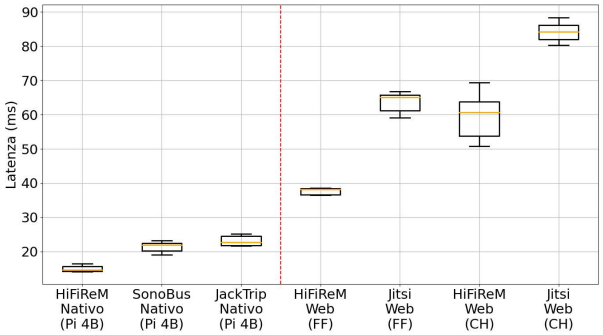


**Figura 3**. Ritardo bocca-orecchio (latenza) delle varie soluzioni analizzate, sia native su Raspberry Pi 4B (a sinistra), sia web su macOS con differenti browser, Mozilla Firefox v.97 (FF) e Google Chrome v.98 (CH) (a destra).

cano un significativo guadagno rispetto al ritardo di altre implementazioni web con compressione audio, che si aggira, nelle stesse condizioni, intorno ai 60 ms per Firefox e agli 80 ms per Chrome su macOS (su Windows e Ubuntu i ritardi eccedono i 150 ms). Il motivo del guadagno in ambiente macOS è verosimilmente dovuto alla migliore integrazione del browser nel sistema di gestione audio: su Windows non è possibile far lavorare il browser con i driver audio ASIO che assicurerebbero un minor ritardo, su Linux, sebbene esistano server audio a bassa latenza come Jack, nelle più comuni distribuzioni non è possibile collegare direttamente i browser a questi ultimi, ma è necessario connetterli attraverso i server audio di default che hanno maggior ritardo [6] .

## 5. CONCLUSIONI E SVILUPPI FUTURI

La realizzazione di un sistema unificato, web e nativo, per la Networked Music Performance presenta al momento ancora dei limiti tecnici da superare. Infatti, mentre non si riscontrano difficoltà rilevanti nel migrare la maggior parte delle applicazioni in ambiente web, nel caso di applicazioni real-time la limitata integrazione del browser con il sistema operativo costituisce ancora un ostacolo per il soddisfacimento dei requisiti di bassa latenza. L'implementazione corrente ha comunque ottenuto il risultato di poter garantire, con ritardi ridotti rispetto alle soluzioni esistenti, un ambiente web per la trasmissione in tempo reale di audio non compresso ad altà fedeltà adeguato per la didattica musicale.

L'integrazione della componente web e della componente nativa in un contesto applicativo indirizzato alla didattica musicale pone le basi per una successiva sperimentazione sul campo a beneficio delle scuole di musica e dei conservatori italiani. Permettere l'esecuzione di queste attività da remoto significa facilitare la partecipazione degli studenti indipendentemente dal luogo di residenza; limitare gli spostamenti riducendo i costi in termini di denaro e tempo, mitigando nel contempo l'inquinamento dovuto alle emissioni dei mezzi di trasporto.

[6] Una soluzione interessante per il futuro è l'integrazione dentro le distribuzioni Linux del server audio PipeWire (https://pipewire.org/) che è specificamente orientato a fornire bassa latenza per l'audio real-time.

## 6. RINGRAZIAMENTI

La parte web di questo progetto è stata avviata come tesi di laurea [10] e pubblicata online su GitHub [11]. Il progetto si è poi evoluto, includendo anche la parte nativa, come attività di dottorato. Si ringraziano le persone che a vario titolo hanno collaborato alle ricerche qui descritte, in particolar modo (in ordine alfabetico): Carlo Barbagallo, Chris Chafe, Leonardo Severi, Luigi Pirisi, Massimiliano Zanoni, Paolo Gastaldi.

Queste ricerche sono state parzialmente finanziate dal Fondo Integrativo Speciale per la Ricerca (FISR) del Ministero dell'Università e della Ricerca (MUR), Programma FISR COVID 2020, tramite il progetto (cod. FISR2020IP_-01206) "Piattaforma ad alta fedeltà per didattica musicale remota e concerti distribuiti su rete Internet".

## 7. BIBLIOGRAFIA

[1] J. Chappell, "SonoBus." https://sonobus.net/, Accesso: giu. 2022.

[2] J.-P. Cáceres and C. Chafe, "JackTrip: Under the hood of an engine for network audio," *Journal of New Music Research*, vol. 39, pp. 183–187, nov. 2010. https://dx.doi.org/10.1080/09298215.2010.481361.

[3] C. Hoene, I. Howell, and A. Carôt, "Networked Music Performance: Developing Soundjack and the Fastmusic Box During the Coronavirus Pandemic," in *AES International Audio Education Conference*, lug. 2021.

[4] L. Vignati, S. Zambon, and L. Turchet, "A comparison of real-time linux-based architectures for embedded musical applications," *J. Audio Eng. Soc*, vol. 70, no. 1/2, pp. 83–93, 2022.

[5] M. Sacchetto, P. Gastaldi, C. Chafe, C. Rottondi, and A. Servetti, "Web-Based Networked Music Performances via WebRTC: a Low-Latency PCM Audio Solution," *Journal of the Acoustical Society of America*, 2022. (in stampa).

[6] M. Sacchetto, A. Servetti, and C. Chafe, "JackTrip WebRTC: Networked Music Experiments in a Web Browser," in *6th International Web Audio Conference (WAC)*, (Barcellona, Spagna), lug. 2021.

[7] C. Rottondi, C. Chafe, C. Allocchio, and A. Sarti, "An overview on networked music performance technologies," *IEEE Access*, vol. 4, pp. 8823–8843, dic. 2016. https://doi.org/10.1109/ACCESS.2016.2628440.

[8] I. Howell, K. J. Gautereaux, J. Glasner, N. Perna, C. Ballantyne, and T. Nestorova, "Preliminary Report: Comparing the Audio Quality of Classical Music Lessons Over Zoom, Microsoft Teams, VoiceLessonsApp, and Apple FaceTime." [online] https://www.ianhowellcountertenor.com/preliminary-report-testing-video-conferencing-platforms, mag. 2020.

[9] W. Munson and M. Gardner, "Standardizing Auditory Tests," *Journal of the Acoustical Society of America*, vol. 22, no. 4, pp. 675–675, 1950. https://doi.org/10.1121/1.1917190.

[10] M. Sacchetto, "JackTrip-WebRTC - Networked Music Performance with Web Technologies," in *Tesi di Laurea Magistrale in Ingegneria Informatica presso il Politecnico di Torino*, 2020. http://webthesis.biblio.polito.it/id/eprint/16659.

[11] M. Sacchetto, "JackTrip-WebRTC." https://github.com/JackTrip-webrtc/JackTrip-webrtc, 2020. Accesso: giu. 2022.