Università degli Studi di Napoli "Parthenope"
Dipartimento di Scienze e Tecnologie
Corso di laurea in Informatica

Elaborato finale di Laurea

Sviluppo di un ambiente virtuale per ridurre il senso di solitudine nella popolazione anziana

Integrazione di agenti virtuali per il sostegno alla socialità

Relatore
Prof.ssa Paola Barra

Candidato
Luca Tartaglia
Matr. 0124002294

Anno Accademico 2024/2025

ABSTRACT

In questa tesi presento **SOULFRAME**, un sistema conversazionale con avatar 3D pensato per rendere l'interazione con l'AI più semplice, naturale e accessibile. L'obiettivo principale è costruire un'esperienza d'uso intuitiva: l'utente deve poter parlare con l'avatar senza complessità tecniche, usando un'interfaccia adattabile sia a desktop (controlli da tastiera) sia a mobile/touch (push-to-talk e navigazione semplificata).

Il progetto è stato sviluppato con un'architettura modulare client-server. Sul lato frontend, **Unity WebGL** gestisce flusso UI, avatar e acquisizione audio. Sul lato backend, microservizi **FastAPI** si occupano di trascrizione vocale con **Whisper (STT)**, risposta contestuale con **LLM + RAG** (Ollama e ChromaDB), sintesi vocale con **Coqui XTTS v2 (TTS)** e gestione/caching degli asset avatar. La memoria è persistente per singolo avatar e può essere arricchita con note, documenti PDF e immagini, anche tramite OCR.

Una parte importante del lavoro riguarda l'automazione operativa: sono stati introdotti script di setup e gestione servizi per installazione e deploy in modo rapido su Windows e Ubuntu, riducendo errori manuali e tempi di configurazione. Durante lo sviluppo sono state affrontate criticità di integrazione tra servizi, latenza end-to-end e compatibilità tra ambienti. I risultati ottenuti mostrano un sistema funzionante, estendibile e sufficientemente robusto per conversazioni vocali contestuali in scenari realistici.

INDICE

1. INTRODUZIONE

1.1 Motivazione e contesto applicativo

La realtà virtuale (VR) si distingue dagli altri media interattivi per la capacità di far sentire l'utente presente in un luogo diverso da quello fisico. Slater e Sanchez-Vives descrivono questa presenza come il risultato di due fattori principali: la *place illusion*, legata alla coerenza tra movimenti e scena virtuale, e la plausibilità degli eventi, cioè quanto l'ambiente reagisce in modo credibile alle azioni dell'utente [1]. Per questo, non basta una buona qualità visiva. Conta anche come il sistema risponde e quanto le interazioni risultano significative.

In questo quadro, la VR viene usata da tempo per training, formazione e simulazioni in ambito medico e professionale, soprattutto quando serve riprodurre situazioni costose, rischiose o difficili da standardizzare. Diversi studi segnalano anche l'interesse per contesti con forte componente sociale, dove realismo e naturalezza dell'interazione influenzano direttamente l'esperienza.

Il limite emerge quando l'ambiente resta statico o gli attori virtuali seguono script rigidi. In questi casi l'interazione si riduce a scelte predefinite e perde adattività. In uno studio sul training in VR, Kan, Rumpelnik e Kaufmann confrontano agenti conversazionali con agenti a audio preregistrato. I risultati mostrano che una pipeline vocale con riconoscimento del parlato e sintesi vocale aumenta in modo significativo la co-presenza percepita [2]. Questo indica che il dialogo non è un elemento accessorio, ma una parte centrale dell'efficacia del sistema.

Gli Embodied Conversational Agents (ECA), cioè agenti conversazionali con corpo virtuale e segnali multimodali, nascono proprio da questa esigenza. La revisione sistematica di Yang e coautori mostra che la maggior parte degli studi in XR si concentra sulla VR, spesso con HMD e con Unity come piattaforma principale [3]. La stessa revisione evidenzia che molte interazioni restano *task-oriented*, ma cresce l'interesse per scambi più dinamici e adattivi grazie a modelli neurali

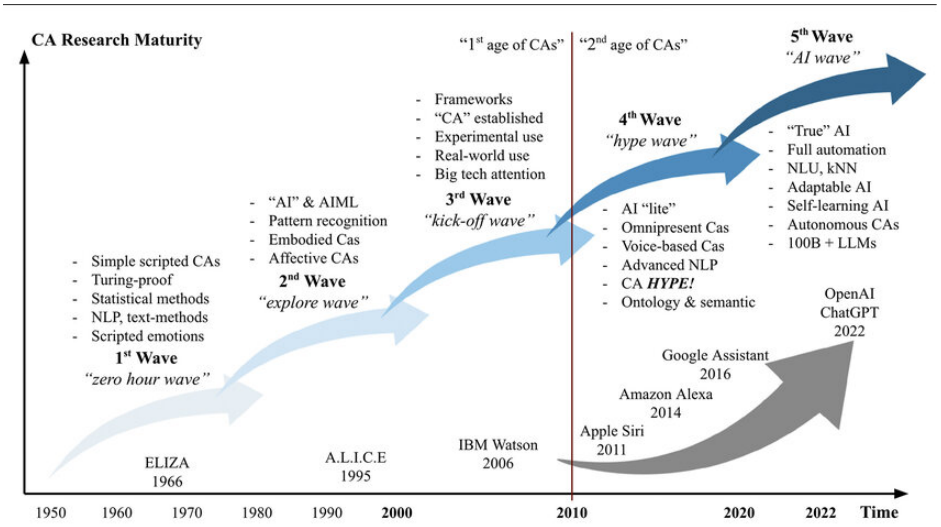

Fig. 1.1: Evoluzione della maturità della ricerca sui Conversational Agents (CA), dalla "zero hour wave" alla "AI wave", con riferimento ai principali passaggi tecnologici. [1]

e, più recentemente, ai *Large Language Models* (LLM). In SOULFRAME, questo filone viene adattato a una configurazione senza HMD, centrata sulla generazione e animazione di avatar a partire da segnali visivi e vocali dell'utente.

Dal punto di vista dei canali, la voce è ormai una scelta frequente sia in input sia in output. Resta però aperta la sfida di integrare dialogo libero e contestuale con un embodiment coerente.

Figura 1.1 sintetizza l'evoluzione dei Conversational Agents (CA) dalla fase iniziale di sistemi scriptati fino all'attuale ondata guidata da modelli linguistici avanzati. In questa traiettoria, l'integrazione di linguaggio naturale, adattività e autonomia rende più rilevante il tema dell'*embodiment* in ambienti immersivi. In questo quadro, SOULFRAME non nasce come semplice demo grafica: punta a collegare presenza sociale e interazione credibile in un ambiente 3D familiare, con la conversazione vocale come asse centrale.

SOULFRAME si inserisce in questo contesto come sistema immersivo con ECA vocale embodied. L'obiettivo è testare una pipeline completa che integra voce e un profilo avatar creato in fase iniziale tramite acquisizione visiva dell'utente, così da replicarne aspetto e stile comportamentale durante il dialogo. Un

[1] Fonte: adattamento da una figura pubblicata su ResearchGate.

punto centrale è la memoria dell'agente: il sistema combina contesto conversazionale e recupero di conoscenza (*RAG*) per generare risposte coerenti con quanto detto e acquisito durante l'interazione. L'ipotesi di fondo è semplice: in ambienti immersivi e familiari, la qualità percepita dipende anche dalla capacità dell'agente di sostenere scambi plausibili, rapidi e contestualmente informati.

1.2 Perimetro e requisiti di progetto

SOULFRAME è un prototipo di interazione immersiva con un ECA vocale embodied in ambiente 3D. Lo scopo è valutare fattibilità tecnica e qualità percepita dell'interazione conversazionale in scenari con componente sociale. Il progetto non vuole essere una piattaforma generale per creare mondi virtuali e non è una soluzione clinica certificata. Questi obiettivi richiedono validazioni regolatorie e studi longitudinali fuori dal perimetro di una tesi triennale.

Il perimetro si concentra quindi sull'integrazione *end-to-end* della pipeline vocale, visiva e dialogica dentro una scena immersiva, con coerenza tra risposta verbale e comportamenti *embodied* dell'agente.

Sul piano tecnologico, il sistema integrato di SOULFRAME usa la fotocamera soprattutto nella fase di configurazione iniziale dell'utente, per creare e salvare il profilo avatar; durante l'interazione ordinaria il sistema si basa principalmente su microfono, memoria di contesto e stato conversazionale. La resa della scena avviene con moduli di generazione/animazione avatar e con un motore real-time 3D. Questa scelta favorisce replicabilità, facilita sostituzioni tra componenti e sostiene un'interazione più familiare, orientata alla replica visiva e comportamentale della persona.

I requisiti funzionali principali derivano dalla necessità di supportare un dialogo naturale in tempo quasi reale. L'input vocale è gestito con logica *push-to-talk*: l'utente tiene premuto un tasto per parlare e rilascia per chiudere il turno. A quel punto il sistema trascrive l'audio con Speech-to-Text (STT) e genera la risposta tramite LLM supportato dalla memoria RAG. La comprensione dell'input emerge dalla combinazione tra modello linguistico, contesto conversazionale e semplici regole applicative. Il testo prodotto viene poi convertito in audio tramite Text-to-Speech (TTS) e restituito attraverso la voce dell'agente. Nella fase di onboarding, il sistema usa la fotocamera per costruire il profilo visivo dell'avatar, che viene

poi riutilizzato durante la conversazione. A questa catena si aggiungono tre funzioni chiave: memoria contestuale con *Retrieval-Augmented Generation* (RAG), descrizione semantica delle immagini per estrarre informazioni dall'ambiente, e acquisizione testuale da documenti tramite OCR, così che l'avatar possa usare anche contenuti visivi e documentali nella conversazione.

I requisiti non funzionali riguardano soprattutto latenza, robustezza e modularità. Per mantenere plausibilità conversazionale, la catena STT–RAG/LLM–TTS deve restare reattiva, anche quando entrano in gioco recupero in memoria, analisi delle immagini e OCR. L'architettura deve anche tollerare guasti parziali senza interrompere l'esperienza immersiva. La modularità è perseguita soprattutto a livello di servizi e interfacce HTTP: i componenti backend possono essere sostituiti in modo relativamente rapido, purché restino compatibili con gli endpoint e i formati attesi dal client.

Per aspetti come la latenza percepita e i criteri di accettabilità, non c'è una soglia unica valida per tutti i contesti applicativi. In questo lavoro i vincoli sono quindi definiti in modo operativo sul prototipo e verificati nei capitoli successivi.

1.3 Obiettivi e contributi di SOULFRAME

Gli obiettivi di SOULFRAME seguono il filone degli studi sugli ECA in XR. Gli studi mostrano una forte presenza di implementazioni VR in Unity, interazioni spesso orientate al compito e una notevole eterogeneità in input, output e metodi di valutazione. Nello stesso tempo, i canali vocali sono molto usati, soprattutto con TTS in uscita [3]. In questo scenario, il progetto punta a costruire un prototipo che renda verificabile l'integrazione *end-to-end* della pipeline vocale con un agente *embodied*, documentando in modo chiaro le scelte tecniche e sperimentali.

Il primo obiettivo (RQ1) riguarda la fattibilità tecnica di un'interazione vocale in tempo reale con un agente embodied in ambiente immersivo 3D. In termini operativi, bisogna dimostrare che la catena STT $\rightarrow$ RAG/LLM $\rightarrow$ TTS $\rightarrow$ output audiovisivo dell'avatar funzioni in modo continuo e stabile, includendo anche descrizione immagini e OCR documentale. La valutazione considera tempi di elaborazione per blocco, completamento dei turni senza errori bloccanti e gestione corretta delle transizioni tra ascolto, elaborazione e risposta. La fattibilità inclu-

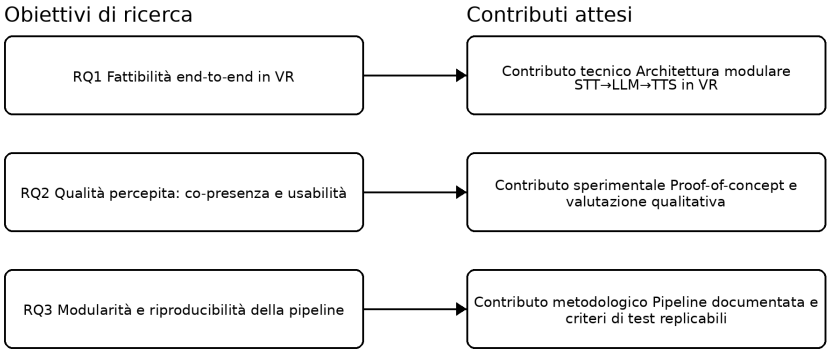

Fig. 1.2: Corrispondenza tra obiettivi di ricerca e contributi del progetto SOULFRAME.

de anche la coerenza dell'embodiment, cioè un avatar che allinei voce, aspetto visivo e segnali comportamentali.

Il secondo obiettivo (RQ2) riguarda la qualità percepita dell'interazione, in particolare co-presenza e naturalezza dialogica. SOULFRAME prevede almeno una valutazione qualitativa guidata, affiancata quando possibile da metriche soggettive usate negli studi VR con ECA: presenza, co-presenza, qualità dell'interazione e qualità della presentazione delle informazioni, oltre a misure di processo come durata dei compiti e performance percepita. Studi comparativi tra agenti conversazionali e audio pre-scriptato mostrano differenze osservabili, soprattutto sulla co-presenza [2].

Il terzo obiettivo (RQ3) riguarda modularità e riproducibilità della pipeline. In pratica, il sistema deve permettere la sostituzione dei componenti principali senza riscrivere l'intera architettura e deve rendere tracciabili configurazioni, tempi e risultati delle prove. Questa sostituzione avviene mantenendo stabili i contratti API tra client e servizi. Questo obiettivo completa i primi due, perché rende il prototipo confrontabile nel tempo e riutilizzabile in test successivi. Figura 1.2 sintetizza la corrispondenza tra i tre obiettivi e i contributi attesi del progetto.

I contributi si distribuiscono su tre livelli. Il contributo tecnico è un'architettura modulare *end-to-end* basata su microservizi (STT, RAG/LLM, TTS e servizi di memoria) con interfacce API esplicite e configurabili. Questo consente di sostituire i componenti con impatto limitato sul resto del sistema, a condizione di mantenere compatibilità dei contratti di scambio. Il contributo sperimentale è un

proof-of-concept verificabile con un set minimo di misure soggettive e osservazioni qualitative, utile a stimare comprensibilità, fluidità e credibilità dell'interazione. Il contributo metodologico è la documentazione riproducibile delle scelte progettuali, con criteri di logging, tracciamento dei tempi e selezione motivata delle misure di *user experience*. Questa impostazione è coerente con i principali lavori sul *dialogue management*, che distinguono approcci *finite-state*, *frame-based* e *agent-based* e raccomandano maggiore standardizzazione nelle metriche tecniche e nella valutazione dell'esperienza utente [4].

1.4 Panoramica del sistema e flusso end-to-end

SOULFRAME adotta un'architettura client–server pensata per supportare dialogo vocale naturale in un ambiente immersivo 3D. Il client gestisce rendering della scena, rappresentazione dell'agente embodied tramite avatar animato e acquisizione dei segnali utente. La cattura audio non è continua: parte quando l'utente tiene premuto il comando *push-to-talk* e termina al rilascio. La cattura video da fotocamera è invece usata in fase iniziale per configurare e salvare il profilo avatar. Il backend mantiene lo stato conversazionale e orchestra la pipeline linguistica e cognitiva: gestione del contesto, recupero di informazioni con *RAG*, generazione con LLM e integrazione di descrizioni di immagini e testi estratti via OCR. Questa separazione permette di tenere sul client le funzioni sensibili al frame-rate e sul backend i moduli più onerosi e soggetti a evoluzione.

Il flusso end-to-end parte dalla voce dell'utente e ritorna alla risposta audiovisiva dell'avatar. Dopo il rilascio del tasto *push-to-talk*, l'audio viene trascritto dal modulo STT. Il testo viene poi inviato al servizio di chat del backend, che funge da orchestratore RAG/LLM. Quando la memoria dell'avatar è presente, il servizio prova a recuperare contesto rilevante con ricerca ibrida; quando non ci sono ricordi utili, la risposta viene generata senza supporto documentale aggiuntivo. Il contesto può essere arricchito con descrizioni delle immagini e con testo proveniente da documenti acquisiti via OCR. Su questa base, il LLM produce la risposta, che viene sintetizzata dal TTS. In parallelo, il client anima l'avatar usando il profilo visivo creato in onboarding e regole di comportamento coerenti con il contesto dialogico, così da mantenere allineamento tra contenuto verbale e resa visiva.

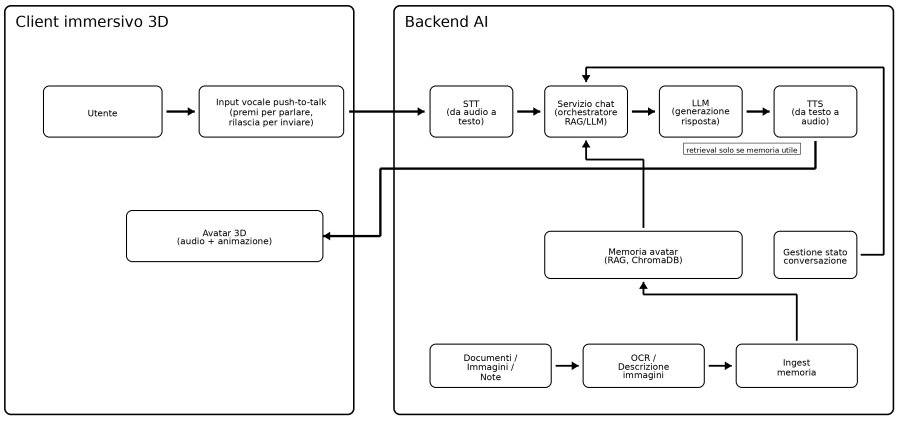

Fig. 1.3: Flusso end-to-end del sistema SOULFRAME: dall'input vocale dell'utente alla risposta dell'agente embodied.

La latenza totale dipende dalla somma delle latenze dei singoli moduli e dalla gestione a turni del *push-to-talk*. L'obiettivo non è eliminare ogni variabilità, ma mantenere continuità tra fine enunciato e risposta dell'agente con tempi percepiti stabili. Il backend coordina gli stati operativi e gestisce interruzioni o riformulazioni dell'utente senza perdere coerenza.

La modularità è un principio guida, ma nel senso operativo del progetto: i servizi sono separati, indirizzabili e configurabili, e possono essere sostituiti mantenendo coerenti endpoint e payload. In questo modo si possono confrontare varianti senza riprogettare tutto il sistema client. Figura 1.3 riassume la catena di elaborazione e la separazione di responsabilità tra client e backend.

1.5 Metodo di lavoro e struttura della tesi

Lo sviluppo di SOULFRAME segue un approccio iterativo basato su prototipazione incrementale. La scelta serve a ridurre il rischio tecnico tipico dei sistemi che combinano vincoli di reattività in interazione a turni (*push-to-talk*), elaborazione linguistica e interazione immersiva. Il lavoro procede per integrazioni successive: prima si valida ogni modulo in isolamento, poi si integra la pipeline completa e si verifica il comportamento in scenari via via più complessi. Questo metodo rende più facile individuare colli di bottiglia e problemi di sincronizzazione

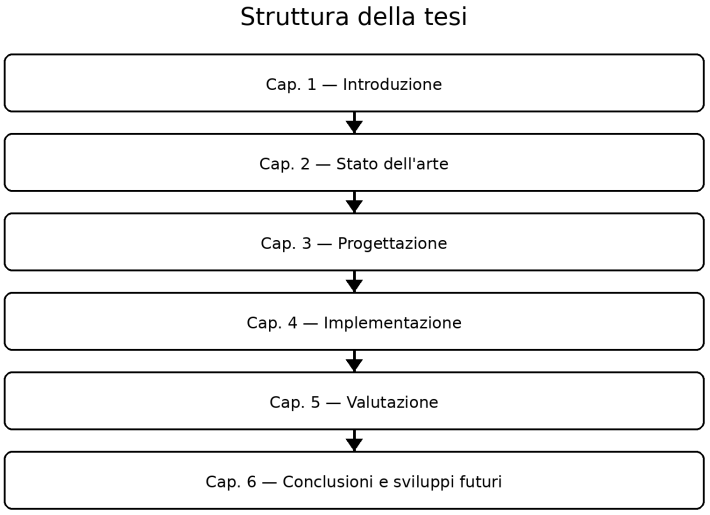

Fig. 1.4: Struttura della tesi e organizzazione dei capitoli.

in fase precoce. Durante tutto il processo vengono tracciate scelte, compromessi e dipendenze per mantenere l'evoluzione del sistema leggibile e replicabile.

La tesi è organizzata in sei capitoli. Il Capitolo 1 introduce problema, perimetro e obiettivi e fornisce la visione d'insieme del sistema. Il Capitolo 2 presenta fondamenti e stato dell'arte su VR, ECA vocali e approcci di memoria conversazionale come RAG, includendo OCR e descrizione immagini come fonti di contesto. Il Capitolo 3 descrive l'architettura di SOULFRAME, i macro-componenti e i criteri progettuali. Il Capitolo 4 entra nelle scelte implementative e tecnologiche, inclusa l'integrazione a microservizi tramite API, la gestione dello stato conversazionale e la resa embodied dell'agente. Il Capitolo 5 riporta test e valutazione, con verifica funzionale della pipeline e stima della qualità percepita. Il Capitolo 6 conclude il lavoro, discute limiti del prototipo e propone sviluppi futuri.

Figura 1.4 offre una vista sintetica della struttura e della progressione logica tra capitoli.

2. FONDAMENTI E STATO DELL'ARTE

2.1 Agenti conversazionali embodied in XR

Gli Embodied Conversational Agents (ECA) sono agenti conversazionali dotati di una rappresentazione corporea, progettati per essere percepiti come interlocutori nello spazio di interazione. In Extended Reality (XR), il corpo virtuale viene collocato in una scena tridimensionale e coordinato con i turni del dialogo, con l'obiettivo di rendere l'interazione più naturale rispetto a un'interfaccia solo testuale.

La letteratura recente mostra che molti sistemi ECA in XR sono sviluppati in Virtual Reality (VR), spesso con Head-Mounted Display (HMD) e con Unity come piattaforma ricorrente. Risulta frequente anche l'uso dell'interazione vocale e di configurazioni uno-a-uno. Molte applicazioni restano orientate a compiti specifici, ma cresce l'interesse verso dialoghi più adattivi supportati da modelli neurali.[3] SOULFRAME si colloca in questo scenario come ECA vocale embodied in un ambiente 3D fruibile via WebGL senza HMD: l'agente conversa in modalità push-to-talk e mantiene un profilo avatar che integra aspetto e voce.

2.1.1 Presenza sociale, co-presenza e ruolo della voce

Quando l'interazione avviene in un ambiente mediato, la presenza sociale descrive la percezione dell'altro come interlocutore con cui si condivide un'esperienza. Una trattazione utile per la valutazione scompone questo fenomeno in componenti distinguibili, includendo la co-presenza (percezione di condividere lo stesso spazio) e forme di coinvolgimento attentivo e comportamentale che rendono il dialogo più contingente e reciproco.[5]

La voce è un canale rilevante per la presenza sociale perché rende immediatamente percepibili tempi di risposta, ritmo e prosodia. In VR questo effetto risulta più marcato quando l'agente è embodied: evidenze sperimentali indicano che un

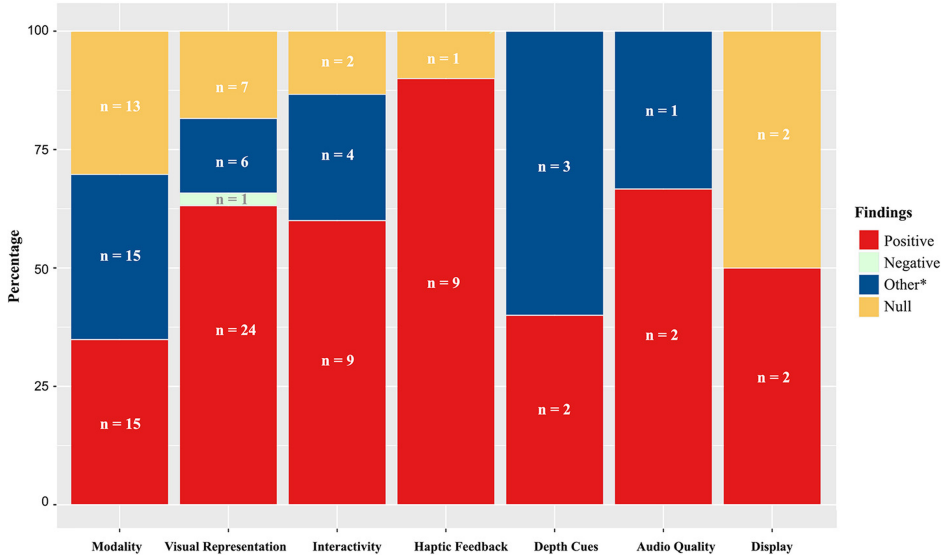

Fig. 2.1: Fattori immersivi associati alla presenza sociale: la qualità audio compare tra le variabili considerate nella letteratura. [1]

ECA con interazione vocale in tempo reale (STT+TTS) aumenta la co-presenza percepita rispetto a una condizione con audio pre-registrato.[2] In SOULFRAME, la scelta del push-to-talk e l'uso di una voce persistente per avatar seguono questa logica, con l'obiettivo di stabilizzare i turni e mantenere coerenza d'identità.

La presenza sociale e la co-presenza dipendono dall'allineamento tra canali comunicativi e tempi dell'interazione. Figura 2.1 mostra che la qualità audio è una delle variabili considerate insieme ad altri fattori immersivi. Per un ECA vocale, qualità della voce e gestione temporale dei turni influenzano la naturalezza percepita; in SOULFRAME, push-to-talk, profilo vocale per avatar, warmup e frasi di attesa sono adottati per mitigare sovrapposizioni e silenzi durante l'elaborazione.

2.1.2 Limiti aperti nei sistemi conversazionali immersivi

L'integrazione del dialogo in ambienti 3D introduce criticità che emergono meno nei sistemi testuali tradizionali. La principale è la latenza end-to-end tra acquisizione audio, trascrizione, generazione e sintesi: ritardi percepibili riducono la naturalezza percepita anche quando il contenuto della risposta è corretto.

[1] Fonte: Oh et al., *A Systematic Review of Social Presence: Definition, Antecedents, and Implications*, Frontiers in Robotics and AI (2018), https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2018.00114/full (Figura 3; licenza/uso: CC BY).

Fig. 2.2: Schema della pipeline conversazionale (STT → RAG/LLM → TTS) adottata in SOULFRAME.

Un secondo limite riguarda la continuità tra turni: senza una memoria affidabile il sistema fatica a mantenere riferimenti e preferenze espresse dall'utente. Un ulteriore nodo è l'integrazione multimodale, che richiede coerenza tra risposta linguistica, tempi e segnali non verbali. Infine, la valutazione resta complessa perché combina metriche soggettive (presenza, co-presenza, naturalezza) e metriche tecniche (latenza, errori di trascrizione), con protocolli non sempre uniformi.

In letteratura, architetture modulari open-source per ECA vocali in VR mostrano che la separazione in servizi STT e TTS semplifica l'integrazione ma rende evidenti i colli di bottiglia temporali, richiedendo strategie come output in streaming.[6] SOULFRAME adotta una pipeline a tre stadi, memoria persistente per avatar e frasi di attesa per ridurre l'impatto dei tempi morti senza dipendenze cloud.

2.2 Pipeline AI adottata in SOULFRAME

SOULFRAME implementa una pipeline conversazionale in tre stadi: riconoscimento del parlato, generazione contestuale e sintesi vocale. Questa scomposizione consente di isolare i vincoli della conversazione a turni, in particolare la latenza, e di aggiornare i singoli moduli senza modificare l'intera architettura. Figura 2.2 mostra il flusso dei dati dal segnale audio in ingresso al testo trascritto, fino al testo di risposta e all'audio sintetizzato.

2.2.1 Speech-to-Text con Whisper

Il primo stadio della pipeline è il riconoscimento automatico del parlato. Nel prototipo, l'audio viene acquisito in push-to-talk dal client WebGL e inviato a un

micro-servizio dedicato che restituisce la trascrizione da inoltrare al modulo di memoria e generazione. L'esecuzione locale riduce dipendenze di rete e supporta requisiti di privacy.

SOULFRAME utilizza Whisper, modello STT basato su architettura Transformer e addestrato su larga scala con supervisione debole. L'addestramento su circa 680.000 ore e l'impostazione multilingue (99 lingue) supportano buone prestazioni zero-shot e robustezza a variabilità di parlanti e condizioni acustiche.[7] Nel prototipo è adottata la versione `medium` come compromesso tra qualità della trascrizione e costo computazionale.

2.2.2 Memoria conversazionale con RAG (LLM + embeddings + retrieval)

Il secondo stadio introduce una memoria conversazionale che combina generazione e recupero di contesto.

In termini operativi, la query testuale viene trasformata in embedding, usata per recuperare dall'indice i passaggi pertinenti e reinserita nel contesto fornito al Large Language Model (LLM). In questo modo la risposta può mantenere continuità tra turni e riutilizzare informazioni già emerse nella conversazione.

L'approccio RAG integra memoria parametrica del modello e memoria non parametrica aggiornata via indice; la conoscenza può quindi evolvere intervenendo sui contenuti recuperabili senza riaddestrare i pesi.[8] Figura 2.3 mostra il flusso retrieval→generation.

Nel prototipo SOULFRAME la memoria è persistente tra sessioni e separata per avatar, per ridurre interferenze tra profili. Il servizio RAG integra LLM via Ollama, modello di embedding e ChromaDB; supporta note testuali e ingestione di documenti (con OCR per PDF) e adotta retrieval ibrido, combinando similarità semantica e segnali lessicali.

2.2.3 Text-to-Speech con Coqui XTTS v2

Il terzo stadio converte il testo in audio e chiude il ciclo conversazionale. In un ECA embodied, la sintesi vocale influisce sulla percezione di continuità e coerenza

[2] Fonte: Kaur et al., *Knowledge, context and personalization in retrieval-augmented generation for healthcare*, Frontiers in Artificial Intelligence (2025), https://www.frontiersin.org/journals/artificial-intelligence/articles/10.3389/frai.2025.1697169/full (Figura 1; licenza/uso: CC BY).

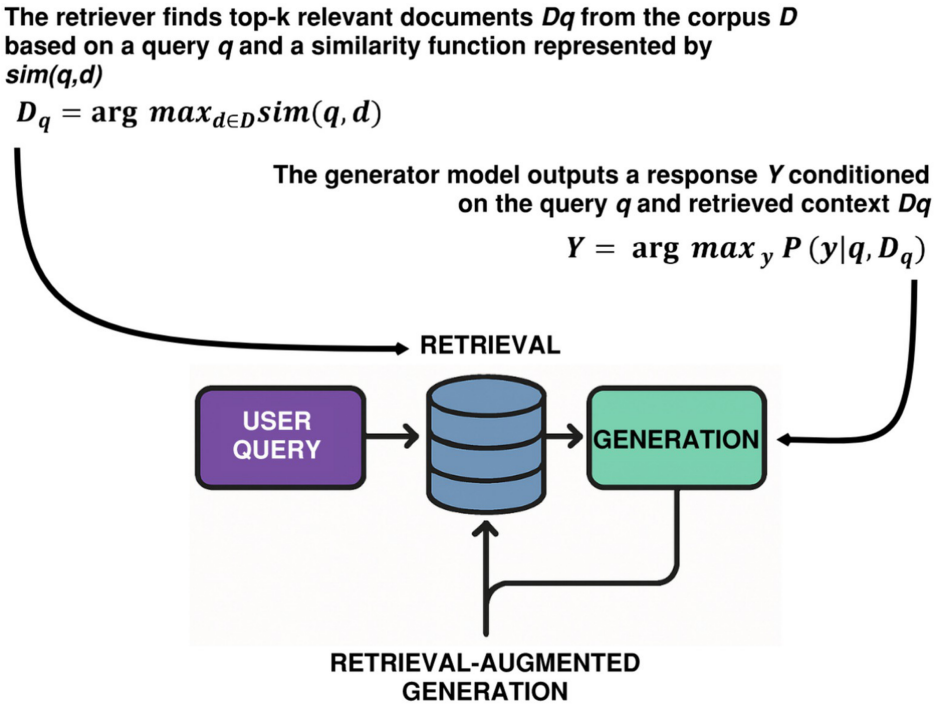

Fig. 2.3: Schema concettuale di Retrieval-Augmented Generation (RAG): recupero di contesto e generazione della risposta. [2]

dell'interlocutore.

SOULFRAME adotta Coqui XTTS v2 per supporto multilingue e voice cloning tramite campione di riferimento, senza addestrare un modello vocale dedicato per ogni utente.[9] Nel prototipo, il campione è validato rispetto al testo atteso e associato al profilo avatar; il servizio supporta sintesi in streaming, warmup e frasi di attesa per mitigare la latenza percepita.

2.3 Integrazione client-server del sistema

L'architettura di SOULFRAME separa il client, responsabile di rendering e interazione, dal backend, responsabile dell'inferenza AI. Questa scelta riduce il carico sul runtime WebGL e permette di gestire i moduli AI come servizi indipendenti. Figura 2.4 mostra il ruolo del reverse proxy come punto di accesso alle API e l'instradamento verso i micro-servizi dedicati.

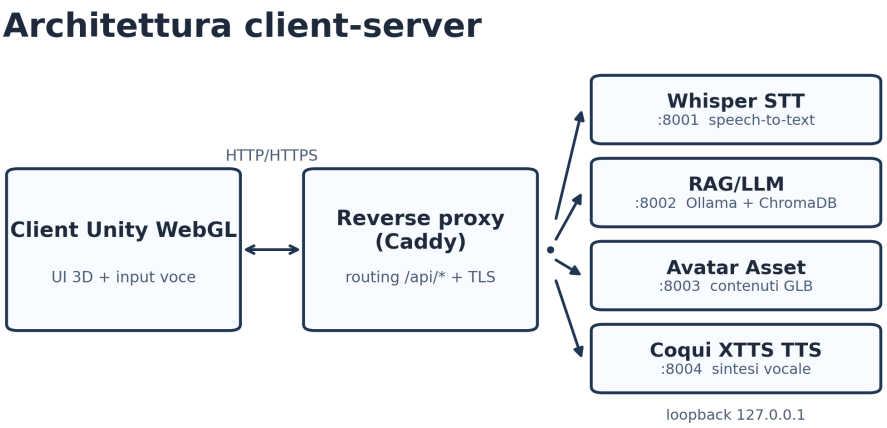

Fig. 2.4: Architettura logica client-server: il frontend Unity WebGL comunica con i servizi applicativi tramite reverse proxy.

2.3.1 *Frontend Unity WebGL e principali vincoli di piattaforma*

Il frontend è una build Unity WebGL eseguita nel browser. Questa scelta favorisce l'accessibilità senza installazione locale, ma introduce vincoli di piattaforma legati a WebAssembly e al main thread del browser.[3] In SOULFRAME, il client gestisce interfaccia e turni conversazionali, mentre STT, RAG/LLM e TTS sono delegati ai servizi backend.

La modalità push-to-talk riduce sovrapposizioni nei turni e semplifica la gestione dei permessi audio in ambiente web. Il client integra inoltre indicatori di stato e schermate di caricamento per rendere esplicite le fasi di inizializzazione.

2.3.2 *Backend FastAPI a micro-servizi*

Il backend è implementato in Python e organizzato in micro-servizi FastAPI, ciascuno dedicato a un sottocompito della pipeline. Questa separazione facilita isolamento delle dipendenze, gestione delle risorse e diagnosi dei guasti.

In sviluppo i servizi sono avviati con `uvicorn`; in esercizio possono essere gestiti in modo indipendente con log separati. FastAPI fornisce interfacce HTTP coerenti e documentazione OpenAPI utile per il collaudo degli endpoint.

[3] Doc ufficiale: Unity WebGL Technical Limitations, https://docs.unity3d.com/6000.2/Documentation/Manual/webgl-technical-overview.html.

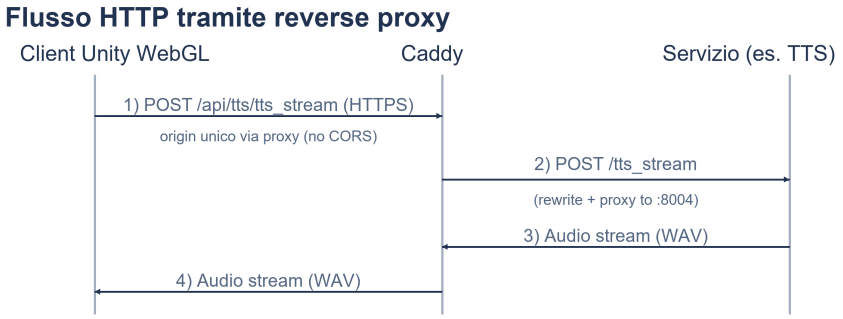

Fig. 2.5: Flusso HTTP semplificato tramite reverse proxy: dal client Unity WebGL al micro-servizio e ritorno dello stream audio.

2.3.3 *Comunicazione HTTP, CORS e proxy applicativo*

La comunicazione tra client e backend avviene tramite richieste HTTP con payload testuali e audio. In ambiente browser, la Same-Origin Policy impone vincoli di origine; quando frontend e servizi risiedono su host o porte diverse, è necessaria una configurazione esplicita del CORS tramite FastAPI.[4] Figura 2.5 mostra una richiesta tipica instradata dal reverse proxy verso un micro-servizio e il ritorno dello stream audio.

In SOULFRAME, il reverse proxy termina TLS e riscrive i path applicativi (`/api/*`) verso i servizi interni. Il client comunica quindi con un unico origin HTTPS, mentre le porte dei micro-servizi restano non esposte; questa configurazione riduce problemi di integrazione lato browser e migliora la stabilità della catena di chiamata.

[4] Doc ufficiale: FastAPI CORS, https://fastapi.tiangolo.com/tutorial/cors/.

3. ARCHITETTURA E TECNOLOGIE UTILIZZATE

3.1 Requisiti del sistema

3.1.1 Requisiti funzionali

Contenuto in preparazione.

3.1.2 Requisiti non funzionali

Contenuto in preparazione.

3.2 Architettura di riferimento di SOULFRAME

3.2.1 Vista d'insieme dei componenti frontend/backend

Contenuto in preparazione.

3.2.2 Flusso end-to-end audio → testo → risposta → audio

Contenuto in preparazione.

3.2.3 Flusso di gestione avatar (creazione, import, cache, rendering)

Contenuto in preparazione.

3.3 Componenti implementati

3.3.1 Integrazione Avaturn nel frontend Unity WebGL

Contenuto in preparazione.

3.3.2 Backend AI a micro-servizi (Whisper, RAG, TTS, Avatar Asset)

Contenuto in preparazione.

3.3.3 Persistenza e gestione dati per avatar

Contenuto in preparazione.

3.4 Setup e deploy operativo

3.4.1 Ambiente locale Windows

Contenuto in preparazione.

3.4.2 Ambiente server Ubuntu

Contenuto in preparazione.

3.4.3 Servizi di supporto (systemd, Caddy, script amministrativi)

Contenuto in preparazione.

4. SVILUPPO DEL PROGETTO: IMPLEMENTAZIONE E CRITICITÀ

4.1 Implementazione frontend

4.1.1 Gestione stati UI e navigazione

Contenuto in preparazione.

4.1.2 Gestione avatar e onboarding con Avaturn

Contenuto in preparazione.

4.1.3 Integrazione Avaturn WebView/SDK nel client Unity

Contenuto in preparazione.

4.1.4 Acquisizione audio e input desktop/touch

Contenuto in preparazione.

4.1.5 MainMode conversazionale

Contenuto in preparazione.

4.2 Implementazione backend

4.2.1 Servizio STT

Contenuto in preparazione.

4.2.2 Servizio RAG e memoria per avatar

Contenuto in preparazione.

4.2.3 Servizio TTS e streaming audio

Contenuto in preparazione.

4.2.4 Servizio asset avatar

Contenuto in preparazione.

4.3 Integrazione end-to-end

4.3.1 Orchestrazione richieste tra client, proxy e micro-servizi

Contenuto in preparazione.

4.3.2 Normalizzazione endpoint locale vs produzione

Contenuto in preparazione.

4.3.3 Gestione errori, retry e fallback

Contenuto in preparazione.

4.4 Criticità affrontate e soluzioni

4.4.1 Latenza e timeout

Contenuto in preparazione.

4.4.2 CORS e routing API

Contenuto in preparazione.

4.4.3 OCR e qualità dell'ingestione

Contenuto in preparazione.

4.4.4 Compatibilità dipendenze/modelli

Contenuto in preparazione.

4.4.5 Differenze operative tra Windows e Ubuntu

Contenuto in preparazione.

4.5 Runbook operativo essenziale

Contenuto in preparazione.

4.6 Affidabilità e sicurezza operativa

Contenuto in preparazione.

5. RISULTATI E VALUTAZIONE

5.1 Impostazione della valutazione

5.1.1 Scenari di prova e setup sperimentale

Contenuto in preparazione.

5.1.2 Metriche tecniche adottate

Contenuto in preparazione.

5.1.3 Metriche di esperienza utente

Contenuto in preparazione.

5.2 Risultati tecnici del prototipo

5.2.1 Prestazioni della pipeline STT-RAG-TTS

Contenuto in preparazione.

5.2.2 Latenza end-to-end e stabilità dei servizi

Contenuto in preparazione.

5.2.3 Osservazioni tra ambiente locale e server

Contenuto in preparazione.

5.3 Risultati qualitativi e casi d'uso

5.3.1 Qualità percepita dell'interazione

Contenuto in preparazione.

5.3.2 Usabilità interfaccia desktop e touch

Contenuto in preparazione.

5.3.3 Analisi di casi e failure cases

Contenuto in preparazione.

5.4 Valutazione utenti (estensione facoltativa)

5.4.1 Risultati SUS

Contenuto in preparazione.

5.4.2 Risultati NPS

Contenuto in preparazione.

5.4.3 Confronti tra gruppi

Contenuto in preparazione.

5.5 Discussione dei risultati

5.5.1 Punti di forza

Contenuto in preparazione.

5.5.2 Limiti emersi

Contenuto in preparazione.

5.5.3 Sintesi rispetto alle research questions

Contenuto in preparazione.

6. CONCLUSIONI E SVILUPPI FUTURI

6.1 Sintesi del lavoro svolto

Contenuto in preparazione.

6.2 Contributi principali

Contenuto in preparazione.

6.3 Limiti attuali del sistema

Contenuto in preparazione.

6.4 Sviluppi futuri prioritari

6.4.1 Miglioramenti tecnici del prototipo

Contenuto in preparazione.

6.4.2 Estensione della valutazione utenti

Contenuto in preparazione.

6.5 Considerazioni finali

Contenuto in preparazione.

RINGRAZIAMENTI

BIBLIOGRAFIA

[1] Mel Slater e Maria V. Sanchez-Vives. «Enhancing Our Lives with Immersive Virtual Reality». In: *Frontiers in Robotics and AI* 3 (2016), p. 74. DOI: 10.3389/frobt.2016.00074. URL: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2016.00074/full.

[2] Peter Kán, Martin Rumpelnik e Hannes Kaufmann. «Embodied Conversational Agents with Situation Awareness for Training in Virtual Reality». In: *ICAT-EGVE 2023 – International Conference on Artificial Reality and Telexistence and Eurographics Symposium on Virtual Environments*. The Eurographics Association, 2023, pp. 27–36. DOI: 10.2312/egve.20231310. URL: https://diglib.eg.org/bitstream/handle/10.2312/egve20231310/027-036.pdf.

[3] Fu-Chia Yang et al. «Embodied Conversational Agents in Extended Reality: A Systematic Review». In: *IEEE Access* 13 (2025), pp. 79805–79824. DOI: 10.1109/ACCESS.2025.3566698. URL: https://doi.org/10.1109/ACCESS.2025.3566698.

[4] Liliana Laranjo et al. «Conversational agents in healthcare: a systematic review». In: *Journal of the American Medical Informatics Association* 25.9 (2018). Free full text via PubMed Central (PMCID: PMC6118869), pp. 1248–1258. DOI: 10.1093/jamia/ocy072. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC6118869/.

[5] Catherine S. Oh, Jeremy N. Bailenson e Gregory F. Welch. «A Systematic Review of Social Presence: Definition, Antecedents, and Implications». In: *Frontiers in Robotics and AI* 5 (2018), p. 114. DOI: 10.3389/frobt.2018.00114. URL: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2018.00114/full.

[6] Michele Yin et al. «Let's Give a Voice to Conversational Agents in Virtual Reality». In: *Interspeech 2023*. 2023, pp. 5247–5248. URL: https://www.isca-archive.org/interspeech_2023/yin23b_interspeech.pdf.

[7] Alec Radford et al. «Robust Speech Recognition via Large-Scale Weak Supervision». In: *Proceedings of the 40th International Conference on Machine Learning (ICML)*. Vol. 202. Proceedings of Machine Learning Research. Also available on arXiv: https://arxiv.org/abs/2212.04356. PMLR, 2023, pp. 28492–28518. URL: https://proceedings.mlr.press/v202/radford23a.html.

[8] Patrick Lewis et al. «Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks». In: *Advances in Neural Information Processing Systems*. Vol. 33. Also available on arXiv: https://arxiv.org/abs/2005.11401. 2020, pp. 9459–9474. URL: https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

[9] Edresson Casanova et al. «XTTS: a Massively Multilingual Zero-Shot Text-to-Speech Model». In: *Interspeech 2024*. 2024, pp. 4978–4982. DOI: 10.21437/Interspeech.2024-2016. URL: https://www.isca-archive.org/interspeech_2024/casanova24_interspeech.pdf.