POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Informatica

Tesi di Laurea Magistrale

Sviluppo di una libreria per Embodied Conversational Agents

Relatore
Prof. Andrea Giuseppe Bottino

Correlatore:
Dott. Francesco Strada

Laureando
Davide Cervella
matricola: 252638

Anno accademico 2018-2019

Indice

Capitolo 1

Introduzione

1.1 Realtà Virtuale e Training

Il progetto di tesi "Sviluppo di una libreria per Embodied Conversational Agents" ha origine dalla volontà di realizzare una libreria che permetta di semplificare lo sviluppo di applicazioni di realtà virtuale con Embodied Conversational Agent. Questa necessità è stata alimentata da diversi fattori: forte crescita tecnologica legata alla materia, numerosi contesti applicativi in cui impiegare gli ECA, mancanza di un software sufficientemente semplice da usare ma allo stesso tempo versatile.
Il contesto applicativo su cui ci si è maggiormente soffermati all'interno dell'elaborato è quello del serious game poiché parte degli obiettivi riguarda la verifica dell'efficacia e della funzionalità di Embodied Conversational Agents in ambito formativo.

Il lavoro di ricerca si focalizza soltanto su alcuni degli aspetti che possono riguardare un ECA, in quanto estremamente complessi e numerosi. Parte dei concetti accennati nell'elaborato vengono inoltre approfonditi in altri progetti di tesi paralleli, che si occupano prevalentemente della gestione delle animazioni del volto e del corpo.
Si è rivolta particolare attenzione al campo del learning e quindi dell'uso di un sistema che possa sostituirsi o affiancarsi ai tradizionali strumenti di apprendimento, implementando un ambiente virtuale che possa rendere più proficua l'acquisizione dei concetti di interesse e di conseguenza più efficiente la loro messa in pratica.

Un sistema con queste caratteristiche non può essere del tutto indipendente dal contesto applicativo. Per questo motivo, si è deciso di implementare un'applicazione specifica sulla quale impostare concetti generali con la possibilità di testarne l'efficacia.
Tra i vari aspetti che possono essere esaminati nel campo dell'apprendimento e dell'istruzione, quello di maggiore interesse risulta il rapporto tra lo studente (l'utente)

e l'insegnante (l'ECA) che, per tale ragione, costituisce l'elemento preponderante dell'analisi. In particolare, si è ritenuto necessario procedere alla realizzazione di un Agente Intelligente che potesse interagire nel modo più naturale possibile con l'utilizzatore del software.

I vantaggi che possono scaturire dal sistema descritto sono notevoli. In primo luogo, si combinano le potenzialità offerte da un mondo virtuale e l'opportunità di interagire con un elemento di riferimento che ricordi la figura di un insegnante. In questo modo, è possibile ad esempio realizzare diversi scenari di apprendimento, riprodurre ambientazioni fittizie o passate e simulare situazioni o ambienti pericolosi senza rischio per l'Utente. Inoltre, il tutto è fruibile senza ulteriori costi e mantenendo costante il contatto umano con l'ECA all'interno del sistema.
Gli aspetti da considerare al fine di rendere l'Agente Conversazionale verosimile e affine al comportamento umano sono molteplici: ad esempio, l'intelligenza artificiale e la capacità di generare risposte coerenti con quanto chiesto dall'utente ma anche la formattazione di tali risposte. Pur trattandosi di aspetti molto diversi tra loro, entrambi risultano di fondamentale importanza per creare un'interazione con l'Utente che conferisca un valore aggiunto all'esperienza di training rispetto ad un normale gioco, nel quale il semplice feedback può essere ottenuto facilmente tramite meccanismi più superficiali (es. pop-up, voce pre-registrata, ecc. . .).
Infatti, ciò che contraddistingue un Agente Intelligente di questo tipo è la capacità di reagire a determinate situazioni come se fosse un essere umano calato nella figura di Trainer. Di conseguenza, le sue risposte sono relazionate con le azioni dell'Utente, a seconda di un buon lavoro o della presenza di errori, del perseverare negli sbagli o l'apprendere da questi ultimi. In quest'ottica assumono fondamentale importanza le emozioni, che rendono un uomo tale e si riversano essenzialmente nel modo di comunicare ed interagire dell'essere umano. Per questo motivo, la tesi approfondisce da una parte l'aspetto informatico e scientifico, riportando le tecnologie disponibili per lo sviluppo di sistemi di questo tipo, e dall'altra le ricerche psicologiche ed umanistiche legate alla definizione e alla rappresentazione delle emozioni umane. Infine, viene fornita una descrizione dettagliata dell'applicazione che si intende realizzare, accompagnata dai risultati ottenuti tramite l'esecuzione di test finali su una gamma di utenti.

1.2 Obiettivi

Il principale obiettivo della Tesi è quello di sviluppare un framework che permetta la realizzazione di applicazioni contenenti un Agente Intelligente e di specializzarne il comportamento in base allo specifico contesto applicativo. Per poter realizzare un sistema di questo tipo è necessario anzitutto uno studio dello stato dell'arte su diversi aspetti, come framework esistenti, librerie e tool per l'implementazione di alcuni aspetti specifici, ricerche relative alle caratteristiche principali di un

Embodied Conversational Agent. Una volta scelti gli strumenti necessari per l'implementazione, il passo successivo è astrarre alcuni concetti fondamentali e trovare una corrispondente rappresentazione digitale tale da poter essere utilizzata per controllare il comportamento dell'ECA in un qualsiasi contesto. Da qui la definizione di **azione** che indica *cosa* l'utente deve fare, di **stato** che tiene traccia di *come* lo si sta svolgendo e di **nodo** che determina *quando* l'azione va eseguita. Tali concetti verranno successivamente approfonditi nel capitolo dedicato all'implementazione.

Come delineato nel paragrafo precedente, parte del lavoro di Tesi è creare i presupposti per l'utilizzo di un Embodied Conversational Agent come mezzo di apprendimento e verificare che l'Agente possa essere effettivamente funzionale per un'applicazione di realtà virtuale incentrata su un percorso di formazione.
Non è scontato che si riesca a stabilire un rapporto tra Utente ed ECA tale da agevolare il processo di apprendimento. L'implementazione di tali meccanismi è particolarmente complessa ed eventuali anomalie nel comportamento dell'Agente, in quanto percepite come artefatti, potrebbero allontanare l'utilizzatore dal contesto applicativo in cui si trova. Questo diminuirebbe il senso di immersione nell'ambiente, ottenendo il risultato opposto a quello sperato. Infatti, ci troviamo in un'epoca in cui l'Utente medio è abituato a nuove tecnologie, le quali hanno sviluppato un forte senso critico nell'occhio umano; pertanto il soggetto è capace di percepire il senso di finzione e tende a non giustificarlo.
In particolare, in questo contesto le difficoltà di riproduzione sono accentuate, poiché i gesti, le espressioni, i comportamenti, i modi di dire e di esprimersi di altri esseri umani sono probabilmente gli elementi che l'occhio umano visualizza e riconosce con maggiore facilità, oltre ad essere quelli ai quali è più frequentemente esposto. Di conseguenza, chiunque può percepire anche una minima alterazione di queste caratteristiche.

Al fine di valutare il lavoro svolto si è deciso di realizzare un'applicazione di RV per apprendere i passi fondamentali al fine di eseguire correttamente una procedura di primo soccorso stradale. L'Utente, non possedendo competenze in materia, può acquisire le conoscenze previste attraverso un'applicazione di training nella quale viene fornito supporto da parte dell'Agente Conversazionale. L'elaborato si propone inoltre di offrire spiegazioni dettagliate sia per quanto riguarda le tecnologie essenziali per l'implementazione sia sulle funzionalità specifiche che l'applicazione deve offrire.

La realizzazione di un sistema che risponda ai requisiti descritti è un procedimento complesso che deve necessariamente tener conto di diversi aspetti, quali: Natural Language Understanding (NLU), Text To Speech (TTS), Speech To Text (STT), animazioni facciali e del corpo, lip sync, gestire ed implementare la logica applicativa, implementare dei meccanismi per rendere l'ECA quanto più simile ad un essere umano. Essendo questo lavoro parte di un progetto più ampio e sviluppato da

più persone, non tutti gli aspetti appena elencati verranno analizzati al suo interno.

Alcuni elementi mancanti e trattati in un progetto di tesi parallelo - quali animazioni facciali e del corpo, modellazione dei modelli e altri aspetti grafici - influenzano in gran parte la percezione dell'utente verso il sistema. Di conseguenza i test e l'analisi dei risultati tengono conto di queste peculiarità.

Capitolo 2

Stato dell'arte: Caratterizzare gli ECA

Prima di analizzare le tecnologie che ad oggi costituiscono lo stato dell'arte è opportuno approfondire il concetto di ECA, offrendone una spiegazione esaustiva. A tale proposito, in questo capitolo sono presentati gli elementi che caratterizzano maggiormente un Embodied Conversational Agent e, di conseguenza, quelli che dovrebbe possedere. Successivamente, si riportano cenni storici ed esempi di applicazione.

2.1 Rapporto tra Utente ed Agente Intelligente

Una delle caratteristiche che più contraddistinguono gli esseri umani da altre specie è la capacità di utilizzare il linguaggio per veicolare informazioni tra un individuo ed un altro. Tale capacità si manifesta non soltanto attraverso l'utilizzo della parola ma anche grazie ad una serie di altre abilità che rafforzano la comunicazione. Di particolare importanza sono gli elementi paralinguistici, come i movimenti del corpo, l'intonazione della voce, lo sguardo, le espressioni e tutto quello che trasmette un'informazione, intenzione o stato d'animo all'interlocutore.

Dal momento che è un metodo di comunicazione a cui l'uomo è abituato e che pertanto risulta essere molto naturale, nel corso del tempo si è cercato di implementare delle interfacce uomo-macchina basate su una metafora di tipo face-to-face con l'obiettivo di rendere l'interazione con la macchina simile a quella che si ha con un altro essere umano.

Data la natura complessa che contraddistingue il modo di comunicare degli esseri umani, per poter implementare un Embodied Conversational Agent in grado di mantenere una conversazione realistica è necessario tenere in considerazione interazioni sia di tipo vocale che di tipo non vocale in modo da attribuire il giusto peso tanto alla componente verbale che a quella relativa al movimento del corpo.

Tuttavia, queste non sono le uniche capacità che un Agente Intelligente deve dimostrare di avere; altri aspetti di grande importanza sono la capacità di comprendere le intenzioni dell'utente, avere consapevolezza del contesto virtuale in cui si è immersi e manifestare uno stato d'animo.
Questo comporta la necessità di ottenere input da parte dell'utente non soltanto tramite gli strumenti di interazione tradizionali ma anche attraverso sistemi che possano catturare e rappresentare in maniera adeguata azioni dell'utente che veicolano informazioni. Esempi di input di questo tipo possono essere la voce (rappresentabile tramite del testo), espressioni e movimenti (rappresentabili ad esempio tramite emozioni o stati d'animo).
Tuttavia, nella maggior parte dei casi la sola rappresentazione del parlato dell'utente in testo può essere ritenuta poco significativa. Pertanto si rende necessaria un'operazione di estrapolazione del concetto e dell'intenzione che l'utente manifesta attraverso la voce così come di ricerca di eventuali parole chiave. Questo si traduce tipicamente nell'utilizzo di Intelligenza Artificiale.

Nel corso del tempo sono stati condotti diversi studi per poter definire le caratteristiche chiave che un ECA dovrebbe possedere. In particolare si riportano le seguenti [10]:

- l'abilità di riconoscere e rispondere ad input sia verbali che non;
- l'abilità di generare output sia verbali che non;
- l'abilità di gestire alcuni aspetti tipici della conversazione come l'alternarsi nel parlare e dare messaggi di feedback;
- l'abilità di fornire dei segnali che facciano percepire l'andamento della conversazione;
- l'abilità di prendere iniziativa e iniziare un discorso;

L'integrazione di tutte queste componenti richiede uno studio interdisciplinare ed approfondito che porta alla creazione di team composti da persone con competenze diverse. Questo diventa necessario dal momento che il processo è particolarmente complesso e richiede operazioni come:

- modellazione 3D;
- animazioni facciali e del corpo;
- Intelligenza Artificiale;
- lip-sync;
- riconoscimento del parlato ed analisi dello stesso;
- generazione di audio a partire da un testo

- cattura dei movimenti ed espressioni dell'utente

Come sarà illustrato nei capitoli successivi, esistono diversi framework che permettono di gestire tutte queste problematiche tramite un unico software che integra diverse tecnologie. Tuttavia, solitamente queste soluzioni presentano poca versatilità a meno di personalizzarne o estenderne le funzionalità.
Dall'altro lato si è visto anche come esistano diverse soluzioni che forniscono supporto per singole (o alcune limitate) funzionalità necessarie per l'implementazione di un sistema che possa gestire in maniera opportuna un Agente Conversazionale. Tuttavia, in questo caso è richiesto uno sforzo notevole legato a diversi aspetti:

- a livello di programmazione, sia per la creazione di un sistema integrato sia per il design di un'architettura scalabile e riutilizzabile;

- a livello di ricerca, volta alla determinazione di come implementare le caratteristiche fondamentali che l'Agente deve avere accennate precedentemente;

Trattandosi di un lavoro particolarmente complesso, la Tesi non si prefigge come obiettivo la realizzazione di un sistema che possieda tutti questi requisiti ma che permetta una facile integrazione ed estensione delle funzionalità implementate ed approfondite nei capitoli successivi.

É stato dimostrato come l'interazione dell'utente con la macchina avviene già con una percezione del sistema come di una entità sociale che porta l'individuo a relazionarsi con esso seguendo delle tipiche regole comportamentali. Infatti, anche utenti che abitualmente utilizzano dispositivi come computer interagiscono conservando ad esempio determinate regole sociali e di buona educazione, riconoscendo differenze tra sistemi più o meno autoritari, mantenendo molti stereotipi di genere, percependo una differenza tra sistemi più esperti contro altri più generalisti. In generale, quindi, reagendo alla macchina come se questa fosse in parte un'entità umana.
Dato che l'utente si relaziona con il sistema seguendo queste modalità, alcuni critici hanno manifestato il proprio scetticismo riguardo all'effettiva necessità di implementare interfacce face-to-face e dunque renderle più simili ad un umano. Il fatto che in passato molti dei tentativi di questo tipo abbiano portato alla creazione di interazioni poco efficienti aumenta il senso di sfiducia verso la creazione di tali interfacce e la preoccupazione che esse non siano particolarmente utili o addirittura che possano portare l'utente ad uno stato di confusione maggiore.
Tuttavia, come già descritto all'inizio del capitolo, una componente chiave che contraddistingue il modo di comunicare degli esseri umani è il linguaggio del corpo insieme a tutti gli elementi della comunicazione esposti. Per cui, fino a quando non si riuscirà ad integrare in modo solido tali elementi all'interno di un qualsiasi sistema non sarà effettivamente possibile verificare la validità di un Embodied Conversational Agent [10].

2.2 Embodied Conversational Agents nella storia

Un Embodied Conversational Agent può essere descritto come un sistema in grado di sostenere una conversazione naturale in funzione non solo del dialogo tenuto dall'utente ma anche delle emozioni e della personalità. Nella figura 2.1 viene mostrato come nel corso della storia si sia verificato un passaggio da soluzioni basate sulla sola analisi sintattica e semantica delle frasi a tecniche che, tramite approcci differenti, hanno portato allo sviluppo di Embodied Conversational Agent.

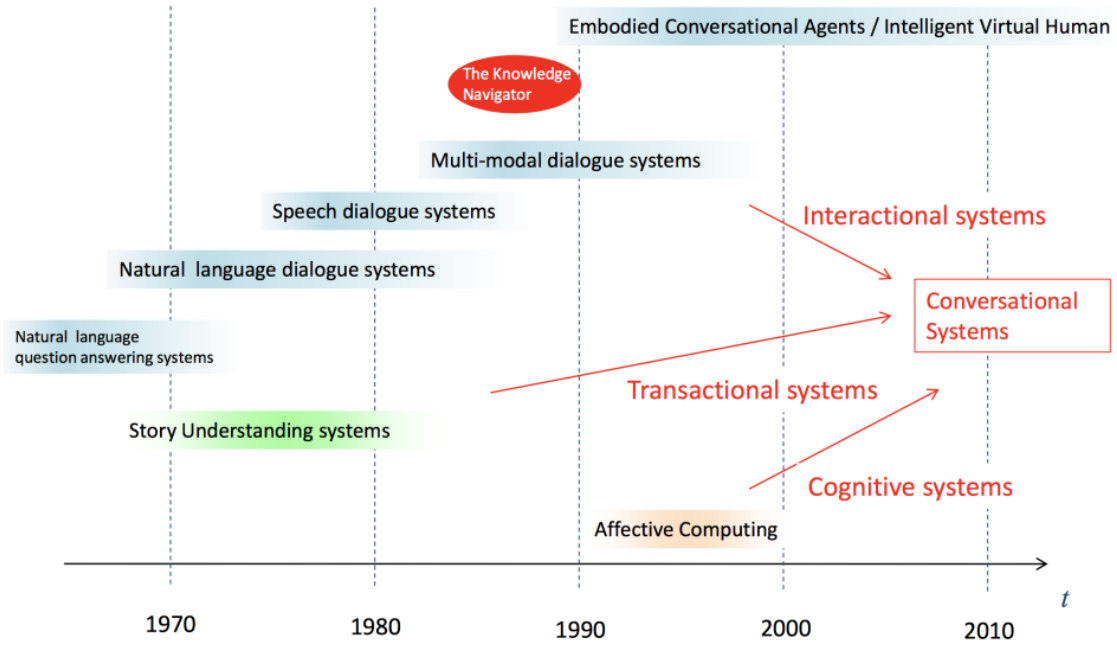


Figura 2.1: Storia dei Sistemi Conversazionali [19]

Tra le prime applicazioni che sono state sviluppate ricordiamo:

LUNAR

Un software sviluppato al fine di consentire un'interazione di tipo domanda-risposta e utilizzato dai geologi per semplificare l'accesso ai dati necessari per l'analisi delle rocce lunari. Per implementare tale meccanismo è necessaria anzitutto un'analisi sintattica per poter estrapolare il valore semantico della richiesta. Tale rappresentazione viene convertita, in modo più formale, in una serie di queries per l'ottenimento delle informazioni che successivamente vengono riformulate per generare la risposta.

SHRDLU

È uno dei primi sistemi di Natural Language Understanding sviluppato da Terry Winograd presso l'M.I.T. Artificial Intelligence Laboratory nel 1968-70 [2] ed è incentrato sull'utilizzo di quelli che vengono chiamati "Blocks". Tramite il programma, l'Utente può porre delle domande, eseguire dei comandi o fornire delle informazioni alla macchina tramite il comune linguaggio orale (in lingua inglese). Alla base dell'implementazione c'è un concetto fondamentale, ovvero la necessaria conoscenza del contesto e della materia in esame volta alla comprensione del linguaggio. L'utilizzo di informazioni sia di contesto sia di tipo semantico permettono di conseguenza la comprensione della frase e la capacità di interpretare correttamente eventuali ambiguità all'interno della stessa. L'interazione tra uomo e macchina avviene pertanto secondo un processo di questo tipo:

- inserimento di un comando testuale;
- analisi sintattica;
- analisi semantica;
- risposta del calcolatore;
- attuazione dell'azione corrispondente (Planning);
- aggiornamento della grafica;

Per meglio comprendere il funzionamento, si riporta di seguito un breve estratto del dialogo relativo ad una nota dimostrazione del software, durante il quale è possibile osservare anche la gestione di eventuali ambiguità grazie alla conoscenza della materia:

"*Computer*: OK. (does it)
Person: GRASP THE PYRAMID.
Computer: I DON'T UNDERSTAND WHICH PYRAMID YOU MEAN.
Person: FIND A BLOCK WHICH IS TALLER THAN THE ONE YOU ARE HOLDING AND PUT IT INTO THE BOX.
Computer: BY "IT", I ASSUME YOU MEAN THE BLOCK WHICH IS TALLER THAN THE ONE I AM HOLDING.
Computer: OK. (does it)
Person: WHAT DOES THE BOX CONTAIN?
Computer: THE BLUE PYRAMID AND THE BLUE BLOCK.
Person: WHAT IS THE PYRAMID SUPPORTED BY?
Computer: THE BOX.
Person: HOW MANY BLOCKS ARE NOT IN THE BOX?
Computer: FOUR OF THEM."

Ci sono molteplici esempi di applicazioni più o meno analoghe: uno di questi è ELIZA [25], sviluppato da Joseph Weizenbaum nel 1966. Si tratta di un sistema di dialogo che mediante regole di trasformazione e parole chiave ricombina le frasi dell'utente, riproponendole sotto forma di domanda in modo naturale e spesso convincente.
Con il passare del tempo acquisisce sempre più importanza il concetto di credibilità e l'esigenza – già avvertita in altri contesti come al cinema, al teatro, nei video game, ecc. – di condurre l'utente ad abbandonare ogni forma di scetticismo al fine di immergersi nella conversazione come se stesse interagendo con un essere umano. Per questo motivo si inizia a tenere conto di ulteriori aspetti che concorrono a rendere un sistema di dialogo credibile e diventano fondamentali elementi come:

- la personalità, poiché il soggetto deve essere unico ed avere dei modi peculiari di svolgere delle azioni;
- le emozioni, che caratterizzano l'agente e che quindi vengono esternate e gestite in modo personale;
- le intenzioni, ossia le volontà e gli obiettivi propri del soggetto;
- i cambiamenti, i quali rendono il personaggio dinamico e variabile nel tempo pur mantenendo una coerenza con le sue caratteristiche;

Questi ed altri aspetti hanno portato gli Agenti Conversazionali ad un livello sempre più alto ed uno dei primi esempi è REA.

REA

REA (Real Estate Agent) [11] è un Embodied Conversational Agent che, tramite funzionalità sia sul piano linguistico che non, implementa un Agente Conversazionale innovativo rispetto ai suoi predecessori. In particolare, da questo punto di vista:

- l'Agente è rappresentato tramite un corpo umano che viene utilizzato per emulare tratti tipici di una normale conversazione. Vengono gestiti lo sguardo, la postura e le animazioni facciali al fine di rendere l'interazione credibile;
- sia gli aspetti riguardanti caratteristiche verbali sia quelli relativi ad elementi non verbali hanno uguale importanza e nessuno dei due prevale sull'altro;
- mentre nelle implementazioni precedenti a REA si tendeva a concentrarsi prevalentemente sugli input generati dall'Utente, da questo punto in poi si attribuisce lo stesso peso sia agli input che agli output. In particolare, non ci si limita a rispondere a segnali visivi o sonori ma questi vengono generati anche da parte dell'ECAstesso.

Il contesto applicativo specifico di REA riguarda la vendita di immobili disponibili a Boston e permette di ottenere informazioni al riguardo tramite l'interazione con un Agente Conversazionale. Il sistema permette così di chiedere informazioni all'Agente che risponde coerentemente a quanto chiesto e, allo stesso tempo, implementa i meccanismi descritti, i quali rendono l'interazione più credibile e simile a quella con un essere umano.
Si possono individuare alcuni aspetti specifici che l'applicazione mette in evidenza:

- vengono gestiti differenti tipi di animazioni come ad esempio gesti che invitano ad avviare una conversazione, movimenti per fornire feedback, enfatizzare dei contenuti o distogliere lo sguardo.

- diversi comportamenti, sia vocali che non, possono veicolare lo stesso messaggio e per questo motivo devono essere riprodotti simultaneamente. Ad esempio, annuire mentre si risponde all'utente manifestando un senso di accordo oppure guardare o indicare un punto nel momento in cui si spiega quale sia il prossimo passo da fare. Questi aspetti vanno però sincronizzati con il contenuto della frase enunciata: infatti, essi possono assumere significati diversi a seconda del contesto e del significato di quello che viene espresso.

- si mette in evidenza come anche dei semplici gesti possono veicolare dell'informazione. Un esempio potrebbe essere il porre le mani in posizione rilassata, ad esempio lungo i fianchi, per indicare la volontà di rimanere in silenzio per cedere il turno all'utente.

- è essenziale tenere in considerazione il fatto che il ripetersi delle stesse azioni costantemente può generare situazioni poco realistiche e che possono essere percepite come degli artefatti. Ad esempio, se l'utente pronuncia una frase della durata di 30 secondi e l'Agente annuisce in continuazione per indicare il proposito di ascolto, l'atteggiamento può essere percepito come artificiale.

- gli ideatori del sistema mettono in luce la necessità di avere una sincronizzazione tra utente ed Agente in modo che questi non si sovrappongano eseguendo delle azioni che nella realtà possono essere eseguite soltanto uno per volta. Durante una conversazione reale questo meccanismo di sincronizzazione avviene costantemente e in particolar modo attraverso dei semplici gesti, che esprimono una chiara intenzione da parte dell'individuo. Attualmente REA non implementa dei meccanismi di sincronizzazione basati su comportamenti non verbali [10].

In 2.2 viene illustrata l'architettura di REA.

Come si può evincere dalla figura, il sistema prevede una grande varietà di input attraverso i quali è possibile elaborare diverse informazioni utili che permettono di rappresentare lo stato e le intenzioni dell'utente. Si ha dunque la possibilità di:

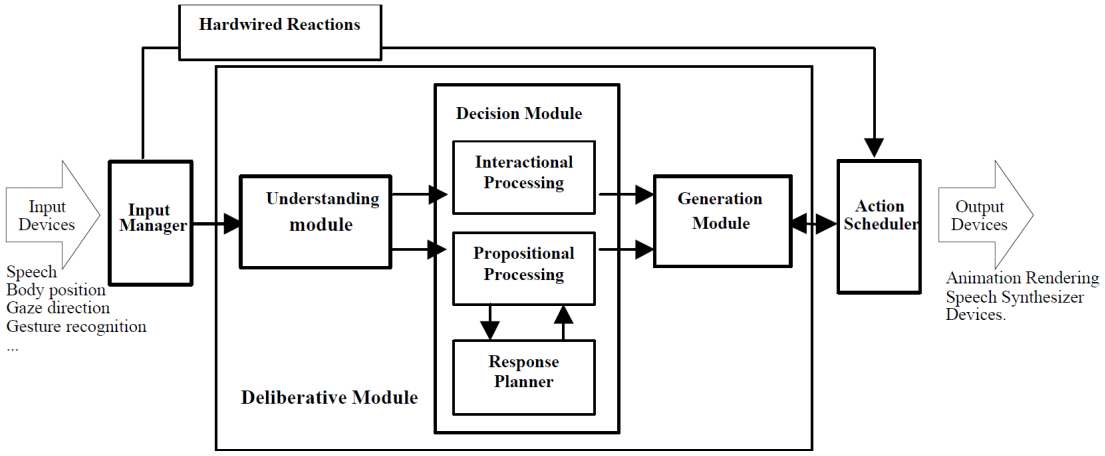


Figura 2.2: REA Architecture [10]

- elaborare la voce dell'interlocutore per studiarne il contenuto;
- catturare la posizione dell'utente, ad esempio per dirigere lo sguardo dell'ECA verso il soggetto quando necessario;
- rilevare la direzione verso cui l'utilizzatore sta guardando per sapere su cosa è focalizzato attualmente;

L'Input Manager colleziona tali informazioni distinguedo il caso in cui queste vadano utilizzate immediatamente o se richiedono successive elaborazioni.
L'ultimo blocco (Action Scheduler) ha un duplice scopo:

- schedulare le varie azioni che possono essere compiute dall'Agente Conversazionale e che riguardano dunque l'animazione ed operazioni di sintesi vocale;
- evitare che si generino collisioni tra i vari blocchi e che dunque vengano richieste simultaneamente più azioni;

Per meglio comprendere il funzionamento di REA, di seguito si riporta un esempio di interazione tra un utente e l'Agente Conversazionale estrapolato dal documento [10]:

" *Mike approaches the projection screen. Rea is currently turned side on and is gazing idly about. As Mike moves within range of the two cameras mounted above the screen, Rea turns to face him and says:*
REA: Hello. How can I help you?
MIKE: I'm looking to buy a place near MIT.
Rea nods, indicating that she is following.
REA: I have a house to show you.

A picture of a house appears on-screen behind Rea who blinks and looks at the house and then at Mike.
REA: It's in Somerville.
MIKE: Tell me about it.
Rea looks up and away while she plans what to say
REA: It's big.
Rea makes an expansive gesture with her hands.
Mike brings his hands up as if he is about to speak and so Rea does not continue; instead waiting for him to speak.
MIKE: Tell me more about it
REA: Sure thing. It has a nice garden.
Rea sketches a curved gesture with her hands indicating that the garden extends along two sides of the house
MIKE: How far is it?
REA: It is five minutes to the Porter Square T station.
Rea makes it clear that it is five minutes on foot from the T station by making a walking gesture with her fingers
MIKE: How big is the house?
REA: It has four bedrooms, three bathrooms. . .
Mike interrupts Rea who stops speaking immediately.
MIKE: Wait. Tell me, where is the master bedroom?
REA:I'm sorry, I didn't catch that. What did you ask me?
MIKE: Where is the master bedroom?
REA: It's upstairs.
Rea points up
MIKE: Where is the master bathroom?
REA: It's next to the bedroom.
Rea brings her hands together to indicate the relationship between the bedroom and the bathroom."

2.3 Emozioni

Come già indicato nei capitoli precedenti, una delle caratteristiche chiave di un Embodied Conversational Agent è la capacità di generare output non verbali. Di particolare rilevanza sono quindi le emozioni che per loro natura tipicamente si manifestano attraverso espressioni facciali, movimenti o azioni piuttosto che tramite parole. In questo capitolo si affronta il problema di come rappresentare le emozioni e come queste possono essere influenzate da eventi di varia natura.

Si tratta di un argomento complesso sul quale sono stati condotti diversi studi di natura interdisciplinare che hanno portato alla nascita di molti modelli computazionali più o meno diversi tra di loro. Tuttavia tali modelli spesso non sono

perfettamente comprensibili sia a causa della complessità intrinseca del problema sia per scelte di design che derivano da assunzioni fatte in base al contesto specifico per cui sono nate. Inoltre, il più delle volte il modello è stratificato in un insieme di blocchi non ampiamente documentati [17].
La mancanza di un linguaggio comune porta ad una problematica di grande rilievo, ovvero l'incompatibilità tra i vari modelli e di conseguenza la difficoltà di rendere un sistema estensibile.
Nella figura 2.3 vengono mostrati alcuni tra i più conosciuti modelli computazionali legati alle emozioni ognuno dei quali è incompatibile con l'altro sia per il diverso funzionamento sia per il diverso formato di input ed output dei dati [17]. Molte

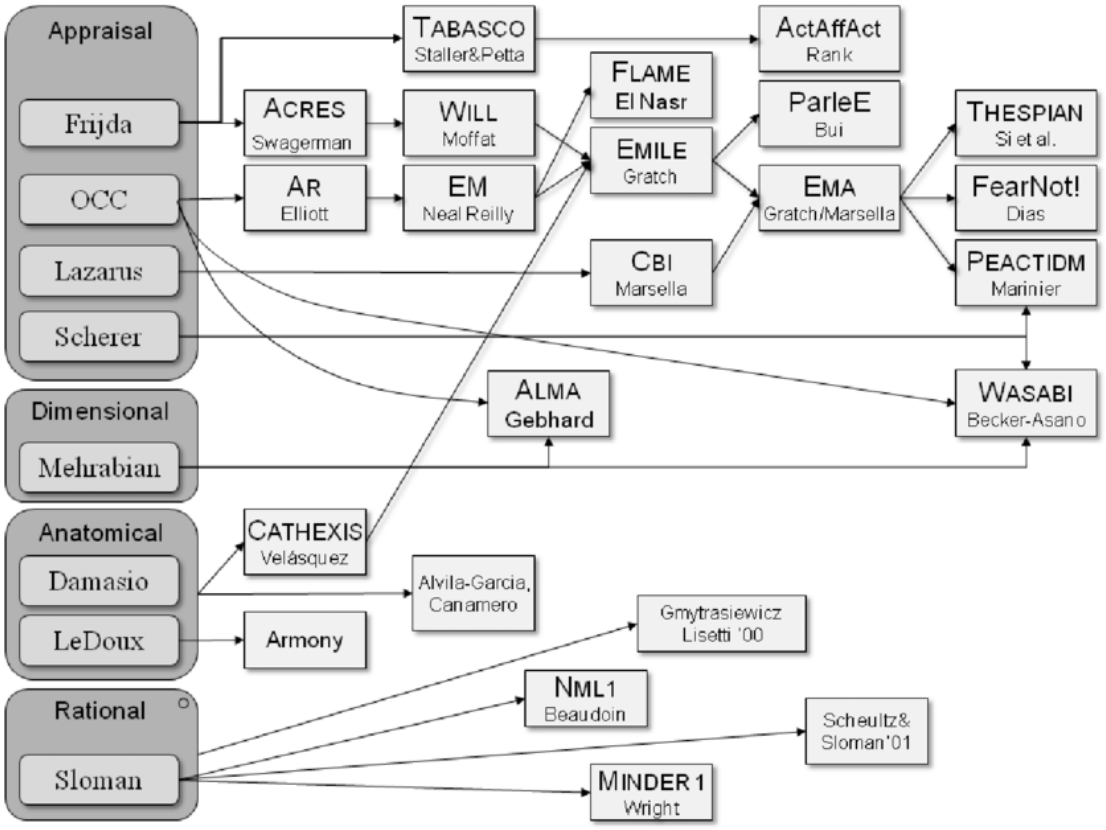


Figura 2.3: Storia dei modelli computazionali relativi alle emozioni [17]

delle componenti dei precedenti sistemi sono frutto di varie teorie consolidate che solitamente si differenziano in base a quali siano ritenute le parti constitutive di un'emozione e sul modo in cui queste sono correlate tra loro.
In generale non è necessario rifarsi ad un'unica teoria nel momento in cui si vuole sviluppare un proprio modello.

Appraisal theory

È una delle teorie più utilizzate per la creazione di modelli computazionali. L'elemento che più caratterizza questo approccio è la presenza di una relazione tra cognizione ed emozione, caratteristica che lo rende particolarmente adatto alla realizzazione di sistemi che fanno uso di Intelligenza Artificiale. Secondo tale teoria le emozioni vengono influenzate da due fattori:

- eventi che accadono;
- credenze, intenzioni e desideri dell'individuo;

Questi due fattori costituiscono quello che viene definito una relazione di tipo persona-ambiente [16]. Una specifica emozione viene generata a seguito di un'analisi e confronto tra quello che sta accadendo e le caratteristiche dell'individuo. Tale teoria si presta ad essere modellata attraverso la cattura di situazioni o eventi che vengono utilizzati per generare specifici comportamenti o reazioni.
In questo caso quindi svolge un ruolo centrale la fase di valutazione di tali variabili e di quali emozioni, espressioni facciali o frasi debbano generare.
Solitamente, tali concetti vengono definiti attraverso l'utilizzo di:

- un set di variabili di valutazione;
- un set di emozioni discreto;
- logiche del tipo if/then per collegare le variabili a specifiche emozioni;

Dimensional Theory

Questa teoria si discosta dalla precedente sia in relazione a quali siano le cause che modificano lo stato emozionale di un individuo sia per il modello utilizzato per la rappresentazione delle emozioni.
Secondo gli studiosi appartenenti a tale scuola di pensiero, tutti gli stati emozionali vengono generati da un unico sistema neurologico e non - come nella teoria precedente - da diversi sistemi neurali che gestiscono emozioni differenti. Questa considerazione ha fatto sì che si abbandonasse l'approccio discreto per prediligere una rappresentazione delle variabili di valutazione attraverso uno spazio continuo e multidimensionale [17], [22], [18], [7], [24]. Dal momento che si perde la relazione tra specifici elementi neurologici e determinate emozioni, il concetto stesso di emozione viene messo da parte per prediligere nuovi elementi quali: mood, affect e core affect [22].

Plutchik's wheel

Di seguito si riporta il modello di rappresentazione delle emozioni denominato Plutchik's whell che è stato scelto per l'implementazione del framework oggetto della

tesi.
Come si può osservare dalla figura 2.4, le emozioni vengo raffigurate attraverso un cono costituito da otto emozioni principali. Ogni emozione ha una sua intensità che viene rappresentata sia dalla saturazione del colore che dalla distanza dal centro della figura.
Più nel dettaglio tale modello è costituito da:

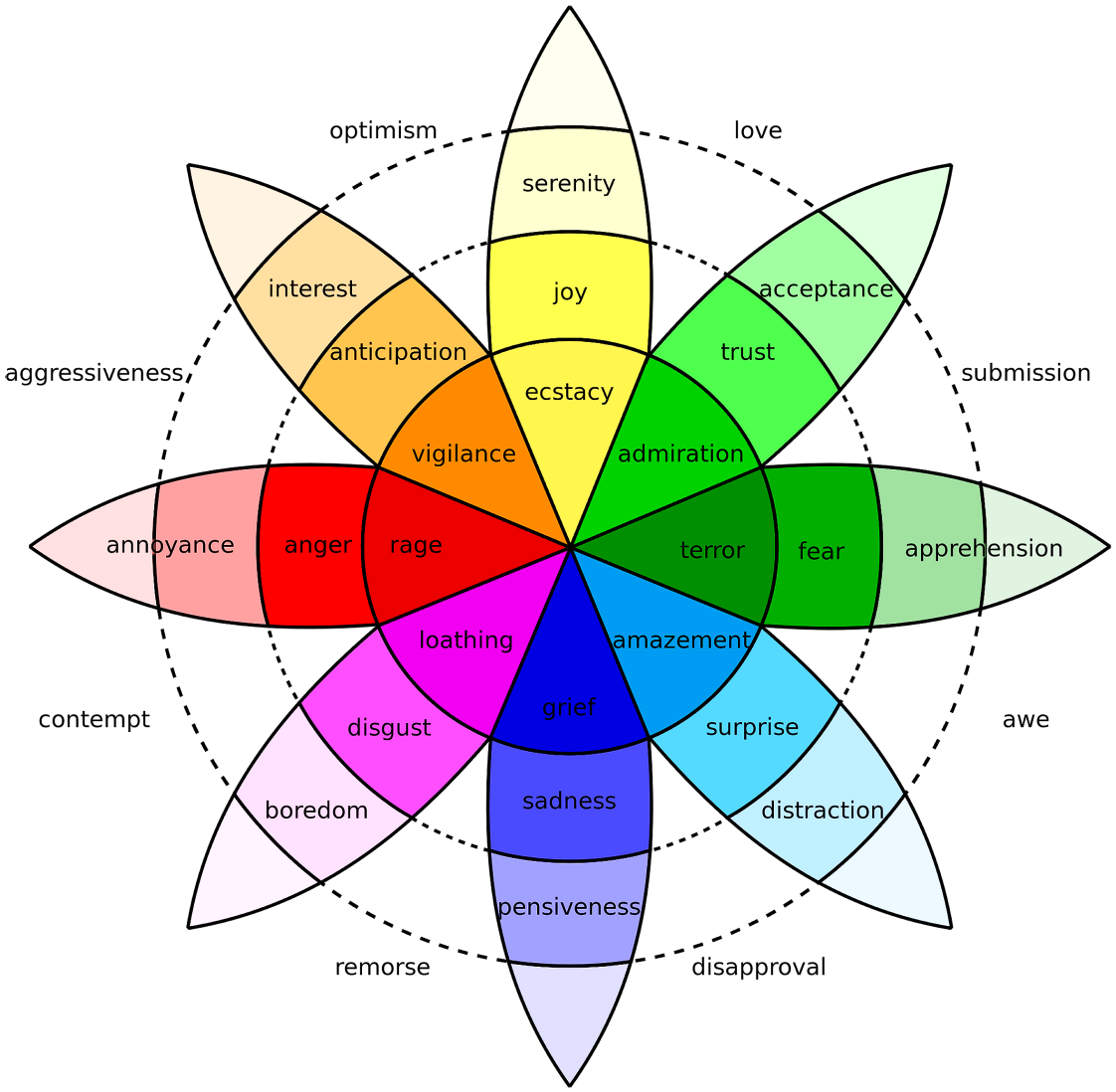


Figura 2.4: Plutchik's wheel

- emozioni primarie: Joy, Trust, Fear, Surprise, Sadness, Disgust, Anger, Anticipation;

- emozioni opposte: ogni emozione primaria ha un'emozione opposta situata nella direzione inversa lungo il raggio del cono. Per cui, ad esempio, Joy sarà l'inverso di Sadness, Anticipation sarà l'inverso di Surpris, ecc;
- emozioni combinate: la combinazione di due emozioni primarie può portare alla generazione di altre emozioni secondarie che vengono rappresentate senza colori nella figura;

Capitolo 3

Stato dell'arte: Frameworks

Nell'elaborato viene fatta una differenziazione tra soluzioni che permettono di gestire tutti gli aspetti fondamentali per l'utilizzo di un ECA in un unico software e soluzioni che invece consentono di implementare singole funzionalità (quali NLU, TTS, STT, ecc.).

Nel primo caso si riscontra solitamente una scarsa versatilità mentre nel secondo è evidente una maggiore elasticità e modularità, pur implicando uno sforzo aggiuntivo a livello di implementazione. In questo capitolo in particolare si fornisce una descrizione dei framework disponibili e più utilizzati per la creazione di tali agenti.

3.1 Virtual Human Toolkit

3.1.1 Architettura VHT

Virtual Human Toolkit(VHT)[13] ha come obiettivo quello di fornire un framework singolo per la creazione di un Embodied Conversational Agent capace di intrattenere delle relazioni sociali realistiche con l'Utente. Si tratta di un toolkit sviluppato presso la University of Southern California (USC) Institute for Creative Technologies (ICT) con altri collaboratori e destinato principalmente a ricercatori che desiderano creare degli ECA o ampliare tale progetto.

Le aree di ricerca sono numerose e tutte fondamentali per la realizzazione di un sistema integrato di questo tipo. In figura 3.1 è possibile osservare i vari blocchi (ognuno dei quali fa riferimento ad una specifica problematica e area di ricerca) e come questi vengano impiegati ad alto livello per analizzare l'input dell'utente e successivamente generare una reazione.
Si tratta di una schematizzazione di alto livello che può essere specializzata in modi differenti e non necessariamente in maniera completa. Si analizzano di seguito alcune componenti fondamentali che consentono la cattura degli input dell'utente, l'analisi e in conclusione la generazione di una reazione da parte del character.

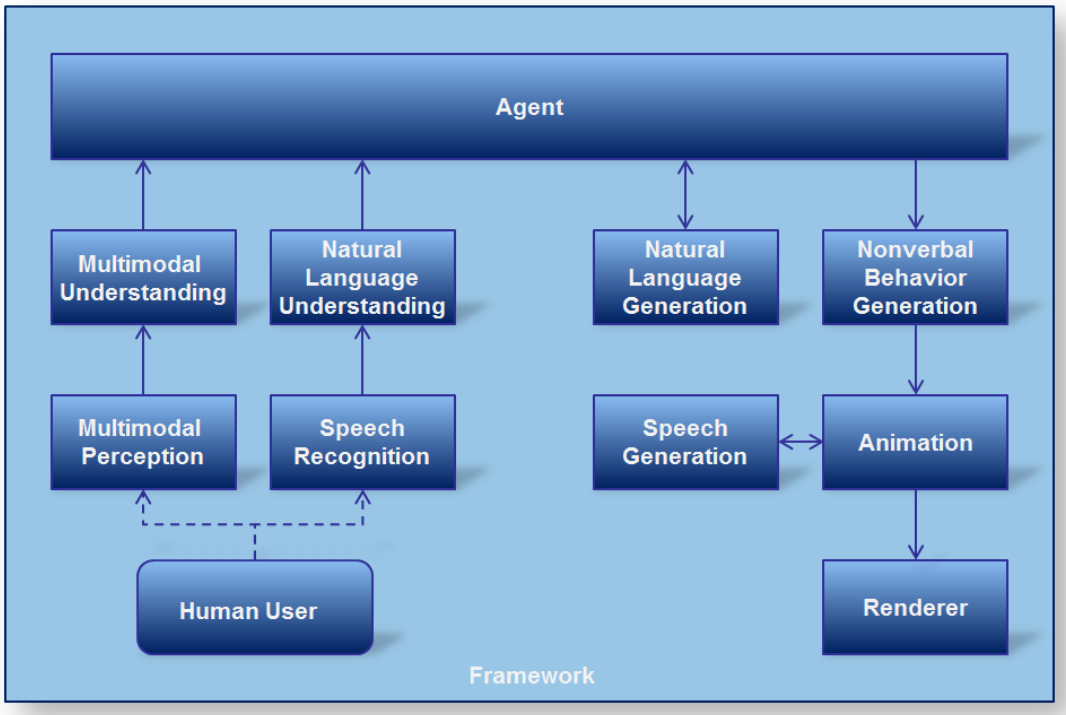


Figura 3.1: Architettura di VHT

- **Speech Recognition**: una funzionalità che permette di interagire con il sistema tramite la voce, convertendo quest'ultima in un formato testuale;
- **Natural Language Understanding**: il testo generato nel passo precedente viene analizzato tramite questo blocco che effettua un'analisi tale da generare una rappresentazione semantica;
- **Audio-Visual Sensing**: insieme di sensori che hanno il compito di individuare delle caratteristiche o delle espressioni tipiche della comunicazione non verbale;
- **Multimodal Understanding**: così come si effettua un'analisi del testo generato dalla voce dell'utente si esaminano gli input non verbali;
- **Nonverbal Behavior Generation/Natural Language Generation**: le analisi degli input precedentemente riportati sono elaborate dall'agente fino a definire un intento che viene comunicato verso l'esterno tramite sistemi sia di tipo verbale che non. Nel caso di comunicazioni verbali si ricorre tipicamente ad un audio pre-registrato oppure a sistemi di **Text To Speech**. Le comunicazioni non verbali sono invece implementate tramite delle **animazioni**.

3.1.2 Moduli VHT

Per l'implementazione delle macro aree appena delineate, Virtual Human Toolkit mette a disposizione una serie di moduli che vengono illustrati nella figura 3.2 e brevemente descritti nel seguito.

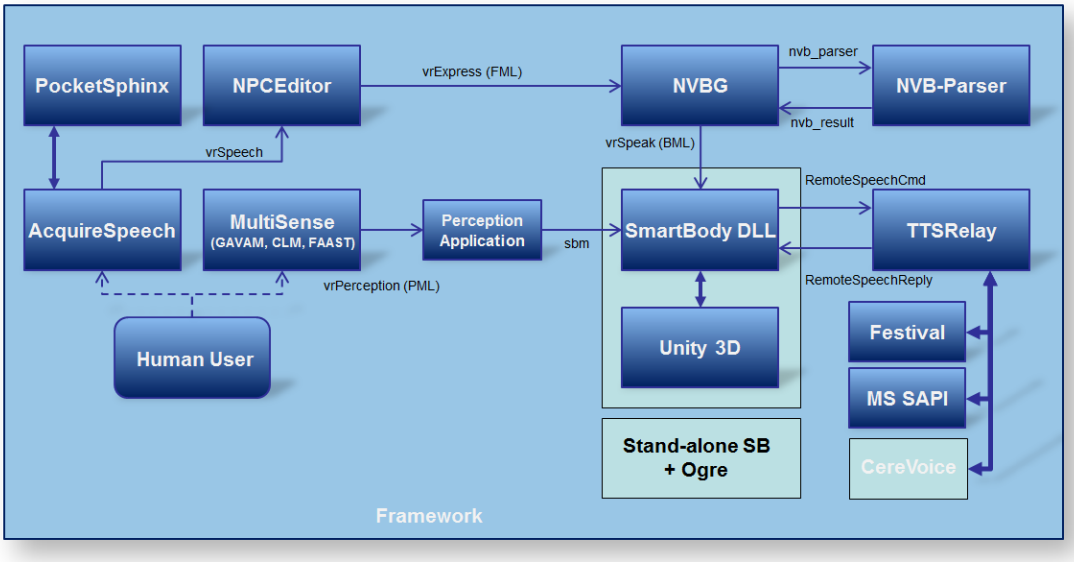


Figura 3.2: Moduli dell'architettura di VHT

Il Toolkit è stato sviluppato per diverse piattaforme ma quella principale è Windows mentre per MacOS e Linux è previsto un supporto limitato. I principali componenti possono essere così riassunti:

- **MultiSense**: fa riferimento a due aspetti chiave mostrati precedentemente nella figura 3.1, ovvero l'acquisizione delle espressioni (Audio-Visual Sensing) e l'elaborazione di tali informazioni per la generazione di un intento (Nonverbal Behavior Understanding). Si tratta pertanto di un meccanismo di più sensori che parallelamente - mediante l'utilizzo di un'architettura multithreading - catturano diverse informazioni. Queste ultime, fuse tra loro, costituiscono una base per la rilevazione di aspetti comportamentali come: l'attenzione, l'agitazione, la tranquillità, ecc. La rappresentazione del risultato di questo task è data da un particolare linguaggio chiamato PML (Perception Markup Language).

- **NPCEditor**: modulo che si occupa di implementare le funzionalità relative al Natural Language Understanding e di conseguenza la gestione del dialogo e delle decisioni dell'Agente. Per consentire il funzionamento dell'algoritmo, è necessaria una fase di configurazione in cui si vanno a specificare una sequenza

di frasi di esempio ed un set di risposte predefinite ad esse associate. Dopo tale fase preliminare si procede con un'analisi statistica per la classificazione del testo e la selezione della risposta più appropriata - tra quelle disponibili - in funzione della frase pronunciata dall'Utente.

- **NVBG** e **SmartBody**: due elementi chiave per implementare la componente non verbale della risposta. Il comportamento non verbale risultante dall'analisi degli input viene codificato tramite un linguaggio denominato BML (Behavior Markup Language). Lo SmartBody è uno degli elementi fondamentali per dare forma visiva all'output e va a generare una sequenza di animazioni guidate dal BML precedentemente prodotto. Questo consente ad esempio di associare a parole particolari dei movimenti o gesti in modo da dare loro maggiore enfasi, di animare la bocca per il parlato, di gestire lo sguardo e molto altro.

3.1.3 Diversi tipi di characters

Dal momento che il Virtual Human Toolkit è sia configurabile che ampliabile e dato che i ricercatori hanno la possibilità di miscelare i vari componenti, è possibile creare delle configurazioni tali da definire una molteplicità di uomini virtuali con caratteristiche differenti. Di seguito se ne descrivono alcune tipologie.

Question-Answering: si tratta di Agenti Conversazionali il cui scopo principale è sostenere una comunicazione con l'Utente analoga ad un'intervista. In questo contesto l'Utente può essere considerato come intervistatore mentre l'ECA come intervistato. Si tratta di un vero e proprio botta e risposta guidato dall'utilizzatore e in cui ad una singola domanda corrisponderà un'unica risposta. Tale sistema è tipicamente impiegato in domini applicativi in cui l'utente vuole accedere ad informazioni riguardanti il dominio stesso.

Virtual Listeners: perfetti per situazioni in cui si vuole riprodurre/simulare un atteggiamento di ascolto da parte dell'agente. Permette infatti di fornire un feedback verso l'interlocutore tramite meccanismi sia verbali che non e scaturiti a partire da input quali i movimenti della testa dell'Utente, il ritmo e l'intonazione del parlato, ecc. Le reazioni dell'ECA saranno tipicamente non verbali e generate a partire da un mapping tra determinati insiemi di input e dei comportamenti predefiniti da generare. Un esempio di animazione di feedback potrebbe essere il movimento della testa per annuire. L'applicazione tipica di tale sistema è quella in cui si vuole creare un rapporto tra un interlocutore ed un pubblico e può essere pertanto utilizzato come allenamento per acquisire naturalezza nel parlare con un pubblico.

Virtual Interviewers: Questa è la prima tipologia - tra quelle elencate - in cui è l'ECA a portare avanti la conversazione. Lo scopo è molto diverso dai precedenti e mira a raccogliere informazioni dall'Utente al fine ad esempio di esaminarlo o

per effettuare delle valutazioni. Per l'implementazione di tale funzionalità è necessario un ottimo meccanismo di Natural Language Understanding e un dominio pre-determinato.

3.1.4 Limiti

Nonostante i diversi pregi discussi precedentemente - in particolar modo il fatto di essere probabilmente il più importante strumento per la creazione di Embodied Conversational Agent che abbia un così elevato numero di features al suo interno -, il software non è privo di difetti. In particolare, VHT è privo di alcuni elementi che sono stati presi in considerazione per il progetto di Tesi come la modellazione delle emozioni e la memoria persistente. Inoltre, come già accennato in precedenza, non tutte le piattaforme sono al momento supportate (es. Android ed iOS). Si è pertanto deciso di optare per altre soluzioni in modo da rendere più agevole lo sviluppo di un'applicazione che possa prevedere anche eventuali sviluppi futuri.

3.2 Virtual Agent Interaction Framework

Virtual Agent Interaction Framework(VAIF)[23] è un framework pensato per sviluppatori o ricercatori inesperti che desiderino realizzare sistemi in cui includere degli Agenti Conversazionali in modo semplificato e centralizzato, mantenendo l'accesso a molteplici funzionalità. Per ottenere tale risultato viene messa a disposizione un'interfaccia apposita accessibile tramite uno dei più famosi game engine: Unity.

Le funzionalità integrate in questo tool sono numerose e permettono di implementare dei meccanismi quali mettere l'ECA in ascolto, farlo parlare, muovere e gesticolare. Inoltre, sono integrati servizi per la gestione della memoria dell'agente - dando la possibilità di regolare le reazioni in base a ciò che è accaduto nel passato - e per la traduzione del parlato dell'Utenteda un formato audio a testo (operazioni di speech recognition).
Gli sviluppatori del framework pongono il software a metà strada tra soluzioni ampiamente più complesse (come VHT) ed altre più semplici (come UTEP), che però possono presentare problemi di affidabiltà.

3.2.1 Architettura VAIF

Alla base del funzionamento c'è il concetto di **timeline** che consente di definire una serie di eventi e di possibili interazioni. Ciascuna timeline può contenere più character, ognuno dei quali può appartenere a diverse timeline. Per poter implementare tale meccanismo all'interno della propria applicazione sono necessari alcuni passaggi che vengono di seguito brevemente descritti.

1. **Scegliere uno o più agenti**: gli agenti possono essere scelti tra quelli direttamente disponibili all'interno del framework oppure inseriti nell'applicazione dopo averli creati tramite dei tool esterni. Nel caso in cui lo sviluppatore voglia utilizzare un modello al di fuori di quelli integrati, si devono considerare due problematiche differenti: la modellazione e l'animazione del character. Per quanto riguarda la modellazione viene consigliato un tool chiamato Adobe Fuse, il quale permette la creazione di nuovi personaggi componendo vari elementi del corpo, vestiti, ecc. presenti nella libreria. Per quanto riguarda le animazioni, vengono consigliati sistemi di motion capture oppure, se non praticabili, tool esterni che forniscono librerie di animazioni direttamente applicabili ad un personaggio, come ad esempio Mixamo. Queste soluzioni permettono di ottenere buoni risultati in modo molto veloce ma non sempre possono essere utilizzate di fronte ad esigenze particolari.

2. **Integrazione** e **configurazione**: I due aspetti fondamentali riguardanti la configurazione sono la **personalità** e le **emozioni**. Dal momento che tipicamente viene decisa dallo sviluppatore nella fase iniziale e non muta durante l'esecuzione dell'applicazione, la personalità è rappresentata tramite una variabile di tipo read-only. In VAIF è rappresentata tramite l'indicatore di personalità di Myers-Briggs. Le emozioni sono invece rappresentate tramite la Plutchik's wheel of emotions e congiuntamente alla personalità possono essere utilizzate per modificare il comportamento dell'agente. Gli stati ottenibili sono diversi e di seguito se ne riportano alcuni esempi:

   - speacking: riproduzione di un file audio e conseguente lip-synch ed animazione;
   - listening/waiting: l'agente è in attesa di un input dell'Utente(verbale nel primo caso o un'azione nel secondo);
   - loockAt: l'utente sta guardando il character;

3. **Interazione**: parte centrale dell'architettura VAIF definita tramite l'Interaction Manager. Creando una timeline di eventi, controlla l'evolversi dello stato dell'ECAin seguito alle varie interazioni. L'interazione è il cuore del sistema e consente al character di muoversi, parlare, ricordare e porre lo sguardo in una particolare direzione. Gli eventi che possono essere generati sono diversi e se ne riportano di seguito alcuni a titolo di esempio.

   - Dialog: permette di specificare il nome di un file audio che verrà quindi riprodotto dall'agente con conseguente lip-synch;
   - Animation: consente di selezionare un'animazione (tra quelle disponibili) affinchè venga riprodotta su uno dei character;

- Response: utilizzato per impostare l'ECA in uno stato di attesa e quindi aspettare che l'Utenteparli. Nel momento in cui l'utente invia un input al sistema, tramite il riconoscimento di keyword predefinite è possibile invocare un nuovo evento sulla timeline associato alla parola chiave pronunciata;

- Gaze: permette di direzionare lo sguardo del character verso un punto di interesse;

- Emote: utilizzato principalmente per modificare lo stato emozionale del personaggio;

- MemoryCheck: implementa il meccanismo di memoria dell'agente. Si tratta di una lista di eventi (inizialmente vuota) alla quale viene aggiunta una entry ogni qualvolta viene lanciato un evento di un certo tipo. Questo permette allo sviluppatore di controllare ad esempio se un evento è stato lanciato in passato, oppure quante volte, ecc.

4. **lip-synch**: tale funzionalità è demandata a librerie esterne che permettono di implementare un meccanismo per il movimento delle labbra del personaggio in sincrono con l'audio. Un esempio è dato dal tool OVRLP di Oculus. Il meccanismo di base consiste nel definire dei blend shape, vale a dire un meccanismo differente dall'animazione tramite rigging che va a manipolare direttamente i vertici del modello tridimensionale. I bland shape così creati dovranno essere associati in parte ai fonemi che saranno pronunciati per l'implementazione del lip-synch, in parte alle espressioni facciali relative alle varie emozioni.

3.2.2 Limiti

Nonostante offra numerose funzionalità previste per il progetto di Tesi, il framework presenta alcune limitazioni non trascurabili. Infatti, se da un lato il sistema offre la possibilità di creare senza troppi sforzi un'applicazione comprendente un Embodied Conversational Agent, dall'altro mostra dei limiti nell'integrazione di tecnologie esterne nel caso in cui le funzionalità offerte non dovessero essere sufficienti.
Un altro elemento negativo è dato dal fatto che le operazioni di riconoscimento vocale sono limitate a sistemi con sistema operativo Windows 10.

Nonostante non si sia scelto di utilizzare VAIF per la realizzazione dell'applicazione oggetto di Tesi, si è cercato in ogni caso di riportare diversi aspetti caratteristici di questo sistema poiché offrono un ottimo spunto per l'implementazione di un sistema nuovo ma più modulare che possa far uso di altre soluzioni per problemi specifici quali TTS, STT e NLU.

3.3 UTEP AGENT System

UTEP AGENT System[20] è un sistema sviluppato presso la University of Texas, El Paso da un team denominato Advanced aGent ENgagement Team(UTEP). Lo scopo principale del gruppo è quello di fornire un sistema che implementi gli strumenti adatti a studiare e approfondire delle ricerche riguardo l'interazione ed i rapporti tra l'ECA e l'essere umano.
Secondo gli studi effettuai, per poter analizzare tale rapporto basato principalmente su aspetti paralinguistici e del parlato è necessario mantenere attiva l'interazione tra i due attori per un tempo prolungato. A tal fine è stata implementata un'applicazione, basata sul sistema UTEP e denominata Jungle. Si tratta di una sequenza di 23 scene attraverso le quali l'Utente, per un periodo di circa 40 - 60 minuti può interagire con l'ECA attraverso una serie di dialoghi, gesti, azioni, scenari virtuali e triggers.

Di particolare interesse sono gli aspetti paralinguistici e come questi influenzano il rapporto tra ECA ed Utente. Per cui saranno importati concetti come il volume della voce, le esitazioni, le pause i selenzi, ecc. che si intrecciano con comportamenti non verbali come le espressioni e i movimenti.

3.3.1 Architettura UTEP

La definizione degli agenti avviene tramite linguaggio XML e dunque con un approccio dichiarativo. Il resto del software è sviluppato in tre livelli differenti che vedono coinvolti tre sistemi fondamentali: Microsoft Kinect, Windows Speech SDK e Unity 3D.

Si analizzano di seguito i tre livelli:

1. **Bottom layer**: costituito dal sensore Kinect, ovvero un dispositivo che permette di catturare video in RGB, l'audio attraverso una serie di microfoni ed informazioni di profondità tramite dei sensori ad infrarossi;

2. **Middle layer**: implementa la logica del comportamento dell'Embodied Conversational Agent attraverso l'interpretazione degli input catturati attraverso i sensori (e dunque il livello sottostante). Fanno dunque parte di questo livello le operazioni di Speech To Text, Text To Speech e gesture recognition;

3. **Top layer**: logica applicativa definita direttamente dentro il game engine Unity 3D. Dunque, Ci si occupa a tale livello della creazione del mondo virtuale, delle fasi di rendering e delle animazioni.

Nella figura 3.3 vengono mostrati i blocchi fondamentali per l'implementazione del sistema ed i collegamenti tra essi.
I tre elementi più importanti all'interno dell'architettura sono animation, markup language e gesture recognition.

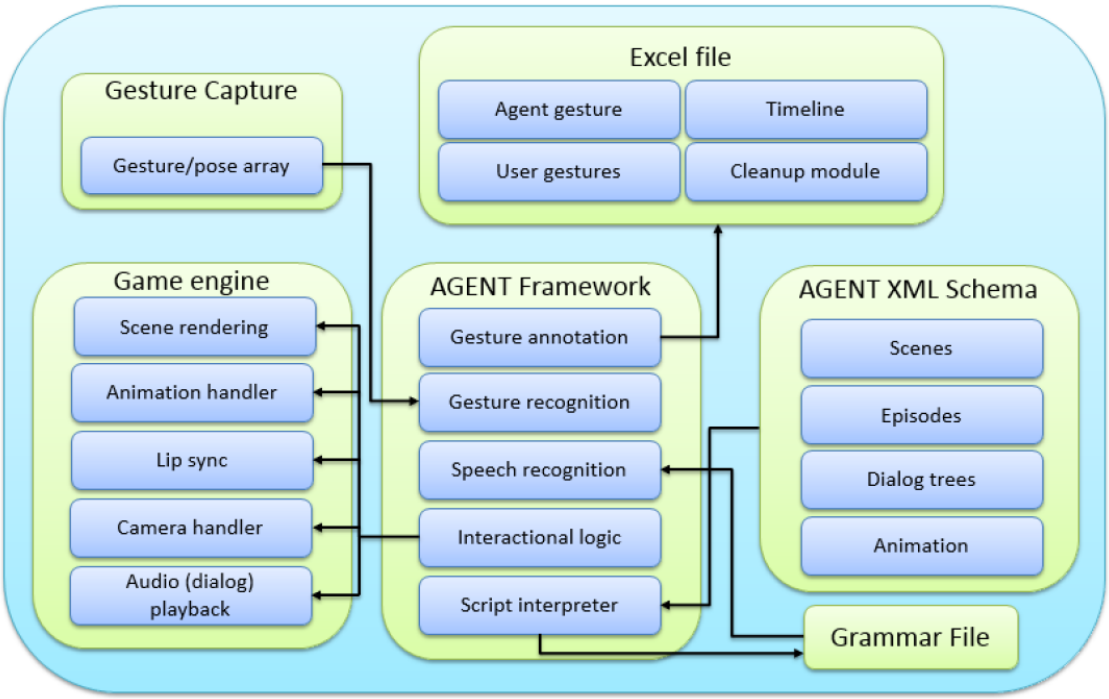


Figura 3.3: Architettura UTEP AGENT System

- **Animation**: Le animazioni dell'Embodied Conversational Agent sono gestite mediante una macchina a stati. Il programmatore può quindi eseguire lo start, end o blending con altra animazione. Quest'ultima possibilità è particolarmente utile per fare in modo che si generino animazioni sempre differenti e dunque evitare che l'utente percepisca movimenti ripetuti e dunque finti. Tali animazioni sono suddivise in layer differenti in modo da poter controllare parti differenti del corpo in modo indipendente;
- **Markup Language**: Al fine di semplificare la gestione di interazioni molto lunghe tra ECA ed Utente, è stato implementato uno strato intermedio in cui vengono interpretati ed eseguiti dei file XML che definiscono la *scena*, ovvero l'ambiente virtuale in cui è immerso l'agente. Ogni scena a sua volta è costituita da uno o più *episodi* che contengono dialoghi e elementi grammaticali per il riconoscimento del parlato. Il passaggio da un episodio ad un altro viene gestito andando ad intercettare e valutare i gesti ed il parlato dell'Utente.
- **Gesture Recognition**: il riconoscimento dei gesti dell'Utente è reso possibile da un'applicazione standalone per Windows. É necessario dunque interfacciare tale software con Unity 3D. Il modulo consente sia il riconoscimento dei gesti sia la creazione di nuovi movimenti e dunque la creazione di una propria libreria personalizzata [12].

Capitolo 4

Tools: Azure Cognitive Services

Una volta descritte le principali soluzioni attualmente disponibili che offrono un pacchetto completo per la creazione di un ambiente che supporti un ECA, in questo capitolo si descrivono le risorse che invece consentono di implementare singoli aspetti fondamentali per la gestione dell'Agente. Rientrano in questa categoria funzionalità come Natural Language Understanding, Text To Speech, Speech To Text, ecc.

Tipicamente questo approccio permette una maggiore versatilità e riduce i limiti introdotti dalle soluzioni esaminate nel capitolo relativo ai framework anche se con un notevole aumento della complessità riguardo l'implementazione.

MS Azure Cognitive Services permette di integrare all'interno della propria applicazione, tool o prodotto dei meccanismi che tramite una comunicazione naturale tra uomo e macchina riescono a comprendere ed interpretare quelli che sono i bisogni dell'Utente. Le funzionalità sono molteplici, tra cui la possibilità di convertire il parlato dell'utente in testo, convertire un testo in una voce sintetizzata e naturale (simile al parlato umano), traduzioni in lingue differenti, riconoscimento e catalogazione di immagini o video, capacità di estrapolare degli obiettivi (Natural Language Understanding) e molto altro.

Alcune di queste features vengono approfondite nelle pagine seguenti e sono state scelte per l'implementazione del progetto di tesi.
Le motivazioni che hanno portato a questa scelta sono diverse ma di particolare importanza sono i seguenti aspetti:

- la possibilità di implementare la maggior parte dei meccanismi necessari allo sviluppo dell'applicazione tramite prodotti e servizi rilasciati da un'unica azienda;

- la possibilità di interfacciarsi facilmente con l'engine Unity 3D grazie ad un SDK (anche se in versione beta);
- la possibilità di poter estendere o modificare dei moduli preservando quanto è stato già sviluppato. Questo è possibile grazie al fatto che ogni macro aspetto (ad esempio STT, TTS, NLU, logica applicativa, ECA, ecc) viene gestito in modo indipendente e poi messo in comunicazione con gli altri;
- la possibilità di sottoscrivere un account gratuito e poter testare la maggior parte dei servizi gratuitamente [†];

4.1 Speech To Text

Il servizio di Speech To Text di Azure [4] consente di generare del testo in real time a partire da uno stream audio che tipicamente è generato a partire dal parlato dell'Utente catturato dall'applicazione. Non è strettamente necessario dunque che l'audio venga catturato attraverso un microfono (anche se è possibile ed il caso più comune) ma è anche possibile utilizzare un file audio che abbia un formato supportato dal tool.

La traduzione in testo avviene tramite le stesse tecnologie che vengono utilizzate per applicazioni come Office o Cortana per cui il livello di accuratezza può essere paragonato a quello che si riscontra durante l'uso di queste ultime.
Se non specificato diversamente, il modello utilizzato per il riconoscimento del parlato è quello denominato Universal language model. Si tratta di un modello generato a partire da un set di dati di proprietà Microsoft e che può essere efficacemente utilizzato in contesti che prevedono delle conversazioni generiche.
Nel caso in cui si necessiti di sviluppare applicazioni che si riferiscono a contesti applicativi molto specifici e che richiedono un linguaggio molto tecnico, è possibile creare ed istruire un modello unico e personale. I vantaggi in questo caso sono sia la possibilità di gestire in maniera più appropriata dei linguaggi specifici sia di poter migliorare il riconoscimento in ambienti con un'acustica particolare (ad esempio ambienti rumorosi).

L'implementazione da parte dello sviluppatore di un'applicazione che faccia uso di tale servizio può avvenire tramite due approcci diversi: utilizzando lo **Speech SDK** oppure tramite le **REST APIs**. Entrambe hanno diversi pro e contro e la loro valutazione ha portato a scegliere, per la realizzazione del progetto di tesi, le Speech

[†]Il piano tariffario gratuito è stato ottenuto tramite sottoscrizione al portale Azure mediante un account studenti previa verifica di status da studente attraverso la mail istituzionale. Tipicamente, tale piano prevede per i vari servizi una limitazione sul numero di richieste totali su un dato periodo di tempo. Ad esempio per LUIS è previsto un massimo di 5 chiamate al secondo, 10K chiamate al mese

SDK. Il motivo principale è dato dalla possibilità di utilizzare in maniera implicita il riconoscimento vocale e le operazioni di Natural Language Understanding tramite LUIS in maniera unificata. Infatti, previa sottoscrizione al servizio LUIS, Le Speech SDK richiamano direttamente l'endpoint LUIS e restituiscono le Entità e gli Intenti risultanti dalla richiesta senza quindi gestire separatamente le operazioni di STT e NLU.

4.2 LUIS

LUIS (Language Understanding Intelligent Service) [1] è un'API appartenente ai servizi cognitivi di Azure di proprietà Microsoft che permette l'elaborazione del linguaggio naturale e dunque la comprensione del linguaggio umano. I domini applicativi in cui questo può costituire un notevole miglioramento nell'interazione uomo macchina sono molteplici: siti Web, dispositivi IoT e molto altro. Lo scopo principale è quello di integrare all'interno di un'applicazione la possibilità di far interagire l'Utentecon un sistema intelligente, capace di reagire coerentemente con le intenzioni dell'interlocutore.
Alla base del funzionamento ci sono servizi di Machine Learning applicati ad un testo generato tramite operazioni di Speech To Text a partire dal parlato dell'Utente, che consentono di estrapolare concetti ed informazioni rilevanti.
Il notevole vantaggio sta nella possibilità di ignorare gli algoritmi di Machine Learning necessari al funzionamento e concentrarsi sulla propria applicazione e sulle attività effettuabili dall'Utente che, come si vedrà in seguito, sono alla base dalla generazione di un modello LUIS.

Si possono individuare alcuni elementi chiave all'interno del sistema:

- **Espressioni/Utterances**: l'input generato dall'Utente, che deve essere successivamente interpretato per estrapolarne dei concetti rilevanti;
- **Finalità/Intent**: l'obiettivo o intenzione che l'utente vuole esprimere attraverso la frase pronunciata;
- **Entità/Entities**: una parola (o insieme di parole) che si vuole estrapolare dalla frase detta. Non è strettamente necessario che tutte le espressioni abbiano delle entità ma spesso costituiscono un valido aiuto per l'algoritmo al fine di individuarne il significato.

Dato un input che consiste nel testo generato dal parlato dell'Utente(tipicamente ottenuto con operazioni di STT), l'API si occuperà di ricavare una Finalità ma sarà cura dell'Utente istruire LUIS indicando espressioni plausibili ed indicando Entità significative.

Di seguito un piccolo esempio per meglio comprendere questi elementi fondamentali alla base del funzionamento.

Si suppone che un Utente voglia prenotare un volo e che pronunci la frase: "Compra 3 biglietti per Berlino per il 5 Dicembre". Da cui:

- *Espressione*: Compra 3 biglietti aerei per Berlino per il 5 Dicembre;
- *Finalità*: prenotare un volo.
- *Entità*: come già detto le entità costituiscono delle parole chiave che devono essere estrapolate dalla frase poiché necessarie per implementare la logica dell'applicazione che si vuole realizzare, in questo caso per la prenotazione di un volo. Per cui, le entità saranno:
  - *Berlino*: classificabile come *Località*;
  - *5 Dicembre*: classificabile come *Data*
  - Numero *3*: in LUIS i numeri sono classificati come *Entità Predefinita*.

Per poter inserire all'interno della propria applicazione delle funzionalità offerte da LUIS sono necessari alcuni passaggi chiave: creare delle risorse, creare un'app LUIS ed infine training, pubblicazione e testing.

Creare delle risorse

Tramite il portale Azure è possibile creare una nuova risorsa che in questo specifico caso sarà di tipo Language Understanding. Il portale è distinto dal sito LUIS che si utilizzerà in seguito per la creazione del servizio vero e proprio. In questa fase pertanto si sta definendo in separata sede una risorsa che si andrà successivamente ad associare all'applicazione LUIS (vedere Training e pubblicazione) e che contiene le informazioni necessarie per l'accesso al modello di linguaggio. †

Creare un'Applicazione LUIS

Esistono tre siti LUIS, uno per ognuna delle aree geografiche possibili (America del Nord, Europa, Australia). Per cui è necessario, in fase di creazione di una nuova risorsa, selezionare la stessa area geografica scelta in fase di creazione del servizio. Una volta creata una nuova app LUIS tramite il sito è necessario eseguire tre passi fondamentali:

- **Progettare uno schema**: consiste nel generare una Finalità per ogni azione prevista dall'applicazione e che deve essere gestita tramite Natural Language

†Ai fini della tesi è stato scelto come piano tariffario il piano gratuito ottenuto tramite sottoscrizione al portale Azure mediante un account studenti previa verifica di status da studente attraverso la mail istituzionale. Tale piano prevede la gestione di 5 chiamate al secondo, 10K chiamate al mese

Understanding. Si determinano in questa fase una serie di frasi possibili e si individuano le Entità;

- **Creare il modello**: si consiglia di definire un numero simile di esempi per ogni Finalità. Il motivo è dovuto al fatto che LUIS effettua delle stime considerando tutti i modelli disponibili, dopodiché seleziona quello con il punteggio migliore. Inoltre le espressioni di esempio utilizzate per una specifica Finalità corrispondono ad esempi negativi per le altre. Questa caratteristica, se non gestita opportunamente (appunto bilanciando il numero di esempi), può portare ad un squilibrio dei dati: le finalità con più esempi positivi hanno una maggiore probabilità di ricevere delle stime positive;
- **Migliorare l'Applicazione**: una volta che l'applicazione sarà pubblicata è possibile monitorare attraverso diversi strumenti messi a disposizione la qualità del riconoscimento del Intent. Durante questa fase di monitoraggio sarà possibile correggere eventuali predizioni errate.

Una volta creata una o più Finalità (ad es. CercaImmagini) si elencano una serie di Espressioni plausibili necessarie per istruire il calssificatore. Il passo successivo consiste nella definizione delle Entità, ovvero delle parole chiave che dovranno essere estrapolate dalle frasi pronunciate dall'Utente.
Una delle principali tipologie di Entità è quella di tipo *Simple* che permette di descrivere un concetto singolo.

Un semplice esempio potrebbe essere il seguente:
Si ha un'applicazione che consente ad un generico utente di richiedere informazioni sulle condizioni metereologiche. Necessariamente queste dipenderanno dal luogo, per cui un Entity di tipo *City* può essere utilizzata per identificare la città specificata dall'utilizzatore dell'app.
É buona prassi utilizzare un Intent particolare, di tipo *None*, per intercettare delle richieste che non sono proprie del contesto applicativo in modo da poterle trattare opportunamente - ad esempio notificando all'utente che l'agente non ha compreso la domanda, oppure comunicare che la richiesta non è confacente al contesto applicativo.

Training e pubblicazione

Il processo di training può richiedere diversi minuti, soprattutto dal momento che la richiesta potrebbe essere accodata e dunque messa in attesa. La procedura consiste nella creazione di un modello che permetta di avere un mapping tra gli obiettivi dell'Utente (Intents) e le Espressioni sfruttando i dati di training impostati nel passo precedente. Inevitabilmente, ogni modifica apportata al modello renderà nuovamente necessaria la procedura di Training.
Il collegamento tra l'applicazione vera e propria con l'app LUIS appena generata avviene tramite quello che viene chiamato **Endpoint**, ovvero una URL tramite la

quale si potrà accedere al servizio (secondo meccanismi indicati successivamente). L'URL conterrà al suo interno tre informazioni chiave:

- **Domain**;
- **Application ID**;
- **Key**;

In automatico viene creato un Endpoint denominato Starter Key che può essere utilizzato fin da subito per effettuare dei test anche se è consigliato generarne uno nuovo nel momento in cui si decide di distribuire l'applicazione. In ogni caso sarà necessario stabilire un collegamento tra la risorsa Endpoint al servizio di Language Understanding in Azure.

A questo punto basterà cliccare sul pulsante *Pubblica* e verrà creato un Endpoint tramite il quale sarà possibile chiamare il modello LUIS.

Testing

Il portale LUIS mette a disposizione un tool per il testing del modello creato. Tramite l'interrogazione del modello mediante delle Espressioni inserite testualmente è possibile visualizzare la finalità generata e il punteggio ad essa associata.
Se per determinate espressioni non si ha un risultato soddisfacente è possibile migliorare il modello andando a selezionare la previsione errata, assegnargli l'Intent corretto e dunque ripetere il Training.

4.3 Text To Speech

I servizi cognitivi Azure di tipo Text To Speech [5] permettono di sintetizzare una voce umana partendo da un qualsiasi testo. Le voci disponibili sono numerose: 75 voci standard per più di 45 lingue differenti e 5 voci neurali (le cui caratteristiche verranno spiegate nel seguito) in 4 lingue differenti (incluso l'Italiano).
Tali servizi consentono dunque di inserire all'interno della propria applicazione dei meccanismi per generare una voce umana che possa pronunciare del testo preimpostato.

Standard voices

Particolarmente utili per permettere all'Utente di accedere alle informazioni contenute dell'applicazione tramite formato audio. L'implementazione si basa su due metodi principali: *Statistical Parametric Synthesis* e *Concatenation Synthesis*, i quali permettono di avere delle voci altamente comprensibili e con un suono naturale e quindi simile ad un vero essere umano. Sono disponibili 45 lingue differenti

ognuna delle quali offre un'elevata accuratezza nella pronuncia, la possibilità di estendere acronimi, supporto alle abbreviazioni, interpretazione dei formati data ed ora, ecc.

Neural voices

Si tratta di un nuovo sistema di sintesi volto a migliorare alcuni difetti tipici dell'approccio precedente più classico. Si fa uso in questo caso di reti neurali per effettuare la traduzione del testo in voce computerizzata e per implementare la sintesi vocale. In questo modo si evita di approcciarsi a caratteristiche come l'intonazione della voce, il ritmo, la durata e gli accenti in modo separato ed indipendente per le varie lingue. Il forte vantaggio introdotto da tale sistema è infatti dato dalla capacità di trattare la prosodia e la sintesi in maniera simultanea fornendo risultati più fluidi e naturali. Tuttavia, questo tipo di sintesi è al momento disponibile soltanto per un numero ristretto di lingue: Tedesco, Inglese, Italiano e Cinese.

Le voci neurali, date le loro caratteristiche appena descritte, sono particolarmente utili alla realizzazione di Embodied Conversational Agent, chatboot, applicazioni vocali utili per situazioni in cui non è possibili digitare (ad esempio in macchina) in quanto permettono un coinvolgimento maggiore ed una minore fatica nell'elaborazione dell'output da parte dell'Utente grazie alla maggiore naturalezza. Le voci neurali permettono allo sviluppatore di utilizzare delle funzionalità non disponibili invece per le voci standard. Ad esempio ci sono delle voci che possono essere configurate per parlare allegramente.
É possibile inoltre configurare altri parametri come il tono e la velocità utilizzando un particolare linguaggio chiamato Speech Synthesis Markup Language (SSML).

Custom voices

I servizi cognitivi di Azure permettono di personalizzare le voci utilizzate per le operazioni di Text To Speech. Per utilizzare questo tipo di servizio è innanzitutto necessario effettuare una registrazione in studio della voce e caricare i file audio, insieme ai relativi file di testo che ne riportano la trascrizione e caricarli sul Custom Voice portal. In questo modo verrà creato un modello personale che può essere successivamente utilizzato per la sintesi vocale. In output vengono fornite anche informazioni che danno indicazioni riguardanti la qualità dell'audio, ovvero un punteggio associato alla pronuncia e il rapporto segnale rumore.
segue una fase di training - fatta sulla specifica lingua di interesse - al termine della quale è possibile effettuare dei test e, se si è soddisfatti, pubblicare il modello.
A questo punto è possibile utilizzare l'endpoint per effettuare le specifiche interrogazioni tramite l'applicazione, tool o prodotto che si sta sviluppando.
Si possono individuare 4 passaggi fondamentali indicati di seguito e schematizzati in figura 4.1.

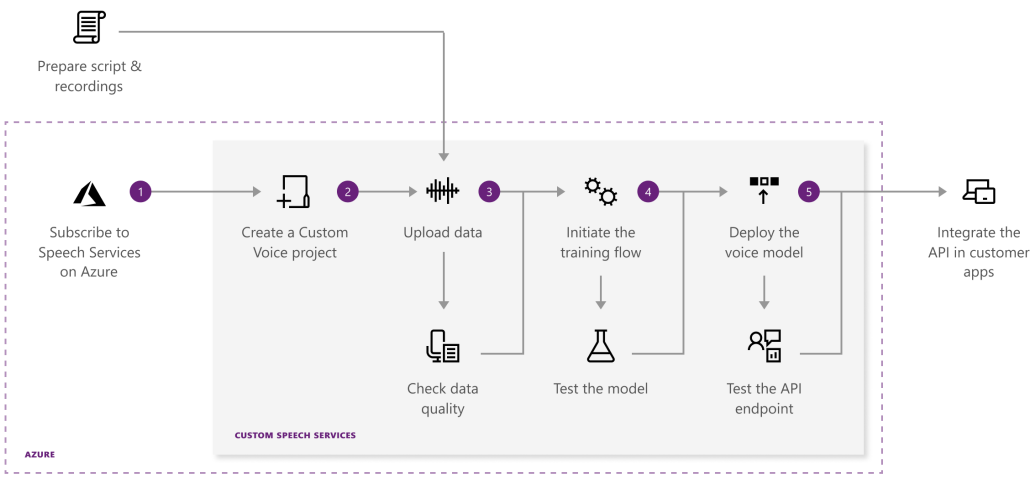


Figura 4.1: Passi per la creazione di un modello Custom Voice

- Creazione account Azure e creazione di un progetto Custom Voice;
- Caricamento dei dati;
- Training del modello;
- Pubblicazione del modello;

SSML

Come già accennato precedentemente, lo Speech Synthesis Markup Language [3] è il linguaggio che viene utilizzato in questo contesto per agire su alcuni parametri che influenzano determinate caratteristiche della voce ottenuta tramite il processo di sintesi vocale. Questo permette dal lato dell'applicazione di implementare logiche che prevedono diversi stati emozionali relativi all'agente conversazionale e di rendere il rapporto tra utente e macchina il più naturale possibile.
Alcuni dei parametri che possono essere definiti tramite XML sono pitch, pronuncia, ritmo del parlato e volume. Altre caratteristiche vengono invece gestite automaticamente dal tool, per cui aspetti come la normale punteggiatura, le pause tra varie frasi o l'intonazione nel caso in cui si stia ponendo una domanda, non vanno guidati dal programmatore.

Si ricorda inoltre che alcune funzionalità sono previste soltanto per pochi modelli vocali. Ad esempio, è stata introdotta precedentemente la possibilità di definire uno stile del parlato che non agisce sulle singole parole (come nel caso standard) ma sull'intera frase da pronunciare, come ad esempio *cheerful* e *sentiment*.

Si riportano di seguito gli elementi (*Elements*) previsti dal tool e utilizzabili all'interno di un file XML.

- **break/pause**: utilizzato per inserire tra una parola e l'altra una pausa o interruzione e andando di fatto a sovrascrivere la gestione automatica delle interruzioni da parte del servizio di Speech To Text. Due sono i parametri utilizzabili:
  - *Strength*: per indicare uno dei valori preimpostati tra none, x-weak, weak, medium (default), strong, xstrong;
  - *Time*: stesso significato del precedente ma permette di indicare il valore associato alla pausa con dei valori assoluti. Alcuni di questi corrisponderanno ai valori preimpostati di cui sopra. Ad esempio: 250ms equivale a x-weak.

  Di seguito, nella tabella 4.1 si riportano le diverse corrispondenze dalle quali è possibile anche comprendere il significato associato ai vari valori di strength.

Tabella 4.1: XML: valori di strength e time.

Strength	**Value**
none	0 ms
x-weak	250 ms
weak	500 ms
medium	750 ms
strong	1000 ms
x-strong	1250 ms

- **Paragraphs** and **sentences**: si utilizzano in questo contesto i tag * * ed *<s></s>*. p ed s vengono utilizzati per organizzare il testo che deve essere riprodotto rispettivamente in paragrafi e frasi. Queste informazioni aiutano il servizio a produrre un audio più naturale. Nel caso in cui non dovessero essere specificati tali elementi, questi vengono automaticamente determinati dal servizio.
- **phonemes**: tramite il tag *ph* è possibile specificare la fonetica delle singole parole in modo da migliorare la pronuncia di determinati vocaboli che vengono riprodotti in modo sbagliato. A differenza degli alfabeti latini, attraverso uno dei tre alfabeti fonetici è possibile indicare un'unica specifica pronuncia associata ad una determinata parola. Si indicano di seguito i vari attributi associati al tag:
  - **alphabet**: per indicare l'alfabeto fonetico che si intende utilizzare. é necessario scegliere tra uno dei tre supportati, ovvero ipa (International Phonetic Alphabet) sapi (Speech API Phone Set) ups (Universal Phone Set);

  - **ph**: indica la pronuncia associata alla parola specificata nel tag *phoneme*. Nel caso in cui si dovesse indicare un carattere non riconosciuto, il servizio di
    tts rigetta l'intero file XML per cui non viene generato alcun output.

- **Prosody**: uno dei tag che maggiormente permette di personalizzare la voce in base al contesto in cui la frase deve essere pronunciata. Fa riferimento a diversi attributi come il pitch, la durata, il volume, il ritmo, ecc. Dal momento che tali valori possono variare su scale anche molto grandi, il servizio di Text To Speech interpreta eventuali valori specificati dallo sviluppatore come dei suggerimenti. Inoltre, nel momento in cui dovesse essere specificato un valore non supportato (ad esempio al di fuori del range previsto) questo viene rimpiazzato dal servizio con un nuovo valore (eventualmente quello limite).
  I parametri possibili sono diversi e tutti opzionali [†]. In particolare:

  - *Pitch*: può essere inteso come il tono della voce e può essere specificato in tre modi differenti:
    * Tramite un valore assoluto espresso in Hertz (es. 400Hz)
    * Tramite un valore relativo con il quale è possibile aggiungere (+) o sottrarre (-) un al valore corrente una quantità espressa in Hertz oppure in semitoni.
    * Tramite un valore costante tra quelli previsti dal tool, ovvero x-low, low, medium, high, x-high, default.
  - *Contour*: In modo simile al caso precedente permette di modificare il valore del pitch della frase [††]. In questo caso è però possibile definire una sorta di array dove ciascuna entry è costituita da una coppia di valori del tipo <tempo, valore>. Il tempo è tipicamente espresso in percentuale mentre il valore in uno dei modi visti per il tag pitch. L'effetto è quello di modificare il pitch durante la sintesi di un unico testo in modo tale che la frase abbia per ogni intervallo specificato un certo valore;
  - *Range*: agisce sempre sul pitch, specificando l'intervallo in cui può assumere dei valori;
  - *Rate*: permette di modificare la velocità con la quale viene pronunciata la frase da sintetizzare. Se viene specificato un valore assoluto questo rappresenterà un fattore moltiplicativo applicato a quello di default utilizzato

[†]Gli attributi rate, volume e pitch possono essere modificati sia a livello di parola che di frase nel caso di voci standard mentre nel caso di voci neurali possono essere modificati soltanto globalmente per l'intera frase.

[††]L'attributo countour non è supportato per le voci neurali.

dal tool. Per cui se il valore è pari a 1 di fatto non comporta alcuna modifica, mentre se pari a 2 raddoppia la velocità e così via. Alternativamente è possibile indicare un valore preimpostato tra quelli messi a disposizione: x-slow, slow, medium, fast, x-fast, default;

– *Duration*: un modo alternativo per influenzare la velocità di riproduzione della frase è tramite questo parametro che permette di specificare il tempo - espresso in millisecondi - che deve trascorrere durante la sintesi vocale;

– *Volume*: permette di modificare il volume della voce che deve pronunciare il testo. Può essere espresso in modo assoluto indicando un numero compreso tra 0 e 100, attraverso un valore relativo per modificare il volume in base al valore corrente, oppure tramite un valore predefinito tra quelli disponibili: silent, x-soft, soft, medium, loud, x-loud, default.

Capitolo 5

Implementazione

Uno dei principali obiettivi del progetto di tesi è quello di realizzare una libreria che fornisca alcuni fondamentali e tipici strumenti per la realizzazione e gestione di un Embodied Conversational Agent. Alcune caratteristiche tipiche del contesto per cui la libreria è stata sviluppata hanno portato all'esigenza della realizzazione di un sistema che fosse semplice da configurare, adattare ed estendere con funzionalità aggiuntive (nuove o di completamento). Tali fattori sono diversi ed indicati di seguito:

- dal momento che le tecnologie impiegate in tale ambito sono in costante sviluppo è particolarmente utile disporre di un sistema che permetta la sostituzione di alcuni elementi senza inficiare il corretto funzionamento delle restanti parti;
- gli aspetti da tenere in considerazione per la realizzazione di un Agente Conversazionale sono fortemente numerosi e di natura interdisciplinare. Questo porta alla necessità di collaborazioni, alla creazione di team potenzialmente numerosi e alla necessità di predisporre il sistema ad ulteriori sviluppi futuri volti a migliorare e/o ad aggiungere nuove funzionalità;
- i contesti applicativi sono potenzialmente illimitati ed ognuno di essi ha caratteristiche specifiche non prevedibili a priori che rendono necessario poter configurare ed estendere il sistema il più possibile;
- nonostante i numerosi studi ed esempi che costituisco lo stato dell'arte ad oggi, molti degli aspetti trattati costituiscono un argomento di ricerca più che mai attuale per i quali dunque non esistono soluzioni ben definite o ottimali. Questo può portare a risultati non sperati ed ad un continuo ciclo di modifiche volte a migliorare il sistema creato;

Per questi motivi si è cercato di realizzare una libreria che rispondesse a due requisiti fondamentali: isolare più possibile gli elementi che possono dipendere da tool o librerie esterne, permettere una facile estensione dell'architettura esistente.

Il software mette a disposizione dell'utilizzatore una serie di strumenti che permettono di integrare in un ambiente di realtà virtuale un Embodied Conversational Agent che abbia le seguenti capacità:

- riconoscere le intenzioni dell'utente attraverso delle interazioni vocali;
- reagire ad eventi legati al parlato dell'utente e definibili dallo sviluppatore;
- utilizzare la sintesi vocale per poter convertire un testo scritto in parlato attraverso cui è possibile comunicare verbalmente con l'utente;
- aggiornare il proprio stato emozionale grazie ad un modello basato su appraisal variables e un set di emozioni discreto rappresentato in un sistema a due dimensioni;
- usare alcuni meccanismi forniti dalla libreria e legati al concetto di gamification che possono essere usati per consentire all'agente di intervenire a seguito di determinati eventi legati al tempo e alla correttezza di ciò che si sta facendo;

la configurazione di molti parametri legati all'ECA, ai tool utilizzati, ai testi che l'Agente dovrà pronunciare ecc. avviene tramite dei file XML di configurazione in modo da semplificare l'utilizzo.

5.1 Architettura Software

Di seguito vengono trattati alcuni dei blocchi fondamentali dell'architettura realizzata e descritte le principali funzionalità. In figura 5.1 vengono riportati i blocchi fondamentali dell'architettura che rappresentano un Embodied Conversational Agent, mentre in 5.2 le classi fondamentali per la gestione degli Intents e in 5.3 le classi manager più rilevanti.

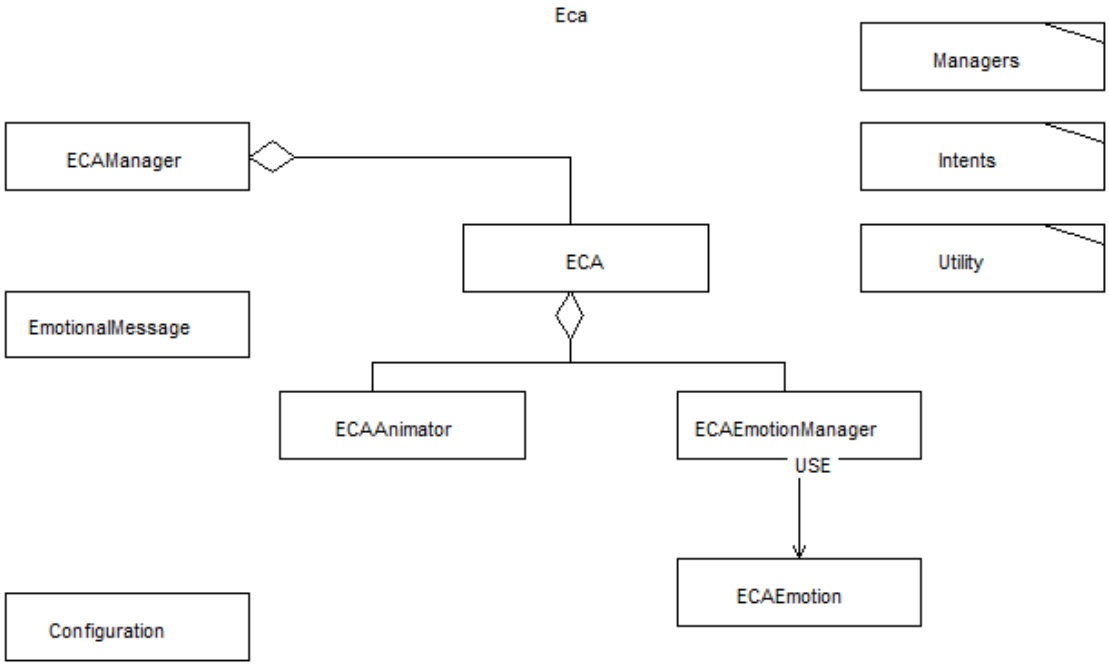


Figura 5.1: Blocchi fondamentali dell'architettura del framework

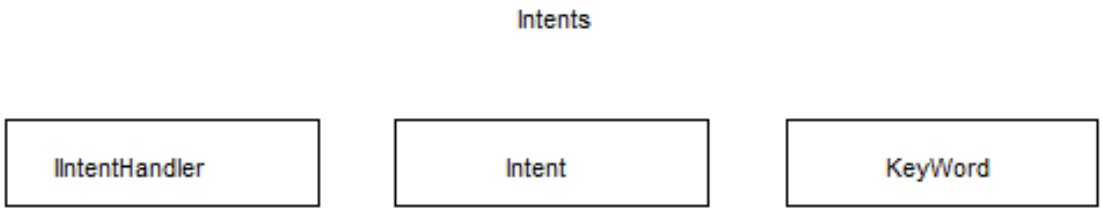


Figura 5.2: Classi per la gestione degli Intents

Figura 5.3: Principali classi Manager

Speech To Text e Natural Language Understanding

Le funzionalità di Speech To Text e Natural Language Understanding sono di fondamentale importanza per permettere un'interazione basata sul parlato dell'utente e correlate tra loro. Per la loro implementazione si è deciso di utilizzare alcuni dei servizi forniti da Microsoft Azure a tale scopo ed in particolare i Servizi Cognitivi Azure per le operazioni di STT e LUIS per le operazioni legate al Natural Language Understanding.
I dettagli sulle funzionalità offerte da questo tool sono affrontate nel capitolo precedente relativo allo stato dell'arte, mentre di seguito si illustrano alcuni dettagli implementativi e di design.
Nella figura vengono mostrate le classi principali relative a questo blocco, tra cui IntentManager, IIntentHandler, Intent, KeyWord. I loro scopi sono i seguenti:

- **IntentManager**: classe singleton che implementa i meccanismi di riconoscimento vocale ed NLU. Invia una richiesta ad un endpoint LUIS per l'elaborazione di un Intent che viene restituito alla classe stessa. I dati così ricevuti vengono dunque formattati secondo gli attributi delle classi Intent e KeyWord;

- **Intent** e **KeyWord**: due classi che racchiudono il risultato di un'interrogazione ai Servizi Cognitivi e Luis. Intent rappresenta il risultato con maggiore punteggio ottenuto dall'interrogazione e conterrà un numero di KeyWord pari al numero di Entity riconosciute;

- **IIntentHandler**: rappresenta la classe di collegamento tra il manager e un qualsiasi altro script della propria applicazione. Permette Infatti di specificare l'Intent di cui si vuole rimanere in ascolto e reagire allo stesso andando ad implementare il metodo Handle dell'interfaccia stessa;

Per eventuali sviluppi futuri e utilizzo di soluzioni alternative per le operazioni di NLU, dal momento che tali operazioni sono confinate al manager, sarà sufficiente estendere o sostituire tale classe. Per preservare l'architettura corrente sarà sufficiente mantenere il meccanismo previsto per l'iscrizione agli Intent di interesse (con le modalità descritte nel capitolo relativo all'utilizzo della libreria) e mantenere o estendere le classi contenitore dei risultati (Intent e KeyWord).

Text To Speech

Anche per quanto riguarda le operazioni di Text To Speech sono stati utilizzati i Servizi Cognitivi Microsoft Azure che sono stati trattati nei capitoli precedenti riguardanti lo stato dell'arte. L'architettura fornice una classe singleton **TtsManager** tramite la quale è possibile effettuare delle richieste di conversione di testo in audio. Tale operazione può essere richiesta tramite il metodo **Speech** specificando una serie di parametri necessari sia per la sintetizzazione dell'audio che per

la riproduzione dello stesso.
Alcuni di tali parametri vengono indicati nel seguito in quanto permettono di comprendere meglio quali sono le funzionalità offerte:

- il contenuto del testo da sintetizzare;
- la lingua;
- la voce con cui viene sintetizzato il testo;
- la sorgente da cui verrà riprodotto l'audio;
- eventuali funzioni da dover eseguire soltanto una volta che l'audio sarà terminato;

Oltre a quelle appena affrontate, un'altra funzionalità è stata percepita come fondamentale durante la fase di design ed è quella di poter gestire un meccanismo di accodamento dei messaggi. In questo modo nel caso in cui l'Agente stesse già parlando sarà comunque possibile accodare nuovi messaggi che verranno riprodotti secondo una strategia FIFO. É inoltre possibile specificare una condizione opzionale che dovrà essere vera nel momento in cui l'audio sarà pronto per essere riprodotto in modo tale da poter verificare se il contesto per cui tale messaggio è stato definito è ancora valido o meno.

Emozioni

Il progetto di tesi non si concentra sulla gestione delle animazioni attraverso le quali è possibile dare una forma visiva alle emozioni provate dall'Agente in quanto oggetto di un progetto parallelo in corso di sviluppo.
Il framework realizzato mette dunque a disposizione un modello logico attraverso il quale è possibile determinare, modificare ed aggiornare lo stato emozionale di un qualsiasi Agente Conversazionale presente all'interno dell'applicazione. Viene dunque fornita un'interfaccia attraverso la quale possono essere facilmente mappate delle animazioni facciali e/o del corpo a specifiche emozioni o combinazioni di esse. L'architettura così strutturata permette di separare due aspetti che tipicamente appartengono a discipline diverse o che comunque possono essere sviluppati in parallelo in quanto indipendenti tra loro o facilmente integrabili. Per l'implementazione di un modello matematico per il trattamento delle emozioni si è deciso di utilizzare un sistema discreto con un numero finito di emozioni che vengono modificate attraverso l'utilizzo di appraisal variables. La teoria riguardante tale modello viene approfondita nel capitolo riguardante lo stato dell'arte.

Il Design prevede tre blocchi fondamentali ed in particolare:

- uno per la definizione delle appraisal variables utilizzabili nel sistema e di quali siano le emozioni supportate dal sistema;

- uno per la definizione di un modello emozionale attraverso il quale è possibile specificare come una specifica appraisal variable influenzi una o più emozioni;
- uno per tenere traccia delle variazioni sulle emozioni relative al singolo Agente e di qual è l'emozione predominante in un determinato istante;

Animazioni

In maniera del tutto analoga al caso delle emozioni appena trattato anche le animazioni non vengono gestite dal punto di vista grafico in maniera esaustiva in quanto oggetto di un altro progetto di Tesi. Tuttavia, l'architettura fornisce un blocco specifico per aspetti come animazioni per il movimento, animazioni facciali, riproduzione dell'Audio, Lip-Sync, gesti ed animazioni del corpo.
Infatti, molte di queste funzionalità sono strettamente correlate ad informazioni logiche trattate nell'elaborato, per questo motivo si fornisce un'apposita interfaccia che permetta di mappare aspetti puramente grafici alle relative rappresentazioni logiche. Nello specifico la classe **ECAAnimator** può essere utilizzata o estesa per la gestione di diverse animazioni in quanto collegata ad eventi ed informazioni riguardanti ad esempio:

- l'audio da riprodurre (collegato ad aspetti di Lip-Sync);
- lo stato emozionale dell'Agente (collegato ad animazioni facciali e del corpo);
- informazioni spaziali (necessari ad esempio per il movimento, indicare oggetti ed altro);

Consapevolezza del contesto

Nella maggior parte dei contesti applicativi l'utente interagisce con l'applicazione non solo verbalmente ma anche svolgendo delle azioni all'interno del mondo virtuale in cui è ambientato lo scenario. Questo implica che le reazioni dell'Embodied Conversational Agent posso dipendere non solo dagli input vocali generati dall'utilizzatore ma anche da altre tipologie di input. In generale, nella maggior parte delle applicazioni l'utente deve svolgere delle attività che hanno delle caratteristiche comuni anche se possono essere di varia natura a seconda del contesto applicativo. All'interno dell'architettura tali elementi costituiscono uno **Stato** il quale è tipicamente correlato a quella che viene definita una **SmartAction**. Tali elementi, di seguito analizzati più in dettaglio, sono particolarmente adatti nel caso di applicazioni incentrate sul concetto di Gamification.

Di seguito vengono dunque riportati gli elementi costitutivi dello stato e successivamente quali sono le funzionalità offerte da una SmartAction.
Elementi dello Stato:

- Start time;

- End time;
- Accuracy;
- Staging;
- Percentage;

Gli attributi Accuracy e Staging sono utili soprattutto in contesti in cui è necessario tenere traccia di come l'Utente sta svolgendo determinate azioni. In particolare l'Accuracy indica la percentuale di accuratezza legata ad un determinato Task da compiere e dunque in corrispondenza di un errore diminuirà fino ad un valore minimo pari a zero. Viceversa l'attributo di Staging aumenta con il passare del tempo fino ad un valore massimo di uno e permette di tenere traccia del tempo impiegato dall'Utente nell'eseguire il proprio compito. Oltre ai valori numerici, la libreria consente di associare a tali criteri delle etichette del tipo Good, Bad o Normal che permettono di avere una rappresentazione più qualitativa dell'andamento dell'applicazione.
Queste informazioni possono essere quindi utilizzate da un Embodied Conversational Agent per poter aggiornare il suo stato emozionale, decidere di intervenire per fornire supporto o dare un feedback all'utente ecc.

Una SmartAction ha il compito di aggiornare tali valori durante l'esecuzione dell'applicazione per cui lo sviluppatore può associare ad un generico task che l'utente deve eseguire tale supporto che può essere esteso in modo da adattarsi ad un contesto specifico.
Pertanto, tipicamente basterà definire quali sono gli stati possibili relativi ad una specifica azione, determinare quali di questi costituiscono uno stato di errore e quanto velocemente deve essere eseguita tale attività.

5.2 Utilizzare la libreria

La libreria fornisce gli strumenti per poter integrare alcune delle funzionalità tipiche di Embodied Conversational Agent. Attraverso alcuni file di configurazione sarà possibile inserire ed utilizzare strumenti quali Speech To Text, Text To Speech, Natural Language Understanding.
Attualmente queste funzioni sono implementate attraverso i servizi cognitivi di **Azure** per STT e TTS e **LUIS** per quanto riguarda la parte di NLU. I moduli che si occupano di questo possono essere facilmente sostituiti con altre tecnologie in quanto trattati in maniera isolata dal resto (purchè rispettino il modo di interfacciarsi con le altri componenti dell'architettura).

Se si desidera invece utilizzare la libreria con i servizi attualmente offerti è necessaria una veloce fase di **configurazione**.

Configurazione

La fase di configurazione prevede due operazioni fondamentali, ovvero la creazione/customizzazione di alcuni file XML e il riempire alcuni metodi dello script *Configuration.cs*. Le principali componenti da configurare riguardano:

1. indicare le credenziali Azure e Luis necessarie per comunicare con i servizi online.
2. definire quali messaggi necessitano dei servizi di TTS e il loro contenuto;
3. (opzionale) definire quali azioni (tipicamente per la Gamification) si vogliono aggiungere all'applicazione e dunque impostarne i paramentri tipici;
4. (opzionale) definire quali messaggi sono abilitati e quali no in base alla tipologia di gioco prevista (es. training, rehearsal o examination);
5. (configurazione di default fornita) definire un Emotion Model che determina come le appraisal variables modifichino le emozioni dell'ECA;
6. registrare gli **Intent** previsti per la propria applicazione (definiti sul portale LUIS).
7. (opzionale) definire le regole del gioco in termini di quali Nodi vadano eseguiti prima di altri;

Credenziali per accesso ai servizi cognitivi

Permette di definire le credenziali necessarie per l'utilizzo dei servizi cognitivi Microsof. Di seguito viene riportata la struttura del file XML.

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>

  !-- By default, LUIS recognizes intents in US English(en-us).
  Other examples: "de-de", "it-it" -->
  <luisAttributes>
    <luisAppId>il tuo AppID</luisAppId>
    <luisAppKey>il tuo AppKey</luisAppKey>
    <luisRegion>region dell'App luis</luisRegion>
    <language>linugua dell'app luis</language>
  </luisAttributes>

  <ttsParameters>
    <serviceId>il tuo service ID</serviceId>
```

```
    <serviceZone>region</serviceZone>
  </ttsParameters>

</configuration>
```

Definizione dei messaggi e SmartAction

Il file permette di definire sia messaggi di carattere generico che messaggi legati allo stato in cui si trova una determinata SmartAction. Riguardo le SmartActions è possibile settare ulteriori parametri che influenzano il comportamento dell'ECA in relazione all'andamento delle attività svolte dall'utente. Di seguito viene riportata la struttura del file XML.

tag **tts**
viene utilizzato per definire dei messaggi di tipo generico che possono essere richiamati in qualsiasi punto del codice. Per aggiungere un messaggio di questo tipo è sufficiente aggiungere un nuovo tag (es. Presentation) che potrà dunque essere richiamato tramite la stringa "Presentation" in questo modo:
EmbodiedConversationalAgent.EcaStt.Speech("Presentation");

tag **actions**
Utilizzato per definire le diverse SmartActions presenti nell'applicazione. in particolare il nome dell'azione dovrà essere lo stesso utilizzato nell'Enum SmartActions.

tag **actions/tts**
Permette di definire messaggi correlati a specifiche fasi del generico scenario.
weight indica l'importanza dell'azione ed influenza la probabilità che un messaggio relativo ad essa venga riprodotto dall'ECA oppure no (in base al livello di presenza dell'ECA stesso). Valori possibili: Low, Medium, High.
Si può inoltre definire un messaggio di **descrizione**, di **conclusione** e di **aiuto** oltre ad altri messaggi che vengono lanciati nel momento in cui si ha un **cambiamento di stato** in Bad, Good o Normal. É possibile definire per quali valori numerici tale passaggio avviene ("xLimit").

```
<?xml version="1.0" encoding="UTF-8"?>
<game>
  <tts>
    <Presentation></Presentation>
    <Misunderstood></Misunderstood>
  </tts>

  <actions>
         <action name="Same action name in SmartActions Enum">
```

```
    <tts weight="High">
      <Description>Descrizione del Task</Description>
      <EndTask> meg finale </EndTask>
      <Help>msg di aiuto</Help>
      <Criteria>
        <Accuracy badTxt ="" badLimit="0.4"
              normalTxt ="" normalLimit ="0.8"
              goodTxt = "" goodLimit ="1">
        </Accuracy>
        <Staging badTxt ="" badLimit ="0.8"
                 normalTxt ="" normalLimit ="0.4"
                 goodTxt ="" goodLimit="0.0">
        </Staging>
      </Criteria>
    </tts>
    </action>
  </actions>
</game>
```

Abilitare o disabilitare messaggi

Nel caso in cui si prevedano diverse modalità di gioco è possibile che in alcune di queste non tutti i messaggi debbano essere riprodotti mentre in altre si. Nel caso in cui questa funzionalità non dovesse essere necessaria è possibile omettere tale file di configurazione. In quest'ultimo caso tutti i messaggi saranno sempre abilitati di default.
Di seguito viene presentata la struttura del file XML.

```
<?xml version="1.0" encoding="UTF-8"?>
<activatedMsgs>

  <gameType name ="Training">
      <EndTask isActive ="YES"> </EndTask>
      <Help isActive ="YES"> </Help>

      <Misunderstood isActive ="YES"> </Misunderstood>
      <Presentation isActive ="YES"> </Presentation>
  </gameType>

  <gameType name ="Rehearsal">
      <EndTask isActive ="NO"> </EndTask>
      <Help isActive ="YES"> </Help>
```

```
    <Misunderstood isActive ="YES"> </Misunderstood>
    <Presentation isActive ="YES"> </Presentation>
  </gameType>

  <gameType name = "Examination">
    <EndTask isActive ="NO"> </EndTask>
    <Help isActive ="NO"> </Help>

    <Misunderstood isActive ="NO"> </Misunderstood>
    <Presentation isActive ="YES"> </Presentation>

  </gameType>
</activatedMsgs>
```

Configurare l'Emotion Model

L'Emotion model definisce come le Appraisal Variables vanno ad influenzare le emozioni percepite dall'Embodied Conversational Agent. Tali variabili possono essere utilizzate in qualunque parte del codice per andare ad etichettare degli eventi come:

- Good;
- Bad;
- UnexpectedPositive;
- UnexpectedNegative;
- Nice;
- Nasty;

Ognuna di queste etichette/variabili influenza una o più delle seguenti emozioni: **Joy, Trust, Fear, Surprise, Sadness, Disgust, Anger, Anticipation**.
Ogni emozione ha un valore compreso tra -1.5 e +1.5 ed un livello (che dipende dal valore stesso) tra **Low, Medium, High** al quale possono essere associate delle emozioni secondarie così come definito nella **Plutchik's Wheel**.
É fornito un file di configurazione di default che può essere personalizzato per generare dei comportamenti diversi.
Di seguito viene riportata la struttura del file XML.

```
<?xml version="1.0" encoding="utf-8"?>
<model>
  <Good>
```

```
  <Joy>+0.3</Joy>
  <Sadness>-0.3</Sadness>
  <Surprise>-0.3</Surprise>
  <Anger>-0.1</Anger>
</Good>

<Bad>
  <Joy>-0.3</Joy>
  <Sadness>+0.3</Sadness>
  <Surprise>-0.3</Surprise>
  <Anger>+0.1</Anger>
</Bad>

<UnexpectedPositive>
  <Surprise>+1</Surprise>
  <Joy>+0.4</Joy>
  <Sadness>-0.3</Sadness>
  <Anger>-0.2</Anger>
</UnexpectedPositive>

<UnexpectedNegative>
  <Surprise>+1</Surprise>
  <Joy>-0.3</Joy>
  <Sadness>+0.4</Sadness>
  <Anger>+0.2</Anger>
</UnexpectedNegative>

<Nice>
  <Surprise>+0.2</Surprise>
  <Joy>+0.2</Joy>
  <Sadness>-0.4</Sadness>
  <Trust>+0.2</Trust>
  <Anger>-0.6</Anger>
</Nice>

<Nasty>
  <Disgust>+1</Disgust>
  <Joy>-0.5</Joy>
  <Sadness>+0.3</Sadness>
  <Surprise>+0.2</Surprise>
  <Trust>-0.2</Trust>
  <Anger>+0.1</Anger>
```

```
  </Nasty>

</model>
```

Registrazione degli Intents creati attraverso il portale LUIS

Una volta creati i vari Intents attraverso il portale LUIS, ogni script che necessita di reagire al riconoscimento di uno uno di essi dovrà estendere l'interfaccia *IIntentHandler* ed implementare i metodi *SubscribeHandlerToIntentManager* ed *Handle*. Il primo è utilizzato per fare in modo che, nel momento in cui venga riconosciuto un particolare Intent si possa invocare il metodo Handle appropriato. Il secondo è utilizzato per reagire all'Intent appena riconosciuto.
Di seguito vengono mostrati due esempi di override dei metodi appena menzionati.

```
public void SubscribeHandlerToIntentManager()
    {
        // Add intents defined in LUIS portal in this way:
        // IntentManager.Instance.AvailableIntents.Add(
             //"Intent NAME", this);

        //EXAMPLE:
        IntentName = new List<string> { "None", "Help", "Presentation" };
        IntentManager.Instance.AddIntentHandler(IntentName[0], this);
        IntentManager.Instance.AddIntentHandler(IntentName[1], this);
        IntentManager.Instance.AddIntentHandler(IntentName[2], this);
    }
```

Dove *Intent name* (string) deve **necessariamente** corrispondere al nome utilizzato sul portale LUIS.

```
public void Handle(Intent intent)
    {
        switch (intent.IntentName)
        {
            case "Presentation":
                SendMessage("Presentation");
                break;
            case "Other":
                // some method
                break;
        }
    }
```

Influenzare emozioni dell'ECA

Per influenzare le emozioni dell'Embodied Conversational Agent è necessario avere un EmotionModel.xml. Questo permette di agire sulle Appraisal Variables piuttosto che direttamente sui valori delle singole emozioni. Per fare questo è necessario accedere al campo EmotionManager dello specifico Embodied Conversational Agent ed aggiornare le sue emozioni classificando l'evento appena avvenuto con una specifica variabile. Un esempio di utilizzo:

```
//get one of available ECAs from EcaManager
ECA myEca = EcaManager.Instance.AvailableEcas[Ecas.Default];
//update emotion of myEca
myEca.Emotion.updateEmotion(AppraisalVariables.Bad);
```

Capitolo 6

Applicazione

In questo capitolo viene descritto il contesto applicativo che è stato scelto per la valutazione del sistema implementato.
Per poter testare l'effettiva utilità dell'Agente Intelligente all'interno di un'applicazione di Realtà Virtuale si è deciso di realizzare un serious game, vero e proprio strumento formativo in cui all'intrattenimento vengono abbinati elementi educativi. In generale l'obiettivo del gioco sarà quello di trasmettere o consolidare delle conoscenze nell'utente dell'applicazione e verificare se questo processo di apprendimento viene reso più efficiente dall'utilizzo di un Embodied Conversational Agent.
I vantaggi derivanti dallo sviluppo di un'applicazione di Realtà Virtuale sono stati menzionati nel capitolo introduttivo, tuttavia è bene ricordare che tali benefici dipendono anche dal grado di immersione dell'individuo all'interno dell'ambiente. Per questo motivo si ritiene che l'utilizzo di un Embodied Conversational Agent possa aumentare il senso di coinvolgimento e, di conseguenza, migliorare i risultati di apprendimento. Questi ultimi dipendono indubbiamente dalle informazioni acquisite ma talvolta anche dalla percezione più o meno realistica dell'ambiente circostante: alcuni contesti possono difatti influenzare fortemente lo stato d'animo o di attenzione dell'utente.
Basti pensare ad applicazioni volte alla formazione di vigili del fuoco, medici o persone comuni che si imbattono in situazioni di emergenza o di panico: il contesto di svolgimento può trasformare l'esecuzione di una procedura anche banale in un compito molto più complesso.

6.1 Contesto applicativo

Il contesto applicativo scelto riguarda l'esecuzione della procedura di primo soccorso nel caso di un incidente stradale. L'applicazione permette di formare un qualsiasi individuo avente conoscenza parziale o nulla riguardo le fasi previste dal Codice Della Strada per poter prestare soccorso ad un ferito.

Nel caso in cui ci si trovi coinvolti in un incidente stradale oppure nei pressi di un incidente e di persone che necessitano di soccorso a causa di malori o traumi, soccorrere il ferito è un obbligo di legge. L'omissione di primo soccorso è infatti perseguibile penalmente indipendentemente dalla gravità delle condizioni di salute dell'individuo coinvolto nel sinistro.
Si riportano di seguito alcune parti salienti del **Codice Della Strada** che mettono in risalto questo aspetto, le conseguenze a cui si può incorrere nel caso di omissione di soccorso e quali sono gli obblighi in queste circostanze.

Art. 189

- **Comma 1**: "L'utente della strada, in caso di incidente comunque ricollegabile al suo comportamento, ha l'obbligo di fermarsi e di prestare l'assistenza occorrente a coloro che, eventualmente, abbiano subito danno alla persona."

- **Comma 6**: "Chiunque, nelle condizioni di cui comma 1, in caso di incidente con danno alle persone, non ottempera all'obbligo di fermarsi, e' punito con la reclusione da sei mesi a tre anni."

- **Comma 7**: "Chiunque, nelle condizioni di cui al comma 1, non ottempera all'obbligo di prestare l'assistenza occorrente alle persone ferite, e' punito con la reclusione da un anno a tre anni."

- **Comma 8**: "Il conducente che si fermi e, occorrendo, presti assistenza a coloro che hanno subito danni alla persona, mettendosi immediatamente a disposizione degli organi di polizia giudiziaria, quando dall'incidente derivi il delitto di lesioni personali colpose, non e' soggetto all'arresto stabilito per il caso di flagranza di reato."

Questi estratti dell'Articolo 189 - che regola il comportamento in caso di incidente - mettono in luce il fatto che non è sufficiente fermarsi nel luogo del sinistro ma, nei casi previsti, anche di dare assistenza al ferito a meno di incorrere in sanzioni di una certa entità.
É dunque di fondamentale importanza acquisire le competenze necessarie per poter eseguire correttamente una procedura di primo soccorso, la quale da una parte potrebbe salvare la vita del ferito e dall'altra potrebbe evitare l'incombere di sanzioni su coloro che, per semplice ignoranza della procedura, non saprebbero prestare soccorso.

La consapevolezza del fatto che non tutti sappiano come agire in queste situazioni e allo stesso tempo di quanto sia fondamentale acquisire le competenze necessarie per prestare soccorso - e averle ben radicate per evitare di commettere errori in situazioni più critiche - ha portato a scegliere questo come contesto per l'applicazione di valutazione.

Il fine del primo soccorso

É anzitutto necessario chiarire quale obiettivo si prefigga una procedura di primo intervento, la quale deve essere accessibile ed eseguibile da qualunque individuo, a prescindere dal possesso di competenze mediche di qualsiasi tipo e complessità. La procedura infatti non ha lo scopo di curare il paziente ma di:

cercare di mantenerlo in vita attraverso dei semplici interventi senza applicare manovre non di propria competenza che potrebbero essere dannose per l'assistito;

fornire assistenza fino all'arrivo dei soccorsi e del personale di competenza.

A tal fine sono stati individuati alcuni passaggi chiave da eseguire in situazioni di emergenza e che possono essere schematizzati in questo modo:

- **proteggere** le persone che hanno bisogno di soccorso mettendo in sicurezza la zona attraverso l'opportuna segnaletica per notificare ad altri conducenti l'incidente;
- **chiamare i soccorsi** tramite il **118** ma non prima di essersi assicurati del numero di feriti presenti sul posto e del luogo esatto in cui ci si trova;
- **dare assistenza** a chi ne ha bisogno cercando di tenere a mente sempre cosa è possibile fare e quali sono invece le aree di competenza medica e le iniziative che non possono essere eseguite (ad esempio la somministrazione di farmaci);

(a) (b) (c)

Figura 6.1: (1) posizionare il triangolo (2) chiamare il 118 (3) prestare i primi soccorsi

Bisogna tenere conto anche di situazioni particolari, come ad esempio il caso in cui la persona da soccorrere sia un motociclista. In questa circostanza è bene ricordare di non sfilare il casco - dal momento che potrebbe portare a danni più gravi - e limitarsi ad aprire la visiera e slacciare il cinturino in modo da rendere la respirazione più agevole. Data la necessità di effettuare prima dei controlli sul ferito per accertarsi delle condizioni di salute, è bene evitare di spostare immediatamente il corpo a meno di situazioni particolari di emergenza come incendi, rischio di annegamento, ecc. Di seguito vengono esaminate più nel dettaglio le fasi della procedura [21].

Verifica dello stato di coscienza

Come prima cosa è necessario verificare lo stato di coscienza dell'individuo in modo da comprendere se vi è il rischio di un arresto cardiaco e respiratorio, che richiede un intervento immediato. Nel caso in cui l'uomo o la donna da soccorrere reagisca a stimoli esterni (ad esempio rispondendo a delle semplici domande o ad una stretta di mano), si può escludere il rischio di arresto cardiaco e respiratorio. In situazioni di incoscienza è necessario agire eseguendo delle manovre standard che possono salvare la vita del paziente o diminuire per quanto possibile i danni. In questo caso sono tre i passi fondamentali da eseguire:

- verificare se vi è respiro portando la propria mano sulla parte bassa del torace oppure sull'addome in modo da potersene accertare;
- se il respiro è affaticato è necessario verificare la presenza di corpi che ostruiscono le vie respiratorie del naso o della bocca. Tra gli elementi tipici che possono causare otturazioni vi sono protesi, saliva ecc.;
- rimanere vicino al ferito fino all'arrivo dei soccorsi in modo da poter monitorare costantemente le condizioni di salute dello stesso;

In figura 6.2 viene riportata schematicamente la procedura da eseguire in base ai possibili stati di coscienza in cui può trovarsi la vittima dell'incidente.

Figura 6.2: Come comportarsi in base allo stato di coscienza dell'infortunato

Stato di shock

Lo stato di shock viene provocato da un abbassamento della pressione arteriosa a causa del quale vi è una diminuzione dell'afflusso di sangue apportato ad organi vitali come cervello, cuore, polmoni, ecc.
Le cause che possono portare ad uno stato di shock sono molteplici e tipicamente legate ad improvvisi e forti cambiamenti. Alcuni esempi tipici sono un dolore lancinante, un forte trauma, emozioni intense di paura, gioia o rabbia, una consistente perdita di sangue ecc.

Più che le cause, è di fondamentale importanza riuscire a diagnosticare uno stato di shock andando ad analizzare lo stato di salute del paziente e confrontandolo con i tipici sintomi che lo caratterizzano. Di seguito vengono dunque riportati sei **sinotmi**:

- pelle molto pallida e fredda;
- brividi;
- sudorazione fredda (soprattutto sulla fronte);
- frequenza cardiaca molto alta;
- stato di agitazione;
- pronuncia di frasi senza senso;

Nel caso in cui venga rilevato uno stato di shock bisogna intervenire eseguendo una specifica procedura composta da poche fasi ma sulle quali spesso vengono commessi errori a causa di errate credenze sociali. Per maggiore chiarezza viene riportata quindi sia la procedura corretta sia alcuni tipici errori che vengono commessi.

Cosa fare:

- disporre il ferito a terra e possibilmente con le gambe sollevate (circa 20-30 cm più in alto rispetto al corpo) in modo da agevolare il flusso di sangue verso cuore e cervello;
- coprirlo nel miglior modo possibile per evitare che si raffreddi;

Cosa non fare:

- mettere il ferito in posizione seduta;
- cercare di riattivare la circolazione attraverso piccoli colpi su guance o gambe;
- somministrare piccole dosi di alcolici;

Ferite ed emorragia

Nel momento in cui il ferito presenta delle lacerazioni della cute che provocano la fuoriuscita di sangue è opportuno intervenire nella maniera corretta. Risulta utile inoltre sapere distinguere tra emorragie venose ed arteriose nel caso fosse necessario fornire informazioni ai soccorsi. Nel primo caso il sangue è di colore rosso cupo e fuoriesce in modo lento e continuo, nel secondo invece ha colore rosso vivo e fuoriesce con una pressione maggiore causando getti intermittenti.

Anche in questo caso, dal momento che si tende a fare confusione su quali siano effettivamente le azioni da compiere e quali non, vengono riportati sia i comportamenti corretti che quelli errati.

Cosa fare:

- bloccare la fuoriuscita del sangue utilizzando del materiale il più possibile sterile applicato sulla ferita;
- usare dell'acqua pulita per eliminare dell'eventuale sporco;
- lasciare eventuali corpi estranei come ad esempio delle schegge di vetro, penetrati all'interno della cute;

Cosa non fare:

- estrarre immediatamente corpi estranei per poter coprire la ferita;
- aspettare che la fuoriuscita del sangue termini spontaneamente e dunque non coprire la ferita;
- utilizzare acqua calda sulla ferita;

Fratture

La frattura di un osso comporta l'interruzione della continuità dello stesso in uno o più punti. A seconda che l'osso rimanga all'interno della cute o che fuoriesca e diventi visibile all'esterno, si possono distinguere due tipi di fratture: interne ed esterne.
É importante saper distinguere i due casi dal momento che le fratture esterne risultano essere più pericolose delle prime in quanto possono da un lato causare forti emorragie esterne e dall'altro esporre l'individuo a infezioni. Nel caso di fratture e traumi che non riguardino la parte toracica, la procedura prevede di non muovere gli arti interessati e provvedere a bloccarli e di coprire eventuali ferite.

Nel caso in cui la parte interessata sia quella del torace - ad esempio a causa di un forte urto contro il volante - è possibile che la respirazione del paziente possa essere compromessa a causa della vicinanza della struttura ossea (gabbia toracica) e dei polmoni.

In questo caso è importante agevolare la respirazione del ferito ponendolo in posizione semi seduta e comprimere con del materiale pulito eventuali ferite profonde.

Ustioni

Le ustioni possono essere classificate in tre gradi differenti a seconda della loro profondità. In particolare, se si considerano le ustioni dalle più superficiali alle più profonde si hanno rispettivamente ustioni di: primo, secondo e terzo grado, ovvero superficiali, intermedie e profonde.
In questo caso la procedura prevede di:

- provvedere a spegnere eventuali fiamme residue;
- non togliere i vestiti o i loro residui che sono rimasti attaccati alla cute del ferito;
- usare acqua fredda per ridurre il dolore del paziente;
- coprire le ustioni con materiale il più sterile possibile;

Rianimazione cardio-polmonare

Nel caso in cui il ferito si trovi in stato di incoscienza con battito cardiaco e respiro assenti è necessario intervenire eseguendo un corretto massaggio cardiaco e respirazione artificiale per evitare o rallentare eventuali danni ad organi vitali provocati dalla mancanza di ossigeno.
L'obiettivo del massaggio cardiaco è infatti quello di mantenere in circolazione il sangue nel paziente. L'abbassamento dello sterno - provocato dalla pressione esercitata dal soccorritore - causa la compressione del cuore contro la colonna vertebrale e l'immissione del sangue in circolo. Nella fase di rilascio che segue ogni compressione, la differenza di pressione generata riporta il sangue nel cuore e nel torace.
Ripetendo queste operazioni per il giusto numero di volte ed alla corretta frequenza si genera un circolo artificiale.
Si descrive di seguito la procedura relativa al massaggio cardiaco:

- verificare che il paziente sia posto su un piano rigido in modo da agevolare il massaggio cardiaco. É inoltre importante scoprire il torace in modo da poter valutare se le mani siano poste correttamente;
- posizionarsi accanto al paziente mantenendo le proprie spalle perpendicolari a quelle della persona da soccorrere;
- posizionare il calcagno di una mano al centro del torace e successivamente portare il calcagno della seconda sopra la prima;

- intrecciare le dita delle mani in modo da non esercitare pressione anche sulle costole e rischiare di provocare delle fratture a causa di eventuale fragilità ossea;
- iniziare le compressioni mantenendo le braccia estese, senza piegare i gomiti, in modo da sfruttare il peso del tronco per esercitare la forza necessaria ad abbassare il torace di 4-5cm. Fare attenzione a non andare mai oltre i 6cm;
- rilasciare senza mai far perdere il contatto tra le proprie mani e lo sterno del paziente;
- ripetere la manovra mantenendo una frequenza di circa 100-120 bpm - la fase di compressione e di rilascio deve avere circa la stessa durata;
- alternare 30 compressioni e 2 ventilazioni;

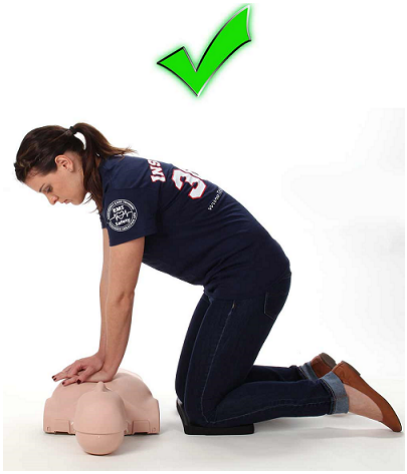


Figura 6.3: Posizione corretta per eseguire un massaggio cardiaco

Per la corretta esecuzione di una rianimazione cardio-polmonare è necessario eseguire anche una fase di ventilazione, il cui scopo è quello di mantenere un'adeguata ossigenazione durante la manovra. Dal momento che l'iperventilazione potrebbe comportare dei peggioramenti, ogni ventilazione deve portare un basso volume d'aria (circa 500-600 cc) per un massimo di un secondo.
Al termine di ogni blocco da 30 compressioni devono essere eseguite 2 ventilazioni di durata inferiore o uguale a 10 secondi totali.
Per poter eseguire correttamente la procedura di ventilazione attraverso la tecnica bocca-maschera bisogna anzitutto posizionare il capo del paziente in iperestensione per poi eseguire una normale ispirazione ed espirazione. L'utilizzo della maschera in sostituzione alla più classica respirazione bocca a bocca introduce una serie di vantaggi: essendo fornita di filtro antibatterico, riduce il rischio di infezioni, evita il contatto diretto con la cute e le secrezioni della vittima e permette il collegamento con una fonte di ossigeno.

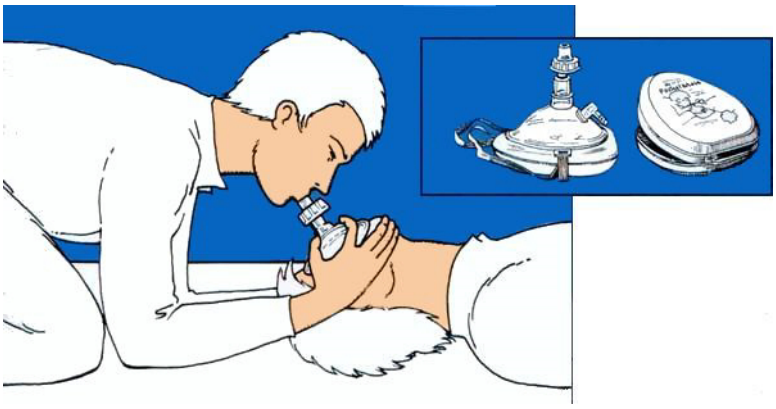


Figura 6.4: Corretta esecuzione di ventilazione tramite tecnica bocca-maschera

6.2 Struttura del gioco

Si distinguono due tipi di interazione principali: quella con gli elementi della scena che costituisce il mondo virtuale in cui l'utente è immerso e quella tramite l'utilizzo delle voce per relazionarsi con l'Agente Conversazionale. L'implementazione dell'interfaccia attraverso la quale è possibile interagire con gli elementi virtuali dipende in gran parte dalla piattaforma scelta per l'applicazione. Tuttavia indipendentemente dalla tecnologia utilizzata, si è deciso di evitare l'uso di comandi vocali rivolti verso l'Embodied Conversational Agent per compiere delle azioni che solitamente si svolgono in prima persona e prediligere strumenti di input più tradizionali quali mouse, tastiera, controllers, ecc come metafora di interazione che simula il prolungamento naturale di un arto.

Nel caso dell'interazione vocale l'intento del giocatore è quello di comunicare con l'Agente per cui si predilige l'uso del parlato in modo da rendere il rapporto il più naturale ed immersivo possibile. [†]

6.2.1 Fasi di gioco

Fase 1: posizionare il triangolo

Ambientazione su di una strada con un motorino a terra ed un ferito. La scena indica chiaramente la presenza di un incidente per cui il primo compito sarà quello di mettere in sicurezza il ferito tramite il segnale apposito (triangolo).

[†]Le principali richieste che possono essere fatte all'ECA vengono gestite in ogni fase del gioco. Un esempio tipo è quello di richiesta di aiuto (se prevista dalla modalità di gioco) attraverso la quale è possibile ottenere un suggerimento dall'Agente.

Fase 2: avvicinarsi al ferito

L'utente deve dirigersi verso il ferito come da obbligo di legg: l'omissione di soccorso è infatti un reato perseguibile penalmente.

Fase 3: sgomberare la zona

Per poter agevolare i soccorsi è necessario rimuovere eventuali elementi di disturbo che rendono difficoltosa l'assistenza al paziente. Sarà cura del giocatore rilevare gli elementi da spostare e dove posizionarli. Anche eventuali passanti che ostruiscono le operazioni di soccorso possono essere considerati come elementi di disturbo e quindi da allontanare.

Fase 4: verifica dello stato di coscienza

L'utente è appena arrivato in prossimità del ferito e deve per prima cosa verificare se è cosciente. Per farlo si rivolgono delle domande al paziente e si controlla se c'è una risposta. Nel caso in cui il ferito non fosse cosciente è necessario capire se respira osservando il petto e controllando se vi è innalzamento ed abbassamento del torace.

Fase 5: chiedere dove si trova

Prima di richiedere assistenza chiamando i soccorsi è doveroso sapere il luogo esatto in cui ci si trova per poter dare indicazioni precise. Si suppone che l'utente stia esplorando una zona che non conosce per cui sarà necessario chiedere indicazioni alle persone vicine.

Fase 6: chiamare il 118

Tramite un tasto l'utente può interagire con lo smartphone ed avviare la chiamata che verrà gestita tramite un botta e risposta vocale. Oltre ad indicare lo stato di salute del paziente è necessario segnalare la posizione (necessaria per l'ambulanza).

Fase 7: rianimazione cardiopolmonare

La rianimazione cardiopolmonare viene effettuata tramite 30 compressioni al ritmo di 100bpm intervallate da due ventilazioni (respirazioni bocca – maschera).
Per iniziare il massaggio cardiaco l'utente dovrà interagire con la zona del petto e sostenere delle compressioni ad un ritmo regolare e con la frequenza corretta.

6.2.2 Modalità di gioco

Training

Dal momento che la procedura di soccorso viene svolta attraverso una simulazione virtuale, le interazioni con gli elementi dell'ambiente vengono effettuate tramite un'interfaccia che deve essere appresa dall'utente. Questa fase preliminare può richiedere un tempo molto diverso a seconda dell'esperienza della persona nell'utilizzo di sistemi simili.
Attraverso la modalità di training l'Utente può acquisire le abilità necessarie per muoversi all'interno dell'ambiente ed interagire con l'Embodied Conversational Agent. In particolare viene richiesto all'Utente di eseguire almeno tre interazioni vocali con l'Agente in modo da metabolizzare tale modalità di interazione, che costituisce uno degli aspetti fondamentali della sperimentazione.

Learning

La modalità di learning ha come obiettivo quello di trasmettere al giocatore le conoscenze necessarie per poter eseguire correttamente la procedura di primo soccorso attraverso le modalità previste dall'applicazione. Questa fase è indispensabile in quanto è necessario che vengano fornite le conoscenze teoriche di base per poter eseguire i passi fondamentali.

In questa modalità di gioco svolge un ruolo centrale l'Embodied Conversational Agent a cui sono assegnati diversi compiti:

- fornire una spiegazione esaustiva di tutte le fasi del gioco in modo propedeutico allo svolgimento delle stesse da parte del giocatore.
- fornire dei feedback che permettano al giocatore di capire se sta commettendo degli errori o meno. Questa peculiarità consente di percepire l'Agente Conversazionale più simile ad un essere umano e quindi più reale, in quanto capace di prendere iniziativa durante il gioco.
- rispondere ad eventuali richieste di aiuto per aiutare il giocatore nel caso in cui non sappia bene cosa fare (nonostante la spiegazione fornita) o in caso di dubbi.

Dato che la procedura è costituita da una serie di passi sequenziali, la fase di learning è gestita in modo tale da impedire al giocatore di eseguire le varie azioni in un ordine diverso da quello corretto. Pertanto si abilita un task per volta in modo da creare una dipendenza tra il task successivo e quello in corso di svolgimento, ovvero il generico task T_{i+1} può essere eseguito solo se si è completato il task corrente T_i.

L'apprendimento può richiedere un tempo variabile da persona a persona per cui l'utente potrà ripetere questa modalità il numero di volte necessario per acquisire

familiarità con la procedura e con le dinamiche di gioco.
Non è previsto un sistema di punteggio o di classifica nella fase di learning dato che lo scopo è quello di acquisire delle competenze e non dimostrare di averle. Tuttavia, dal momento che la rapidità di esecuzione di determinate azioni e la precisione con cui vengono svolte possono essere cruciali per la corretta esecuzione dell'intera procedura, l'Embodied Conversational Agent avvertirà di eventuali errori di questo tipo durante il gioco.

Capitolo 7

Test

7.1 Protocollo sperimentale

Partecipanti

Per la sperimentazione sono state effettuate 22 esecuzioni complete dell'applicazione di addestramento alla procedura di primo soccorso stradale. Tutti i partecipanti selezionati non avevano nessuna esperienza pregressa rilevante riguardo il dominio applicativo scelto, per cui nessuno aveva svolto corsi di primo soccorso stradale. Questa peculiarità di partenza è stata considerata tassativa per evitare che le valutazioni soggettive dei partecipanti – in particolare quelle positive – fossero condizionate da una eventuale familiarità con la procedura.

Materiale

L'esperimento è stato condotto all'interno di una stanza con poco riverbero e sufficientemente silenziosa al fine di evitare eccessive difficoltà nel riconoscimento vocale. Per il medesimo scopo si è deciso di utilizzare un microfono esterno in modo da consentire al partecipante di poterlo posizionare nei pressi della bocca.
La versione dell'applicazione utilizzata durante l'esperimento è di tipo Desktop, per cui le interazioni sono state svolte dall'utente per mezzo di tipici dispositivi di input quali mouse e tastiera.
Per quanto concerne i dispositivi di output, è stato utilizzato un monitor LCD per la visualizzazione di informazioni visive e delle cuffie per la riproduzione dell'audio in modo da isolare l'utente dall'ambiente esterno.

Design

Ogni Utente ha eseguito l'intera procedura un'unica volta in quanto un'eventuale seconda sperimentazione sarebbe stata invalidata dalle conoscenze acquisite nell'esecuzione precedente, che avrebbero ridotto automaticamente l'utilità dell'istruttore.
Inoltre, dal momento che non tutti hanno la stessa esperienza nell'utilizzo di applicazioni di Realtà Virtuale, prima di svolgere la procedura di primo soccorso ogni utente è stato sottoposto ad una fase di Training nella quale apprendere le interazioni fondamentali previste nella fase di gioco. A tale scopo è stata implementata un'applicazione di Training attraverso la quale acquisire le seguenti competenze:

- muoversi all'interno dell'ambiente virtuale;
- imparare a comunicare con un Agente Conversazionale presente nell'applicazione;
- metabolizzare alcune features proprie dell'istruttore, tra le quali la capacità di rispondere ad eventuali richieste di aiuto da parte dell'Utente.

Questa fase preliminare termina nel momento in cui l'utente si sente sicuro e padrone delle modalità di gioco ed è pertanto pronto per la sperimentazione.

Dati oggettivi

Durate l'esecuzione dell'applicazione vengono generati diversi file di Log, che permettono di tracciare l'andamento della simulazione in merito ad alcuni elementi di interesse. Sono tre gli aspetti principali che sono stati presi in considerazione:

- aspetti legati al grado di sicurezza e di errore commesso dall'Utente durante la procedura. Rientrano in questa categoria informazioni quali: accuratezza, tempo di esecuzione, richieste di aiuto, ecc. I valori legati a tali parametri sono molto rilevanti dal momento che vengono utilizzati dall'Agente Conversazionale per avere informazioni sul contesto. Di conseguenza, l'istruttore virtuale sarà tanto più presente quanto più numerose saranno le variazioni di tali parametri.
- aspetti legati allo stato emozionale dell'Agente. Nel progetto di tesi non vengono gestite animazioni facciali per l'esternazione dei sentimenti provati dall'ECA in quanto oggetto di studio di una tesi parallela. Tuttavia il framework sviluppato presenta tutti gli elementi necessari per definire, aggiornare e monitorare lo stato emozionale dell'utente, che viene analizzato attraverso uno specifico file di Log in forma testuale.
- aspetti legati alle interazioni vocali. Uno specifico file di Log viene aggiornato ad ogni richiesta vocale effettuata dall'Utente durante la sperimentazione. In

forma del tutto anonima vengono così memorizzate le frasi utilizzate e il contesto in cui sono pronunciate, gli Intent riconosciuti ed il punteggio associato a ciascuno di essi. Tali informazioni possono essere utili per effettuare un'analisi riguardo l'efficacia dei Tools impiegati per l'implementazione del framework. Inoltre, il linguaggio utilizzato dall'Utente per relazionarsi con l'Agente Conversazionale può essere analizzato per fare delle considerazioni in merito al livello di immersione dell'Utente nel contesto.

7.2 Questionario

La scelta dei questionari è stata fatta considerando le peculiarità del framework di cui si vuole verificare l'efficacia. In particolare gli elementi di maggiore rilevanza individuati riguardano le interazioni vocali, per cui operazioni come Text To Speech, Speech To Text e Natural Language Understanding ricoprono un ruolo fondamentale. Altri aspetti - che richiedono un'integrazione futura con un altro progetto parallelo in corso di sviluppo dedicato ad animazioni di vario tipo - non vengono analizzati tramite dei questionari ma, come già visto, attraverso dei file di log in maniera testuale.
Tuttavia non si vuole effettuare una ricerca riguardo la robustezza delle tecnologie utilizzate ma verificare l'efficacia di un Embodied Conversational Agent all'interno di un'applicazione di Training integrato nell'ambiente virtuale attraverso il Framework realizzato. Per questo motivo si è più volte specificato nel questionario di valutare principalmente le interazioni con i vari ECA presenti nella scena e di considerare principalmente la pertinenza delle informazioni fornite dall'Agente e l'utilità di tali interventi.

Per la scelta di questionari adatti a tale scopo si è fatto riferimento ad una ricerca basata sulla comparazione di 6 questionari specifici per interfacce conversazionali [15]. Diversi studi hanno portato alla definizione di alcune user experience (UX) dimensions che risultano particolarmente utili per differenziare i questionari a seconda dei parametri trattati.
Di seguito vengono elencate le principali UX [6]:

- Generic UX;
- Affect, emotion;
- Enjoyment, fun;
- Aesthetics, appeal;
- Hedonic quality;
- Engagement, flow;

- Motivation,
- Enchantement;
- Frustration;

Le principali dimensioni di cui si vuole valutare l'efficacia sono quelle di tipo Generic, Enjoyment, Emotion e Pragmatic. Per cui, sulla base di uno studio condotto sulla classificazione di alcuni questionari in relazione alla copertura di diverse UX dimensions [15], sono stati scelti: **SASSI**, **TRINDI**, **SUS**. In particolare, di seguito vengono brevemente descritti il SASSI ed il SUS.

SASSI

Subjective Assessment of Speech System Interfaces (SASSI) [14] è un questionario molto diffuso per la valutazione di interfaccie conversazionali. Si tratta di una raccolta di 34 domande suddivise in 6 categorie di seguito elencate [14]:

- System Response Accuracy: domande relative alla capacità di riconoscere gli input vocali dell'utente e il loro significato;
- Likeability: domande relative alla gradevolezza dell'applicazione;
- Cognitive Demand: domande relative sia allo sforzo mentale necessario per utilizzare il sistema sia alle sensazioni provate dall'Utente durante l'esperienza.
- Annoyance: domande relative alla capacità del sistema di essere poco noioso, ripetitivo, ecc;
- Habitability: domande relative alla facilità di comunicazione con il sistema in relazione ai termini da utilizzare;
- Speed: domande relative alla velocità del sistema;

Per focalizzare la valutazione di tali fattori sull'Embodied Conversational Agent nel questionario proposto ai partecipanti è stato più volte ripetuto che per "Sistema" si intende l'Agente Conversazionale.

SUS

System Usability Scale [9] è uno dei questionari più diffusi per la valutazione dell'usabilità di un generico sistema. Si tratta di una raccolta di 10 domande per le quali l'utente può esprime un certo grado di accordo tramite una *Likert-Scale* del tipo in figura 7.1. Le domande che costituiscono tale questionario sono facilmente generalizzabili a qualsiasi sistema, per cui esistono una moltitudine di sperimentazioni condotte utilizzando questo set di domande in cui il termine "System" viene

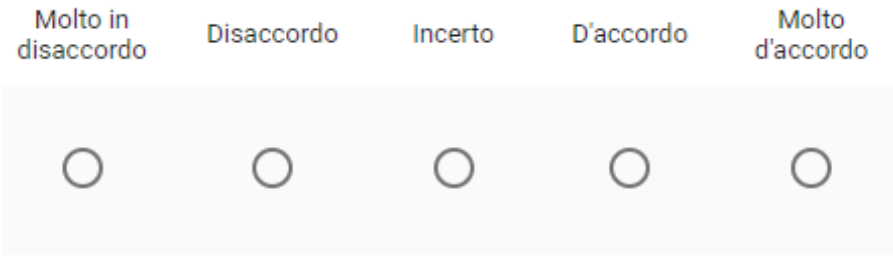


Figura 7.1: Likert scale a 5 livelli

modificato per adattarsi al contesto specifico [15]. Anche nella sperimentazione oggetto della tesi si è seguito questo approccio in modo tale da poter applicare il SUS in maniera mirata per la valutazione dell'Agente Conversazionale.

Domande generiche

Oltre ai tre questionari standard menzionati precedentemente, sono state formulate altre domande di completamento utili per valutare il sistema implementato. Si tratta principalmente di quesiti di valutazione dell'Embodied Conversational Agent e di come questo sia stato percepito dall'utente. Alcune di queste domande sono state ispirate dalla sperimentazione effettuata per valutare GRETA [8]. Le qualità dell'Agente Conversazionale sottoposte a valutazione in questo caso sono:

- Utilità;
- Intelligenza;
- Credibilità;
- Piacevolezza;
- Affidabilità;
- Competenza;

L'Utente andrà a valutare tali caratteristiche attraverso una scala di valori da 1 a 6.

7.3 Risultati

Di seguito vengono riportati i risultati di maggiore rilevanza ottenuti dal protocollo sperimentale, suddivisi tra quelli di tipo oggettivo e quelli di tipo soggettivo.

Risultati oggettivi

Attraverso l'analisi dei file di Log generati durante l'esecuzione di ciascun test sono state isolate alcune informazioni di particolare interesse. Uno degli aspetti più rilevanti riguarda la capacità dell'Embodied Conversational Agent di intervenire in seguito a situazioni o eventi particolari. Nel caso specifico l'Agente deve essere in grado di fornire un supporto concreto nel momento in cui l'utente dovesse commettere degli errori durante l'esecuzione della procedura o nel caso stesse impiegando un tempo eccessivo. Si è così analizzato il livello di intervento dell'ECA in ogni fase della procedura per verificare in media quante volte ha preso iniziativa per correggere il comportamento dell'Utente. Il risultato di tale analisi è riportato nel grafico 7.2. Come si può osservare dal grafico il numero di interventi medi si concentra principalmente in:

- **1**: posizionare il triangolo;
- **3**: sgomberare la zona da persone inopportune;
- **11**: rianimazione cardio-polmonare;

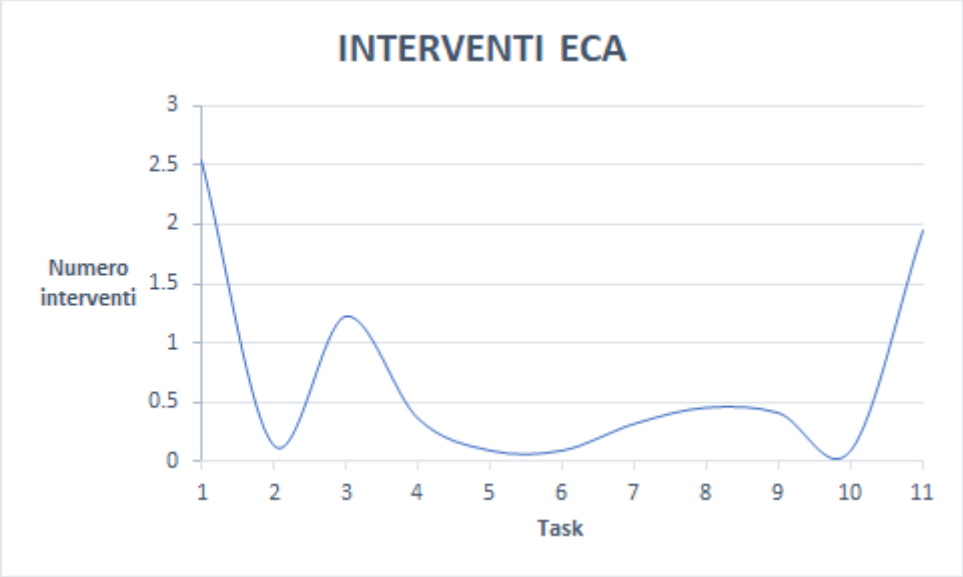


Figura 7.2: Numero di interventi medi dell'Embodied Conversational Agent per Task

Queste sono le azioni in cui gli utenti hanno avuto maggiore difficoltà e che di conseguenza necessitano di un supporto aggiuntivo da parte dell'ECA. Alcune delle azioni centrali sono molto simili a livello di interazione alle prime, per cui il migliore andamento del gioco può essere dovuto all'esperienza pregressa ottenuta tramite l'esecuzione delle prime attività.
Queste informazioni vengono corredate da un ulteriore dato significativo che indica

quanto tempo è mediamente necessario per completare il Task dopo che l'Agente ha fornito supporto all'utente. Tali informazioni sono indicate nel grafico in figura 7.3 dove, per i punti 1, 3 e 11, viene indicato:

- tempo impiegato per completare l'azione dopo il supporto dell'Agente;
- tempo totale impiegato per completare l'azione;

Come si può notare dal grafico, il tempo impiegato una volta ottenuto il supporto è significativamente inferiore rispetto al tempo totale impiegato. Questo implica che l'utente riesce velocemente a recuperare da eventuali errori commessi e a completare il task in seguito all'intervento dell'Agente.

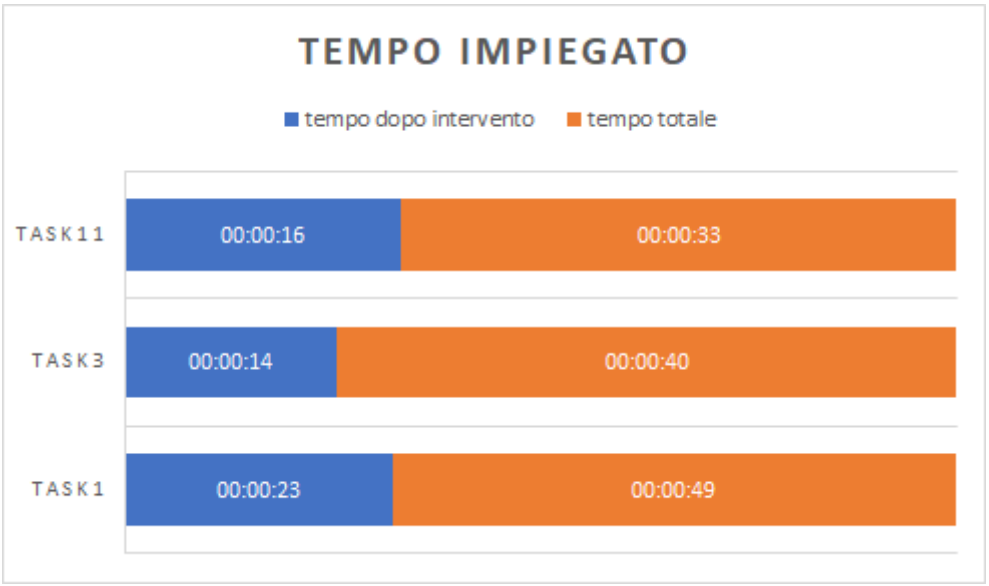


Figura 7.3: Tempo impiegato in totale e tempo impiegato dopo l'intervento dell'ECA

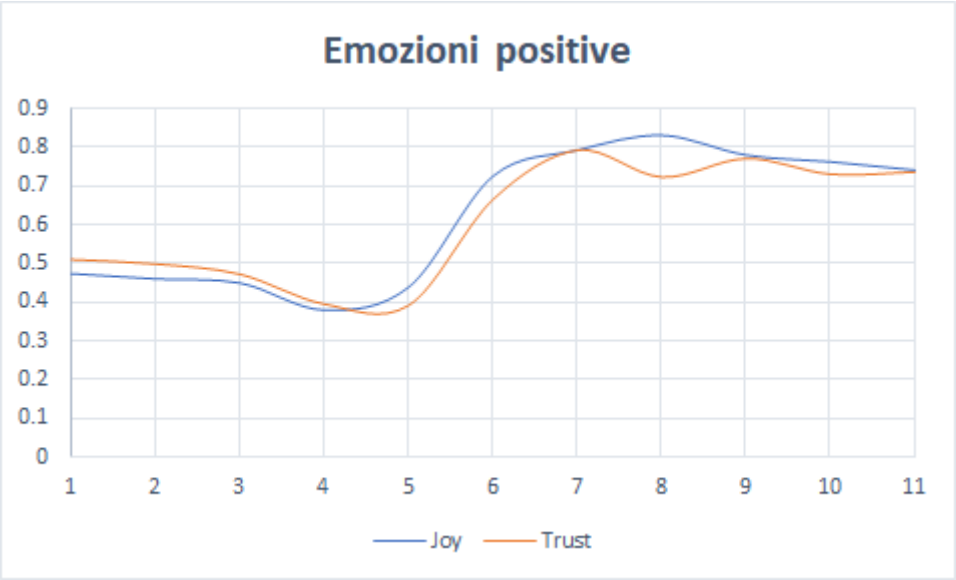
Un'altra peculiarità dell'istruttore è quella di riuscire a gestire richieste di aiuto da parte dell'utente. Questo implica una forte percezione del contesto in modo da poter fornire un aiuto coerente con le esigenze dell'utente in quel determinato istante ed in base alla fase corrente della procedura di primo soccorso. Dato che le capacità di iniziativa dell'Embodied Conversational Agent fanno sì che questo intervenga in maniera considerevole anche senza una domanda diretta da parte dell'utente, le richieste di aiuto non sono molto numerose 7.4. Tuttavia si può constatare una certa correlazione tra le fasi in cui l'utente ha riscontrato maggiori difficoltà (come già indicato in 7.2 e il numero di richieste di aiuto. Infatti entrambi hanno un picco sui Task già precedentemente indicati, ovvero 1, 3 ed 11. L'ultimo aspetto che si vuole sottolineare è quello legato alle emozioni provate dall'istruttore durante le fasi della procedura. Di seguito si riporta l'andamento delle emozioni

Figura 7.4: Numero di richieste di aiuto per Task

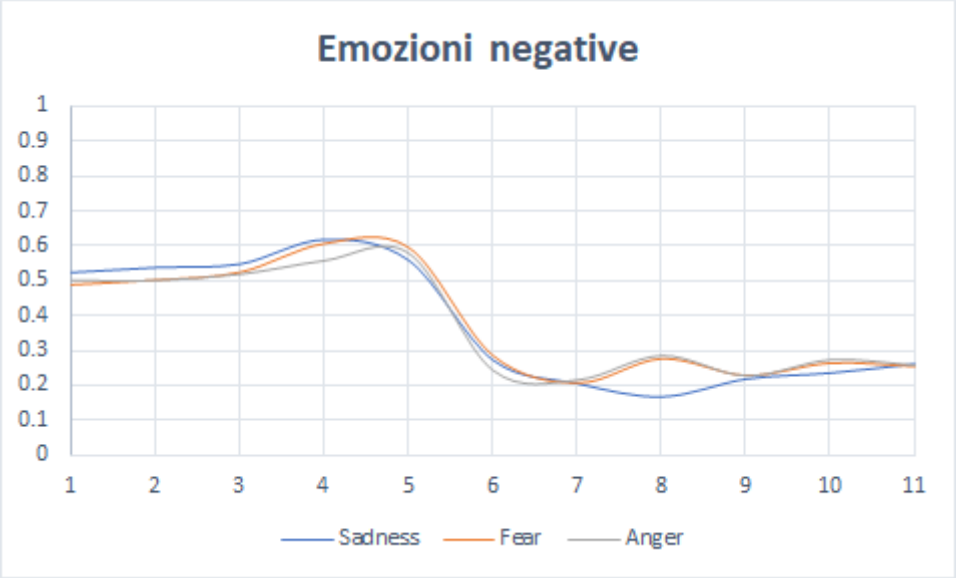
suddivise tra emozioni positive ed emozioni negative. Come si può verificare dai due grafici in 7.5, le emozioni positive e quelle negative sono bilanciate fra di loro per cui quando un tipo di emozione aumenta l'altra diminuisce. Inoltre si può osservare come sia nella fase iniziale che in quella finale del gioco si rileva un andamento decrescente della curva delle emozioni positive mentre si ha una pendenza crescente durante l'esecuzione dei Task centrali. Questo andamento è del tutto compatibile con le informazioni riportate nei grafici precedenti che hanno già messo in luce le fasi della simulazione risultate più complesse. Inoltre è possibile notare come la variazione delle emozioni positive sia opposta a quella delle emozioni negative.

Risultati soggettivi

Per risultati soggettivi si intendono quelli ottenuti tramite l'analisi dei questionari somministrati agli utenti che hanno effettuato la sperimentazione. L'obiettivo del questionario è valutare l'interazione con gli agenti principalmente dal punto di vista vocale. I risultati che vengono presentati di seguito sono relativi alle domande che sono state ritenute più significative e suddivise per tipologia di questionario.
Attraverso il questionario TRINDI è stato possibile valutare le capacità del sistema di interpretare le frasi dell'utente e di gestire situazioni in cui si ha difficoltà nel comunicare a causa ad esempio di limiti tecnologici legati alle operazioni di Text To Speech e/o Natural Language Understanding. Nella figura 7.6 si riportano quattro risposte selezionate in quanto considerate più rilevanti. Tali risposte hanno permesso di constatare che l'agente possiede le capacità necessarie per poter proseguire la conversazione anche in seguito a difficoltà nella comunicazione verbale.



(a)



(b)

Figura 7.5: (1) Andamento emozioni positive (2) Andamento emozioni negative

Le funzionalità che permettono di riproporre una domanda o fornire suggerimenti per portare avanti la conversazione sono particolarmente utili nel momento in cui, a causa di limiti tecnologici, l'Agente ha difficoltà nell'interpretare una frase dell'utente.

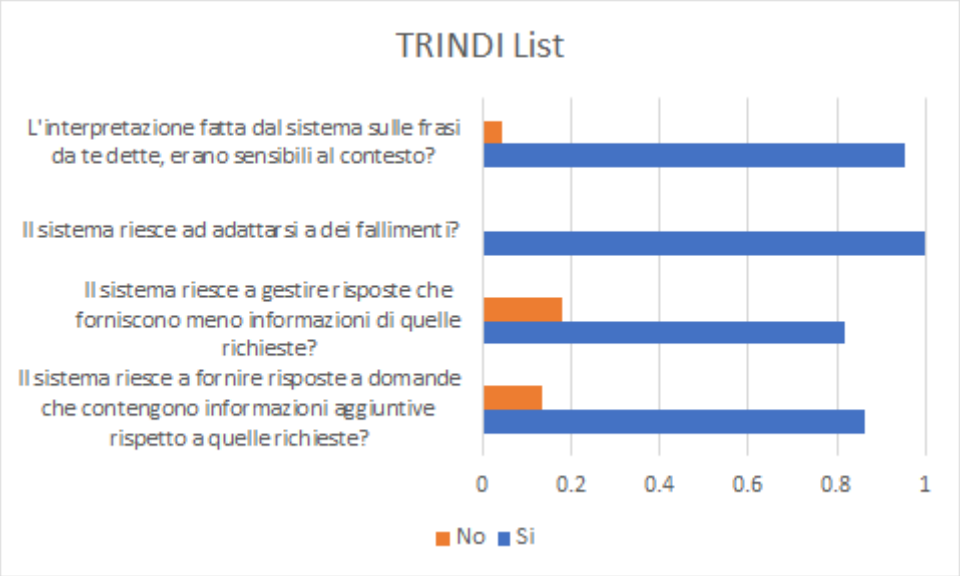


Figura 7.6: Risultati: questionario TRINDI

Di seguito vengono riportate alcune delle domande di maggiore interesse appartenenti al questionario SASSI tramite il quale è stato possibile valutare altre caratteristiche relative all'interazione vocale con l'Embodied Conversational Agent. I risultati mostrati in figura 7.7 dimostrano che:

- risulta effettivamente utile avere degli Agenti all'interno dell'applicazione;
- l'interazione vocale è efficace;
- il sistema è semplice da utilizzare;

Attraverso domande di carattere generale è stato possibile riscontrare come la presenza di Embodied Conversational Agent all'interno dell'ambiente virtuale sia utile tanto per immedesimarsi nel contesto quanto per procedere correttamente nell'esecuzione della procedura. Inoltre, ogni utente è stato in grado di completare la procedura senza l'intervento di una terza persona che spiegasse all'utente come agire o cosa dire. Questo è stato possibile grazie a due caratteristiche fondamentali dell'ECA:

- notificare quando una frase dell'utente non è stata compresa;
- fornire frasi di esempio per l'interazione vocale quando tale problema si prolunga per troppo tempo;

Di seguito (7.8) i risultati del questionario in merito a tali caratteristiche.

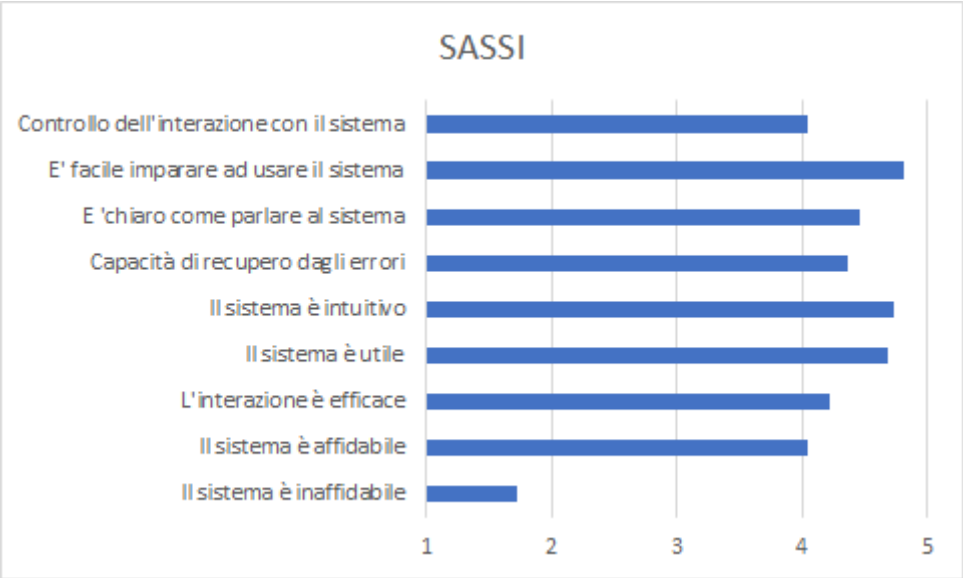


Figura 7.7: Risultati positivi del SASSI

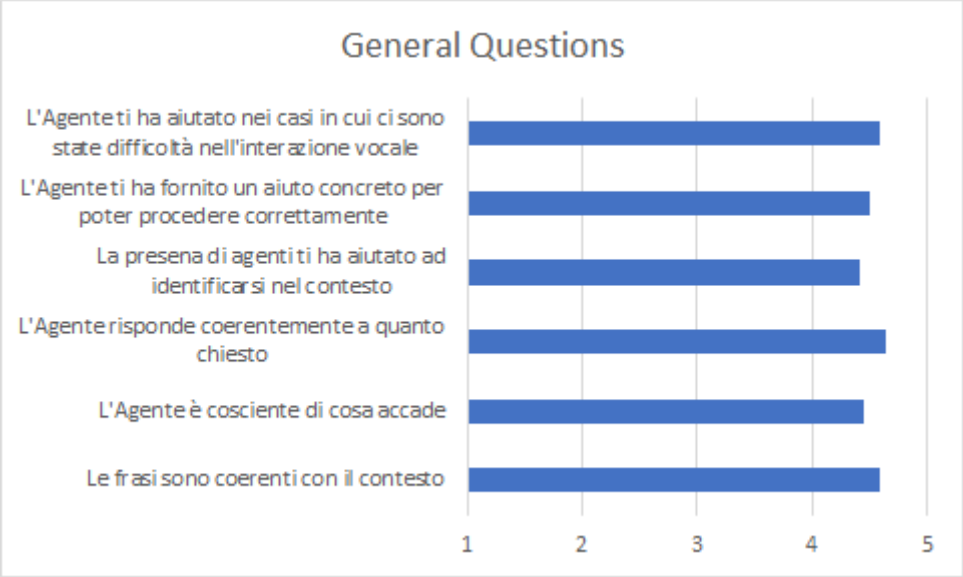


Figura 7.8: Risultati relativi a domande generali

Un ulteriore set di domande, ispirato al questionario utilizzato per la valutazione dell'Embodied Conversational Agent GRETA [8], è stato utilizzato per valutare alcune qualità degli ECA presenti nell'applicazione. I risultati sono riportati nel grafico in figura 7.9.

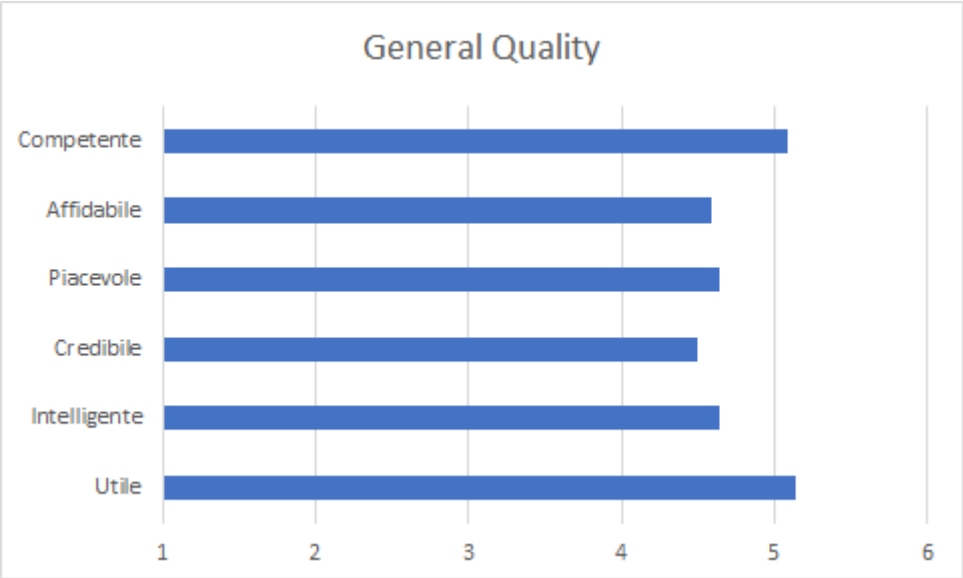


Figura 7.9: Qualità generali dell'Embodied Conversational Agent

Nella figura 7.10 viene inoltre riportato il valore ottenuto tramite il questionario SUS (84.43) all'interno della scala di valutazione prevista.

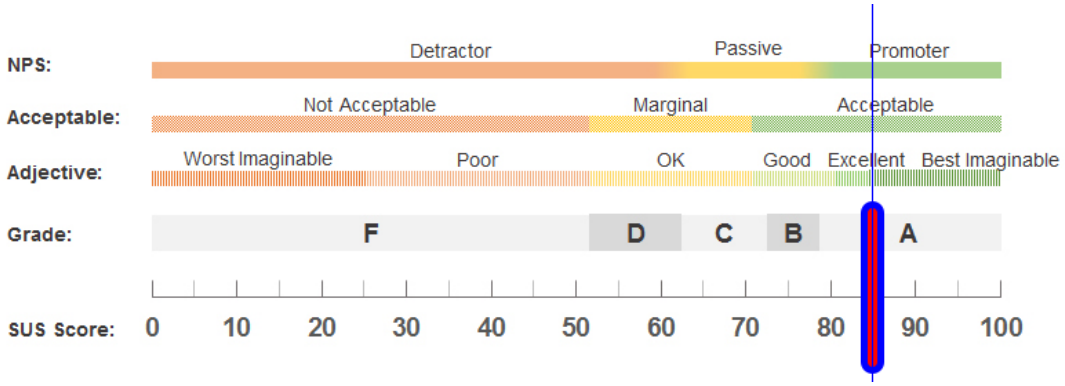


Figura 7.10: Risultato del questionario SUS riportato sulla scala di valutazione

Capitolo 8

Conclusioni

Il lavoro di Tesi può essere suddiviso in tre punti fondamentali: sviluppo del framework, sperimentazione ed analisi dei risultati. La libreria permette di integrare all'interno di un ambiente virtuale uno o più Embodied Conversational Agent aventi alcune delle caratteristiche più importanti che contraddistinguono il modo di comunicare degli esseri umani. Di particolare importanza sono aspetti come:

- percezione del contesto e di ciò che accade nell'ambiente;
- operazioni di Text To Speech per permettere all'Agente di comunicare verbalmente con l'utente;
- operazioni di Speech To Text e NLU per l'elaborazione del parlato dell'utente e conseguente estrapolazione di un concetto;
- gestione dello stato emozionale dell'Agente attraverso l'uso di un modello emozionale;
- facile integrazione di elementi visivi legati principalmente ad animazioni facciali e del corpo tramite estensione dell'architettura esistente;

Una volta terminato il design e lo sviluppo del framework, il lavoro è stato valutato sia in termini di facilità di utilizzo che in termini di efficacia del generico ECA all'interno di un'applicazione utente.
Per valutare la versatilità e semplicità d'uso della libreria è stata sviluppata un'applicazione costituita da un unico Embodied Conversational Agent, che rappresenta un paziente con il quale è possibile interagire vocalmente. L'integrazione dell'Agente nell'ambiente virtuale con le caratteristiche sopra elencate ha richiesto circa due ore di tempo.
Per quanto concerne la valutazione dell'applicazione utente, il protocollo sperimentale adottato ha messo in luce alcuni aspetti carenti dell'architettura realizzata e che costituiscono uno spunto per eventuali sviluppi futuri. Di seguito (8.1) si riportano alcuni dati di particolare interesse estrapolati dal questionario SASSI usato

durante la sperimentazione.

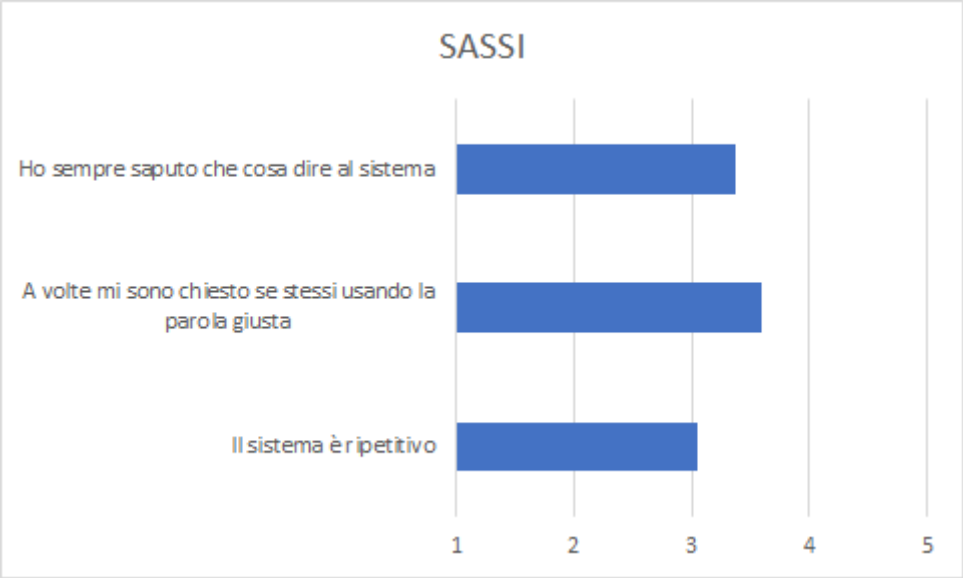


Figura 8.1: Risultati SASSI utili per sviluppi futuri

Da questi risultati emergono due aspetti che possono essere migliorati:

- attualmente per avviare l'interazione vocale è richiesto che l'utente prema un tasto. Per ridurre la ripetitività dell'operazione sarebbe produttivo implementare un sistema di tracciamento del corpo e del labiale utili per capire quando l'utente ha intenzione di parlare;
- dal momento che è stata riscontrata una certa indecisione riguardo quali parole l'utente dovesse utilizzare per poter essere compresi, sarebbe conveniente sviluppare un sistema capace di gestire un linguaggio più vasto e specifico per un determinato contesto applicativo;

L'integrazione di queste due funzionalità contribuirebbe a rendere la conversazione tra l'utente e l'Embodied Conversational Agent più naturale e simile alle modalità con cui gli esseri umani comunicano tra di loro.

Elenco delle figure

Bibliografia

[1] Mr and azure 303: Natural language understanding (luis). https://docs.microsoft.com/en-us/windows/mixed-reality/mr-azure-303.

[2] Shrdlu project. http://hci.stanford.edu/ winograd/shrdlu/.

[3] Speech synthesis markup language (ssml). https://docs.microsoft.com/en-gb/azure/cognitive-services/speech-service/speech-synthesis-markup.

[4] What is speech-to-text? https://docs.microsoft.com/en-gb/azure/cognitive-services/speech-service/speech-to-text.

[5] What is text-to-speech? https://docs.microsoft.com/en-gb/azure/cognitive-services/speech-service/text-to-speech.

[6] Javier A Bargas-Avila and Kasper Hornbæk. Old wine in new bottles or novel challenges: a critical analysis of empirical studies of user experience. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 2689–2698. ACM, 2011.

[7] Lisa Feldman Barrett. Are emotions natural kinds? *Perspectives on psychological science*, 1(1):28–58, 2006.

[8] Dianne C Berry, Laurie T Butler, and Fiorella De Rosis. Evaluating a realistic agent in an advice-giving task. *International Journal of Human-Computer Studies*, 63(3):304–327, 2005.

[9] John Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[10] Justine Cassell. More than just another pretty face: Embodied conversational interface agents. *Communications of the ACM*, 43(4):70–78, 2000.

[11] Justine Cassell, Timothy Bickmore, Mark Billinghurst, Lee Campbell, Kenny Chang, Hannes Vilhjálmsson, and Hao Yan. Embodiment in conversational interfaces: Rea. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 520–527. ACM, 1999.

[12] Ivan Gris, Adriana Camacho, and David Novick. Full-body gesture recognition for embodied conversational agents: The utep agent gesture tool. In *Conference on Gesture and Speech in Interaction*, pages 131–136, 2015.

[13] Arno Hartholt, David Traum, Stacy C. Marsella, Ari Shapiro, Giota Stratou, Anton Leuski, Louis-Philippe Morency, and Jonathan Gratch. All Together Now: Introducing the Virtual Human Toolkit. In *13th International Conference on Intelligent Virtual Agents*, Edinburgh, UK, August 2013.

[14] Kate S Hone and Robert Graham. Towards a tool for the subjective assessment of speech system interfaces (sassi). *Natural Language Engineering*, 6(3-4):287–303, 2000.

[15] A Baki Kocaballi, Liliana Laranjo, and Enrico Coiera. Measuring user experience in conversational interfaces: a comparison of six questionnaires. In *Proc. 32nd British Computer Society Human Computer Interaction Conference, Belfast, Northern Ireland*, 2018.

[16] Richard S Lazarus and Richard S Lazarus. *Emotion and adaptation*. Oxford University Press on Demand, 1991.

[17] Stacy Marsella, Jonathan Gratch, Paolo Petta, et al. Computational models of emotion. *A Blueprint for Affective Computing-A sourcebook and manual*, 11(1):21–46, 2010.

[18] Albert Mehrabian and James A Russell. *An approach to environmental psychology.* the MIT Press, 1974.

[19] Toyoaki Nishida, Atsushi Nakazawa, Yoshimasa Ohmoto, and Yasser Mohammad. *Conversational informatics*. Springer, 2014.

[20] David Novick, Iván Gris Sepulveda, Diego A Rivera, Adriana Camacho, Alex Rayon, and Mario Gutierrez. The utep agent system. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 383–384. ACM, 2015.

[21] Domenico Di Clemente Antonello Ganau Giuseppe Nusdeo Stefania Piga Silvio Romano Giancarlo Roscio Maurizio Santomauro Pier Sergio Saba, Natale Daniele Brunetti. Manuale di rianimazione cardio-polmonare di base e defibrillazione nell'adulto. pages 9–24.

[22] James A Russell. Core affect and the psychological construction of emotion. *Psychological Review*, 110(1):145–172, 2003.

[23] Ivan Gris Sepulveda and David Novick. Virtual agent interaction framework (vaif): A tool for rapid development of social agents. In *AAMAS*, 2018.

[24] David Watson and Auke Tellegen. Toward a consensual structure of mood. *Psychological bulletin*, 98(2):219, 1985.

[25] Joseph Weizenbaum et al. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.