DOI: 10.17469/O2101AISV000028

MASSIMILIANO TODISCO, FABIO POROLI, MAURO FALCONE

# Uno strumento per la prototipizzazione rapida di "dialoghi pratici"

By the term "practical dialogue" we mean what is necessary to a voice interaction with-out particular limitations regarding the naturalness and freedom of the language used, but on the fulfillment of a specific goal. The tool, or better the tools, that arise refer to two well-defined and specific phases of development of a speech dialogue: a first step having the purpose of evaluation of the tools and characterization of dialogue and voice interaction with the user through the simulation of a hypothetical and ideal system simulated in part or in whole by an operator with the technique of the Wizard of Oz; a second which allows, quickly and easily, to provide a system of voice dialog exploiting third-party technologies for both voice technologies and language processing.

## 1. *Introduzione*

L'utilizzo delle tecnologie di riconoscimento vocale (*ASR Automatic Speech Recogni-tion*) e di sintesi vocale da testo (*TTS Text to Speech Synthesis*) trova oramai un'ampia serie di possibili applicazioni in servizi e automatizzazioni. L'ambizione ultima è certamente quella di realizzare un qualcosa con cui si possa dialogare e che riesca a capirci ed aiutarci nella risoluzioni di richieste o compiti più o meno complessi. Se questo obiettivo è stato per anni mitizzato nella comune opinione, basti ricordare le scene di dialogo con il computer HAL del film di Kubrick "2001 Odissea nello spazio" del 1968, solo nel 2012 con il rilascio da parte di Apple del "iPhone 4S" con il software SIRI si è per la prima volta osato fornire a livello di generico consumatore "*an intelligent assistant that helps you get things done just by asking. Siri understands context allowing you to speak naturally when you ask it questions* (Apple, 2011)". A onore del vero dobbiamo ricordare che sofisticati sistemi ad interazione vocale erano già utilizzati con successo nell'automatizzazione di servizi telefonici ed in particolare nell'ambito del CRM (*Customer Relationship Management*) (Minker, 2011), ma questi operavano su compiti ed in scenari specifici. La barriera tecnologica e concettuale da oltrepassare era proprio quella di un 'unico assistente virtuale tuttofare' a cui ci si possa rivolgere parlando normalmente. Nell'arco di pochi anni, si è visto come i principali attori interessati stiano dotando i loro sistemi di 'assistenti' che possano interagire vocalmente con l'utente. Siri per Apple, Cortana per Microsoft, Nina per Nuance sono solo alcuni esempi, vedi figura 1, molte altre soluzioni sono fornite da società o centri di ricerca che propongono i loro 'assistenti vocali' per specifici compiti o, con grande ambizione, totalmente aperti a qualsiasi tipo di interazione.

Figura 1 - *Sistemi di dialogo vocale*


La 'lampada' che ospita questa specie di 'géni' al nostro servizio e con cui possiamo interagire con la voce è, nei casi citati, un semplice *smartphone*, ma potrebbe essere anche un *tablet*, un *personal computer* o altro dispositivo che utilizziamo comunemente per svolgere altre attività. È comunque una soluzione di tipo software che va ad aggiungersi come un servizio o una funzionalità supplementare. Tuttavia l'importanza di un 'assistente vocale' sta crescendo sempre più negli scenari tecnologi, fino a divenire esso stesso l'elemento fondante. Sparisce quindi ogni altro elemento interferente e l'assistente vocale si materializza semplicemente in entità vagamente antropomorfe (Jibo, 2014), o in una specie di neutro totem familiare (Amazon, 2014), sino a dematerializzarsi del tutto come auspicato da soluzioni ancora in fase di studio (Dirha, 2014), e che ricordano, citando nuovamente un immaginario collettivo, quanto disponibile nella serie televisiva "*Star Trek: The Next Generation*".

Auspicare che tali soluzioni potessero essere realtà, è stato possibile solo dopo che la ricerca nella sintesi vocale e nel riconoscimento del parlato avessero raggiunto prestazioni ragguardevoli e comunque sufficienti per affrontare l'ultimo è più difficile problema: il dialogo vocale. Sebbene la ricerca sul riconoscimento del parlato e sulla sintesi vocale non si siano concluse (anzi, ad esempio, il riconoscimento sta vivendo una interessante rinascita con i paradigmi di "*deep learning*" (Yu, 2015), oggi è certamente il dialogo il problema che aziende e centri di ricerca devono risolvere. Se da un lato sono chiari gli obiettivi da raggiungere, ovvero naturalezza del dialogo e ampiezza del dominio di dialogo, dall'altro non è ancora evidente la strategia da adottare: rimangono infatti sulla cresta dell'onda sistemi basati su regole e sull'analisi del testo trascritto dal segnale vocale, ma al contempo si spera che, come è successo per il riconoscimento del parlato, il problema possa essere risolto con un approccio totalmente statistico sulla base di un apprendimento di una grande mole di dati di esempio (Thomson, 2013). Quest'ultima soluzione, forse più interessante, richiede tuttavia risorse che, come abbiamo visto per il riconoscimento vocale, solo grandi aziende come Google, Amazon, etc. possono permettersi. È quindi facile profetizzare che soluzioni di questo tipo nella risoluzione del dialogo vocale potranno a breve essere fornite da

queste o da analoghe aziende, similmente a quanto, ad esempio, Google ha fatto per le tecnologie di ASR e di TTS.

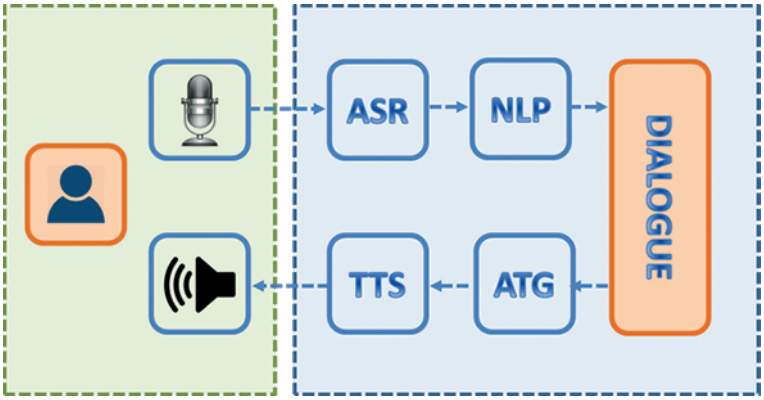
Chi voglia oggi trovare una soluzione per un sistema basato su interazione vocale, può quindi utilizzare con soddisfazione soluzioni di riconoscimento e di sintesi vocale di terzi, ma dovrà risolvere autonomamente il problema dialogo o utilizzare, qualora lo ritenga opportuno, sistemi di sviluppo specifici e vincolati a soluzioni proprietarie come nei casi già citati. Qualora invece si voglia ricorrere a strumenti terzi a supporto dello sviluppo dell'interfaccia di dialogo, il panorama delle soluzioni offerte non è così promettente (McTear, 2004), specialmente se consideriamo il caso piuttosto frequente dove si voglia realizzare, con uno sforzo ragionevole o meglio minimo, un dialogo vocale per comandare un sistema già esistente (ad esempio un televisore), o si voglia trasformare un servizio esistente in un servizio fruibile con la sola interazione vocale (ad esempio la prenotazione di un treno, di un aereo, di un albergo, etc.), o ancora si voglia sostituire un operatore di "*call center*" con un sistema automatico, ad esempio un centro della Pubblica Amministrazione che informi e guidi l'utente relativamente a nuove disposizioni e/o regolamentazioni, etc.

Precisamente in questo ambito si inquadra il sistema in oggetto: avere la possibilità di sviluppare, ottimizzare e realizzare un dialogo vocale reale per compiere uno specifico compito sfruttando, quando possibile, le tecnologie di riconoscimento, di sintesi, di elaborazione del testo riconosciuto (NLP *Natural Language Processing*) e di generazione automatica del testo per le risposte (ATG *Automatic Text Generation*) disponibili da terze parti. Questa attività è stata condotta nell'ambito del progetto 'Speaky-Acutattile' di "Industria 2015", il cui obiettivo è la realizzazione di una piattaforma a supporto di utenti 'deboli' (anziani, ipovedenti, non vedenti, disabili motori, etc.) anche attraverso l'acceso tramite dialogo vocale a servizi, applicazioni e sistemi.

## 2. *Disegno, valutazione e realizzazione del dialogo*

Premesso che lo scopo ultimo è un sistema di dialogo vocale che permetta all'utente di raggiungere il completamento di un obiettivo in uno specifico 'dominio', dove con questo termine intendiamo l'utilizzo di un servizio, il controllo di un sistema, l'accesso a delle informazioni, la guida o indirizzamento ('*steering*') dell'utente in ambienti o servizi complessi, etc., è evidente che il raggiungimento di questo obiettivo deve, o comunque può, passare attraverso una serie di specifiche fasi.

Figura 2 - *Schema a blocchi della interfaccia di dialogo vocale*




In particolare qualora ci si trovi ad affrontare il problema della realizzazione di un dialogo utilizzando risorse tecnologiche nuove, nel senso da noi mai utilizzate prima, o senza aver ancora maturato sufficiente esperienza nel disegno del dialogo, è bene avere strumenti che ci permettano il controllo e la valutazione dei singoli elementi, dialogo compreso.

Insomma ogni qual volta nel nostro complesso sistema entrino in gioco elementi che non abbiamo sperimentato prima, ad esempio un nuovo sistema di ASR, oppure più semplicemente un diverso tipo di interfaccia microfonica, e così via, oppure dove la classe di utenti è caratterizzabile da elementi ben noti, ad esempio turisti con scarsa conoscenza della lingua italiana, ed in generale ogni qual volta abbiamo delle variabili, ovvero delle parti del sistema, si veda lo schema di figura 2, che possono influenzare le prestazioni finali del nostro sistema e di cui non abbiamo sufficienti informazioni sulle prestazioni, è bene avere uno strumento per valutare al meglio ciascuna di queste variabili, le loro mutue influenze e infine l'impatto sull'intero sistema.

Per meglio chiarire questa problematica supponiamo di avere un sistema di dialogo 'ideale' e che vogliamo studiare l'impatto che un approccio all'acquisizione del segnale vocale di tipo "*push to talk*" (in questo caso l'utente deve premere un pulsante per attivare il microfono e inviare il segnale vocale al riconoscitore) rispetto a uno "*open mic*" (in questo caso invece il microfono è sempre attivo e quindi il sistema acquisisce continuamente il segnale vocale) ha sul dialogo. Oppure supponiamo di voler valutare come due o più sistemi ASR si comportano, o meglio impattano sul dialogo. Avremo quindi bisogno di configurare un sistema di dialogo dove solo alcune componenti sono effettivamente quelle che vogliamo valutare, ovvero le nostre variabili, mentre tutto il resto deve essere 'ideale', teoricamente ad errore zero (o comunque non variabile), e questo può essere simulato solo attraverso l'ausilio di un operatore umano che svolge, in modalità nascosta all'utente, questa funzionalità ideale a errore zero. Questa tecnica è ben nota nella valutazione dei sistemi, vocali e non, ed è comunemente detta del 'Mago di Oz' (WOz *Wizard of Oz*) dall'omonimo romanzo di Frank Baum del 1900 dove i personaggi del romanzo, raggiunto il paese

di Oz, si trovano a parlare con una 'entità astratta' che credono impersonale, ma che in realtà non era altro che uno uomo nascosto dietro una tenda.

Il primo software realizzato è Speaky-WOz, un sistema che permette ad un operatore remoto di controllare il dialogo pratico di un sistema dove ogni singolo elemento può essere controllato o sostituito dall'operatore. Tipicamente l'operatore sostituisce il sistema di riconoscimento e comprensione del parlato in modo che non vi siano errori di comprensione, mentre la sintesi vocale è operata con messaggi preregistrati o generati online, eventualmente anche con l'ausilio di un avatar per rendere l'interazione più gradevole. Il dialogo invece è codificato secondo una strategia di "*frame-box filler*" dove ogni compito è scomposto in singoli "*frame*" elementari, ciascuno dei quali si considera risolto solo dopo che una sequenza di "*box*" sono state correttamente riempite. Il dialogo vocale, si veda un esempio in figura 3, opera in maniera tale che l'utente possa fornire tutte le informazioni per risolvere un "*frame*", quindi decide come procedere, saltando al successivo o ad altro "frame", ed eventualmente operando delle azioni attuative reali o simulate se lo stato del dialogo lo richiede. È questa una strategia procedurale ben nota, e che possiamo esemplificare come riportato di seguito.

Figura 3 - *Un esempio di interazione vocale con il sistema*

```
COMPITO: Prenotazione voli
FRAME01: [CITTA_PARTENZA] [CITTA_ARRIVO]

U: salve vorrei prenotare un volo {per [Milano Malpensa]} per la
    prossima settimana
S: da che città vuole partire?
U: ah... si... ehm parto da casa... {da [Roma]}
...
```

Una volta disegnato il dialogo e valutato il sistema con Speaky-WOz, possiamo operare le necessarie modifiche al dialogo e scelte tecnologiche, e quindi valutare nuovamente l'intero insieme in modo iterativo fino ad arrivare ad una soluzione soddisfacente del nostro sistema, ovvero del disegno del dialogo e delle tecnologie da utilizzare. A questo punto dobbiamo mettere in atto i frutti della nostra ricerca sperimentale e realizzare un reale sistema di dialogo che, sulla base delle prestazioni e delle interazioni studiate e messe a punto in precedenza e utilizzando quelle tecnologie già valutate nel contesto di nostro interesse, sia in grado di realizzare un sistema ad interazione vocale reale e non più simulato tramite l'ausilio di un operatore, ovvero tramite il Mago di Oz. Per fare questo abbiamo realizzato un secondo software Speaky-RD (*Speaky-Real Dialogue*), che lavora con la medesima strategia di "*frame-box filler*" per quanto riguarda la codifica e gestione del dialogo, e che utilizza risorse di ASR, di TTS e di NLP terze, disponibili in rete o che è possibile portare localmente sul nostro sistema.

Figura 4 - *Schema a blocchi della interfaccia di dialogo vocale "distribuita in rete"*


Chiaramente qualora si voglia fisicamente attuare il risultato di un dialogo, ad esempio il controllo di un sistema di domotica o di intrattenimento, sarà necessario sviluppare del software specifico aggiuntivo. Vedremo successivamente come procedere in tal senso e come sia stato possibile realizzare il controllo di un sistema "*media-center*" con uno sforzo minimo. Il precedente schema generale è stato da noi realizzato utilizzando risorse distribuite in rete e quindi meglio schematizzabile secondo la figura 4.

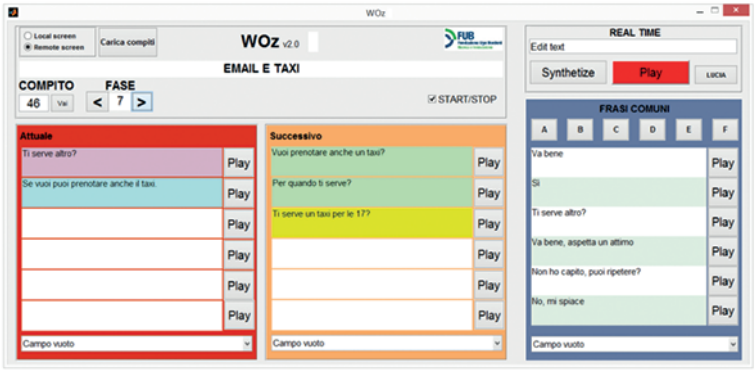
## 2.1 Speaky-WOz: simulare per valutare

Figura 5 - *Un esempio dell'interfaccia lato utente del sistema (domotica)*

Il software Speaky-WOz è sviluppato in linguaggio MatLab ed è in pratica un 'cruscotto di lavoro' dal quale l'operatore controlla il dialogo vocale in maniera autonoma, ma seguendo rigidamente quanto definito negli schemi di dialogo del tipo "*frame-box filler*".

Il cruscotto comanda remotamente il terminale, un semplice personal computer dotato di video e microfono, con cui interagisce l'utente. È sufficiente quindi una connessione di rete di media/buona qualità, e il WOz può gestire l'interazione vocale anche con utenti situati in città diverse da quella in cui si trova (Poroli, 2014).

Figura 6 - *Un esempio della di interfaccia lato "WOz" del sistema*




Sul terminale di interazione con l'utente è possibile inviare sia immagini, sia video, sia messaggi audio. Nel nostro caso, tipicamente, si inviava una immagine video a descrizione dello scenario di lavoro e un filmato relativo alla interazione vocale con un avatar. Nella figura 5 è mostrato un esempio del terminale che l'utente si trova davanti nel caso di una simulazione di casa domotica, uno degli scenari da noi utilizzati. Il WOz invece si trova a lavorare utilizzando un cruscotto come quello riportato nella figura 6.

In un file Excel sono codificati tutti i passaggi della interazione ovvero del dialogo. Possono raggrupparsi in un unico file un numero qualsiasi di 'compiti', ovvero schemi di dialogo con l'utente per raggiungere uno specifico obiettivo più o meno semplice, dalla prenotazione di un viaggio o di un ristorante, all'inserimento di un farmaco nei promemoria, ad un consulto medico e successiva prenotazione di visita, e così via. Ogni compito è quindi diviso in 'sotto compiti' o "*frame*" di più semplice gestione dove attraverso il dialogo è necessario riempire le "*box*" ovvero risolvere le variabili, tipicamente da due a quattro, dello specifico "*frame*". Sul cruscotto è evidenziato, al centro dello schermo, l'argomento del compito, in alto a destra il numero del compito ed il numero del "*frame*" in cui ci troviamo. Il WOz può avanzare o in genere navigare tra i "*frame*" con un semplice click. A schermo compariranno le possibili frasi che il WOz può, secondo gli schemi di dialogo prestabiliti nel file Excel, presentare all'utente nel "*frame*" corrente e in quello successivo in modo che il WOz abbia visione di come evolverà il dialogo. In funzione di quanto pronunciato dell'utente il WOz può scegliere la risposta premendo il bottone "*play*" corrispondente. A questo punto sul terminale dell'utente saranno trasmesse le immagini e l'audio corrispondente a quella scelta. Poiché ovviamente non è possibile prevedere e codificare tutte le possibili risposte da dare all'utente, in alcuni casi è ne-

cessario fornire delle risposte al di fuori dello schema prestabilito. A tal fine il WOz ha a disposizione, a sinistra del cruscotto, una sezione con cui interagire in tempo reale ovvero dove può gestire situazioni di errore, di attesa e altro o, se necessario, realizzare *on line* risposte specifiche da dare all'utente in modo da essere in grado di gestire qualsiasi tipo di situazione. Nella nostra esperienza (Poroli, 2013) l'utilizzo di questa sezione del cruscotto si è rilevata piuttosto rara, ma molto efficace quando necessaria. Il sistema registra in un file di testo lo storico di tutti i passaggi, ovvero dei tempi e delle scelte che il WOz effettua. Ovviamente sarà necessario non solo definire nel file Excel tutti i compiti e i "*frame*" relativi, ma anche le immagini e i le risposte audio (o audio-video nel caso di utilizzo di avatar) per ciascuna delle possibili scelte che il WOz può operare. Il frutto di questo lavoro è un sistema che simula un'interazione vocale di alto livello, come mostrato ad esempio nel video (CSP, 2013), e che permette la raccolta di dati relativi a interazioni vocali in diverse città italiane da un unico punto di controllo come descritto in (Poroli, 2013). Una volta effettuate le simulazioni e le relative valutazioni, sarà quindi possibile avere maggior confidenza nel disegnare i dialoghi per una reale interfaccia vocale per un sistema o un servizio.

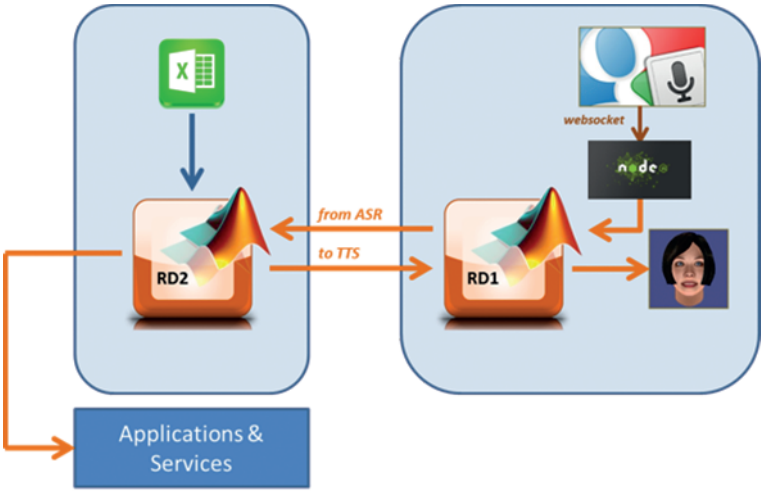
### 2.2 Speaky-RD: per la prototipizzazione rapida

Il software Speaky-RD è composto, come vedremo, da una serie di moduli, i principali e più complessi in MatLab. Il suo obiettivo è emulare il sistema "Speaky-WOz" senza che vi sia l'operatore umano, e in tempo reale. La codifica dei dialoghi, delle azioni da intraprendere e quanto altro è sempre contenuta in file Excel, il che rende di semplice e immediata realizzazione l'implementazione dei dialoghi pratici.

## 3. *L'infrastruttura e le scelte operate*

L'infrastruttura del sistema deve ora includere l'utilizzo di un sistema ASR e di un sistema di TTS che comunicano con il modulo di gestione del dialogo, che a sua volta può utilizzare moduli di NLP e attuare fisicamente dei comandi su dispositivi esterni o alternativamente simularli. Tutto questo implica che i vari moduli comunichino tra loro su rete secondo protocolli ben definiti. Nel nostro caso il sistema di ASR prescelto è quello offerto da Google, mentre il sistema di sintesi vocale è quello Microsoft nella declinazione della voce femminile di "Silvia", le stringhe riconosciute dal sistema ASR vengono inviate al modulo di dialogo attraverso un '*socket TCP*', analogamente il sistema di dialogo comunica con il modulo di sintesi inviando il testo da pronunciare attraverso un '*socket TCP'.* I sistemi ASR e TTS sono stati raggruppati in un unico modulo software Speaky-RD1 da noi sviluppato e configurabile, se necessario, con altri sistemi di riconoscimento e sintesi. Tutti gli altri moduli sono raggruppati in Speaky-RD2 che acquisisce le informazioni per gestire il dialogo e i comandi attuativi da i file Excel.

Figura 7 - *Schema a blocchi del sistema di dialogo vocale*




Nel dettaglio, RD1 apre una pagina html nel browser Chrome che è in grado di acquisire il segnale audio dal microfono del computer e di operare la trascrizione di quanto pronunciato. Il microfono può rimanere sempre aperto (*default*) o aprirsi solo quando il sistema di sintesi non stia parlando.

Questa ultima modalità di funzionamento da un lato facilita il riconoscimento del parlato, ma al contrario è mal accettata dagli utenti in quanto non naturale nei normali dialoghi parlati. Per motivi di sicurezza la pagina html può comunicare solo attraverso la nuova tecnologia di "*websocket*", pertanto si è dovuto realizzare un livello intermedio di comunicazione attraverso un semplice programma in linguaggio "*node js*" che legge i messaggi da un protocollo e li rispedisce nell'altro al sistema, come delineato in figura 7. Aver sviluppato il modulo RD1 permette di sostituire il sistema di ASR o di TTS facilmente senza dover operare alcuna modifiche nei file Excel o negli altri moduli. Riassumendo, possiamo dire che i moduli RD1 e RD2 si scambiano in RD1 invia a RD2, le stringhe relative a quanto riconosciuto dal sistema di ASR, mentre nel verso contrario, RD2 invia a RD1, le stringhe relative ai messaggi di testo da far pronunciare al sistema di TTS o all'avatar. In aggiunta il sistema RD2 potrà inviare stringhe a dispositivi per attuare comandi o fruire di servizi (e.g. accendere una luce, ascoltare un programma radio o televisivo, etc.).

### 3.1 Speaky-RD1: gestione del "ASR" e del "TTS"

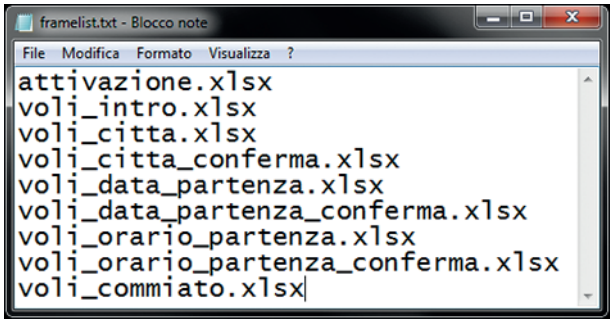
Questa parte del sistema si occupa della gestione del "*front-end*" audio che si presenta all'utente e della integrazione dei sistemi di riconoscimento del parlato e di sintesi. Il riconoscimento del parlato è operato attraverso una pagina html che sfrutta il riconoscitore di Google per parlato continuo. Una volta eseguita la pagina html, tutto quanto acquisito dal microfono viene spedito al sistema di riconoscimento che inizia a elaborare il segnale e propone una serie di ipotesi di riconoscimento che evolvono dinamicamente sino a quando il segnale audio non presenti una pausa

sufficiente ed il sistema di riconoscimento abbia risolto la trascrizione con determinato grado di affidabilità. Solo a questo punto il segnale audio viene 'consolidato' e la relativa trascrizione viene comunicata attraverso "*websocket*" all'applicativo "*node js*" che a sua volta lo comunica al pro-gramma MatLab, che lo invierà al modulo di gestione del dialogo. Sebbene il sistema di riconoscimento in questione funzioni molto bene, abbiamo riscontrato delle situazioni che necessitano una particola attenzione. Un problema che è subito risultato evidente consiste nel fatto che il riconoscitore può attendere anche diversi secondi quando deve consolidare segnali audio di brevissima durata, come ad esempio per la frase "si" che gli utenti utilizzano spesso per confermare. In questi casi infatti il sistema riconosce immediatamente il segnale, ma tarda nel consolidamento o addirittura fallisce nel consolidamento. Per eliminare questo problema abbiamo modificato il sorgente html derivato da quelli dei dimostrativi di Google, e abbiamo intercettato il testo riconosciuto ma non consolidato, e se questo corrisponde ad un determinato messaggio (al messaggio "si" nel nostro caso) il testo viene forzatamente marcato come consolidato simulando quanto avrebbe dovuto fare il sistema. Strategie diverse e più sofisticate per la forzatura del consolidamento del testo, ad esempio basate sul tempo di attesa o di non operabilità del riconoscitore, potrebbero anche essere operate, ma per ora questa semplice operazione è stata sufficiente a risolvere la maggior parte delle situazioni. Il secondo problema incontrato è anche legato al consolidamento e si manifesta quando vi sia un forte rumore di fondo o quando l'utente continui a parlare anche dopo aver comunicato con il sistema, ad esempio con dei commenti tra sé e sé oppure parlando con altre persone presenti. Il sistema di riconoscimento non è ovviamente in grado di capire se l'utente stia parlando al sistema o con altri, né tantomeno è in grado di capire se il segnale audio che arriva al microfono debba essere analizzato dal riconoscitore o no (come nel caso di rumore di fondo). Essendo il sistema nel nostro caso sempre in ascolto, tutto quanto arriva al microfono viene mandato al riconoscitore che cercherà di trascrivere fino a che non si riconosca una pausa di silenzio opportuna. Se per i motivi sopra elencati tale pausa di silenzio non si verifica il sistema tarderà a consolidare quanto pronunciato dall'utente. Purtroppo una soluzione tecnica a questo tipo di problema non è di banale realizzazione, e quindi si è cercato di evitare il problema, piuttosto che risolverlo, utilizzando il sistema solo in ambienti chiusi di ufficio e con un microfono di prossimità o comunque mai posizionato a distanze maggiori di qualche decina di centimetri dal parlatore. Per quanto riguarda il sistema di sintesi da testo questo può considerarsi, tipicamente, come una risorsa interna al computer che ospita RD1. Nel nostro caso si sono esaminate diverse sintesi da testo (Microsoft, Loquendo, etc.) e l'avatar 'Lucia' sviluppato dal CNR (Cosi, 2003) ed utilizzato in Speaky-WOz. Altre risorse di sintesi da testo, ad esempio la sintesi da testo offerta da Google, possono facilmente essere integrate in Speaky-RD1. Come abbiamo detto, un sistema da sintesi da testo che permetta di conoscere l'istante il cui il sistema ha terminato la comunicazione può risultare utile nel caso di un più stretto controllo dell'acquisizione audio, a svantaggio ovviamente di maggior naturalezza del dialogo.

### 3.2 Speaky-RD2: gestione del dialogo e attuazione

Speaky-RD2 è il modulo che controlla il dialogo con l'utente e che permette l'attuazione o la simulazione di semplici operazioni che realizzano quanto richiesto dall'utente. Il testo generato dal sistema di ASR arriva a RD2 che come prima cosa lo normalizza eliminando eventuali caratteri spuri e di punteggiatura, riportandolo tutto a minuscolo ed altro, secondo una serie di regole che possono facilmente essere modificate. A questo punto una prima elaborazione del testo consiste in una ulteriore e più forte normalizzazione riportando ciascuna parola riconosciuta al lemma corrispondente. Si crea quindi, con l'ausilio di un 'lemmatizzatore' (Zanchetta, 2005), un vettore di termini dello stesso ordine di quello relativo al testo trascritto ma i cui elementi corrispondono ai lemmi delle corrispondenti parole. Sebbene analisi ed elaborazioni del testo più complesse (Pianta, 2008) possano essere operate per facilitare la comprensione del testo, la riduzione in lemmi è risultata sufficiente, come vedremo, alla gestione di dialoghi pratici. Il sistema legge da un semplice file di testo la lista dei file Excel che contengono i dialoghi pratici, o meglio le singole parti di uno o più dialoghi pratici. In assenza di un comando, ad esempio richiesto dall'utente, che forza il dialogo a saltare in un preciso stato, il programma esegue linearmente la lista dei file Excel, che quindi si dovrà aprire sempre con un file di attivazione, ad esempio su 'parola chiave', e di introduzione. Ad esempio il file lista per un dialogo pratico per un ipotetico compito legato all'utilizzo di un servizio di prenotazioni voli, potrà essere il seguente, riportato in figura 8.

Figura 8 - *Un esempio di file di comando per il sistema*




L'attivazione fa sì che il sistema sia operativo solo dopo che l'utente abbia pronunciato una parola chiave. Nel nostro caso l'utente era invitato ad attivare il sistema con il comando 'ciao speaky', ma la parola chiave utilizzata era, per ovvi motivi, solamente 'ciao', in quanto la parola 'speaky' non è contenuta in un vocabolario standard dell'italiano. Nel file 'attivazione' possono essere definite una serie di parole o frasi chiave diverse per attivare il sistema. Il file 'voli_intro', analogamente a 'voli_commiato', non prevede l'interpretazione del parlato pronunciato dall'utente, ma comunica semplicemente dei contenuti di saluto, istruzione, etc. a seconda dei casi, per lo specifico compito. Gli altri file Excel contengono i dialoghi per acquisire le informazioni su città di partenza e di arrivo, giorno e mese di partenza, orario di partenza, e

per ciascuna di queste informazioni è possibile avere conferma con un ulteriore file Excel. Nel caso di risposta positiva il dialogo prosegue, tipicamente con file successivo ma potrebbe puntare ad altro se necessario, mentre nel caso di risposta negativa il dialogo prosegue puntando, tipicamente, al file Excel che richiede nuovamente i dati non confermati dall'utente ma anche in questo caso potrebbe puntare ad altro se necessario. Ciascun file Excel è composto da quattro fogli: "*grammar*", "*tts*", "*jump*" e "*action*", che devono essere opportunamente riempiti a seconda delle funzionalità che si vogliono realizzare.

Nel foglio "*grammar*", figura 9, sono definite le varie "*box*" che devono essere riempite per completare correttamente la parte di dialogo pratico corrispondente al "*frame*", ovvero al file Excel, in elaborazione. Per ciascuna "*box*" avremo due colonne: una prima corrispondente al testo che deve trovarsi nella stringa riconosciuta dal trascrittore, nella seconda il contenuto o comando da associare a quanto riconosciuto ed individuato nella grammatica.

Figura 9 - *Un esempio di file Excel per la definizione di una grammatica*

| S- | [BOX1] grammar | [BOX1] command | [BOX2] grammar | [BOX2] command |
|---|---|---|---|---|
| DEFAULT VALUE | | | | |
| | roma | roma | roma | roma |
| | da roma | roma | a roma | roma |
| | aosta | aosta | per roma | roma |
| | da aosta | aosta | aosta | aosta |
| | | | a aosta | aosta |
| | | | ad aosta | aosta |
| | | | per aosta | aosta |

Figura 10 - *Un esempio di file Excel per la definizione delle risposte per l'utente*

| [BOX1] status | [BOX2] status | Answer1 | Answer2 |
|---|---|---|---|
| | | | |
| 0 | 0 | dimmi città di partenza e città di arrivo | per iniziare dimmi da che città vuoi partire |
| 0 | 1 | la destinazione è £2£ , ora dimmi da dove vuoi partire | per il tuo volo a £2£ , ora dimmi da che città vuoi partire |
| 1 | 0 | la città di partenza è £1£, ora dimmi dove vuoi andare | hai scelto un volo con partenza da £1£, ora scegli la città di destinazione |
| 1 | 1 | mi confermi partenza da £1£ e destinazione £2£ ? | partenza da £1£ ed arrivo a £2£, è corretto? |

Se vi sono 'n *box*' nella grammatica, lo stato del dialogo potrà essere in 2^n condizioni diverse a seconda che ciascuna singola *box* sia stata risolta o no. Queste condizioni sono enumerate nel foglio "*tts*" dove per ciascuna situazione, da zero a (2^n)-1, sono riportate le frasi che il sistema deve rivolgere all'utente per completare il dialogo pratico, si veda figura 10. Per ciascuno stato è possibile avere un numero arbitrario di possibili risposte: il sistema ne sceglierà casualmente una tra quelle disponibili. Al fine di rendere il dialogo più naturale è possibile inserire nel testo da sintetizzare il contenuto delle *box* già risolte utilizzando sia il valore della cella di grammatica, sia quella di comando.

Il foglio "*jump*", si veda figura 11, interpreta un particolare contenuto dei comandi associati alle grammatiche.

Figura 11 - *Un esempio di file Excel per la gestione dei flussi di dialogo condizionati*

| N | GRAMMAR A | COMMAND A |
|---|---|---|
| DEFAULT VALUE | | |
| | si | >1 |
| | bene | >1 |
| | va bene | >1 |
| | confermo | >1 |
| | ok | >1 |
| | no | >2 |
| | sbagliato | >2 |
| | errato | >2 |

| JUMP |
|---|
| DEFAULT VALUE |
| voli_data_partenza.xlsx |
| voli_citta.xlsx |

Quando nel foglio delle grammatiche la casella associata alla parola riconosciuta, ovvero la casella di '*command*', è del tipo numerico, preceduto dal simbolo '>' questo viene interpretato dal sistema come un salto condizionato e più specificatamente al file di grammatiche Excel contenuto nel foglio "*jump*" ed indicizzato dal valore associato alla casella del foglio "*grammar*" riconosciuta. In questo caso si possono, ad esempio, predisporre nel dialogo dei momenti di verifica e/o di conferma che fanno procedere il dialogo a seconda della risposta dell'utente, come nell'esempio in cui se si ha una risposta affermativa (">1") il dialogo prosegue con la richiesta della data di partenza, altrimenti nel caso di risposta negativa (">2") si torna a chiedere la città di partenza e di destinazione.

Un formalismo del tutto analogo vale per il foglio "*action*", si veda figura 12.

Figura 12 - *Un esempio di file Excel per la attuazione di semplici comandi*

| ACTION |
|---|
| |
| sys:RAI1.vbs |
| sys:RAI2.vbs |
| sys:RAI3.vbs |
| sys:RAI-RADIO1.vbs |
| .... |

Sempre sulla base dei valori numerici associati nel riconoscimento e codificati nel foglio "*grammar*" oltre a saltare secondo quanto definito nel foglio "*jump*" appena descritto, il sistema esegue il comando corrispondente del foglio "*action*". Ad esempio con il suffisso "*sys:*" il sistema interpreta la stringa seguente come un comando di sistema, e nel nostro caso, predisponendo specifici file di comando in *visual basic* per il controllo di un *media-center*, è stato possibile attuare delle scelte operate vocalmente dall'utente direttamente sul dispositivo televisivo.

Utilizzando opportunamente le funzionalità sopra descritte e limitatamente alla realizzazione di dialoghi pratici, è stato possibile realizzare interfacce vocali per dei test in cui l'utente, dopo aver attivato vocalmente il sistema con una parola chiave, entrava in un menù che permetteva: la prenotazione di un biglietto aereo su tratta nazionale, servizio esclusivamente vocale tipo IVR (*Interactive Voice Response*); il controllo di una casa domotica, simulata su schermo; il controllo reale di un *media-center*, dove l'utente poteva accedere ai canali tv e radio, a contenuti multimediali audio e video. Come abbiamo visto un ruolo centrale è svolto dal foglio "*grammar*" a cui è demandata l'analisi della stringa di testo riconosciuta del sistema ASR e opportunamente elaborata da un semplice sistema di NLP, un lemmatizzatore nel nostro caso. La ricerca dei termini contenuti nelle varie colonne di '*grammar*' procede dapprima cercando nella stringa le caselle che contengono il più alto numero di termini nell'intero insieme della grammatica (ad esempio si cerca prima la casella "per roma fiumicino", poi "per roma" e infine "roma" nel caso ovviamente tutti e tre le caselle siano presenti nella grammatica). Il secondo termine di ricerca è ovviamente dalla prima colonna di grammatica a sinistra all'ultima a destra. Una volta individuata una corrispondenza, questa viene catalogata opportunamente e il testo della stringa viene epurato del testo relativo. In realtà è possibile definire una strategia specifica su come operare sulla stringa riconosciuta immettendo un codice nella prima casella del foglio "*grammar*" (ad esempio il codice "S-" nella figura 9). In particolare è possibile comandare il sistema in modo che i termini delle grammatiche riconosciute siano cancellati o no dalla stringa, che la stringa sia ereditata da un foglio Excel al successivo, che la stringa al contrario sia azzerata prima di procedere con la nuova acquisizione e analisi e altro ancora come dallo schema di figura 13.

Figura 13 - *Liste delle possibili modalità di controllo del testo riconosciuto*

| codice | azione operata sulla stringa dell'ARS |
|---|---|
| N | Non elabora la stringa in ingresso e non ne salva una nuova in uscita |
| D | Non elabora la stringa in ingresso e la cancella in uscita |
| S+ | Non elabora la stringa in ingresso e la salva completa in uscita |
| S- | Non elabora la stringa in ingresso e la salva priva delle *keyword* in uscita |
| P | Elabora la stringa in ingresso e non ne salva una nuova in uscita |
| PD | Elabora la stringa in ingresso e la cancella in uscita |
| PS+ | Elabora la stringa in ingresso e la salva completa in uscita |
| PS- | Elabora la stringa in ingresso e la salva priva delle *keyword* in uscita |

Ovviamente sarà cura del '*dialog designer*' sfruttare le potenzialità e le funzionalità offerte dai fogli Excel appena descritti, al fine di progettare al meglio i dialoghi pratici di interesse. Considerando che quella qui presentata è la prima versione del sistema da noi realizzato, questo si è mostrato già piuttosto efficace e ci ha permesso agilmente di realizzare una serie di diversi dimostrativi sviluppati in breve tempo e facilmente. Futuri studi saranno mirati alla formalizzazione delle attuali funzionalità, all'integrazione di altre funzioni per supportare il disegno dei dialoghi o per sfruttare sistemi di NLP più sofisticati.

## 4. *Conclusioni*

Si sono sviluppati due sistemi a supporto della realizzazione e valutazione di interazioni vocali in linguaggio naturale attraverso dialoghi pratici. I software sviluppati permetto l'utilizzo e la facile integrazione di risorse terze per il riconoscimento vocale e per la sintesi vocale, in modo che chiunque possa personalizzare il sistema a proprio piacimento. Anche l'infrastruttura scelta per la realizzazione del sistema è completamente aperta e può pertanto adattarsi a qualsiasi configurazione: da un sistema su un singolo computer, a un sistema distribuito su rete locale o pubblica. Il primo sistema si basa sulla tecnica del Mago di Oz e permette ad un operatore di controllare il dialogo in modo che si possano valutare le strategie più opportune e caratterizzare i comportamenti dell'utenza durante l'interazione vocale. Il secondo sistema permette la realizzazione di reali interfacce vocali in tempo reale, basate su dialogo naturale per gestire a voce dei servizi o per controllare, sempre con il dialogo vocale, sistemi o dispositivi. La codifica delle parole chiave, del dialogo, e della eventuale attuazione di comandi sui dispositivi è predisposta in dei normali file Excel. Il primo sistema è stato utilizzato per raccogliere un corpus di dati sul territorio italiano su un campione di circa 80 parlatori che dovevano svolgere un numero limitato tra quaranta compiti diversi. Il secondo sistema è stato utilizzato su un campione di circa 20 soggetti che dovevano operare compiti quali la prenotazione di un volo, il controllo di un media center, la gestione di una casa domotica.

## *Bibliografia*

Amazon (2014). *Introducing Amazon Echo*. http://www.amazon.com/oc/echo.

Apple (2011). Apple Press Info. *Apple Launches iPhone 4S, iOS 5 & iCloud*. https://www.apple.com/pr/library/2011/10/04Apple-Launches-iPhone-4S-iOS-5-iCloud.html.

Cosi, P., Fusaro, A. & Tisato, G. (2003). LUCIA a new italian talking-head based on a modified Cohen-Massaro's labial coarticulation model. In *Proceedings of Eurospeech 2003*, Geneve, Switzerland.

CSP (2013). *Progetto Speaky Acutattile*. https://www.youtube.com/watch?v=8sIOorZ7w7c.

Dirha (2014). *Distant-speech Interaction for Robust Home Applications*. https://dirha.fbk.eu/sites/dirha.fbk.eu/files/docs/Dirha_Brochure_def.pdf

Jibo (2014). *Welcome to the Jibo Blog on Social Robots*. https://www.jibo.com.

McTear M. (2004). *Spoken dialogue technology: toward the conversational user interface*. London: Springer-Verlag.

Minker, W. et al. (2011). *Spoken dialogue systems technology and design*. London: Springer Science Business Media, Springer-Verlag.

Pianta, E., Girardi, C. & Zanoli, R. (2008). *The TextPro tool suite*. In *Proceedings of LREC, 6th edition of the Language Resources and Evaluation Conference*, 28-30 May 2008, Marrakech, Morocco.

Poroli, F., Paoloni, A. & Todisco, M. (2013). Gestione degli errori in un corpus di dialogo uomo-macchina: strategie di riformulazione. In *Atti del X convegno nazionale AISV*, Torino, gennaio 22-24, 2014.

Poroli, F. et al. (2013a). Prime indagini su un corpus di dialogo uomo-macchina raccolto nell'ambito del progetto Speaky Acutattile. In *Atti del IX convegno nazionale AISV*, Venezia, gennaio 11-13, 2013.

Poroli, F. et al. (2014). Il corpus Speaky. In *Proceedings of the First Italian Conference on Computational Linguistics CLiC-it 2014*, 9-11 December, Pisa.

Thomson, B. (2013). *Statistical Methods for Spoken Dialogue Management.* London: Springer Theses, Springer-Verlag.

Yu, D., Deng, L. (2015). *Automatic speech recognition a deep learning approach*. London: Signals and Communication Technology, Springer-Verlag.

Zanchetta, E., Baroni, M. (2005). *Morph-it! A free corpus-based morphological resource for the Italian language*, proceedings of Corpus Linguistics 2005, University of Birmingham, Birmingham, UK.