Secure Login

**Politecnico di Bari**

**I Facoltà di Ingegneria**

**Ingegneria Informatica Magistrale**

**Sicurezza Informatica**

**A.A. 2010/2011**

Di Leo Carlo

Salatino Angelo Antonio

Report del progetto tema d’anno per la disciplina Sicurezza Informatica basato sul riconoscimento vocale.

# Sommario

[Sommario 1](#_Toc342485392)

[1. Introduzione 3](#_Toc342485393)

[2. Estrazione delle feature 3](#_Toc342485394)

[2.1. Acquisizione 3](#_Toc342485395)

[2.2. Pre-enfasi 3](#_Toc342485396)

[2.3. Framing e Windowing 4](#_Toc342485397)

[2.4. Trasformata di Fourier 5](#_Toc342485398)

[2.5. Filtraggio alle frequenze Mel 6](#_Toc342485399)

[2.6. Trasformata Coseno Discreta 6](#_Toc342485400)

[2.7. Settings 7](#_Toc342485401)

[2.8. Clustering 7](#_Toc342485402)

[2.8.1. Clustering Model-Based 7](#_Toc342485403)

[2.8.2. Gaussian Mixture Model 8](#_Toc342485404)

[2.8.3. Expectation Maximization 10](#_Toc342485405)

[2.9. Decisore 11](#_Toc342485406)

[3. Risultati 12](#_Toc342485407)

[3.1. Settaggi 12](#_Toc342485408)

[3.2. Metriche di valutazione 12](#_Toc342485409)

[3.3. Risultati con wekaGMM 12](#_Toc342485410)

[3.4. Risultati con GMM 12](#_Toc342485411)

[4. Implementazione 14](#_Toc342485412)

[4.1. Installer Tool 14](#_Toc342485413)

[4.1.1. Visualizzazione strutturale 17](#_Toc342485414)

[4.2. Administration Secure Tool 18](#_Toc342485415)

[4.2.1. Add User 18](#_Toc342485416)

[4.2.2. Delete User 19](#_Toc342485417)

[4.2.3. Update Model 20](#_Toc342485418)

[4.2.4. Login User 21](#_Toc342485419)

[4.2.5. MODELLO del PARLATORE 21](#_Toc342485420)

[4.2.6. Visione strutturale 23](#_Toc342485421)

[4.3. Pannello Login/Logout – User Sight 25](#_Toc342485422)

[5. Classi o librerie off-the-shelf 26](#_Toc342485423)

[5.1. WEKA 26](#_Toc342485424)

[5.2. WavFile 26](#_Toc342485425)

[6. UML 28](#_Toc342485426)

[6.1. Diagramma dei casi d’uso 28](#_Toc342485427)

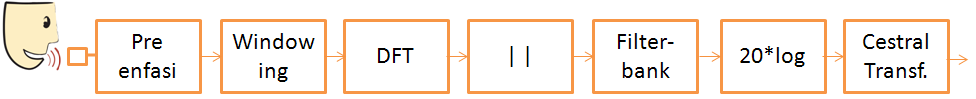
[7. Reference 29](#_Toc342485428)

# Introduzione

Con la tecnologia attualmente a disposizione, le richieste di ausilio tecnologico per il supporto alla vita quotidiana aumentano sempre di più. Con queste ultime nasce anche l’esigenza di alcuni sistemi di sicurezza robusti ed efficienti per tutelare sia degli oggetti, sia le persone. *Secure Login* nasce come progetto accademico, nonché tema d’anno del modulo di Sicurezza Informatica, dove si è proposti di creare un software che garantisca l’accesso sicuro ad un sistema attraverso il riconoscimento vocale. Uno scenario tipico che meglio rappresenta l’idea di questo progetto è descritto dall’accesso controllato ad un azienda dove i dipendenti devono registrare i loro ingressi e le loro uscite attraverso le proprie carte magnetiche. Dai capitoli successivi si cercherà di descrivere al meglio quali sono i dettagli di implementazione del sistema, come l’integrazione di un modulo per l’estrazione delle features e un modulo per la clusterizzazione delle features.

# Estrazione delle feature

Partendo dall’acquisizione del segnale audio, l’estrazione delle feature consiste dei seguenti step:



## Acquisizione

Il primo passo nel processing è convertire il segnale analogico (il segnale elettrico nel microfono) in un segnale digitale x[n] dove n è un indice del tempo. L’analisi dello spettro del segnale audio mostra come tutta l’energia risieda nella banda sino a 4 kHz.

Settaggi del microfono:

* 8000 Hz
* 16-bit, Signed
* Little Endian
* Mono Channel
* Uncompressed PCM

## Pre-enfasi

Il segnale viene filtrato con un filtro passa-alto al fine di rendere disponibile le informazioni presenti alle alte frequenze.

Viene utilizzato un filtro passa-alto del primo ordine. Nel dominio del tempo, con un segnale audio x[n] avremo la seguente equazione:

y[n]=x[n]-αx[n-1]

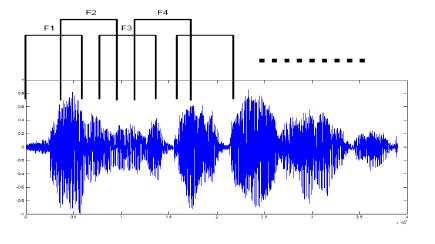
con α compreso tra 0.9 e 1.

Amplificando le alte frequenze rendiamo disponibili le informazioni portate da tali frequenze al fine di migliorare l’accuratezza del modello da creare.

## Framing e Windowing

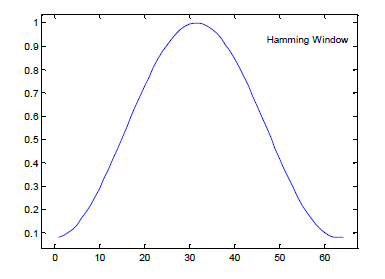
Essendo la voce un segnale non-stazionario, le sue proprietà statistiche non sono costanti nel tempo. Tuttavia faremo l’assunzione che il segnale sia stazionario in un intervallo di tempo limitato, quindi divideremo il segnale i piccoli frame in modo poter estrarre le caratteristiche spettrali del segnale.

Ogni frame sarà largo 32ms con un overlap del 50% tra un frame e l’altro (256 campioni per frame e un overlap di 128 campioni).

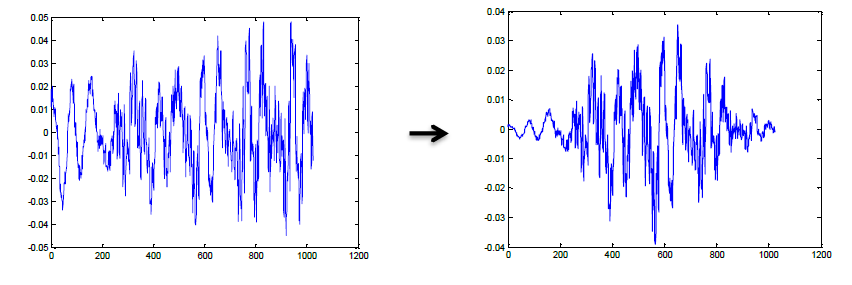


La finestra rettangolare può causare distorsioni allo spettro del segnale, a causa del brusco taglio del segnale ai suoi alti. Una funzione finestra buona ha un lobo principale stretto e bassi valori di lobi laterali nelle loro funzioni di trasferimento riducendo i valori del segnale ai bordi della finestra, evitando discontinuità. La funzione finestra più comunemente usata nella elaborazione del segnale vocale è la finestra di Hamming definito come segue:

1≤n≤N



L’estrazione del frame avviene moltiplicando il valore del segnale al tempo n con il valore della finestra al tempo n.



## Trasformata di Fourier

In questa fase viene utilizzata la trasformata di Fourier del segnale finestrato per estrarre lo spettro di ciascun frame. La trasformata di Fourier prende in ingresso i campioni di ciascun frame di lunghezza N e restituisce in output un frame, anch’esso di lunghezza N contenente un numero complesso per ogni banda di frequenza (N bande) rappresentante l’ampiezza e la fase della componente alla specifica frequenza nel segnale originale:

k=0,1,2,…,N-1

Lo spettro risulta essere simmetrico rispetto alla frequenza di *folding* e solo la metà di questi punti sono realmente utili.

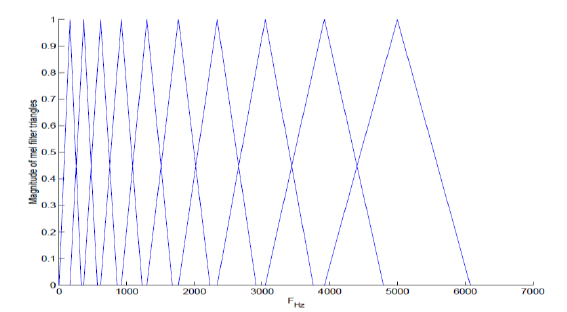
## Filtraggio alle frequenze Mel

Lo spettro presenta molte fluttuazioni mentre l’inviluppo spettrale, privo di queste fluttuazioni, risulta essere una buona approssimazione dello spettro, oltre a ridurre le dimensioni dei frame.

Per ottenere l’inviluppo spettrale abbiamo moltiplicato lo spettro con un banco di filtri.

Un banco di filtri è costituito da una serie di filtri passa-banda moltiplicati uno per uno con lo spettro in modo da averne l’inviluppo spettrale per ogni data banda. Un banco di filtri è definito dalla forma dei filtri e dalla posizione delle frequenze(di sinistra, di destra e centrale). I filtri possono essere triangolari, o avere altre forme, e possono avere varie disposizioni nella scala delle frequenze. In questo progetto è stata usata la scala delle frequenze Bark/Mel. La posizione delle frequenze centrali di questi filtri è data dalla formula:

La scala Mel/Bark è basata sulla scala non lineare delle frequenze dell’orecchio umano:



## Trasformata Coseno Discreta

Per ottenere i coefficienti cepstrali, si applica la trasformata coseno discreta ai vettori spettrali calcolati.

L’n-esimo coefficiente cepstrale in un frame sarà dato dalla seguente formula:

Dove K è il numero di coefficienti log-spettrali in un frame mentre L è il numero di coefficienti cepstrali che si vuole ottenere.

## Settings

Ogni frame sarà rappresentato da :

* 12 coefficienti cepstrali statici
* 1 coefficienti che rappresentano *e*

## Clustering

In questa fase i frame precedentemente estratti dal segnale audio vengono analizzati e viene lanciato un algoritmo di clustering al fine di trovare una struttura ordinata che descriva le caratteristiche del parlatore.

### Clustering Model-Based

Si è scelto di utilizzare un approccio Model-Based che non soltanto opera la clusterizzazione dei dati ma fornisce modelli complessi in grado di trovare correlazioni e dipendenze degli attributi.

In pratica, ogni cluster viene rappresentato da una distribuzione, per esempio una Distribuzione Normale. L’intero dataset viene quindi modellato da un insieme di queste distribuzioni.

Se i parametri delle distribuzioni sono ben stimati, un modello siffatto copre bene i dati e ogni componente presenti un picco.

L’algoritmo utilizzato per l’addestramento del modello è l’algoritmo *Expectation-Maximization* (EM).

L’idea base è quella di assumere di conoscere i parametri delle distribuzioni e quindi calcolare la probabilità che ciascun dato appartenga a ciascuna componente. A questo punto si riadattano i parametri ai dati con peso pari alla probabilità che essi appartengano alla distribuzione e in modo tale da massimizzare la likelihood dell’intero dataset dei dati rispetto al modello.

EM è un algoritmo iterativo che dovrebbe convergere verso i valori reali dei parametri.

Nel progetto è stato scelto per ogni Cluster una distribuzione gaussiana, è stato settato a 32 il numero di cluster, per evitare il problema dell’overfitting, e si è settato il numero di iterazioni dell’algoritmo EM a 10 come compromesso tra una buon addestramento e tempi di addestramento contenuti.

### Gaussian Mixture Model

La distribuzione Gaussiana, anche conosciuta come distribuzione normale, è una funzione con curva a campana. Una distribuzione Gaussiana è una funzione parametrizzata da una media (mu) e da una varianza (sigma quadro). La seguente formula rappresenta la distribuzione Gaussiana:

In statistica, un modello misto è un modello probabilistico che assume i dati sottostanti di appartenere a una distribuzione mista. In una distribuzione mista, la sua densità è una combinazione convessa (una combinazione lineare nella quale tutti i coefficienti o pesi sommati danno uno) di altre funzioni di densità di probabilità:

Ogni funzione di densità pi(x) che è combinata per dare la densità mista p(x) sono chiamati componenti misti, i pesi w1, w2, …, wn associati ad ogni componente sono chiamati pesi misti o coefficienti misti.

Una mistura di distribuzioni Gaussiane (GMM) è una densità di probabilità parametrica comunemente usata come modello di dati continui e nella caratteristiche biometriche meglio note nei sistemi di riconoscimento del parlatore grazie alla loro capacità di rappresentare una vasta gamma di distribuzioni campione.

Come la K-Means, la GMM può essere considerato come un modello di clustering o di apprendimento non supervisionato. GMM è basato sulle tecniche di cluster, dove l'intero set di dati sperimentali è basato su una mistura di Gaussiane. Non proprio come le K-Means, le GMM sono abili a costruire soft clustering boundaries, per esempio punti nello spazio che appartengono a una qualsiasi classe con una probabilità data.

In una mistura di distribuzioni Gaussiane, la sua densità è una combinazione convessa (una combinazione lineare nella quale tutti i coefficienti o pesi sommati danno uno) di altre funzioni di densità di probabilità Gaussiana:



Matematicamente, una GMM è la somma pesata delle M densità delle componenti Gaussiane dati dall'equazione:

dove

* è un vettore di dimensione k
* è il peso della m-esima gausssiana. Il vettore w dei pesi soddisfa il vincolo .
* m=1,2,…,M sono le densità di probabilità. Ciascuna componente rappresenta una funzione gaussiana di dimensione k:
* è il vettore media di lunghezza k della m-esima gaussiana
* è la matrice di covarianza quadrata di dimensione k della m-esima gaussiana

Cosi il modello di misture Gaussiane completo è parametrizzato dai pesi misti, vettore delle medie e matrice delle covarianze per tutti i componenti di densità. I parametri sono rappresentati dalla notazione,

Questi parametri sono stimati in una sezione di training. Per un sistema di riconoscimento del parlatore, ogni parlatore è rappresentato da una GMM ed è riferito da un suo modello (gamma).

GMM è ampiamente utilizzato nella modellazione del parlatore e nella classificazione grazie ai suoi due vantaggi: come primo, ogni singolo componente Gaussiano in un GMM dipendente dal parlatore è interpretato per rappresentare alcune classi come tratti vocali dipendenti dal parlatore che sono utili per identificare il modello del parlatore stesso. La voce di un parlatore può essere caratterizzata da un set di classi che rappresentano alcuni eventi fonetici ampi come le vocali, nasali, fricative. Queste classi acustiche riflettono alcune configurazione di tratti vocali dipendenti dal parlatore che sono utili per caratterizzare l'identità del parlatore.

Queste forme spettrali dell'i-esima classe acustica può essere rappresentata da una media dell'i-esimo componente densità, la varianza della forma spettrale media può essere rappresentata dalla matrice di covarianza. Queste classi acustiche sono nascoste prima del training. Secondariamente, la densità mista Gaussiana fornisce una approssimazione smussata della distribuzione dei campioni a lungo termine dell'apprendimento delle caratteristiche vocali di un dato parlatore. Il modello unimodale Gaussiano del parlatore rappresenta la distribuzione delle caratteristiche del parlatore da un vettore delle medie e una matrice delle covarianze e il modello VQ che rappresenta la distribuzione del parlatore attraverso un set discreto di template caratteristici. GMM si attua come un ibrido tra questi due modelli appena descritti usando una set discreto di funzioni gaussiane, ognuna delle quali con la loro propria media e matrice di covarianza per acconsentire meglio la funzionalità di modellazione.

### Expectation Maximization

L'approccio di Massima-Verosimiglianza è una tecnica molto popolare per stimare i parametri del modello. Esso fornisce buoni risultati quando l'insieme di dati di training è largo. La verosimiglianza del modello p(X|λ) indica quanto verosimile è che i parametri λ siano i veri parametri del modello presunto che ha generato i dati X. Si noti che λ rappresenta la variabile indipendente mentre il set di dati X è noto e fisso. Il corretto insieme di parametri può essere trovato in modo che la p(X| λj) del modello sia più alta rispetto alla verosimiglianza di qualsiasi altro modello. Ovviamente il massimo che può essere fatto è trovare un insieme di parametri che massimizza la funzione di verosimiglianza. Per questa ragione, è utile avere un ampio set di dati in modo da coprire l'insieme delle caratteristiche acustiche di una persona. Formalmente, i parametri λ stimati con l'approccio della Massima Verosimiglianza sono ottenuti dal set di dati X={xt : t=1,2,...,T} massimizzando la verosimiglianza del modello rispetto ai parametri:

La modellazione delle caratteristiche acustiche assume che vettori di caratteristiche **X** di un certo parlatore siano statisticamente indipendenti. La verosimiglianza congiunta dei vettori di caratteristiche diventa quindi il prodotto delle loro funzioni di verosimiglianza:

Per semplicità di calcoli, è spesso utile lavorare con la log-verosimiglianza:

Sfortunatamente, questa espressione è una funzione non lineare del parametro λ e non è possibile una diretta massimizzazione. Tuttavia tali parametri possono essere ottenuti iterativamente utilizzando un caso speciale dell'algoritmo di Expectation-Maximization:

L’algoritmo Expectation-Maximization si compone di due fasi: una prima fase in cui si calcola la probabilità che ciascun punto appartenga a ciascuna componente gaussiana e una seconda parte in cui si sfruttano i dati così ottenuti per aggiornare i parametri del modello in modo da massimizzare l’aderenza del modello ai dati.

Di seguito viene presentato l’algoritmo:

* Passo 1: vengono inizializzati i pesi a 1/M, la matrici di covarianza a 1 e i vettori delle medie ad un valore random.
* Passo 2 (Estimation Step): Calcolo della probabilità che ogni punto appartenga alla componente :

* Passo 3 (Maximization Step): sfruttando la matrice ottenuta al passo 2 vengono aggiornati i parametri del modello:

* Ritornare al passo 2 ripetendo l’intero processo per un numero *it* fissato di iterazioni.

## Decisore

Il decisore è una funzione integrata all’interno del modello che deve decidere se la registrazione di una voce di un parlatore non noto appartiene o meno ad un dato utente.

Nel progetto si è scelto di confrontare il punteggio ottenuto fornito dal modello del presunto parlatore con il massimo punteggio ottenuto da una schiera di modelli, addestrati su un set di parlatori scelti in modo tale da coprire tutto il dominio delle caratteristiche vocali.

E’ stato quindi scritto un applicativo che permetta di addestrare un set di N modelli i cui parlatori dovranno essere scelti accuratamente in modo da coprire lo spazio delle caratteristiche acustiche.

La fase di decisione prevede che se il punteggio ottenuto dal modello del presunto parlatore è maggiore del punteggio fornito dall’UBM la voce viene assegnata al presunto parlatore.

# Risultati

E’ stato testato il motore per il riconoscimento del parlatore su un database di 30 individui le cui registrazioni sono state gentilmente concesse da una radio locale.

## Settaggi

Sono state utilizzate solo registrazioni con frequenza di campionamento 8000 Hz in modo da rendere conforme le prove con il sistema descritto in precedenza.

Per ogni parlatore vengono estratti due file di cui uno di durata di circa 30s verrà utilizzato per l’addestramento dei modelli, l’altro, di durata di circa 10s verrà utilizzato per il test dei modelli.

## Metriche di valutazione

Per l’analisi delle prestazioni del sistema verranno valutati, in percentuale, il numero di falsi negativi e il numero di falsi positivi.

Un falso negativo viene definito come il caso in cui la voce registrata di un utente sconosciuto appartiene realmente ad un presunto parlatore, ma il sistema non l’ha riconosciuto come tale.

Un falso positivo viene definito come il caso in cui una voce registrata di un utente sconosciuto non appartiene ad un presunto parlatore , ma il sistema ha etichettato la voce come appartenente al presunto parlatore.

Per i test sono state utilizzate due classi per la creazione del modello e il calcolo della verosimiglianza: una è wekaGMM che si avvale della libreria Weka, l’altra, GMM creata da noi, che implementa da principio il GMM.

## Risultati con wekaGMM

Con wekaGMM è stata ottenuta la seguente matrice di confusione:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Valori Predetti | | |
| Valori  Effettivi |  | p’ | n’ |
| p | TP = 96,6% | FP = 3,3% |
| n | FN = 4,4% | TN = 95,6% |

## Risultati con GMM

Con GMM si è ottenuta la seguente matrice di confusione:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Valori Predetti | | |
| Valori  Effettivi |  | p’ | n’ |
| p | TP = 76,67% | FP = 23,33% |
| n | FN = 4,4% | TN = 95,6% |

Come si nota dai risultati, l’utilizzo di una libreria esterna porta dei risultati nettamente migliori per quanto riguarda i falsi negati, a fronte però di un leggero aumento del tempo di calcolo dovuto alle scritture su disco necessarie per la comunicazione tre l’applicativo e le classi di weka.

# Implementazione

## Installer Tool

Il tool di installazione è un vero e proprio wizard che consente il setting di alcuni parametri utili per il funzionamento del sistema di riconoscimento vocale.

Eseguendolo si accede al pannello mostrato in seguito:



Per la completezza è stato aggiunta l'opzione di disinstallazione del sistema, il quale provvede a rimuovere tutti i dati creati con il tempo (Cliccando su Check e Uninstall).

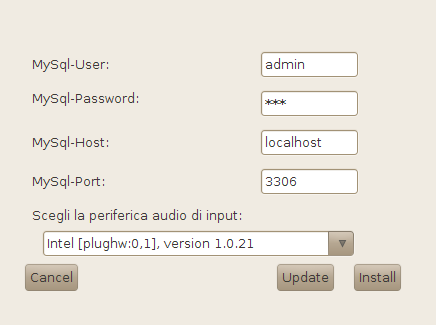
Cliccando sul tasto Continue si procede con il wizard, dove appare il seguente pannello:



Tra i parametri richiesti vi sono Username e Password per accedere al DBMS installato sulla macchina, hostname che indica la locazione del database (nel caso in cui è sito sulla stessa macchina è localhost) e la porta per la connessione (default 3306).

Un ulteriore parametro utile al funzionamento è la periferica di input da utilizzare per l'acquisizione dell'audio per il calcolo del modello del parlatore. La lista delle periferiche di input viene stilata all'avvio del tool il quale legge le periferiche attive e le stampa in una combobox affinché sia selezionabile la periferica desiderata.

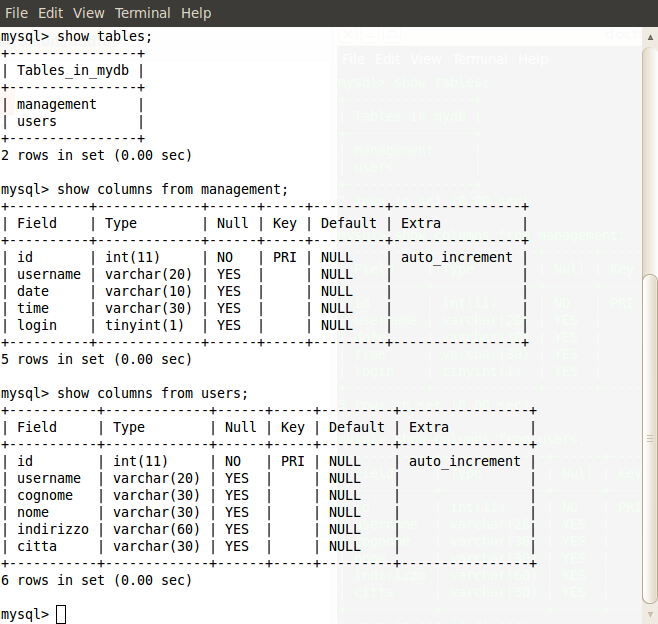
Es:



Procedendo con il tasto Install vengono create le tabelle all'interno del database e viene creato un file di configurazione (configuration.bin). Il pulsante Update serve ad aggiornare i dati di accesso al database qualora vi siano modifiche al database, o cambi di periferiche. Con il tasto Update si evita di creare nuovamente le tabelle, alla quale il DBMS restituirebbe errore, mentre modifica il file configuration.bin.

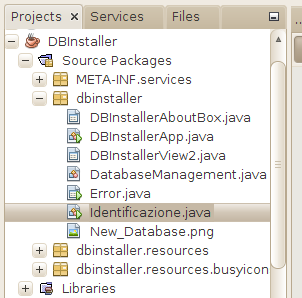
La disinstallazione effettua un reverse dell'installazione in quanto cancella il file di configurazione e le tabelle presenti nel database.

Utilizzando una macchina sulla quale è installata una distro Ubuntu di Linux si usera il terminale per la controprova dell'effettiva creazione delle tabelle nel database:



Dalla figura appena mostrata si apprende che l'installazione è andata a buon fine, tra l'altro confermata in precedenza dai messaggidi notifica che il wizard lancia. Al termine dell'installazione avremo che nel nostro DBMS ci sarà il database “mydb”, creato opportunamente e all'interno del quale saranno presenti le tabelle management e users le quali rispettano tutti i canoni di creazione di una tabella (es. presenza di chiavi primarie, chiavi esterne in caso di dipendenza, ecc.) e rispettivamente tengono traccia dei login/logout e degli utenti.

### Visualizzazione strutturale



La figura mostra il tree del progetto DBInstaller che racchiude il programma appena descritto, il package dbinstaller è il fulcro di tutto in quanto contiene tutte le classi che lo implementano.

Tra le classi presenti molte vengono create automaticamente per via dell'interfaccia grafica, quelle fondamentali sono DBInstallerView2.java che è il motore di questo progetto in quanto gestisce tutta l'applicazione creando query SQL per il database, lettura dei paramentri immessi dall'utente per l'installazione del sistema. Un'altra classe fondamentale è Identificazione.java che si preoccupa di scandagliare tutte le periferiche audio della macchina in cui viene eseguito il tool per listarle e dare la possibilità all'utente di selezionarle per l'installazione. La classe Error.java è stata creata come classe ad-hoc per mostrare a video le notifiche siano esse messaggi di errore, siano essi messaggi di operazioni avvenute con successo. La classe DatabaseManagement.java si comporta come bridge che connette l'applicazione al database la quale interpreta le query create e le trasforma in SQL per il DBMS. Le altre classi sono state create dal tool Swing per le interfacce grafiche, le quali non hanno alcun compito ai fini esecutivi dell'applicazione.

## Administration Secure Tool

Il tool Administration Secure è una suite completa, che fornisce tutt'una serie di servizi utili alla gestione del sistema. Tra le operazioni più importanti ci sono l'aggiunta, la cancellazione e la modifica di un utente, ovvero il fulcro del sistema. Un operazione aggiunta è il log user.

Quest'ultima operazione è una last resort qualora il sistema di riconoscimento vocale dorebbe creare problemi e dovrebbe essere cosi mantenuto. Essa permettere di effettuare il login e il logout manuale da parte di un amministratore senza il riconoscimento vocale.



Le tre operazioni fondamentali citate precedentemente sono simili alle classiche operazioni di aggiunta, modifica e cancellazione di utenti classico, presente in ogni sistema di registrazione, alla quale viene aggiunta la possibilità di aggiungere un modello del parlatore. Quest'ultima feature viene calcolata dinamicamente attraverso la registrazione di un file audio o l'acquisizione di un file audio come citato in letteratura.

### Add User



Come già accennato, il form di iscrizione di un nuovo utente è molto simile a quello presente su tutti i sistemi di registrazione degli utenti. Si noti che il tasto Salva è temporaneamente opaco e non cliccabile, perché al click di tale pulsante viene generata una query che aggiunge un utente nel database. Per questioni di sicurezza e ai fini dell'integrità delle informazioni all'interno del database, il tasto Salva diventerà cliccabile non appena il modello sarà stato calcolato e i campi siano tutti opportunamente compilati. Per evitare doppioni sugli username, il sistema è capace, al volo, di capire se l'username è stato già utilizzato, obbligando a sceglierne un altro.

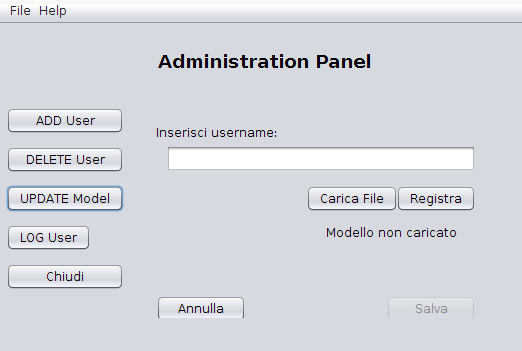
### Delete User



Questo form è di costituzione semplice in quanto vi è un'unica textbox dove inserire l'username dell'utente da cancellare e cliccare successivamente su Cancella Utente. Al click di quest'ultimo pulsante viene generata una query SQL che permette di liberare il database di una sola tupla. Se il campo testuale è vuoto e viene premuto Cancella, viene lanciata una notifica nella quale si avverte che il campo username è vuoto.

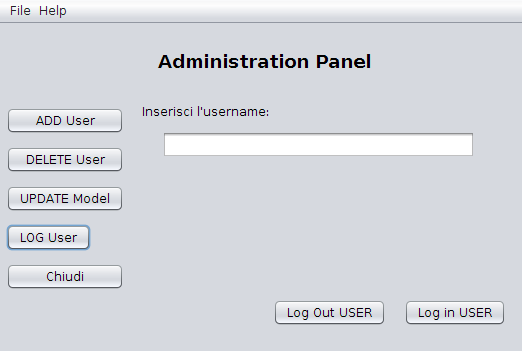
Nel caso in cui l'username da cancellare è stato già cancellato e/o non è presente nel database, una notifica molto simile alla precedente avverte che l'operazione non è andata a buon fine in quanto l'username non è stato trovato.

### Update Model



La suddetta sezione del pannello permette di aggiornare il modello del parlatore e quindi dell'utente che deve essere riconosciuto dal sistema. Questa è un opportunità che nasce dalla necessità di un possibile non riconoscimento. L'operazione di update del modello ha senso in quanto la voce dell'utente potrebbe non essere la stessa negli anni. Si pensi alla voce di un fumatore, la quale è soggetta a una frequente mutazione in funzione di quanto fuma. Il cambiamento della voce da parte di un utente e quindi la possibilità di non essere riconosciuto dal sistema ha portato alla realizzazione del pannello Log User descritto in seguito.

### Login User



Nell'occasione in cui il modello dell'utente calcolato in fase di login che si vuole loggare al sistema e il modello dello stesso utente presente nel database hanno dei missmatch, l'utente non potrà essere loggato ai fini della sicurezza. Quando l'amministratore del sistema, in possesso di questo tool avrà riconosciuto l'effettiva corrispondenza tra l'utente loggante e l'utente presente nel database, potrà espletare la funzione del riconoscitore ovvero loggarlo. Di li in poi sarà sicuro che il sistema non sarà in grado di riconoscerlo e perciò si dovrà provvedere ad un update del modello.

L'operazione di logging demandata all'amministratore può essere effettuata sia in fase di login sia in fase di logout, per via della progettazione del sistema.

### MODELLO del PARLATORE

Si è già discusso in molte occasioni che per ogni utente vi sarà un modello utile alla fase di logging. Attualmente è previsto un supporto minimo all'operazione perché l'unico formato riconosciuto è il \*.wav. Il sistema sià nella fase di ADD USER e di UPDATE MODEL prevede comunque due modalità, le quali si basano sulla:

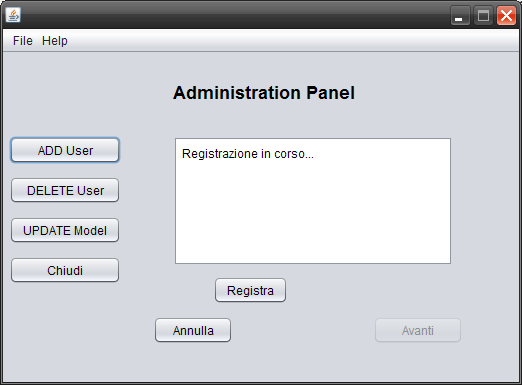
- registrazione

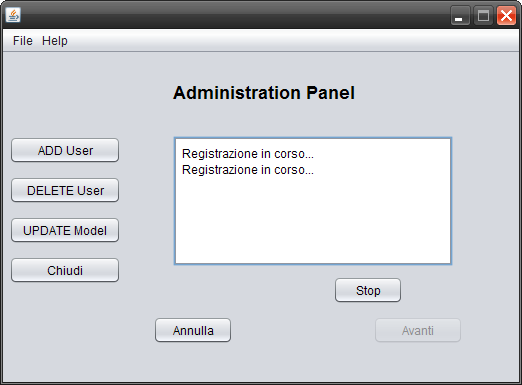
- caricamento

Quindi nei pannelli appena citati vi si troveranno sempre due tasti che consentono le due modalità attuabili. Il caricamento comporta l'apertura di un selettore, il quale fornisce un explorer per spostarsi all'interno delle directory fino a raggiungere il file \*.wav (condizione obbligatoria in questa versione del software) per poi calcolare il modello.

Se non vi è alcun file audio su cui calcolare il modello, è stato implementato un pannello che consente di registrare attraverso la periferica audio settata durante la fase di installazione la voce dell'utente. Al termine della registrazione verrà creato un file temp.wav e su di esso calcolato il modello del parlatore.

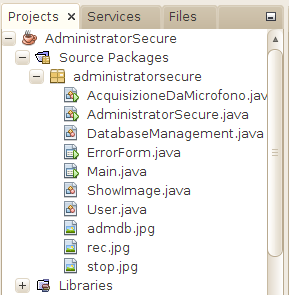
Ecco un esempio di pannello di registrazione nel quale vi è il tasto record sul quale premere per avviare la registrazione. Ad atto compiuto il tasto di record scompare dando spazio al tasto di stop per consentire successivamente il calcolo del modello.





Si noti che anche qui il tasto avanti è opaco, esso diventerà cliccabile non appena il file wav sarà salvato, cosicché sarà possibile tornare al pannello precedente e calcolare il modello.

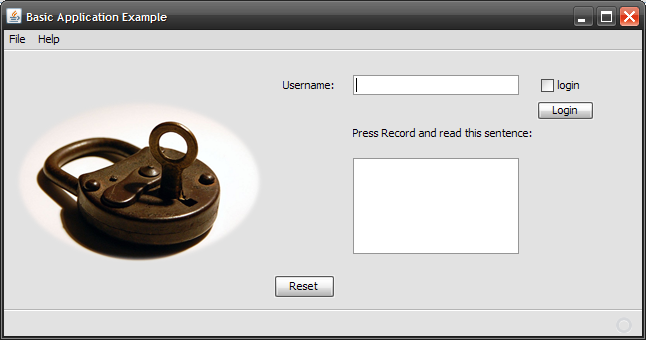
### Visione strutturale



La precedente figura mostra il tree del progetto che conserva l'applicatore per l'amministratore e administrationsecure è il package fondamentale. La classe centrale che gestisce tutto l'applicativo è Main.java, provvedendo a lanciare le notifiche, eccezione e query quando necessario. La classe ErrorForm e DatabaseManagement sono le stesse usate per il wizard di installazione. La classe AcquisizioneDaMicrofono.java permette di interfacciarsi alla periferica audio leggere lo stream e incanalarlo in un unico contenitore quale il file temp.wav. Le altre classi sono create appositamente per l'interfaccia grafica non influendo con le altre classi.

## Pannello Login/Logout – User Sight

Questo modulo del sistema si interfaccerà con i dipendenti dell’azienda e permetterà loro di segnalare l’inizio o la fine della loro giornata lavorativa.



L’utente deve inserire il suo username di registrazione nel campo ID e spuntare il flag login se intende fare il login al sistema.

Cliccando quindi sul tasto Login verrà caricata una frase random che l’utente potrà leggere.

Al fine del login non è importante che l’utente legga proprio quella frase ma che parli in maniera costante per almeno 5s: tanto è infatti la durata minima che deve avere la registrazione per ottenere dei validi risultati nel riconoscimento.

Cliccando sui tasti “Record” e “Stop” si effettua la registrazione della voce che verrà processata.

Se il decisore riterrà che il parlatore è realmente chi dichiara di essere, il database verrà aggiornato aggiungendo un record contenente la tupla (username, giorno, mese, anno, ora, 8flagLogin) ove flagLogin è posto pari a 1 se è stato effettuato il login al sistema, viceversa è posto pari a 0.

Questo modulo del programma si avvale della classe “Cepstral.java” per il pre-processing del file audio e l’estrazione delle feature che il modello del parlatore andrà ad analizzare.

Per la creazione del modello è stata utilizzata la classe EM delle librerie Weka ma anche una classe GMM creata ex-novo.

# Classi o librerie off-the-shelf

## WEKA



Weka è un software scritto in Java che offre una serie di tool per il Machine Learning. Esso è l’acronimo di Waikato Environment for Knowledge Analysis ed è stato sviluppato dall’Università di Waikato in Nuova Zelanda. Il set completo di Weka contiene una vasta collezione di strumenti per l’analisi dei dati e modellazione predittiva, tutti con una GUI per facilitare l’accesso a queste funzionalità.

Weka supporta diverse operazioni di data mining, come data processing, clustering, classification, regression, visualization e selezione delle feature. Tutte le tecniche Weka sono predicate sull’assunzione che i dati siano disponibile come un singolo file o relazione, dove ogni dato è descritto da un numero o un attributo. Weka fornisce l’accesso ai database SQL usando le opportune classi java e può processare i risultati provenienti dal database.

Oltre al software in se, Weka viene distribuito sotto forma di classi Java, importabili nel proprio codice java che si sta implementando.

Le classi utilizzate per i fini del progetto sono:

1. Cluster Evaluation

E’ stata usata per farci restituire la somiglianza tra le feature estratte dal parlatore non conosciuto e il modello del parlatore noto.

1. Data Source

Interfaccia a qualcosa che è capace di essere interfaccia di ingresso.

1. Instance

Classe che gestisce le istanze. Fornisce un approccio object-oriented elegante perché è molto veloce.

1. EM ExpectationMaximization

La classe Em assegna una distribuzione di probabilità per ogni istanza che indica la probabilità di ogni appartenenza ai cluster. La classe EM può decidere quanti cluster creare usando la tecnica del cross validation, oppure è possibile specificarlo a priori.

## WavFile

Classe java ad-hoc prelevata dal web che permette l’interazione con i file audio aventi estensione dot wav. I membri della classe principalmente utilizzati sono quelli di lettura e scrittura, perché questo tipo di file fungevano da contenitori della voce dell’utente al quale verrà successivamente calcolato il modello.

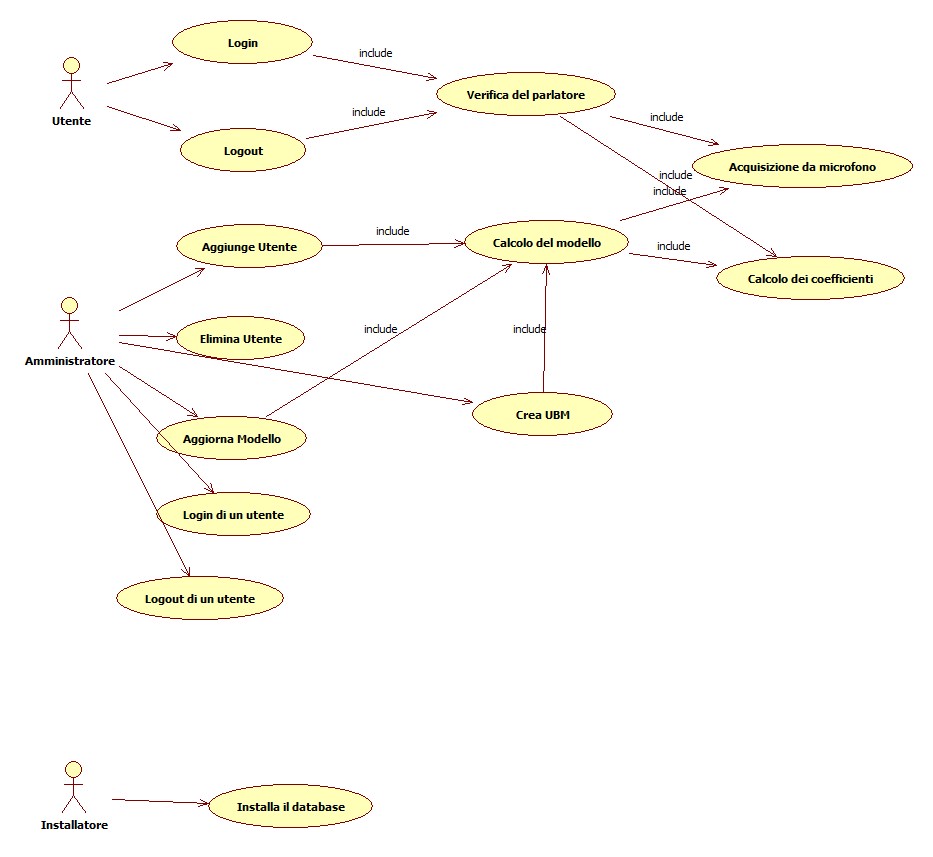
Come supporto a questa classe è stata usata la classe di servizio WavFileException che serve a gestire eventuali eccezioni generate a runtime.

# UML

## Diagramma dei casi d’uso

I diagrammi UML sono noti in ambito di progettazione software in quanto forniscono un valido supporto alla documentazione. L’uml è composto da molti diagrammi che tendono a descrivere il sistema in ogni suo livello di astrazione.

Tra i diagrammi uml più utilizzati si annovera quello dei casi d’uso il quale mette in mostra come gli attori interagiscono con i casi d’uso e le relazioni che vi sono tra di essi. Il caso d’uso è definito come una modalità di utilizzo del sistema, mentre un attore è l’utilizzatore e quindi colui che interagisce con il sistema.

******

# Reference

1. A Tutorial on Text-Independent Speaker Verification, Bimbot, Bonastre, Fredouille, Gravier, Magrin-Chagnolleau, Meignier, Merlin, Ortega-Garcia, Petrovska-Delacrétaz, Reynolds, 8th August 2003.