

MyTalk

Software di comunicazione tra utenti senza
requisiti di installazione



clockworkTeam7@gmail.com

Specifica Tecnica

v 1.0

Informazioni sul documento

Nome documento	Specifica Tecnica
Versione documento	v 1.0
Data creazione	2013/01/14
Data ultima modifica	2013/01/29
Uso documento	Esterno
Redazione	<i>Fase2:</i> Bain Giacomo Ceseracciu Marco Gavagnin Jessica Zohouri Haghian Pardis <i>Fase3:</i> Ceseracciu Marco Furlan Valentino Gavagnin Jessica
Verifica	Bain Giacomo La Bruna Agostino
Approvazione	Palmisano Maria Antonietta La Bruna Agostino
Lista distribuzione	gruppo <i>Clockwork</i> Zucchetti SPA Prof. Tullio Vardanega

Sommario

Questo documento vuol definire l'architettura generale che il prodotto dovrà avere.

Diario delle modifiche

Autore	Modifica	Data	Versione
La Bruna Agostino	Verifica documento e approvazione	2013/01/29	v 1.0
Palmisano Maria Antonietta	Controllo dei tracciamenti	2013/01/29	v 0.16
Bain Giacomo	Controllo concettuale capitolo 9 e 10	2013/01/28	v 0.15
Palmisano Maria Antonietta	Controllo concettuale fino al capitolo 6	2013/01/28	v 0.14
Bain Giacomo	Controllo ortografico, strutturale e sintattico	2013/01/26	v 0.13
Furlan Valentino	Stesura capitolo 9	2013/01/25	v 0.12
Gavagnin Jessica	Stesura capitolo 10	2013/01/25	v 0.11
Ceseracciu Marco	Stesura capitolo 7	2013/01/24	v 0.10
Furlan Valentino	Stesura capitolo 8	2013/01/24	v 0.9
Ceseracciu Marco	Bozza capitolo 7	2013/01/23	v 0.8
Bain Giacomo	Stesura capitolo 2	2013/01/22	v 0.7
Gavagnin Jessica	Stesura capitolo Server	2013/01/21	v 0.6
Zohouri Haghian Pardis	Stesura capitolo Client	2013/01/21	v 0.5
Bain Giacomo	Stesura capitolo 4	2013/01/21	v 0.4
Gavagnin Jessica	Bozza capitolo 3	2013/01/17	v 0.3
Ceseracciu Marco	Bozza capitolo 2	2013/01/16	v 0.2
Zohouri Haghian Pardis	Creazione documento, stesura sezione Introduzione	2013/01/15	v 0.1

Indice

1	Introduzione	1
1.1	Scopo del documento	1
1.2	Scopo del prodotto	1
1.3	Glossario	1
1.4	Riferimenti	1
1.4.1	Normativi	1
1.4.2	Informativi	1
2	Architettura Generale	2
2.1	Client	2
2.2	Server	3
2.3	Architettura dell'intero sistema	3
3	Design Pattern	5
3.1	Multitier	5
3.2	MVP	6
3.3	Observer	7
4	Strumenti Utilizzati	8
4.1	Java	8
4.2	HTML5	8
4.3	JavaScript	8
4.4	SQLite	8
4.5	Backbone.js	9
5	Client	10
5.1	View	11
5.1.1	mytalk.client.view.LoginView	11
5.1.2	mytalk.client.view.TutorialView	11
5.1.3	mytalk.client.view.UserListView	11
5.1.4	mytalk.client.view.IpCallView	12
5.1.5	mytalk.client.view.AudioCallView	12
5.1.6	mytalk.client.view.VideoCallView	12
5.1.7	mytalk.client.view.NotificationView	13
5.1.8	mytalk.client.view.ChatView	13
5.1.9	mytalk.client.view.ViewAccountDataView	13
5.1.10	mytalk.client.view.RecordMessageView	13
5.1.11	mytalk.client.view.CallStatisticsView	14
5.1.12	mytalk.client.view.SignUpView	14
5.1.13	mytalk.client.view.ModifyAccountDataView	14
5.1.14	mytalk.client.view.DefaultView	15
5.1.15	mytalk.client.view.SendFileView	15
5.2	Presenter	15
5.2.1	mytalk.client.presenter.AuthenticationControl	15

5.2.2	mytalk.client.presenter.UserListControl	16
5.2.3	mytalk.client.presenter.CallControl	16
5.2.4	mytalk.client.presenter.ChatControl	16
5.2.5	mytalk.client.presenter.NotificationControl	17
5.2.6	mytalk.client.presenter.UserDataControl	17
5.2.7	mytalk.client.presenter.StatisticsControl	17
5.2.8	mytalk.client.presenter.RecordMessageControl	18
5.2.9	mytalk.client.presenter.TutorialControl	18
5.2.10	mytalk.client.presenter.SendFileControl	18
5.3	Model	18
5.3.1	mytalk.client.model.User	19
5.3.2	mytalk.client.model.Contact	19
5.3.3	mytalk.client.model.Statistics	19
5.3.4	mytalk.client.model.TextMessage	19
5.3.5	mytalk.client.model.Tutorial	20
6	Server	21
6.1	Transfer Layer	22
6.1.1	mytalk.server.transfer.CallTransfer	22
6.1.2	mytalk.server.transfer.AuthenticationTransfer	22
6.1.3	mytalk.server.transfer.UserDataTransfer	22
6.1.4	mytalk.server.functionManager.TutorialTransfer	23
6.1.5	mytalk.server.transfer.RecordMessageTransfer	23
6.1.6	mytalk.server.transfer.NotificationTransfer	23
6.2	Manager Layer	24
6.2.1	mytalk.server.functionmanager.CallManager	24
6.2.2	mytalk.server.functionmanager.RecordMessageManager	24
6.2.3	mytalk.server.functionmanager.TutorialManager	24
6.2.4	mytalk.server.usermanager.AuthenticationManager	25
6.2.5	mytalk.server.usermanager.UserDataManager	25
6.3	Data Layer	25
6.3.1	mytalk.server.data.UserData	26
6.3.2	mytalk.server.data.RecordMessageData	26
6.3.3	mytalk.server.data.TutorialData	26
7	Tracciamento componenti-requisiti	27
8	Tracciamento requisiti-componenti	29
9	Diagramma delle attività	31
10	Prototipo interfaccia utente	33

Elenco delle figure

1	Architettura generale	2
2	Architettura generale dell'intero sistema	4
3	Rappresentazione Three Tier	5
4	Rappresentazione MVP	6
5	Client	10
6	Server	21
7	Diagramma della attività	31
8	Pagina iniziale	33
9	Pagina dopo login	34
10	Schermata videochiamata	35
11	Schermata chiamata audio	36
12	Schermata comunicazione testuale	37
13	Schermata Videoconferenza	38

Elenco delle tabelle

1	Tracciamento tra componenti e requisiti	28
2	Tracciamento tra requisiti e componenti	30

1 Introduzione

1.1 Scopo del documento

Il presente documento ha lo scopo di definire la progettazione ad alto livello che il prodotto dovrà avere. Verranno presentati i vari design pattern utilizzati nella creazione del prodotto, l'architettura generale secondo la quale saranno organizzate le varie componenti software e il tracciamento tra le componenti software ed i requisiti.

1.2 Scopo del prodotto

Il prodotto denominato **MyTalk** si propone di fornire un software per un sistema di comunicazione audio e video tra utenti. Lo scopo del progetto è poter comunicare con altri utenti tramite il browser, utilizzando solo componenti standard, senza dover installare plugin o programmi esterni. L'utilizzatore dovrà poter chiamare un altro utente, iniziare la comunicazione sia audio che video, svolgere la chiamata e terminare la chiamata ottenendo delle statistiche sull'attività.

1.3 Glossario

I termini tecnici o di uso non comune sono presenti nel documento allegato Glossario_v2.0.pdf. Tali riferimenti vengono evidenziati tramite sottolineatura alla prima occorrenza del termine nel documento.

1.4 Riferimenti

1.4.1 Normativi

- Norme di Progetto (allegato Norme_di_Progetto_v2.0.pdf)

1.4.2 Informativi

- Analisi dei Requisiti (allegato Analisi_dei_Requisiti_v2.0.pdf)
- SWEBOK - Chapter 3: Software Design: <http://www.computer.org/protal/web/swebok/html/ch3>
- SWEBOK - Chapter 11: Software Quality: <http://www.computer.org/protal/web/swebok/html/ch11>
- Software Engineering - Chapter 11: Architectural design - Ian Sommerville - 8th ed. (2006)

2 Architettura Generale

L'applicativo è stato diviso in due sistemi, ovvero il client e il server, come indicato nella figura 1. L'indice di accoppiamento tra questi due sottosistemi sarà tenuto al minimo necessario.

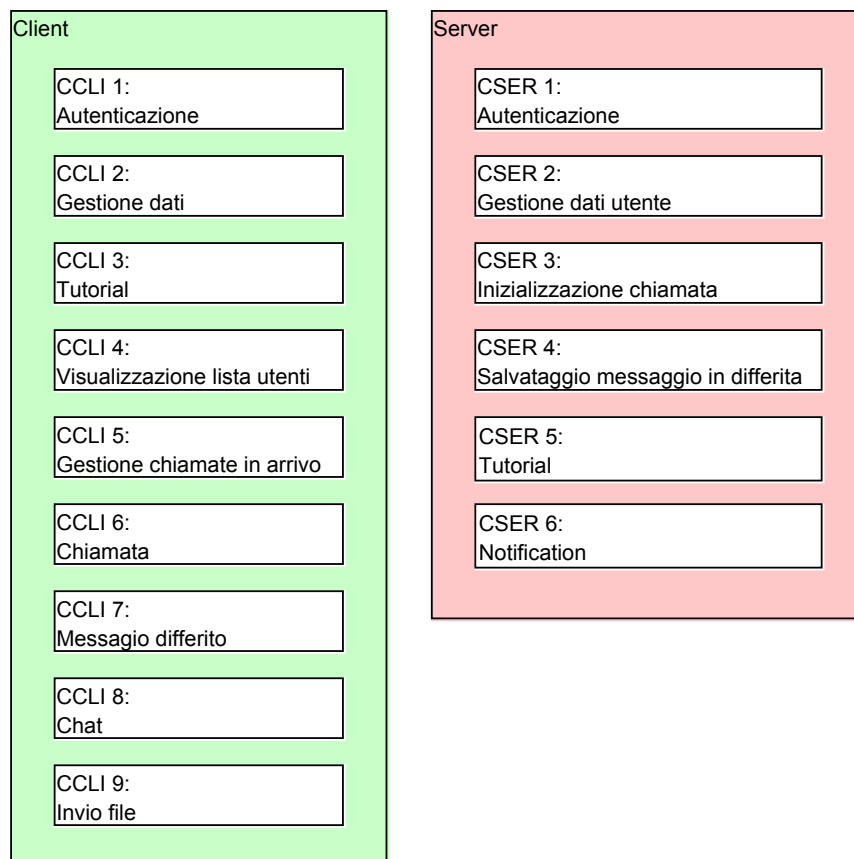


Figura 1: Architettura generale

2.1 Client

Il client si occuperà di gestire l'interfaccia visibile all'utente le cui operazioni verranno poi inviate al server. Il lato client è stato suddiviso nel seguente modo:

- **CCLI1:** autenticazione
- **CCLI2:** gestione dati
- **CCLI3:** tutorial

- **CCLI4:** visualizzazione lista utenti
- **CCLI5:** gestione notifiche
- **CCLI6:** chiamata
- **CCLI7:** messaggio differito
- **CCLI8:** chat
- **CCLI9:** invio file

2.2 Server

Il server si occuperà di gestire le richieste fatte e di inviare una risposta al client. Il server è stato suddiviso in

- **CSER1:** autenticazione
- **CSER2:** gestione dati utenti
- **CSER3:** inizializzazione chiamata
- **CSER4:** salvataggio messaggio in differita
- **CSER5:** tutorial
- **CSER6:** notification

2.3 Architettura dell'intero sistema

Facendo riferimento alla figura 2 si può avere quindi una visione dell'architettura e delle relazioni tra gli strati:

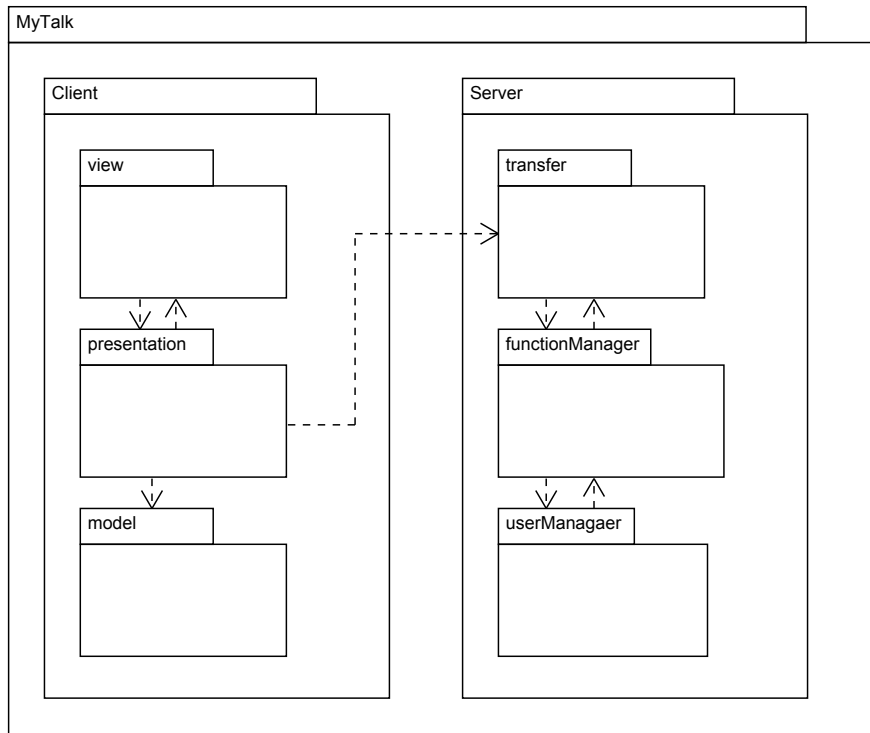


Figura 2: Architettura generale dell'intero sistema

- **mytalk:** incorpora l'intero sistema
- **mytalk.client** racchiude al suo interno il lato client di **MyTalk**
- **mytalk.client.view** racchiude al suo interno l'interfaccia grafica
- **mytalk.client.presenter** gestisce il collegamento tra client e server
- **mytalk.client.model** contiene dati locali
- **mytalk.server** racchiude al suo interno il lato server di **MyTalk**
- **mytalk.server.transfer** gestisce il collegamento tra server e client
- **mytalk.server.functionmanager** gestisce le operazioni di chiamata e di modifica della base di dati
- **mytalk.server.usermanager** gestisce le operazione di autenticazione
- **mytalk.server.data** contiene le informazioni del database

3 Design Pattern

Presentiamo qui di seguito i vari design pattern che andremo ad utilizzare per la rappresentazione dell'architettura del sistema.

3.1 Multitier

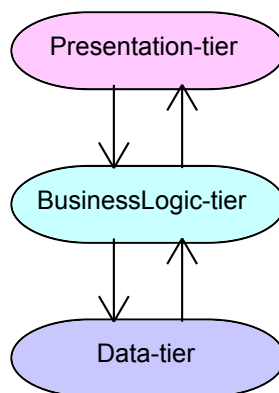


Figura 3: Rappresentazione Three Tier

Il design pattern di tipo Multitier verrà implementato nella forma Three Tier:

- **Descrizione:** tale design pattern permette una disgiunzione fra i vari gruppi di entità che cooperano nell'erogazione del servizio. Un primo livello si occuperà della comunicazione con il client, un secondo livello invece si occuperà di effettuare le operazioni logiche ed elaborare le informazioni salvandole e prelevandole da un terzo livello inferiore
- **Motivazione:** il beneficio principale di questo design pattern è la possibilità di aggiornare/cambiare un livello senza andare a modificare i livelli adiacenti. Nonostante i vari livelli si occupino di specifiche operazioni è presente una forte comunicazione rigidamente strutturata tra di loro, questo lo rende un ottimo modello per applicazioni client-server, poiché ogni livello non esiste come unità logica a se stante, ma si adegua allo specifico ambiente di rete in cui esegue
- **Contesto applicativo:** tale design pattern verrà utilizzato come struttura portante del server e sarà suddiviso come segue
 - Transfer Layer: offrirà un'interfaccia di comunicazione con il client e i livelli sottostanti del server

- Logic Layer: si occuperà di svolgere le funzionalità di comunicazione e di gestire la visualizzazione e la modifica delle informazioni presenti nello strato Data
- Data Layer: sarà un contenitore di informazioni riguardanti gli utenti e di messaggi registrati e collegamenti

3.2 MVP

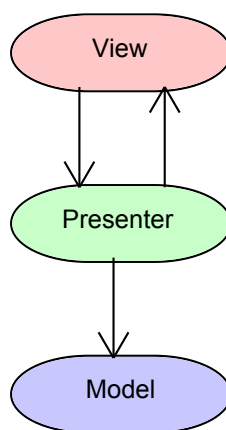


Figura 4: Rappresentazione MVP

Il design pattern di tipo MVP verrà implementato per rappresentare l'architettura generale e più specificamente nel lato client.

- **Descrizione:** tale design pattern permette una completa disgiunzione tra le funzionalità che il prodotto deve offrire ed è strutturato in tre livelli
 - **Model:** che rappresenta la parte in grado di recuperare le informazioni presenti nel sistema
 - **View:** che rappresenta l'interfaccia grafica del sistema con la quale gli utenti possono interagire
 - **Presenter:** rappresenta la componente che riceve le richieste dalla view e le traduce in operazioni che agiranno sia sui dati che sui modelli; si occupa inoltre della comunicazione col server
- **Motivazione:** questo design pattern permette un buon livello di disaccoppiamento tra la vista ed il modello. L'architettura risulta facilmente manutenibile poiché le modifiche ad uno dei tre componenti non comporta modifiche ai restanti

- **Contesto applicativo:** il design pattern verrà utilizzato nello sviluppo dell'architettura generale e come struttura rappresentante il client. Sono state individuate tre macro componenti:
 - **Model:** ha il compito di astrarre i dati presenti nel database recuperandone o aggiornandone le informazioni e disporli in modo tale da essere visibili alla view. Questo sarà il compito di parte del data layer e parte del manager layer del server
 - **View:** rappresenta l'interfaccia grafica che consente l'interazione tra gli utenti ed il sistema
 - **Presenter:** si occupa di ricevere le richieste dalla view e della comunicazione tra il lato client ed il lato server del sistema

3.3 Observer

- **Descrizione:** tale design pattern si occupa di gestire i cambiamenti che avvengono in un oggetto riflettendone le conseguenze su tutti gli oggetti ad esso legato
- **Motivazione:** questo design pattern modifica delle componenti nel caso altre componenti ad esse legate vengano modificate
- **Contesto applicativo:** si rivela utile per quanto riguarda la manutenibilità della lista utenti connessi o meno o nel caso nuovi utenti si registrassero

4 Strumenti Utilizzati

4.1 Java

L'utilizzo di Java è legato ai requisiti di capitolato; verrà utilizzato per l'avvio della comunicazione tra due utenti e per l'utilizzo del protocollo di comunicazione WebSocket.

- **Vantaggi:**

- **Multipiattaforma:** Grazie alla presenza della JVM si ha la sicurezza che il programma sarà eseguibile indipendentemente dal sistema operativo installato sulla macchina
- **Indipendenza delle risorse:** per lo stesso motivo sopra elencato l'utilizzo delle risorse fisiche sarà indipendente dal sistema operativo installato

- **Svantaggi:**

- Nessun svantaggio rilevato

4.2 HTML5

L'utilizzo di HTML5 è legato ai requisiti di capitolato e andrà a costituire insieme a CSS3 l'interfaccia web del prodotto.

- **Vantaggi:**

- HTML5 supporta le ultime tecnologie riguardanti la creazione di applicazioni web
- Grafica più leggera e valida evitando l'utilizzo di tecnologia Flash

- **Svantaggi:**

- HTML5 e CSS3 non attualmente standard

4.3 JavaScript

Andremo ad utilizzare JavaScript per l'utilizzo di WebRTC e quindi per la comunicazione tra gli utenti.

4.4 SQLite

Considerando la complessità relativamente limitata del database che occorrerà per la gestione degli utenti abbiamo deciso di utilizzabile SQLite.

- **Vantaggi:**

- Conoscenza del linguaggio SQL

- Gestibile attraverso una libreria di Java
- Non richiede installazione di un server
- Leggero e veloce

- **Svantaggi:**

- Alcune limitazioni rispetto alla versione MySql

4.5 Backbone.js

Per la gestione del lato client abbiamo deciso di utilizzare il framework Backbone.js, che risulta comodo ed efficiente per la gestione di applicazioni che utilizzino pesantemente il linguaggio JavaScript.

- **Vantaggi:**

- Maggiore facilità nella gestione del lato client e nella sua programmazione
 - Backbone.js è un progetto open source

- **Svantaggi:**

- Dover seguire la struttura data dal framework

5 Client

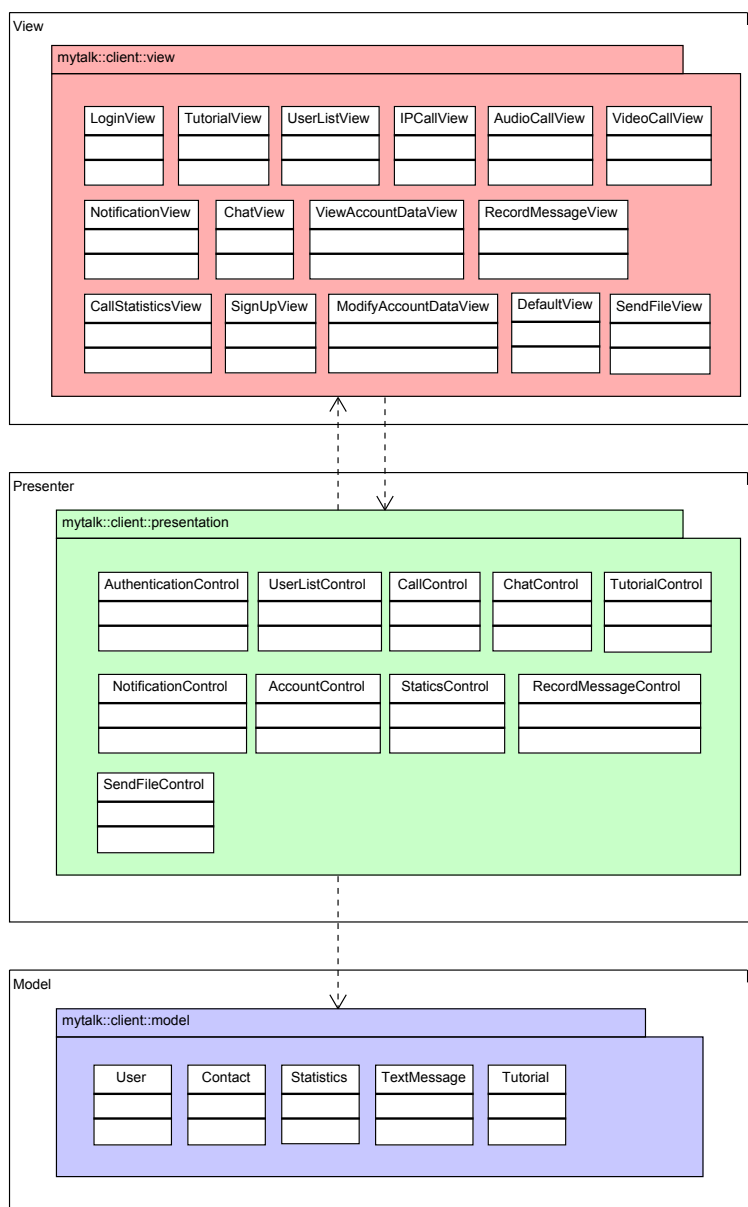


Figura 5: Client

5.1 View

- **Tipo, obiettivo e funzione del componente:** costituisce la parte del sistema che definisce ed implementa l'interfaccia web usufruibile dagli utenti mediante pagine web
- **Relazioni d'uso con altre componenti:** il componente è costituito dal package `view`, e andrà a comunicare con il package `mytalk.client.presenter` che farà da tramite per la comunicazione tra il lato client e il lato server e controllerà i dati in transito

5.1.1 `mytalk.client.view.LoginView`

- **Tipo, obiettivo e funzione del componente:** la classe `LoginView` definisce la struttura, e la conseguente visualizzazione, delle pagine web che consentono di effettuare l'accesso e la disconnessione al sistema
- **Relazioni d'uso con altre componenti:** la classe `LoginView` comunicherà con la classe `mytalk.client.presenter.AuthenticationControl` per poter verificare se le credenziali inserite siano corrette o meno e quindi effettuare l'accesso al sistema o mostrare i relativi messaggi di errore
- **Attività svolte e dati trattati:** la classe fa in modo che vengano visualizzati i componenti grafici necessari per le operazioni di autenticazione e terminazione della sessione in modo che possano essere inviati alla classe `mytalk.client.presenter.AuthenticationControl`, che svolgerà i vari controlli

5.1.2 `mytalk.client.view.TutorialView`

- **Tipo, obiettivo e funzione del componente:** la classe `TutorialView` definisce la struttura, e la conseguente visualizzazione, delle pagine web che consentono la visualizzazione dei video tutorial del prodotto **MyTalk**
- **Relazioni d'uso con altre componenti:** la classe `TutorialView` comunicherà con la classe `mytalk.client.presenter.TutorialControl` al fine di visualizzare il video scelto dall'utente
- **Attività svolte e dati trattati:** la classe fa in modo che vengano visualizzati i componenti grafici necessari alla selezione e visualizzazione di un tutorial video

5.1.3 `mytalk.client.view.UserListView`

- **Tipo, obiettivo e funzione del componente:** la classe `UserListView` definisce la struttura, e la conseguente visualizzazione, delle pagine web che consentono la visualizzazione della lista utenti iscritti al server

- **Relazioni d'uso con altre componenti:** la classe `UserListView` comunicherà con la classe `mytalk.client.presenter.UserListControl` che restituirà gli iscritti al server connessi in quel momento o meno
- **Attività svolte e dati trattati:** la classe fa in modo che venga visualizzata una lista di utenti registrati presso il server indicando anche quali di essi sono online e quali offline, inoltre premendo con il mouse su un utente della lista lo si seleziona come possibile destinatario di una delle funzionalità messe a disposizione

5.1.4 `mytalk.client.view.IpCallView`

- **Tipo, obiettivo e funzione del componente:** la classe `IpCallView` definisce la struttura, e la conseguente visualizzazione, delle pagine web che consentono di effettuare chiamate inserendo unicamente l'indirizzo IP da chiamare
- **Relazioni d'uso con altre componenti:** la classe `IpCallView` comunicherà con le classi `mytalk.client.presenter.CallControl` e `mytalk.client.presenter.ChatControl` per permettere la comunicazione audio, video e testuale tra due utenti
- **Attività svolte e dati trattati:** la classe fa in modo che vengano visualizzate le componenti grafiche che permettono l'inserimento di un indirizzo IP e la scelta delle funzionalità di comunicazione disponibili

5.1.5 `mytalk.client.view.AudioCallView`

- **Tipo, obiettivo e funzione del componente:** la classe `AudioCallView` definisce la struttura, e la conseguente visualizzazione, delle pagine web che consentono di effettuare chiamate audio
- **Relazioni d'uso con altre componenti:** la classe `AudioCallView` comunicherà con la classe `mytalk.client.presenter.CallControl` per permettere la gestione della comunicazione audio tra due o più utenti
- **Attività svolte e dati trattati:** la classe fa in modo che vengano visualizzate le componenti grafiche che permettono la gestione delle funzionalità legate alla comunicazione audio

5.1.6 `mytalk.client.view.VideoCallView`

- **Tipo, obiettivo e funzione del componente:** la classe `VideoCallView` definisce la struttura, e la conseguente visualizzazione, delle pagine web che consentono di effettuare chiamate video
- **Relazioni d'uso con altre componenti:** la classe `VideoCallView` comunicherà con la classe `mytalk.client.presenter.CallControl` per permettere la gestione della comunicazione audio e video tra due utenti

- **Attività svolte e dati trattati:** la classe fa in modo che vengano visualizzate le componenti grafiche che permettono la visualizzazione dei video e la gestione delle funzionalità legate alla comunicazione audio/video

5.1.7 mytalk.client.view.NotificationView

- **Tipo, obiettivo e funzione del componente:** la classe `NotificationView` definisce la struttura, e la conseguenti visualizzazioni, delle pagine web che consentono di visualizzare notifiche di chiamate in entrata
- **Relazioni d'uso con altre componenti:** la classe `NotificationView` verrà utilizzata dalla classe `mytalk.client.presenter.NotificationControl` che invierà notifiche di richiesta di comunicazione
- **Attività svolte e dati trattati:** la classe fa in modo che vengano visualizzate le notifiche relative a richieste in arrivo da parte di altri utenti

5.1.8 mytalk.client.view.ChatView

- **Tipo, obiettivo e funzione del componente:** la classe `ChatView` definisce la struttura, e la conseguente visualizzazione, delle pagine web che consentono di effettuare comunicazioni testuali
- **Relazioni d'uso con altre componenti:** la classe `ChatView` comunicherà con la classe `mytalk.client.presenter.ChatControl` per permettere la comunicazione testuale tra due utenti
- **Attività svolte e dati trattati:** la classe fa in modo che vengano visualizzate le componenti grafiche necessarie alla composizione di messaggi di testo e alla visualizzazione dei messaggi finora inviati e ricevuti

5.1.9 mytalk.client.view.ViewAccountDataView

- **Tipo, obiettivo e funzione del componente:** la classe `ViewAccountDataView` definisce la struttura, e la conseguente visualizzazione, delle pagine web che consentono di visualizzare i dati dell'account selezionato
- **Relazioni d'uso con altre componenti:** la classe `ViewAccountDataView` comunicherà con la classe `mytalk.client.presenter.UserDataControl` per poter visualizzare i dati relativi al proprio account
- **Attività svolte e dati trattati:** la classe fa in modo che vengano visualizzati i dati personali dell'utente

5.1.10 mytalk.client.view.RecordMessageView

- **Tipo, obiettivo e funzione del componente:** la classe `RecordMessageView` definisce la struttura, e la conseguente visualizzazione, delle pagine web per effettuare registrazioni audio da inviare

- **Relazioni d'uso con altre componenti:** la classe `RecordMessageView` comunicherà con la classe `mytalk.client.presenter.RecordMessageControl` che con i suoi metodi permetterà la registrazione di un video da inviare ad un utente registrato
- **Attività svolte e dati trattati:** la classe definisce quindi la struttura della pagina web che consente la registrazione di chiamate video da inviare verso utenti registrati

5.1.11 `mytalk.client.view.CallStatisticsView`

- **Tipo, obiettivo e funzione del componente:** la classe `CallStatisticsView` definisce la struttura, e la conseguente visualizzazione, delle pagine web per la visualizzazione di statistiche durante lo svolgimento di una chiamata
- **Relazioni d'uso con altre componenti:** la classe `CallStatisticsView` comunicherà con la classe `mytalk.client.presenter.CallStatisticsControl` che avrà lo scopo di visualizzare le statistiche della chiamata
- **Attività svolte e dati trattati:** la classe definisce la struttura della pagina web che visualizza i dati della chiamata che viene effettuata

5.1.12 `mytalk.client.view.SignUpView`

- **Tipo, obiettivo e funzione del componente:** la classe `SignUpView` definisce la struttura, e la conseguente visualizzazione, delle pagine web per la registrazione di un nuovo utente al server
- **Relazioni d'uso con altre componenti:** la classe `SignUpView` comunicherà con la classe `mytalk.client.presenter.UserDataControl` che avrà lo scopo di effettuare la registrazione al server tenendo in considerazione l'univocità dell'username
- **Attività svolte e dati trattati:** la classe fa in modo che vengano visualizzati i definisce quindi la struttura della pagina web per l'iscrizione di un nuovo utente al server

5.1.13 `mytalk.client.view.ModifyAccountDataView`

- **Tipo, obiettivo e funzione del componente:** la classe `ModifyAccountDataView` definisce la struttura, e la conseguente visualizzazione, delle pagine web per la modifica dei dati del proprio account
- **Relazioni d'uso con altre componenti:** la classe `ModifyAccountDataView` comunicherà con la classe `mytalk.client.presenter.UserDataControl` che avrà lo scopo di effettuare le modifiche dei dati dell'utente
- **Attività svolte e dati trattati:** la classe definisce quindi la struttura della pagina web per la modifica dei dati personali dell'utente

5.1.14 mytalk.client.view.DefaultView

- **Tipo, obiettivo e funzione del componente:** la classe `DefaultView` definisce la struttura, e la conseguente visualizzazione, delle pagine web di default
- **Relazioni d'uso con altre componenti:** essendo la pagina di base, non ha relazioni d'uso con altre componenti
- **Attività svolte e dati trattati:** la classe si occupa di far vedere una schermata di default nel momento in cui non ci siano nessun tipo di chiamata in corso o visualizzazione di dati di altri utenti.

5.1.15 mytalk.client.view.SendFileView

- **Tipo, obiettivo e funzione del componente:** la classe `SendFileView` definisce la struttura, e la conseguente visualizzazione, delle pagine web che consentono di selezionare un file salvato in locale per mandarlo all'utente autenticato selezionato
- **Relazioni d'uso con altre componenti:** la classe `SendFileView` comunicherà con la classe `mytalk.client.presenter.SendFileControl` per comunicare il file e il destinatario selezionato
- **Attività svolte e dati trattati:** la classe fa in modo che vengano visualizzati i componenti grafici necessari per le operazioni selezione di un file salvato in locale e di scelta del destinatario ed invia tali dati a `mytalk.client.presenter.SendFileControl`, che svolgerà le operazioni di invio del file

5.2 Presenter

- **Tipo, obiettivo e funzione del componente:** costituisce la parte del sistema che definisce ed implementa la parte del sistema che restituisce il risultato delle elaborazioni effettuate dall'utente
- **Relazioni d'uso con altre componenti:** il componente è costituito dal package `presenter`, e andrà a comunicare con il package `view` riportando i risultati delle elaborazioni dei dati e con il package `model`. Inoltre il package `presenter` si occuperà della comunicazione tra client e server

5.2.1 mytalk.client.presenter.AuthenticationControl

- **Tipo, obiettivo e funzione del componente:** la classe `AuthenticationControl` si occupa di controllare le credenziali che un utente inserisce trasmetterle al server, e, in base alla risposta, aggiornare la pagina web che l'utente vede ed aggiornare `mytalk.client.model.User`

- **Relazioni d'uso con altre componenti:** utilizzando i metodi presenti nella classe `mytalk.client.view.LoginView` stabilisce una connessione col server dal quale riceve una risposta che poi andrà trasmessa al livello view che aggiornerà la pagina di conseguenza, ed in caso di risposta affermativa ci sarà un collegamento a `mytalk.client.model.User` al quale verranno passate i dati del proprio account
- **Attività svolte e dati trattati:** la classe si occupa quindi della comunicazione tra client e server per quanto riguarda l'autenticazione degli utenti

5.2.2 `mytalk.client.presenter.UserListControl`

- **Tipo, obiettivo e funzione del componente:** la classe `UserListControl` si occupa di controllare la lista degli utenti iscritti al server comunicando con il server e restituire il risultato al package presenter
- **Relazioni d'uso con altre componenti:** la classe si interfaccia con la classe `mytalk.client.view.UserListView` alla quale invia gli utenti iscritti al server e se sono connessi o meno una volta fatto accesso al server, ed inoltre salva tali informazioni in `mytalk.client.model.Contact`
- **Attività svolte e dati trattati:** la classe si occupa quindi di restituire la lista di tutti gli altri utenti iscritti al server

5.2.3 `mytalk.client.presenter.CallControl`

- **Tipo, obiettivo e funzione del componente:** la classe `CallControl` si occupa di avviare la comunicazione tra utenti attraverso una comunicazione con il server
- **Relazioni d'uso con altre componenti:** la classe si interfacerà in base al tipo di chiamata alla classe `mytalk.client.view.VideoCallView` se si tratta di una chiamata video ed audio, `mytalk.client.view.AudioCallView` se si tratta di una chiamata solamente audio, o `mytalk.client.view.IpCallView` se la chiamata è stata effettuata inserendo direttamente un indirizzo IP
- **Attività svolte e dati trattati:** la classe si occupa quindi di avviare una chiamata tra due diversi utenti

5.2.4 `mytalk.client.presenter.ChatControl`

- **Tipo, obiettivo e funzione del componente:** la classe `ChatControl` si occupa di controllare se sono in arrivo messaggi di testo da parte di altri utenti attraverso una comunicazione con il server
- **Relazioni d'uso con altre componenti:** la classe si interfacerà con la classe `mytalk.client.view.ChatView` se la comunicazione avviene tra

due utenti iscritti, altrimenti alla classe `mytalk.client.view.IpCall` se è stata effettuato inserendo direttamente un indirizzo IP

- **Attività svolte e dati trattati:** la classe si occupa quindi di avviare una comunicazione testuale tra due utenti

5.2.5 `mytalk.client.presenter.NotificationControl`

- **Tipo, obiettivo e funzione del componente:** la classe `NotificationControl` si occupa di inviare la presenza di notifiche di chiamata ad un altro utente
- **Relazioni d'uso con altre componenti:** la classe `NotificationControl` si interfaccia con la classe `mytalk.client.view.Notification` e segnala la presenza di una chiamata in arrivo da parte di un altro utente
- **Attività svolte e dati trattati:** la classe si occupa quindi di avvisare l'utente che è in arrivo una chiamata in arrivo

5.2.6 `mytalk.client.presenter.UserDataControl`

- **Tipo, obiettivo e funzione del componente:** la classe `UserDataControl` si occupa della gestione dei dati degli utenti
- **Relazioni d'uso con altre componenti:** la classe `UserDataControl` si interfaccia con la classe `mytalk.client.view.ViewAccountDataView` che si occuperà della visione dei dettagli degli account degli utenti, `mytalk.client.view.ModifyAccountDataView` per la modifica dei dati e a `mytalk.client.view.SignUpView` per la registrazione
- **Attività svolte e dati trattati:** la classe si occupa quindi della ricezioni di richieste riguardanti i dettagli di account degli utenti

5.2.7 `mytalk.client.presenter.StatisticsControl`

- **Tipo, obiettivo e funzione del componente:** la classe si occupa di gestire l'invio delle statistiche delle chiamate alla classe `mytalk.client.view.CallStatisticsView`
- **Relazioni d'uso con altre componenti:** la classe si interfaccia alla classe `Statistics` dalla quale riceverà le informazioni sulla chiamata che poi andranno passate alla classe `mytalk.client.view.CallStatisticsView`
- **Attività svolte e dati trattati:** la classe quindi si occupa di avere al suo interno le informazioni sulla chiamata in corso che verranno visualizzate all'utente grazie alla classe `mytalk.client.view.CallStatisticsView`

5.2.8 mytalk.client.presenter.RecordMessageControl

- **Tipo, obiettivo e funzione del componente:** la classe `RecordMessageControl` si occupa di gestire i video messaggi salvati inviando al server il compito di salvarlo nel server
- **Relazioni d'uso con altre componenti:** la classe si interfaccia con `mytalk.client.view.RecordMessageView` da cui riceve il video salvato che verrà inviato al server
- **Attività svolte e dati trattati:** la classe quindi si occupa di ricevere la richiesta di avvio registrazione e di mandare al server il compito di salvarla

5.2.9 mytalk.client.presenter.TutorialControl

- **Tipo, obiettivo e funzione del componente:** la classe `TutorialControl` si occupa di gestire la visualizzazione dei tutorial
- **Relazioni d'uso con altre componenti:** la classe `TutorialControl` si interfaccia con `mytalk.client.view.Tutorial` per la visione a livello grafico dei tutorial, con `mytalk.client.model.Tutorial` per la modellazione dei tutorial e con `mytalk.client.model.Tutorial` per ricevere l'indirizzamento al video
- **Attività svolte e dati trattati:** la classe fa sì che vengano visualizzati i tutorial richiesti dalla parte grafica del client

5.2.10 mytalk.client.presenter.SendFileControl

- **Tipo, obiettivo e funzione del componente:** la classe `SendFileControl` si occupa di gestire l'invio di file esterni a **MyTalk**
- **Relazioni d'uso con altre componenti:** la classe `SendFileControl` si interfaccia con `mytalk.client.view.SendFileView` per la scelta del file da inviare e con `mytalk.server.transfer.NotificationTransfer` per inviare al destinatario l'avviso di tentato trasferimento, inoltre con `mytalk.client.presenter.NotificationControl` nel caso si stia ricevendo una richiesta di trasferimento file.
- **Attività svolte e dati trattati:** la classe si occupa quindi di effettuare il trasferimento tra client di un file

5.3 Model

- **Tipo, obiettivo e funzione del componente:** ha lo scopo di rappresentare localmente le informazioni presenti nel database del server al fine di popolare le viste in modo dinamico
- **Relazioni d'uso con altre componenti:** viene utilizzato dal presenter che invierà le richieste di gestione e modifica dei dati

5.3.1 mytalk.client.model.User

- **Tipo, obiettivo e funzione del componente:** la classe `User` si occuperà di tenere aggiornato localmente le informazioni del proprio account
- **Relazioni d'uso con altre componenti:** la classe `User` riceverà da `mytalk.client.presenter.AuthenticationControl` le informazioni riguardanti il proprio account nel qual caso l'utente riesca a connettersi
- **Attività svolte e dati trattati:** la classe quindi si occupa di tenere le informazioni del proprio account restituendo i dati all'utente

5.3.2 mytalk.client.model.Contact

- **Tipo, obiettivo e funzione del componente:** la classe `Contact` si occuperà di gestire la lista utenti
- **Relazioni d'uso con altre componenti:** la classe `Contact` riceverà da `mytalk.client.presenter.UserListControl` la lista degli utenti che sono presenti nella lista e il loro stato attuale una volta effettuato l'accesso mentre da `mytalk.client.presenter.AuthenticationControl` aggiornamenti sullo stato degli utenti una volta autenticatisi
- **Attività svolte e dati trattati:** la classe si occupa di gestire aggiornata la lista utenti connessi o meno e di restituirla all'utente

5.3.3 mytalk.client.model.Statistics

- **Tipo, obiettivo e funzione del componente:** la classe `Statistics` si occuperà di gestire le statistiche di chiamata
- **Relazioni d'uso con altre componenti:** la classe `Statistics` riceverà la richiesta da `mytalk.client.presenter.StatisticsControl` per l'aggiornamento delle statistiche della chiamata
- **Attività svolte e dati trattati:** la classe quindi si occupa di gestire le informazioni sulla chiamata restituendo quindi le statistiche

5.3.4 mytalk.client.model.TextMessage

- **Tipo, obiettivo e funzione del componente:** la classe `TextMessage` si occupa della comunicazione testuale
- **Relazioni d'uso con altre componenti:** la classe `TextMessage` riceverà da `mytalk.client.presenter.ChatControl` le richieste per la comunicazione testuale tra due utenti
- **Attività svolte e dati trattati:** la classe si occuperà di gestire le comunicazioni testuali tra utenti restituendo il messaggio testuale

5.3.5 mytalk.client.model.Tutorial

- **Tipo, obiettivo e funzione del componente:** la classe `Tutorial` si occuperà di conservare i collegamenti ai tutorial
- **Relazioni d'uso con altre componenti:** la classe `Tutorial` riceverà richieste da parte di `mytalk.client.presenter.TutorialControl` per il caricamento di un video tutorial riguardante le funzionalità del prodotto **MyTalk** restituendone l'indirizzo del video
- **Attività svolte e dati trattati:** la classe si occuperà quindi di conservare i collegamenti da dove prelevare i video tutorial

6 Server

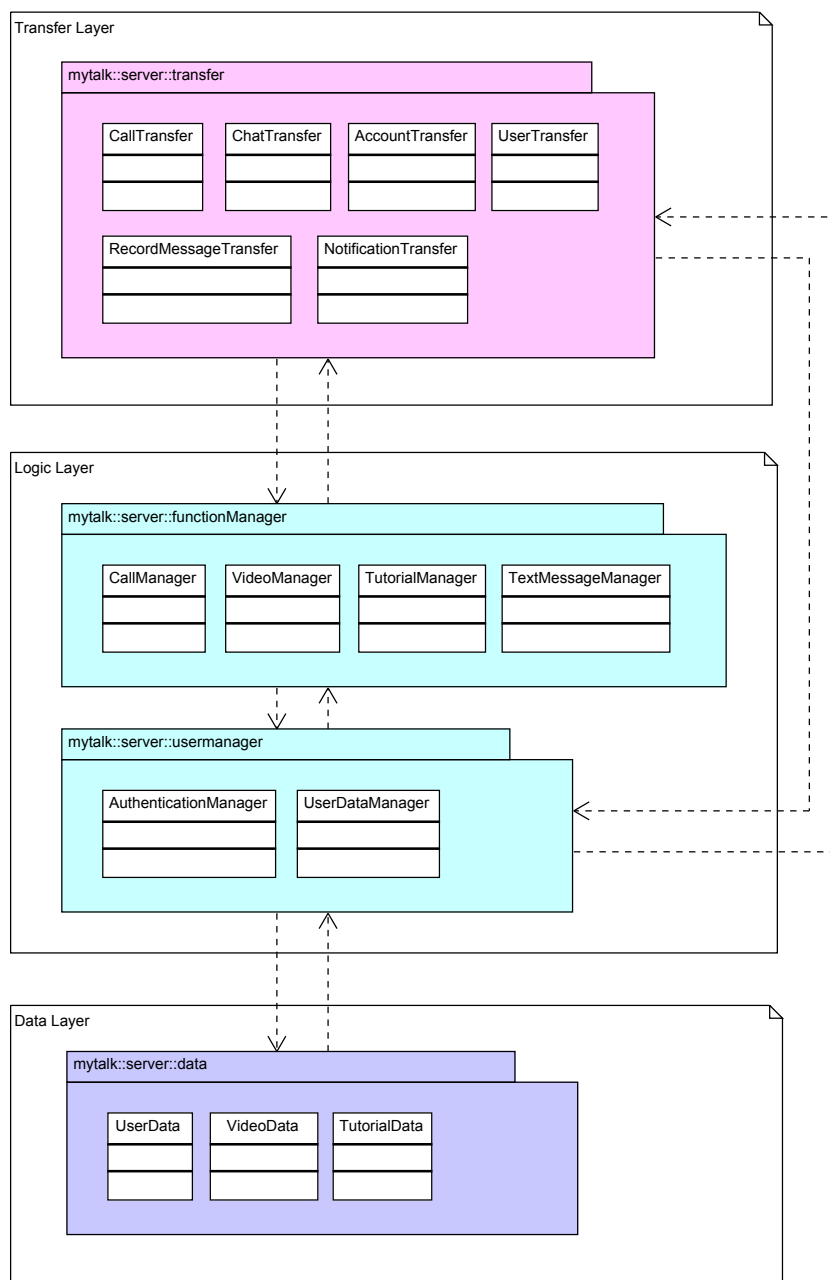


Figura 6: Server

6.1 Transfer Layer

- **Tipo, obiettivo e funzione del componente:** lo strato Transfer si occupa della comunicazione tra il client e il server
- **Relazioni d'uso con altre componenti:** lo strato Transfer interagisce con i package `mytalk.server.functionmanager` e `mytalk.server.usermanager` e con lo strato `mytalk.client.presenter` allo scopo di interfacciare il server col client

6.1.1 `mytalk.server.transfer.CallTransfer`

- **Tipo, obiettivo e funzione del componente:** la classe `CallTransfer` si occupa di far arrivare i dati all'apparato logico al fine di inizializzare la chiamata
- **Relazioni d'uso con altre componenti:** la classe `CallTransfer` viene contattata e si rivolge a `mytalk.client.presenter.CallControl` e `mytalk.server.functionmanager.CallManager`
- **Attività svolte e dati trattati:** la classe `CallTransfer` si occupa di gestire il trasferimento della richiesta di chiamata dal client al server e riferisce al chiamante se la comunicazione è stata accettata

6.1.2 `mytalk.server.transfer.AuthenticationTransfer`

- **Tipo, obiettivo e funzione del componente:** la classe `AuthenticationTransfer` si occupa del trasferimento delle richieste di autenticazione al server
- **Relazioni d'uso con altre componenti:** la classe `AuthenticationTransfer` viene utilizzata da `mytalk.client.presenter.AuthenticationControl` e `mytalk.server.usermanager.AuthenticationManager`
- **Attività svolte e dati trattati:** la classe `AuthenticationTransfer` ha lo scopo di ricevere da `mytalk.client.presenter.AuthenticationControl` richieste di autenticazione e manda i dati a `mytalk.server.usermanager.AuthenticationManager` che si occuperà di verificarli e dare una risposta che verrà quindi inviata al client.

6.1.3 `mytalk.server.transfer.UserDataTransfer`

- **Tipo, obiettivo e funzione del componente:** la classe `UserDataTransfer` si occupa del trasferimento dei dati riguardanti gli utenti al fine di effettuare nuove registrazioni o di visualizzare ed eventualmente modificare i dati persistenti
- **Relazioni d'uso con altre componenti:** la classe `UserDataTransfer` comunica con `mytalk.client.presenter.UserDataControl`, `mytalk.client.presenter.UserListControl` e `mytalk.server.usermanager.UserDataManager`

- **Attività svolte e dati trattati:** la classe riceve da `mytalk.client.presenter.UserDataControl` richieste di visualizzazione o modifica dei dati presenti nella base di dati e dialoga con `mytalk.server.usermanager.UserDataManager` per eseguire le operazioni richieste; inoltre in caso di effettuata autenticazione invia a `mytalk.client.presenter.UserListControl` i nomi utente ed eventualmente l'indirizzo IP identificativo degli utenti connessi

6.1.4 `mytalk.server.functionManager.TutorialTransfer`

- **Tipo, obiettivo e funzione del componente:** la classe `TutorialTransfer` classe si occupa del passaggio delle richieste di visualizzazione dei video tutorial dal client al server
- **Relazioni d'uso con altre componenti:** la classe `TutorialTransfer` viene chiamata da `mytalk.client.presenter.TutorialControl` e contatta `mytalk.server.functionmanager.TutorialManager`
- **Attività svolte e dati trattati:** la classe ha una funzione di collegamento, si occupa di far arrivare una richiesta a `mytalk.server.functionmanager.TutorialManager` e in seguito restituire a `mytalk.client.presenter.TutorialControl` il collegamento richiesto

6.1.5 `mytalk.server.transfer.RecordMessageTransfer`

- **Tipo, obiettivo e funzione del componente:** la classe `RecordMessageTransfer` si occupa del trasferimento del messaggio registrato dal client al server e del seguente invio al destinatario
- **Relazioni d'uso con altre componenti:** la classe `RecordMessageTransfer` comunica con `mytalk.server.manager.RecordMessageManager` e `mytalk.client.presenter.RecordMessageControl`
- **Attività svolte e dati trattati:** la classe si occupa quindi di ricevere dal client il messaggio registrato e l'utente a cui mandarlo e passa tali dati a `mytalk.server.manager.RecordMessageManager`. In seguito una volta ricevuto il video da `mytalk.server.manager.RecordMessageManager` lo manda al destinatario

6.1.6 `mytalk.server.transfer.NotificationTransfer`

- **Tipo, obiettivo e funzione del componente:** la classe `NotificationTransfer` si occupa di avvisare i client di eventuali notifiche dovute ad eventi generati da altri utenti
- **Relazioni d'uso con altre componenti:** la classe `NotificationTransfer` comunica con `mytalk.client.presenter.NotificationControl` e viene chiamata da `mytalk.server.functionmanager.RecordMessageManager`, `mytalk.server.functionmanager.CallManager`

- **Attività svolte e dati trattati:** la classe si occupa quindi di mandare al client messaggi che riguardano eventi esterni come una chiamata in corso o la presenza sul server di un messaggio registrato in attesa

6.2 Manager Layer

- **Tipo, obiettivo e funzione del componente:** lo strato Manager del server si occupa delle funzionalità logiche del server e della lettura e della modifica della base di dati
- **Relazioni d'uso con altre componenti:** lo strato Manager agisce sullo strato Data per visualizzare o modificare i dati e comunica in modo bidirezionale con lo strato Transfer

6.2.1 mytalk.server.functionmanager.CallManager

- **Tipo, obiettivo e funzione del componente:** la classe CallManager ha il compito di inizializzare la chiamata tra due utenti
- **Relazioni d'uso con altre componenti:** la classe CallManager viene contattata e comunica con `mytalk.server.transfer.CallTransfer` e `mytalk.server.transfer.NotificationTransfer`
- **Attività svolte e dati trattati:** la classe si occuperà quindi di inizializzare la chiamata tramite WebSocket, comunicando a `mytalk.server.transfer.CallTransfer` la situazione della chiamata che sta venendo effettuata e a `mytalk.server.transfer.NotificationTransfer` che tipo di operazione sia stata rivolta verso il destinatario e da parte di quale utente o indirizzo IP

6.2.2 mytalk.server.functionmanager.RecordMessageManager

- **Tipo, obiettivo e funzione del componente:** la classe RecordMessageManager ha l'obiettivo di gestire la persistenza dei messaggi da mandare in differita
- **Relazioni d'uso con altre componenti:** la classe RecordMessageManager riceve dati da `mytalk.server.transfer.RecordMessageTransfer` e salva i messaggi in `mytalk.server.data.RecordMessageData`
- **Attività svolte e dati trattati:** la classe si occuperà quindi del salvataggio, della cancellazione e dell'invio a `mytalk.server.transfer.RecordMessageTransfer` dei messaggi registrati e del loro invio

6.2.3 mytalk.server.functionmanager.TutorialManager

- **Tipo, obiettivo e funzione del componente:** la classe TutorialManager si occupa della gestione dei collegamenti ai tutorial

- **Relazioni d'uso con altre componenti:** la classe `TutorialManager` utilizza `mytalk.server.data.TutorialData` per scrivere e leggere i collegamenti ai video tutorial e `mytalk.server.transfer.TutorialTransfer` per far giungere tali link al client
- **Attività svolte e dati trattati:** la classe si occuperà quindi di accedere ai collegamenti dei tutorial presenti nella base di dati e del loro invio alla classe `mytalk.server.transfer.TutorialTransfer`

6.2.4 `mytalk.server.usermanager.AuthenticationManager`

- **Tipo, obiettivo e funzione del componente:** la classe `AuthenticationManager` si occupa di stabilire la riuscita o il fallimento di un tentativo di autenticazione da parte di un utente
- **Relazioni d'uso con altre componenti:** la classe `AuthenticationManager` utilizza `mytalk.server.data.UserData` e comunica con `mytalk.server.transfer.AuthenticationTransfer`
- **Attività svolte e dati trattati:** la classe si occuperà quindi di dover controllare i dati presenti nel database in `mytalk.server.data.UserData` per verificare se un tentativo di autenticazione sia riuscito o meno, e restituire a `mytalk.server.transfer.AuthenticationTransfer` l'opportuna risposta, e per effettuare il logout

6.2.5 `mytalk.server.usermanager.UserDataManager`

- **Tipo, obiettivo e funzione del componente:** la classe `UserDataManager` si occupa delle operazioni di aggiornamento delle informazioni sugli utenti, quali nuove registrazioni e modifiche degli account
- **Relazioni d'uso con altre componenti:** la classe `UserDataManager` viene utilizzata da `mytalk.server.transfer.UserDataTransfer` e agisce su `mytalk.server.data.UserData`
- **Attività svolte e dati trattati:** la classe si occuperà quindi di ricevere le istruzioni da `mytalk.server.transfer.UserDataTransfer`, controlla ed eventualmente modifica la base di dati e ritorna a `mytalk.server.transfer.UserDataTransfer` il successo o meno dell'eventuale cambiamento

6.3 Data Layer

- **Tipo, obiettivo e funzione del componente:** lo strato Data rappresenta la base di dati su cui vengono salvati le informazioni persistenti. Questo include i dati degli utenti registrati, i link ai tutorial video che saranno salvati su un sito esterno e i messaggi registrati

- **Relazioni d'uso con altre componenti:** lo strato Data verrà utilizzato da alcune classi all'interno dei package `mytalk.server.usermanager` e `mytalk.server.functionmanager`

6.3.1 `mytalk.server.data.UserData`

- **Tipo, obiettivo e funzione del componente:** la classe `UserData` è un semplice contenitore di informazioni
- **Relazioni d'uso con altre componenti:** la classe `UserData` viene utilizzata da `mytalk.server.usermanager`; essendo un semplice contenitore di informazioni la classe non utilizza componenti
- **Attività svolte e dati trattati:** la classe preserva le informazioni riguardanti gli utenti, ovvero username, password, nome, cognome e indirizzo IP

6.3.2 `mytalk.server.data.RecordMessageData`

- **Tipo, obiettivo e funzione del componente:** la classe `RecordMessageData` è un semplice contenitore di informazioni
- **Relazioni d'uso con altre componenti:** la classe `RecordMessageData` viene utilizzata dalla classe `mytalk.server.functionmanager.RecordMessageManager`; essendo un semplice contenitore di informazioni la classe non utilizza componenti
- **Attività svolte e dati trattati:** la classe preserva i messaggi registrati da mandare in differita

6.3.3 `mytalk.server.data.TutorialData`

- **Tipo, obiettivo e funzione del componente:** la classe `TutorialData` è un semplice contenitore di informazioni
- **Relazioni d'uso con altre componenti:** la classe `TutorialData` è utilizzata da `mytalk.server.functionmanager.TutorialManager`; essendo un semplice contenitore di informazioni la classe non utilizza componenti
- **Attività svolte e dati trattati:** la classe `TutorialData` preserva i collegamenti ai tutorial video

7 Tracciamento componenti-requisiti

Componente	Classi	Requisiti
CCLI1	mytalk.client.view.LoginView mytalk.client.presenter.AuthenticationControl	RUFO 2 RUFO 2.1 RUFO 2.2
CCLI2	mytalk.client.view.ViewAccountDataView mytalk.client.view.SignUpView mytalk.client.view.ModifyAccountDataView mytalk.client.presenter.UserDataControl mytalk.client.model.User	RUFO 1 RUFO 1.1 RUFO 1.2 RUFO 1.3 RUFO 1.4 RUFO 1.5 RUFF 3 RUFF 3.1 RUFF 3.2 RUFF 3.3 RUFF 3.4 RUFF 3.5
CCLI3	mytalk.client.view.TutorialView mytalk.client.presenter.TutorialControl mytalk.client.model.Tutorial	RUFF 4 RUFF 4.1
CCLI4	mytalk.client.view.UserListView mytalk.client.presenter.UserListControl mytalk.client.model.Contact	RUFO 5 RUFF 5.1 RUFO 6.3
CCLI5	mytalk.client.view.NotificationView mytalk.client.presenter.NotificationControl	RUFF 7 RUFF 7.1 RUFF 7.2 RUFF 7.3 RUFF 7.4 RUFF 7.7 RUFF 7.8 RUFF 7.9
CCLI6	mytalk.client.view.IpCallView mytalk.client.view.AudioCallView mytalk.client.view.VideoCallView mytalk.client.view.CallStatisticsView mytalk.client.presenter.CallControl mytalk.client.presenter.StatisticsControl mytalk.client.model.Statistics	RUFO 6 RUFO 6.1 RUFO 6.2 RUFO 6.4 RUFO 6.5 RUFF 6.6 RUFF 6.7 RUF 6.8 RUF 6.9 RUF 6.13 RUFO 6.15
CCLI7	mytalk.client.view.RecordMessageView mytalk.client.presenter.RecordMessageControl	RUFF 6.10 RUFF 6.11

		RUFF 7.5 RUFF 7.6
CCLI8	mytalk.client.view.ChatView mytalk.client.presenter.ChatControl mytalk.client.model.TextMessage	RUFF 6.12
CCLI9	mytalk.client.view.SendFileView mytalk.client.presenter.SendFileControl	RUFD 6.14
CSER1	mytalk.server.transfer.AuthenticationTransfer mytalk.server.userManager.AuthenticationManager	RUFO2
CSER2	mytalk.server.transfer.UserDataTransfer mytalk.server.userManager.UserDataManager mytalk.server.data.UserData	RUFO 1 RUFO 1.1 RUFF 3 RUFF 3.1 RUFF 3.2 RUFF 3.3 RUFO 5 RUFF 5.1
CSER3	mytalk.server.functionmanager.CallManager mytalk.server.transfer.CallTransfer	RUFO 6 RUFO 6.1 RUFO 6.2 RUFO 6.3 RUFO 6.4 RUFO 6.5 RUFF 6.6 RUFF 6.7
CSER4	mytalk.server.transfer.RecordMessageTransfer mytalk.server.functionmanager.RecordMessageManager mytalk.server.data.RecordMessageData	RUFF 6.10 RUFF 6.11 RUFF 7.5 RUFF 7.6
CSER5	mytalk.server.functionmanager.TutorialTransfer mytalk.server.functionmanager.TutorialManager mytalk.server.data.TutorialData	RUFF 4 RUFF 4.1
CSER6	mytalk.server.transfer.NotificationTransfer	RUFF 7 RUFF 7.1 RUFF 7.2 RUFF 7.3 RUFF 7.4 RUFF 7.7 RUFF 7.8 RUFF 7.9

Tabella 1: Tracciamento tra componenti e requisiti

8 Tracciamento requisiti-componenti

Requisito	Componente Client	Componente Server	
RUFO 1	CCLI2	CSER2	
RUFO 1.1		-	
RUFO 1.2			
RUFO 1.3			
RUFO 1.4			
RUFO 1.5			
RUFO 2	CCLI1	CSER1	
RUFO 2.1		-	
RUFO 2.2			
RUFF 3		CSER1	
RUFF 3.1			
RUFF 3.2			
RUFF 3.3			
RUFF 3.4		-	
RUFF 3.5			
RUFF 4		CCLI3	CSER5
RUFF 4.1			
RUFO 5	CCLI4	CSER2	
RUFF 5.1			
RUFO 6	CCLI6	CSER3	
RUFO 6.1			
RUFO 6.2			
RUFO 6.3			
RUFO 6.4			
RUFO 6.5			
RUFF 6.6			
RUFF 6.7			
RUFD 6.8			
RUFD 6.9			
RUFF 6.10	CCLI7		-
RUFF 6.11			
RUFF 6.12	CCLI8		
RUFD 6.13	CCLI6		
RUFD 6.14	CCLI9		
RUFO 6.15	CCLI6		
RUFF 7	CCLI5	CSER6	
RUFF 7.1		-	
RUFF 7.2			

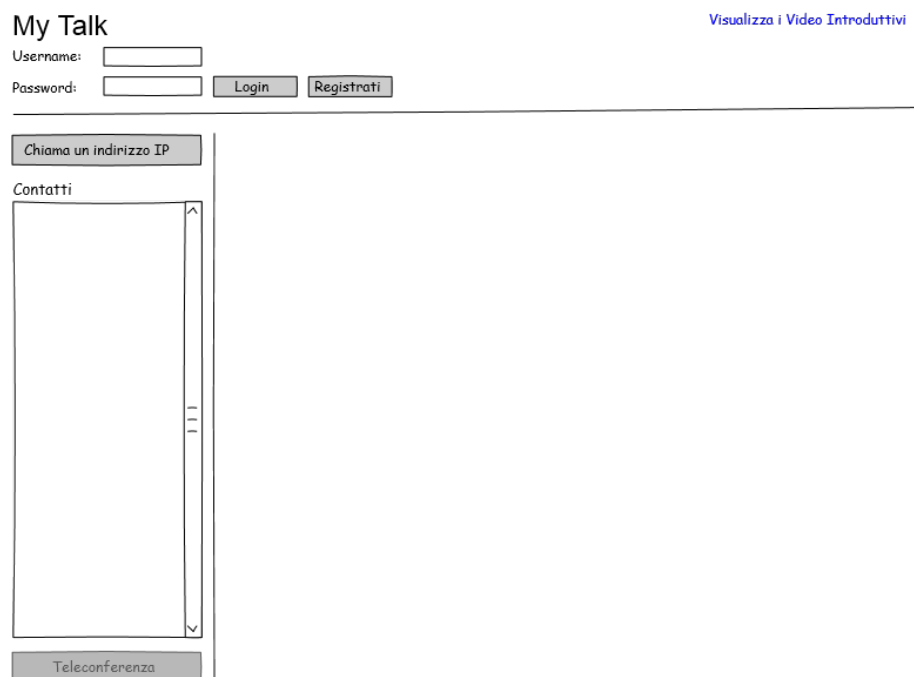
RUFF 7.3		
RUFF 7.4	CCLI7	CSER4
RUFF 7.5		
RUFF 7.6	CCLI5	CSER6
RUFF 7.7		
RUFF 7.8		-
RUFF 7.9		
RUFO 8	CCLI1	CSER1

Tabella 2: Tracciamento tra requisiti e componenti

La figura 7 rappresenta il diagramma della attività eseguibili dall'utente quando si avvia l'applicazione. Una volta aperta la pagina web del programma **MyTalk** l'utente potrà decidere di iscriversi al sito, se già non lo è, guardare i video tutorial riguardante il funzionamento del programma, effettuare l'accesso con le proprie credenziali oppure semplicemente contattare direttamente un utente se è a conoscenza del suo indirizzo IP, potendo così scrivere, effettuare una chiamata audio o video, cambiare modalità di chiamata e registrare la conversazione. Nel caso in cui l'utente decidesse di accedere inserendo dati corretti, avrà la possibilità, oltre di chiamare inserendo un indirizzo IP e vedere video tutorial, di poter modificare i dati relativi al proprio account, visualizzare i messaggi differiti in memoria nel server, accettare o rifiutare chiamate in ingresso e di poter selezionare i contatti della lista da cui sarà possibile verificarne i dati. Se si decidesse di selezionare un contatto, come per la chiamata IP, sarà possibile comunicarci testualmente o effettuare una chiamata audio e video o solo audio; sarà possibile registrare tale chiamata e sarà possibile inoltre inviare messaggi registrati che andranno caricati nel server che potranno quindi essere successivamente visualizzati dall'utente che si vuole contattare ma non presente in linea, inoltre sarà possibile inviare file. Sarà possibile anche selezionare più contatti contemporaneamente, in questo caso sarà possibile effettuare sia una comunicazione testuale o una chiamata video e audio o solo audio tra più utenti.

10 Prototipo interfaccia utente

In questo capitolo andremo a descrivere come sarà l'applicazione a livello visivo e le operazioni che l'utente avrà modo di effettuare.



The screenshot shows the 'My Talk' web application interface. At the top left, the text 'My Talk' is displayed. To its right is a blue link labeled 'Visualizza i Video Introduttivi'. Below the title, there are two input fields: 'Username:' and 'Password:'. To the right of the 'Password:' field are two buttons: 'Login' and 'Registrati'. A horizontal line separates the header from the main content area. On the left side of the main area, there is a vertical sidebar. At the top of the sidebar is a button labeled 'Chiama un indirizzo IP'. Below it is a section titled 'Contatti' which contains a large, empty rectangular box with a vertical scrollbar on its right side. At the bottom of the sidebar is a button labeled 'Teleconferenza'.

Figura 8: Pagina iniziale

Come si può notare dalla la figura 8 una volta acceduti al sito le uniche operazioni che potremo fare saranno la possibilità di effettuare chiamate sapendo l'indirizzo IP del destinatario, guardare i video tutorial riguardanti il prodotto **MyTalk** e ricevere infine le chiamate da parte di terzi. Guardando la figura 9 le operazioni che possiamo quindi eseguire sono le stesse nel caso in cui non si è autenticati ma è possibile chiamare gli utenti presenti nella lista e eseguire videoconferenze con più utenti.

MyTalk

[Visualizza i Video Introduttivi](#)

Ciao, Clockwork Team!

[Modifica dati](#)[Logout](#)

Chiama un indirizzo IP

Contatti

Contatto 1
Contatto 2
Contatto 3
Contatto 4
Contatto 5
Contatto 6
Contatto 7
Contatto 8
Contatto 9
Contatto 10

Teleconferenza

Figura 9: Pagina dopo login

La figura 10, la figura 11 e la figura 12 mostrano le varie schermate di comunicazioni possibili con i vari utenti. Presente anche la possibilità di registrare le comunicazioni in tempo reale, o inviare messaggi anche se l'utente non è connesso.

MyTalk

[Visualizza i Video Introduttivi](#)

Ciao, Clockwork Team!

Modifica dati

Logout

Chiama un indirizzo IP

Contatti

Contatto 1
Contatto 2
Contatto 3
Contatto 4
Contatto 5
Contatto 6
Contatto 7
Contatto 8
Contatto 9
Contatto 10

Teleconferenza

Stai chiamando Contatto 3

Disattiva video

Condividi schermata

Termina

☐ Registra chiamata

Chat

Clockwork Team: posso chiamarti?
Contatto 3: si, certo
Contatto 3: ma videochiamata o solo audio?
Clockwork Team: anche solo audio, tanto è una cosa veloce

Invia

Figura 10: Schermata videochiamata

MyTalk

[Visualizza i Video Introduttivi](#)

Ciao, Clockwork Team!

[Modifica dati](#)[Logout](#)

Chiama un indirizzo IP

Contatti

Contatto 1

Contatto 2

Contatto 3

Contatto 4

Contatto 5

Contatto 6

Contatto 7

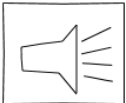
Contatto 8

Contatto 9

Contatto 10

Teleconferenza

Stai chiamando Contatto 3



Attiva video

Condividi schermata

Termina

☐ Registra chiamata

Chat

Clockwork Team: posso chiamarti?

Contatto 3: sì, certo

Contatto 3: ma videochiamata o solo audio?

Clockwork Team: anche solo audio, tanto è una cosa veloce

Invia

Figura 11: Schermata chiamata audio

MyTalk

[Visualizza i Video Introduttivi](#)

Ciao, Clockwork Team!

Modifica dati

Logout

Chiamata un indirizzo IP

Contatti

- Contatto 1
- Contatto 2
- Contatto 3
- Contatto 4
- Contatto 5
- Contatto 6
- Contatto 7
- Contatto 8
- Contatto 9
- Contatto 10

Teleconferenza

Contatto 3

Chiamata
Videochiamata
Condividi schermata
Invia un Videomessaggio

☐ Registra chiamata

Chat

Clockwork Team: posso chiamarti?
Contatto 3: sì, certo
Contatto 3: ma videochiamata o solo audio?
Clockwork Team: anche solo audio, tanto è una cosa veloce

Invia

Figura 12: Schermata comunicazione testuale

la figura 13 rappresenta la schermata di videoconferenza.

MyTalk

[Visualizza i Video Introduttivi](#)

Ciao, Clockwork Team!

Modifica dati

Logout

Chiama un indirizzo IP

Contatti

Contatto 1
Contatto 2
Contatto 3
Contatto 4
Contatto 5
Contatto 6
Contatto 7
Contatto 8
Contatto 9
Contatto 10

Teleconferenza

Disattiva video

Termina

☐ Registra chiamata

Chat

Clockwork Team: posso chiamarti?
Contatto 3: si, certo
Contatto 3: ma videochiamata o solo audio?
Clockwork Team: anche solo audio, tanto è una cosa veloce

Invia

Figura 13: Schermata Videoconferenza