Convertitore Audio  
Matvej Rossi

# Indice

1 Indice 2

2 Introduzione 3

2.1 Informazioni sul progetto 3

2.2 Abstract 3

2.3 Scopo 3

3 Analisi 4

3.1 Analisi del dominio 4

3.2 Analisi e specifica dei requisiti 4

3.3 Use case 7

3.4 Pianificazione 7

3.5 Analisi dei mezzi 9

3.5.1 Software 9

3.5.2 Hardware 9

4 Progettazione 9

4.1 Design dell’architettura del sistema 9

4.2 Design dei dati e database 9

4.3 Design delle interfacce 12

4.4 Design procedurale 12

5 Implementazione 13

6 Test 13

6.1 Protocollo di test 13

6.2 Risultati test 15

6.3 Mancanze/limitazioni conosciute 15

7 Consuntivo 15

8 Conclusioni 17

8.1 Sviluppi futuri 17

8.2 Considerazioni personali 17

9 Bibliografia 18

9.1 Bibliografia per articoli di riviste: 18

9.2 Bibliografia per libri 18

9.3 Sitografia 18

10 Glossario 19

11 Indice delle figure 19

12 Allegati 19

# Introduzione

## Informazioni sul progetto

* Data di inizio progetto: 04.09.2024
* Allievo: Matvej Rossi
* Scuola: SAMT
* Sezione: Informatica
* Materia: Modulo 306
* Docente responsabile: Geo Petrini
* Data di consegna: 18.12.24

## Abstract

Avere un programma locale che anche in modalità offline consente di convertire vari tipi di file audio ha vari campi di applicazione. Il programma può essere usato in varie situazioni e ha vari formati per la conversione e si ha la possibilità di inserire un’illimitata grandezza dei file.

Il programma è molto efficiente sia perché è di semplice utilizzo ma anche perché consente sia la conversione tra molteplici files e con la possibilità di scegliere la qualità e il formato di conversione.

## Scopo

Principalmente il progetto ha come scopo di facilitare la conversione tra file audio diversi (MP3, FLAC, WAV, OGG, AAC, M4A, …) e così semplificare la ricerca all’utente del file audio del tipo corretto. Attraverso ad una interfaccia grafica di semplice utilizzo l’utente può facilmente convertire molteplici file audio nel tipo di file che preferisce. Il progetto è a scopo didattico. L’utilizzo di questo tipo di programmi è molto comune e utile, anche con una gamma di formati audio ridotta.

# Analisi

## Analisi del dominio

Il prodotto funziona su GUI / CMD e viene utilizzato nel bisogno di convertire un file audio nel tipo corretto. Esistono sul Web molteplici prodotti simili a questo, tutti creati per aiutare gli utenti nel sostituire il tempo di ricerca del file corretto nel tempo di aspettare di scaricare un semplice file del formato corretto, il mio prodotto serve ad aiutare l’utente localmente. L’utente ha il semplice bisogno di avere il file del formato corretto, possono essere dall’utente comune o un’utente con competenze informatiche avanzate. Per poter realizzare questo prodotto serve la conoscenza riguardo a come sono creati i vari formati di file, cosa contengono e la conoscenza del bitrate dei file audio.

## Analisi e specifica dei requisiti

L’applicazione base fatta su command-line dovrà avere principalmente i seguenti requisiti:

* Realizzazione dell’input per l’inserimento dei file e cartelle.
* Filtrazione dei files/cartelle sorgenti.
* Realizzazione dell’output.
* Scelta della qualità dell’audio.
* Scelta della cartella di destinazione.
* Controllare se il file di output è già esistente e far scegliere all’utente cosa fare in quel caso
* Controllo dell’avanzamento della conversione (gestire eventuali errori nel corso della conversione)
* Gestire eventuali errori.
* Gestione del file di configurazione.
* Controllo compatibilità file.
* Creazione della GUI

Il programma parte tramite CMD e utilizza la libreria FFMPEG che verrà scaricata dall’utente quando scaricherà il programma. Consente la conversione di file audio, funziona come un classico programma di conversione ma in questo caso il programma senza l’interfaccia grafica resta non molto “user friendly” ma molto diretto nel suo utilizzo, quindi consente la scelta della cartella di destinazione e consente di convertire non solo un singolo file ma una cartella intera. Il file di configurazione del programma è utile perché serve a memorizzare le scelte dell’utente.

|  |  |
| --- | --- |
| **Req-01** | |
| **Nome** | Realizzazione dell’input per i file e cartelle sorgenti. |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | L’input può essere sia di un file singolo che una cartella. |

|  |  |
| --- | --- |
| **Req-02** | |
| **Nome** | Filtrare l’input (sia file singolo che cartella) |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Senza il Req-01 non è possibile realizzarlo  I tipi di file da filtrare sono: MP3, FLAC, WAV, OGG, AAC, M4A  Altri tipi di file non sono accettati. |
| **Sotto requisiti** | |
| **Req-02\_1** | Filtrare l’input su file |
| **Req-02\_2** | Filtrare l’input su cartella/e |
| **Req-02\_3** | Consentire all’utente di scegliere la filtrazione del tipo di file. |

|  |  |
| --- | --- |
| **Req-03** | |
| **Nome** | Realizzare l’output con i file convertiti |
| **Priorità** | 2 |
| **Versione** | 2.0 |
| **Note** | Fa vedere a schermo e anche nella conversione il risultato della conversione, se essa ha funzionato o no. |
| **Sotto requisiti** | |
| **Req-03\_1** | Realizzare il salvataggio dei file convertiti localmente. |
| **Req-03\_2** | Consentire all’utente della scelta del tipo di file per l’output. |
| **Req-03\_3** | Controllo dei permessi di scrittura nella cartella di destinazione. |

|  |  |
| --- | --- |
| **Req-04** | |
| **Nome** | Consentire scelta del Bitrate all’utente. |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Fa scegliere all’utente la qualità del file di output se ne ha la possibilità. |

|  |  |
| --- | --- |
| **Req-05** | |
| **Nome** | Consentire scelta della cartella di destinazione dei files. |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | La possibilità di scegliere tramite CMD (poi GUI) la cartella di destinazione.  (Default: cartella sorgente) |

|  |  |
| --- | --- |
| **Req-06** | |
| **Nome** | Controllo se il file output è già esistente nella cartella di destinazione. |
| **Priorità** | 2 |
| **Versione** | 2.0 |
| **Note** | - |
| **Sotto requisiti** | |
| **Req-04\_1** | Sovrascrivere il file con lo stesso nome se già ne esiste uno. |
| **Req-04\_2** | Sovrascrivere file / files in base alla scelta dell’utente. |

|  |  |
| --- | --- |
| **Req-07** | |
| **Nome** | Controllo dell’avanzamento della conversione. |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | - |

|  |  |
| --- | --- |
| **Req-08** | |
| **Nome** | Gestione degli eventuali errori |
| **Priorità** | 2 |
| **Versione** | 2.0 |
| **Note** | Errori che possono essere generati tramite input, output o altro.  Testo dell’errore deve essere comprensibile all’utente. |

|  |  |
| --- | --- |
| **Req-09** | |
| **Nome** | Gestione e aggiunta del file di configurazione |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Il file serve per contenere un “pattern” per il salvataggio del nome dei files, il formato di Output dei file, la qualità (bitrate) e la scelta di cosa fare in caso di errore nella conversione. |

|  |  |
| --- | --- |
| **Req-10** | |
| **Nome** | Controllo compatibilità dei file |
| **Priorità** | 1 |
| **Versione** | 2.0 |
| **Note** | I file che vengono presi dall’input devono essere controllati per gestire eventuali errori che possono succedere durante la conversione.  Il controllo di compatibilità viene fatto tramite la libreria o tramite il Magic Number. |

|  |  |
| --- | --- |
| **Req-11** | |
| **Nome** | Creazione dell’interfaccia grafica. |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | - |

## Use case

L’attore principale che comunica con il sistema è l’utente finale che utilizza il programma.

Il sistema quindi prevede solamente l’utente come attore attivo che rispetta i requisiti.

Come allegato ho inserito il Visio che spiega tutti gli Use-Case.

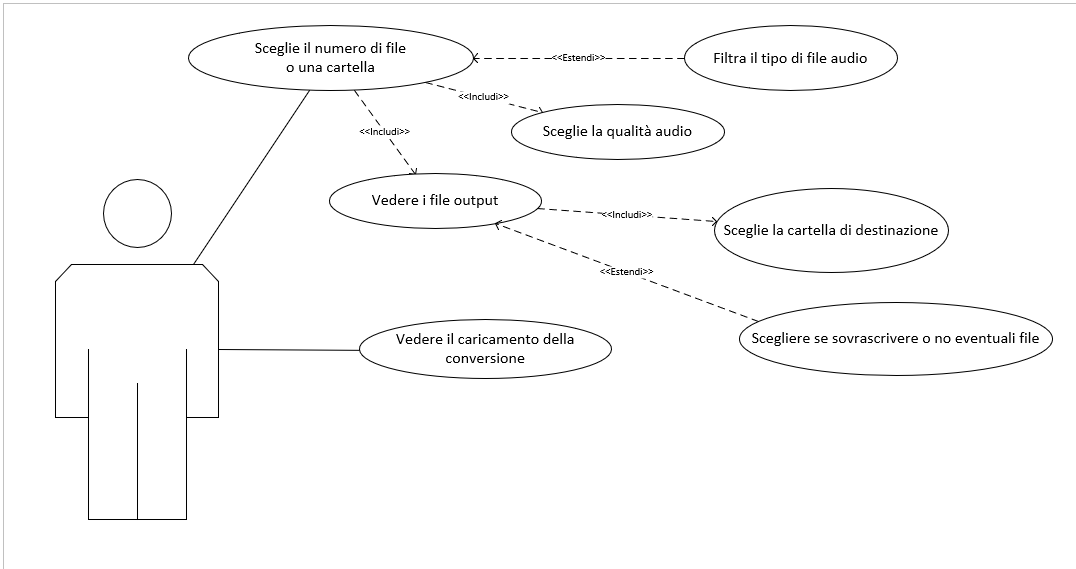


Figura 1: Use Case

## Pianificazione

La pianificazione del mio progetto è stata rappresentata mediante un diagramma di Gantt, dove vede come data di partenza del progetto il 04.09.24 e come fine il 18.12.2024. Immagine che contiene testo, schermata, numero, Parallelo

Descrizione generata automaticamenteFigura 2 Gantt Preventivo

## Analisi dei mezzi

Per la creazione del progetto mi è servito solamente un IDE e l’accesso a Internet.

### Software

Di seguito ci sono tutti i software e le relative versioni che mi sono servite per realizzare il progetto e la documentazione:

* Apache NetBeans IDE 20
* JDK: 19.0.2
* Word: 2019
* PlantUML
* Visual Studio Code

### Hardware

Per eseguire il programma basta che su qualsiasi dispositivo, indipendentemente dal sistema operativo, sia installato:

* FFMPEG
* JDK (Almeno versione 19)

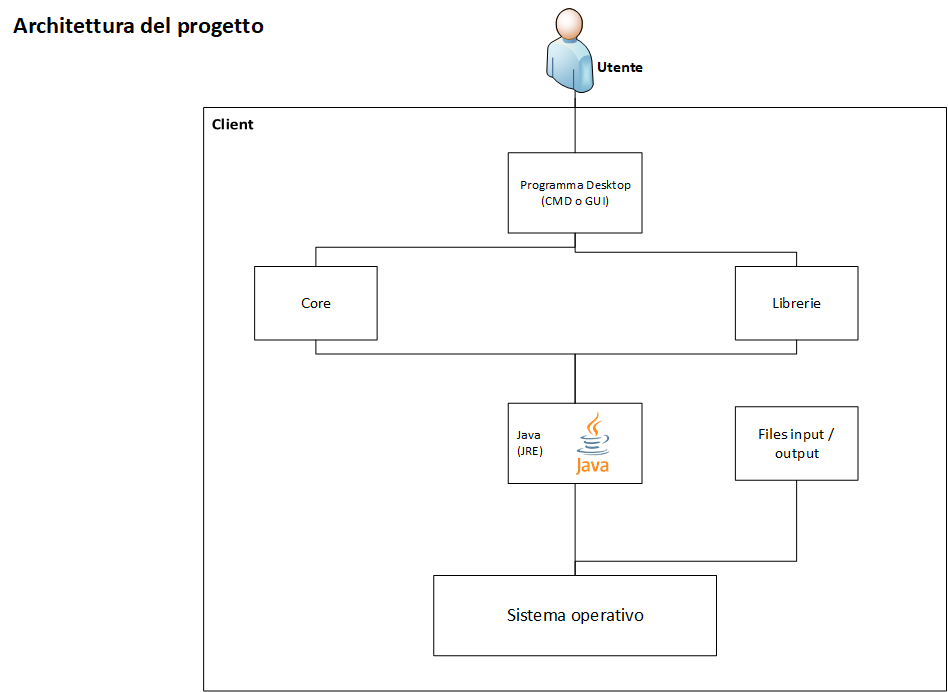
Il programma usa una particolare potenza quindi un qualsiasi PC odierno anche di bassa qualità può utilizzare il programma. Il programma può funzionare su qualsiasi sistema operativo e non presenta limitazioni.

Durante lo sviluppo, il PC scolastico presentava queste specifiche:

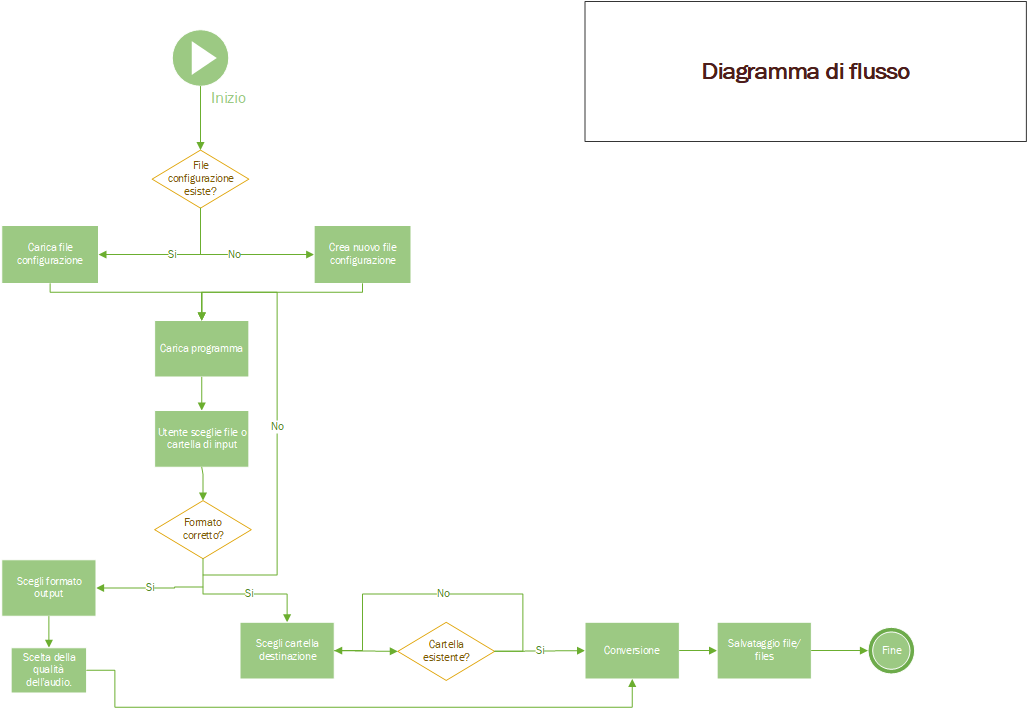
*DA AGGIUNGERE*

# Progettazione

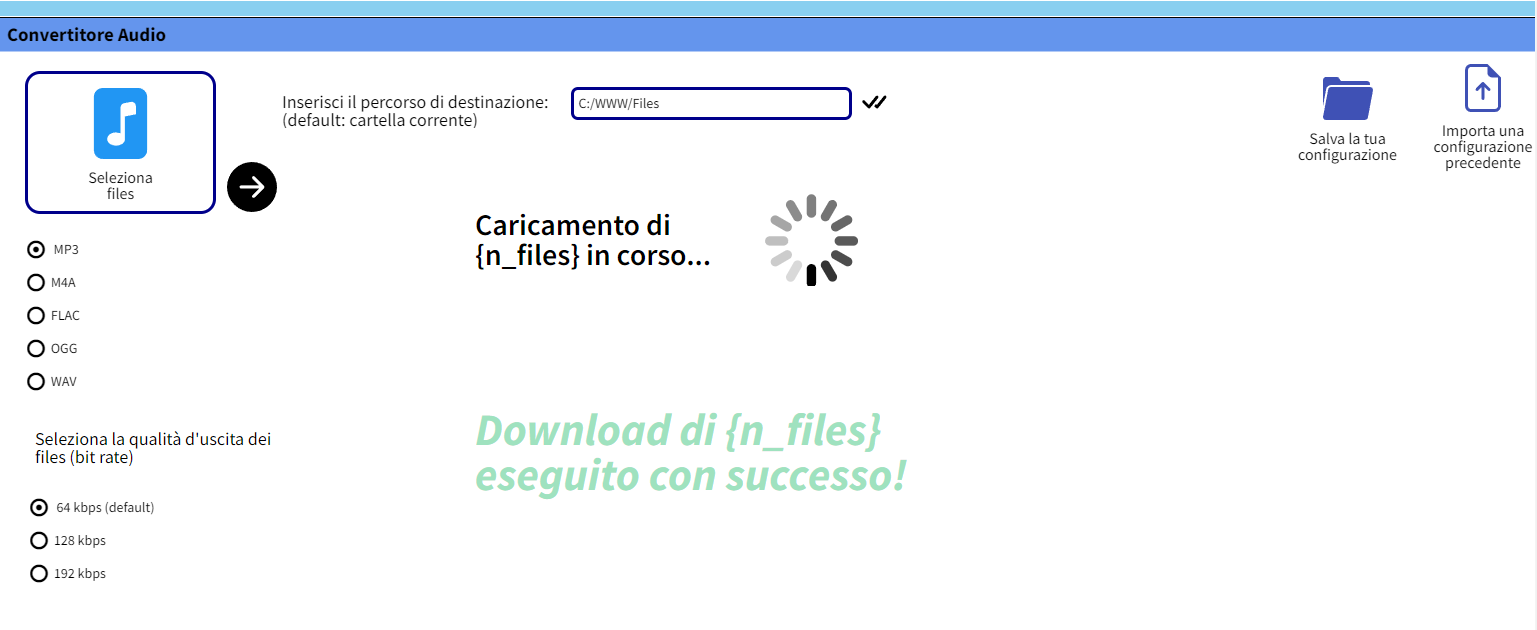
## Design dell’architettura del sistema



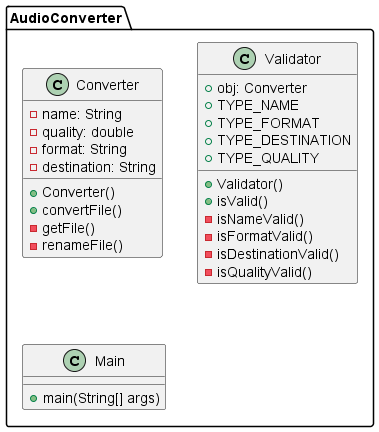
## Diagramma di flusso



## Design delle interfacce



## Design delle classi



# Implementazione

Il progetto è stato creato utilizzando Java, con l’utilizzo di 3 classi:

* **Classe Main:** classe che si occupa di gestire tramite il classico metodo main() il ricevimento dei parametri attraverso gli “args” di Java.
* **Classe Converter**: crea l’oggetto **Converter** che mi permette attraverso i suoi attributi e i metodi privati e pubblici di prendere e memorizzare temporaneamente le informazioni del Main.
* **Classe Validator:** si occupa di controllare i dati dell’oggetto Converter e fa da ultimo step la parte principale quella di eseguire il comando passato dall’utente e passarlo ad FFMPEG che farà la conversione per noi.

Poi ho scelto di utilizzare FFMPEG come programma esterno perché penso sia la soluzione più utile che fa al caso per una applicazione desktop; quindi, il mio programma di per sé fa da Wrapper attorno a FFMPEG.

## Classe Converter

Prima di tutto ho creato la classe Converter che contiene questi attributi:  
Immagine che contiene testo, Carattere, linea, schermata

Descrizione generata automaticamente

Contiene anche tutti i getter e setter più un costruttore di default e un costruttore con tutti gli argomenti: Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

## Classe Validator

La classe Validator come detto prima si occupa di controllare ed eseguire i dati ricevuti dalla classe Converter. Ha 3 metodi principali molto autoesplicativi: **isValidConverter(), buildFFMPEGcommand(), executeCommand()**. Questi 3 metodi vengono racchiusi in un metodo unico chiamato **convertAudio()** che viene richiamato dalla classe Main.

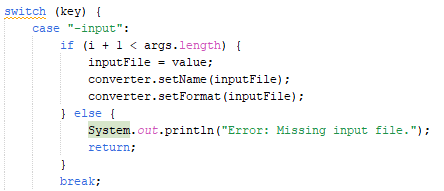
* isValidConverter: ha il compito di controllare tutti gli input ricevuti tramite “args” dalla classe Main attraverso l’oggetto **Converter**, metodo strutturato principalmente su una base di “if”
* buildFFMPEGCommand: questo metodo letteralmente “costruisce” il comando FFMPEG, quindi, richiama il programma e costruisce la stringa da passargli.
* executeCommand(): questo programma attraverso la classe ProcessBuilder di Java

prende la stringa del comando e la fa eseguire a Java attraverso un processo.

## Classe Main

La classe Main è la classe principale che alla fine richiama tutte e 2 le classi.

Attraverso un ciclo **for** controlla ogni argomento con uno **switch:**



Alla fine del for e dello switch viene passato il tutto alla classe Validator tramite il suo metodo **convertAudio(converter).**

# Test

## Protocollo di test

Definire in modo accurato tutti i test che devono essere realizzati per garantire l’adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-001  REQ-012 | **Nome** | Import a card with KIC, KID and KIK keys, but not shown with the GUI |
| **Descrizione** | Import a card with KIC, KID and KIK keys with no obfuscation, but not shown with the GUI | | |
| **Prerequisiti** | Store on local PC: Profile\_1.2.001.xml (appendix n\_n) and Cards\_1.2.001.txt (appendix n\_n).  PIN (OTA\_VIEW\_PIN\_PUK\_KEY) and ADM (OTA\_VIEW\_ADM\_KEY) user right not set. | | |
| **Procedura** | 1. Go to “Cards manager” menu,  in main page click “Import Profiles” link, Select the “1.2.001.xml” file, Import the Profile 2. Go to “Cards manager” menu,  in main page click “Import Cards” link, Select the “1.2.001.txt” file, Delete the cards,  Select the “1.2.001.txt” file, Import the cards 3. Research the “41795924770” Card, Click the imsi card link Check the card details 4. Execute the SQL: SELECT imsi, dir, keyset, cntr, rawtohex(kickey), rawtohex(kidkey), rawtohex(kikkey), rawtohex(chv), rawtohex(dap)FROM otacardkey a where imsi='340041795924770' ORDER BY keyset; | | |
| **Risultati attesi** | Keys visible in the DB (OtaCardKey) but not visible in the GUI (Card details) | | |

## Risultati test

Tabella riassuntiva in cui si inseriscono i test riusciti e non del prodotto finale. Se un test non riesce e viene corretto l’errore, questo dovrà risultare nel documento finale come riuscito (la procedura della correzione apparirà nel diario), altrimenti dovrà essere descritto l’errore con eventuali ipotesi di correzione.

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Risultato ottenuto** | **Stato** |
| TC-001 | Testo o immagine a conferma del test | Passato |
| TC-002 | Testo o immagine a conferma del test | Fallito |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

# Consuntivo

Consuntivo del tempo di lavoro effettivo e considerazioni riguardo le differenze rispetto alla pianificazione (cap. 1.7) (ad esempio Gantt consuntivo).

Stile orizzontale



CONSUNTIVO

Figura Esempio di diagramma di Gantt consuntivo

# Conclusioni

Il programma si è rivelato arduo da implementare, non come me l’aspettavo. Sono convinto che il progetto alla fine sia un buon prodotto, anche se vari requisiti sono stati messi da parte per colpa di vari fattori come il tempo o la difficoltà. Grazie a questo progetto mi sono reso conto di un aspetto fondamentale in quanto la “teoria”, ovvero la fase di preparazione del codice, sia una delle parti più complicate e che secondo me richiede molta attenzione.

Ho avuto anche la possibilità di sviluppare ulteriori conoscenze riguardo ai vari File Audio e le proprie particolarità.

## Sviluppi futuri

Molte migliori che si possono applicare al prodotto sono già elencate come requisiti visto che certi non ho avuto la possibilità nel realizzarli.

## Considerazioni personali

Cosa ho imparato in questo progetto? ecc.

# Bibliografia

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

**Esempio:**

* http://standards.ieee.org/guides/style/section7.html, *IEEE Standards Style Manual*, 07-06-2008.

# Glossario

|  |  |
| --- | --- |
| **Termine** | **Significato** |
| ASCII | American Standard Code for Information Interchange, codice per la codifica di caratteri. |
|  |  |
|  |  |
|  |  |
|  |  |
| ZFS | Zettabyte File System, filesystem opernsource a 128 bit con funzionalità avanzate. |

# Indice delle figure

[Figura 1: Use Case 8](#_Toc185367015)

[Figura 2 Gantt Preventivo 9](#_Toc185367016)

[Figura 3 Esempio di diagramma di Gantt consuntivo 18](#_Toc185367017)

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o QdC
* Prodotto
* …