SiVoDiM

Sintesi Vocale per Dispositivi Mobili

Piano di Qualifica

Versione	3.0.0
Redattori	Alberto Andriolo Enrico Chiara Gino Zaidan
Verificatori	Riccardo Rizzo
Responsabili	Federico Rossetto
Uso	Esterno
Lista di distribuzione	Stark Labs Prof. Tullio Vardanega, Prof. Riccardo Cardin

Documento riguardante l'insieme di strategie di verifica adottate dal gruppo Stark Labs per il conseguimento di requisiti qualitativi per il progetto SiVoDiM

Registro delle modifiche

Versione	Data	Attività	Autori
3.0.0	10/06/2016	Accettazione	Federico Rossetto
2.1.0	10/06/2016	Verifica di modifiche e incrementi apportati	Riccardo Rizzo
2.0.4	09/06/2016	Incrementata appendice E con nuovi risultati delle metriche	Alberto Andriolo
2.0.3	04/06/2016	Aggiunti test di unità in appendice D	Gino Zaidan
2.0.2	01/06/2016	Rimosso capitolo 4 e spostati i contenuti in appendice E	Enrico Chiara
2.0.1	01/06/2016	Spostati i contenuti di appendice E in appendice F	Enrico Chiara
2.0.0	15/05/2016	Accettazione	Riccardo Rizzo
1.2.0	14/05/2016	Verifica	Alberto Andriolo
1.1.5	13/05/2016	Appendice D: Aggiunti test di Integrazione	Francesco Bizzaro
1.1.4	11/05/2016	Appendice D: Aggiunti test di Validazione	Gino Zaidan
1.1.3	09/05/2016	Aggiunta appendice E	Enrico Chiara
1.1.2	09/05/2016	Appendice D: Aggiunti test di Sistema	Alberto Andriolo
1.1.1	09/05/2016	Correzione errori rilevati nelle appendici	Enrico Chiara
1.1.0	03/05/2016	Appendici A, B e C: verifica	Enrico Chiara
1.0.4	28/04/2016	Appendici A, B e C: stesura contenuti	Federico Rossetto
1.0.3	27/04/2016	Sezione 3: stesura	Riccardo Rizzo, Enrico Chiara
1.0.2	26/04/2016	Sezione 2: stesura	Riccardo Rizzo, Enrico Chiara
1.0.1	26/04/2016	Riorganizzazione struttura del documento e creazione di appendici	Riccardo Rizzo, Enrico Chiara
1.0.0	31/03/2016	Accettazione	Enrico Chiara

0.2.0	31/03/2016	Verifica	Riccardo Rizzo
0.2.0	31/03/2016	Calcolo indice Gulpease dei documenti	Federico Rossetto
0.1.1	24/03/2016	Correzione errori	Federico Rossetto
0.1.0	23/03/2016	Verifica	Riccardo Rizzo
0.0.4	19/03/2016	Stesura standard di qualità	Federico Rossetto
0.0.3	18/03/2016	Stesura gestione amministrativa	Federico Rossetto
0.0.2	12/03/2016	Stesura strategie di verifica	Federico Rossetto
0.0.1	11/03/2016	Stesura della struttura	Federico Rossetto

Indice

1 Introduzione **1**
1.1 Scopo del documento . . . 1
1.2 Scopo del progetto . . . 1
1.3 Glossario . . . 1
1.4 Riferimenti . . . 1
1.4.1 Normativi . . . 1
1.4.2 Informativi . . . 1

2 Visione generale della strategia di gestione della qualità **3**
2.1 Definizione obiettivi . . . 3
2.1.1 Qualità di processo . . . 3
2.1.1.1 Miglioramento continuo - OQPC1 . . . 3
2.1.1.2 Quantificazione dei ritardi - OQPC2 . . . 3
2.1.1.3 Quantificazione dei costi - OQPC3 . . . 4
2.1.2 Qualità di prodotto . . . 4
2.1.2.1 Indice di qualità dei documenti - OQPDD1 . . . 4
2.1.2.2 Qualità del software . . . 5
2.1.2.2.1 Copertura dei requisiti obbligatori - OQPDS1 . . . 5
2.1.2.2.2 Copertura dei requisiti desiderabili - OQPDS2 . . . 5
2.1.2.2.3 Manutenibilità e comprensibilità del codice - OQPDS3 5
2.1.2.2.4 Superamento dei test - OQPDS4 . . . 7
2.1.2.2.5 Robustezza - OQPDS5 . . . 7
2.1.2.2.6 Funzionamento privo di interruzioni - OQPDS6 . . . 7
2.2 Organizzazione . . . 7
2.3 Pianificazione strategica e temporale . . . 8
2.4 Responsabilità . . . 8

3 Visione di dettaglio della strategia di gestione della qualità **9**
3.1 Risorse . . . 9
3.2 Misure e metriche . . . 9
3.2.1 Metriche per i processi . . . 9
3.2.1.1 Software Process Improvement and Capability Determination - MPC1 . . . 9
3.2.1.2 Schedule Variance (SV) - MPC2 . . . 9
3.2.1.3 Budget Variance (BV) - MPC3 . . . 10
3.2.2 Metriche per i prodotti . . . 10
3.2.2.1 Metriche per i documenti . . . 10
3.2.2.1.1 Indice di Gulpease - MPDD1 . . . 10
3.2.2.2 Metriche per il software . . . 10
3.2.2.2.1 Copertura dei requisiti obbligatori - MPDS1 . . . 10
3.2.2.2.2 Copertura dei requisiti desiderabili - MPDS2 . . . 11
3.2.2.2.3 Numero di livelli di annidamento - MPDS3 . . . 11
3.2.2.2.4 Numero di attributi per classe - MPDS4 . . . 11
3.2.2.2.5 Numero di parametri per metodo - MPDS5 . . . 11
3.2.2.2.6 Linee di codice per linee di commento - MPDS6 . 12
3.2.2.2.7 Copertura di codice - MPDS7 . . . 12
3.2.2.2.8 Percentuale di test superati - MPDS8 . . . 12

3.2.2.2.9 Failure Avoidance - MPDS9 . . . 12
3.2.2.2.10 Breakdown Avoidance - MPDS10 . . . 13
3.3 Tecniche di analisi . . . 13
3.3.1 Analisi statica . . . 13
3.3.1.1 Walkthrough . . . 13
3.3.1.2 Inspection . . . 13
3.3.2 Analisi dinamica . . . 13
3.3.2.1 Test di unità . . . 14
3.3.2.2 Test di integrazione . . . 14
3.3.2.3 Test di sistema . . . 14
3.3.2.4 Test di regressione . . . 14
3.3.2.5 Test di validazione . . . 14

A Standard ISO/IEC 15504 15

B Ciclo di Deming 17

C Standard ISO/IEC 9126 18
C.1 Modello della qualità del software . . . 18
C.2 Modello della qualità in uso . . . 20
C.3 Metriche per la qualità del software . . . 20
C.3.1 Metriche per la qualità esterna . . . 20
C.3.2 Metriche per la qualità interna . . . 20
C.3.3 Metriche per la qualità in uso . . . 20

D Pianificazione dei test 21
D.1 Test di Sistema . . . 21
D.1.1 Descrizione dei test di Sistema . . . 21
D.2 Test di Integrazione . . . 24
D.2.1 Descrizione dei test di Integrazione . . . 24
D.2.2 Tracciamento componenti-test di Integrazione . . . 26
D.3 Test di Unità . . . 28
D.3.1 Descrizione dei test di Unità . . . 28
D.4 Test di Validazione . . . 43
D.4.1 Descrizione dei test di Validazione . . . 43

E Resoconto delle attività di verifica 53
E.1 Periodo di Analisi . . . 53
E.1.1 Dettaglio della verifica di analisi . . . 53
E.1.1.1 Verifica sui processi . . . 53
E.1.1.2 Verifica sui prodotti . . . 54
E.1.1.2.1 Documenti . . . 54
E.1.1.3 Esito della revisione di analisi . . . 55
E.2 Periodo di Analisi di Dettaglio . . . 55
E.2.1 Dettaglio della verifica di analisi di dettaglio . . . 55
E.2.1.1 Verifica sui processi . . . 55
E.2.1.2 Verifica sui prodotti . . . 55
E.2.1.2.1 Verifica sui documenti . . . 55
E.3 Periodo di Progettazione Architetturale . . . 56
E.3.1 Dettaglio della verifica di progettazione architetturale . . . 56

E.3.1.1 Verifica sui processi 56
E.3.1.2 Verifica sui prodotti 57
E.3.1.2.1 Verifica sui documenti 57
E.4 Periodo di Progettazione di dettaglio e codifica 57
E.4.1 Dettaglio della progettazione di dettaglio e codifica 57
E.4.1.1 Verifica sui processi 57
E.4.1.2 Verifica sui prodotti 59
E.4.1.2.1 Verifica sui documenti 59
E.4.1.2.2 Verifica sul codice 59

F Tracciamento obiettivi di qualità - Metriche 60

Elenco delle tabelle

2 Test di Sistema 23
3 Test di Integrazione 26
4 Tracciamento Componenti - test di Integrazione 27
5 Test di Unità 42
6 Test di Validazione 52
7 Metriche dei processi 53
8 Indici Gulpease dei documenti 54
9 Metriche dei processi 55
10 Indici Gulpease dei documenti 56
11 Metriche dei processi 56
12 Indici Gulpease dei documenti 57
13 Metriche dei processi 57
14 Indici Gulpease dei documenti 59
15 Metriche sul codice del package Libraries 59
16 Metriche sul codice del package Drama 59
17 Copertura dei requisiti 60
18 Tracciamento obiettivi di qualità - Metriche 60

Elenco delle figure

1 Ciclo di Deming 17

1 Introduzione

1.1 Scopo del documento

Questo documento contiene le strategie che il gruppo di lavoro ha deciso di adottare per raggiungere gli obiettivi qualitativi nel progetto **SiVoDiM**. All'interno di tale ottica, è necessario un continuo processo di verifica, con lo scopo di correggere tempestivamente e senza spreco di risorse le anomalie riscontrate sulle attività svolte.

1.2 Scopo del progetto

Lo scopo del progetto risiede nello sviluppo di un'applicazione utile a dimostrare efficacemente le potenzialità del motore di sintesi vocale FA-TTS$_G$, realizzato dall'azienda MIVOQ s.r.l. e messo a disposizione del gruppo di lavoro. Si devono realizzare due applicazioni per sistemi Android$_G$:

- **Applicazione di configurazione**: deve permettere all'utente di interfacciarsi direttamente con il sistema operativo per configurare, salvare e modificare le voci ereditate dal motore di sintesi FA-TTS di MIVOQ;
- **Applicazione per la creazione di sceneggiati**: permette la creazione e il salvataggio di racconti e sceneggiati, che possono essere esportati in formato audio attraverso l'utilizzo del motore FA-TTS.

Entrambe le applicazioni devono interfacciarsi con un modulo di basso livello:

- **Modulo di sistema**: permette di interfacciarsi tramite connessione di rete al motore FA-TTS. Fornisce una libreria contenente tutte le funzionalità offerte da FA-TTS, utile nell'ottica di un riuso futuro del *software*.

Lo sviluppo di tutte e quattro le suddette componenti è a carico del gruppo Stark Labs.

1.3 Glossario

Al fine di aumentare la comprensione del testo ed evitare eventuali ambiguità, viene fornito un glossario (*Glossario v2.0.0*) contenente le definizioni degli acronimi e dei termini tecnici utilizzati nel documento. Ogni vocabolo contenuto nel glossario è contrassegnato dal pedice "$_G$".

1.4 Riferimenti

1.4.1 Normativi

- *Norme di Progetto v2.0.0*;
- **Capitolato C6** – SiVoDiM: Sintesi Vocale per Dispositivi Mobili.

1.4.2 Informativi

- *Glossario v2.0.0*;
- *Piano di Progetto v2.0.0*;
- **Slide dell'insegnamento Ingegneria del Software modulo A**:

  - Ingegneria dei requisiti;
  - Diagrammi dei casi d'uso
    http://www.math.unipd.it/~tullio/IS-1/2015/.
- Software Engineering - Ian Sommerville - 9th Edition 2010:
  - Chapter 24: Quality management;
  - Chapter 25: Configuration management;
  - Chapter 26: Process improvement.
- SWEBOK - Version 3 (2004): capitolo 11 - Software Quality;
- IEEE 830-1998: https://en.wikipedia.org/wiki/Software_requirements_specification;
- Indice Gulpease: http://it.wikipedia.org/wiki/Indice_Gulpease.

2 Visione generale della strategia di gestione della qualità

2.1 Definizione obiettivi

Di seguito vengono descritti gli obiettivi di qualità che il gruppo si impegna a raggiungere, relativi al prodotto commissionato e ai processi necessari per una sua corretta realizzazione. Per ogni obiettivo si farà uso di standard, modelli e metriche al fine di valutarne l'effettivo raggiungimento. Per ogni criterio di valutazione utilizzato vengono definiti dei valori minimi che si intende raggiungere nell'arco dell'intero progetto. Inoltre vengono definiti dei valori ottimali che dovranno essere sperabilmente raggiunti.

2.1.1 Qualità di processo

Per garantire la qualità del prodotto si deve perseguire la qualità dei processi che lo definiscono. Assicurare la qualità dei processi permette di ottimizzare l'uso delle risorse, contenere i costi e migliorare la stima dei rischi. A fronte di tali motivazioni, desideriamo che i processi abbiano le seguenti caratteristiche:

- Un processo dovrebbe essere in grado di migliorare continuamente le proprie *performance*;
- Le attività di un processo dovrebbero rispettare i tempi indicati nel *Piano di Progetto v2.0.0*;
- I costi di ogni processo dovrebbero rispettare quanto dichiarato nel *Piano di Progetto v2.0.0*.

Di seguito vengono riportati gli obiettivi quantitativi che il gruppo Stark Labs intende raggiungere.

2.1.1.1 Miglioramento continuo - OQPC1 Si è deciso di adottare il modello SPICE$_G$ per quantificare la capacità dei processi di misurare le proprie *performance* e di porsi obiettivi quantitativi di miglioramento. In particolare si desidera raggiungere almeno il livello 2 previsto dalla scala, mentre l'obiettivo ottimale da raggiungere è il livello 4.
Per una descrizione più dettagliata del modello SPICE fare riferimento all'appendice A Standard ISO/IEC 15504.
Criteri utilizzati:

- **Modello**: SPICE;
- **Soglia di accettazione**: livello 2 previsto da SPICE;
- **Soglia ottimale**: livello 4 previsto da SPICE.

2.1.1.2 Quantificazione dei ritardi - OQPC2 Per quantificare i ritardi in relazione a quanto pianificato nel Piano di Progetto, viene utilizzata la metrica *Schedule Variance* (SV). Attraverso la *Schedule Variance* si può valutare se si è in linea, in anticipo o in ritardo rispetto alla pianificazione delle *milestone*$_G$. Se il valore di *Schedule Variance* è SV>0 significa che il team sta procedendo con maggior velocità rispetto alla pianificazione, viceversa se negativo.
Criteri utilizzati:

- **Metrica**: Schedule Variance;

- **Range di accettazione**: [> -(Costo preventivo fase x 5%)];
- **Range ottimale**: [>=0].

2.1.1.3 Quantificazione dei costi - OQPC3 Per controllare se i costi di un processo rientrano nel budget preventivato nel Piano di Progetto, viene utilizzata la metrica *Budget Variance* (BV). Attraverso la *Budget Variance* si valuta, nella data corrente, la differenza tra la spesa attuale e il costo pianificato. Se il valore di *Budget Variance* è BV>0 significa che si sta consumando il budget più lentamente di quanto previsto, viceversa se negativo.
Criteri utilizzati:

- **Metrica**: Budget Variance;
- **Range di accettazione**: [> -(Costo preventivo fase x 10%)];
- **Range ottimale**: [>=0].

2.1.2 Qualità di prodotto

Per aumentare il valore commerciale di un prodotto *software*, e per garantire il corretto funzionamento dello stesso, è necessario fissare degli obiettivi qualitativi e verificare che questi vengano rispettati. Lo standard ISO_G/IEC_G 9126 è stato redatto allo scopo di definire questi obiettivi e delineare alcune metriche capaci di misurare il raggiungimento degli stessi.
Nei paragrafi che seguono sono esposti gli obiettivi che il gruppo Stark Labs intende raggiungere, suddivisi per due tipologie di prodotto: documenti e *software*. Per ogni obiettivo vengono specificati i criteri con i quali si effettuano le misurazioni sulla qualità.

2.1.2.1 Indice di qualità dei documenti - OQPDD1 Per misurare la qualità della documentazione prodotta si valuta il valore ottenuto dal calcolo dell'indice $Gulpease_G$. Esso è un indice di leggibilità tarato sulla lingua italiana. Viene preferito rispetto ad altri poiché utilizza la lunghezza in lettere anziché in sillabe, semplificandone il calcolo. Tale indice evidenzia la complessità dello stile del documento.
L'indice è calcolato secondo la seguente formula:

$$89 + \frac{300 * (numero\ frasi) - 10 * (numero\ lettere)}{numero\ parole}$$

I risultati sono compresi tra 0 e 100, con 100 la migliore leggibilità e 0 la peggiore.

Per una descrizione più dettagliata dell'indice Gulpease fare riferimento all'appendice C Standard ISO/IEC 9126.
Criteri utilizzati:

- **Metrica**: Gulpease;
- **Range di accettazione**: [40-100];
- **Range ottimale**: [50-100].

2.1.2.2 Qualità del software Gli obiettivi di qualità del *software* che il gruppo Stark Labs desidera raggiungere sono i seguenti:

- **Copertura dei requisiti obbligatori**: il prodotto possiede tutte le funzionalità obbligatorie descritte nell'*Analisi dei Requisiti v2.0.0*;
- **Copertura dei requisiti desiderabili**: il prodotto possiede tutte le funzionalità desiderabili descritte nell'*Analisi dei Requisiti v2.0.0*;
- **Manutenibilità**: il codice deve risultare manutenibile e facilmente comprensibile;
- **Superamento dei test**: tutte le funzionalità del prodotto sono state testate e rientrano nei *range* di accettabilità;
- **Robustezza**: il prodotto non interrompe la sua corretta esecuzione in seguito a situazioni anomale.

A seguire, gli obiettivi trattati nel dettaglio.

2.1.2.2.1 Copertura dei requisiti obbligatori - OQPDS1 Il prodotto deve implementare tutte le funzionalità obbligatorie descritte nell'*Analisi dei Requisiti v2.0.0*. Per quantificare lo stato di completamento del prodotto, si è deciso di rapportare il numero di requisiti completati con quelli ancora da implementare.
Per una descrizione più dettagliata della metrica fare riferimento all'appendice C Standard ISO/IEC 9126.
Criteri utilizzati:

- **Metrica**: copertura dei requisiti obbligatori;
- **Soglia di accettazione**: 100%;
- **Soglia ottimale**: 100%.

2.1.2.2.2 Copertura dei requisiti desiderabili - OQPDS2 Il prodotto deve implementare tutte le funzionalità desiderabili descritte nell'*Analisi dei Requisiti v2.0.0*. Per quantificare lo stato di completamento del prodotto, si è deciso di rapportare il numero di requisiti completati con quelli ancora da implementare.
Per una descrizione più dettagliata della metrica fare riferimento all'appendice C Standard ISO/IEC 9126.
Criteri utilizzati:

- **Metrica**: copertura dei requisiti desiderabili;
- **Soglia di accettazione**: 70%;
- **Soglia ottimale**: 100%.

2.1.2.2.3 Manutenibilità e comprensibilità del codice - OQPDS3 Il *software* deve risultare manutenibile e di facile comprensione. Al fine di misurare tali obiettivi di qualità, vengono utilizzate le seguenti metriche.

- **Numero di livelli di annidamento**: tale indice rappresenta il numero di livelli di annidamento dei metodi, ossia l'annidamento delle strutture. Un alto livello di questo indice può essere sintomo di un elevata complessità del codice o di un basso livello di astrazione.
  Per una descrizione più dettagliata della metrica fare riferimento all'appendice C Standard ISO/IEC 9126.
  Criteri utilizzati:
  - **Range di accettazione**: [1-6];
  - **Range ottimale**: [1-3].
- **Numero di attributi per classe**: tale indice rappresenta il numero di attributi contenuti in una classe. Un valore elevato può indicare la necessità di suddividere la classe in più classi. Pertanto, tale valore potrebbe essere sintomo di errori progettuali.
  Per una descrizione più dettagliata della metrica fare riferimento all'appendice C Standard ISO/IEC 9126.
  Criteri utilizzati:
  - **Range di accettazione**: [0-16];
  - **Range ottimale**: [3-8].
- **Numero di parametri per metodo**: tale indice rappresenta il numero di parametri contenuti nei metodi. Un valore elevato potrebbe indicare un metodo con funzionalità eccessivamente complesse, sintomo di errori progettuali.
  Per una descrizione più dettagliata della metrica fare riferimento all'appendice C Standard ISO/IEC 9126.
  Criteri utilizzati:
  - **Range di accettazione**: [0-8];
  - **Range ottimale**: [0-4].
- **Linee di codice per linee di commento**: tale indice viene calcolato come il rapporto tra le linee di commento e le linee di codice. Valori ottimali di questo parametro indicano un codice più manutenibile.
  Per una descrizione più dettagliata della metrica fare riferimento all'appendice C Standard ISO/IEC 9126.
  Criteri utilizzati:
  - **Range di accettazione**: [>0.25];
  - **Range ottimale**: [>0.30].
- **Copertura del codice**: tale indice indica la percentuale di istruzioni eseguite durante i test. Maggiore è questo parametro, minore sarà la probabilità di errori nel codice. Questo valore può essere abbassato dalla presenza di metodi semplici, come *setter* o *getter*.
  Per una descrizione più dettagliata della metrica fare riferimento all'appendice C Standard ISO/IEC 9126.
  Criteri utilizzati:
  - **Range di accettazione**: [42%-100%];
  - **Range ottimale**: [65%-100%].

2.1.2.2.4 Superamento dei test - OQPDS4 Il prodotto viene testato sulla base dei singoli requisiti contenuti nell'*Analisi dei Requisiti v2.0.0*. Affinché il prodotto funzioni correttamente, è necessario che i test vengano superati con esito positivo. La quantità di test superati con successo viene espressa in percentuale.
Per una descrizione più dettagliata della metrica fare riferimento all'appendice C Standard ISO/IEC 9126.
Criteri utilizzati:

- **Metrica**: percentuale di test superati;
- **Range di accettazione**: [80%-90%];
- **Range ottimale**: [90%-98%].

2.1.2.2.5 Robustezza - OQPDS5 Il prodotto è robusto quando non interrompe la sua corretta esecuzione in seguito a situazioni anomale. Pertanto, il *software* deve essere in grado di reagire a situazioni di errore e scenari imprevisti.
Per una descrizione più dettagliata della metrica *Failure Avoidance* fare riferimento all'appendice C Standard ISO/IEC 9126.
Criteri utilizzati:

- **Metrica**: Failure Avoidance;
- **Range di accettazione**: [80%-90%];
- **Range ottimale**: [>90%].

2.1.2.2.6 Funzionamento privo di interruzioni - OQPDS6 Per garantire un corretto funzionamento, il prodotto deve eseguire senza interruzioni. Durante l'esecuzione possono avvenire al massimo il 20% di interruzioni a seguito di situazioni anomale.
Per una descrizione più dettagliata della metrica *Breackdown Avoidance* fare riferimento all'appendice C Standard ISO/IEC 9126.
Criteri utilizzati:

- **Metrica**: Breakdown Avoidance;
- **Range di accettazione**: [80%-90%];
- **Range ottimale**: [>90%].

2.2 Organizzazione

L'organizzazione della strategia di verifica è basata sull'utilizzo di attività di controllo per ogni processo attuato. Per ognuno di questi viene verificata la qualità ed eventualmente si verifica anche la qualità del prodotto ottenuto. Questo tramite i metodi di verifica descritti nella sezione Tecniche di analisi sui prodotti. Ognuna delle fasi del progetto descritte nel *Piano di Progetto v2.0.0* necessita di diverse attività di verifica dipendenti dai differenti *output*:

- **Analisi:** durante questo periodo si sono verificati tutti i documenti prodotti;
- **Analisi di Dettaglio:** durante questo periodo si è posta l'attenzione sulla verifica dell'*Analisi dei Requisiti*;

- **Progettazione Architetturale:** durante questo periodo il team si è occupato di verificare il *Piano di Progetto*, il *Piano di Qualifica*, la *Specifica Tecnica*, le *Norme di Progetto* e l'*Analisi dei Requisiti*.

In ogni documento viene inoltre incluso il registro delle modifiche che permette di mantenere uno storico delle attività svolte e delle relative responsabilità.

2.3 Pianificazione strategica e temporale

Dato che l'obiettivo è di rispettare le scadenze fissate nel *Piano di Progetto v2.0.0*, è necessario che l'attività di verifica sia ben organizzata. Pertanto l'individuazione e la correzione di errori dovrà essere tempestiva, in modo da impedire che si diffondano. Ogni attività di redazione di documenti o di codifica deve essere preceduta da un'analisi della struttura e dei contenuti. Questo allo scopo di evitare imprecisioni concettuali o tecniche, rendendo l'attività di verifica più semplice, con conseguente riduzione del numero di correzioni richieste. La metodologia da seguire per individuare e correggere eventuali errori è descritta nel documento *Norme di Progetto v2.0.0*.

2.4 Responsabilità

Per garantire che il processo di verifica sia efficace e sistematico, vengono attribuite tali responsabilità a *Verificatore* e *Responsabile di Progetto*. La suddivisione dei compiti e le modalità sono definite nel documento *Norme di Progetto v2.0.0*.

3 Visione di dettaglio della strategia di gestione della qualità

3.1 Risorse

Per assicurarsi che gli obiettivi vengano raggiunti sono necessarie risorse umane e tecnologiche. Coloro che detengono la maggiore responsabilità per le attività di verifica e validazione sono *Verificatore* e *Responsabile di Progetto*. I ruoli sono descritti nel dettaglio nelle *Norme di Progetto v2.0.0*. Per risorse tecniche e tecnologiche sono intesi tutti gli strumenti *software* e *hardware* che il gruppo intende utilizzare. Affinché il lavoro dei *Verificatori* venga semplificato, sono stati impostati alcuni strumenti di controllo sistematico. Questi sono descritti in modo accurato nelle *Norme di Progetto v2.0.0*.

3.2 Misure e metriche

Il processo di verifica deve essere quantificabile per risultare informativo. Pertanto, è necessario stabilire a priori delle metriche su cui basare le misurazioni del processo di verifica. Essendo le metriche di natura variabile, vengono definite due tipologie di intervalli:

- **Accettazione:** valori che vengono richiesti affinché il prodotto sia accettato;
- **Ottimale:** valori entro cui è consigliabile che la misurazione si collochi.

Non sono intervalli vincolanti, ma consigliati. Se tali valori si discostano, è necessaria una verifica approfondita.

3.2.1 Metriche per i processi

Come metriche per valutare i processi si è scelto di utilizzare degli indici che analizzino i costi e i tempi. Ogni metrica è caratterizzata da un codice identificativo pensato per aumentare il livello di tracciabilità di ogni singola metrica. Le metodologie di assegnazione del codice sono descritte nel documento *Norme di Progetto v2.0.0*.

3.2.1.1 Software Process Improvement and Capability Determination - MPC1 Per porsi obiettivi quantitativi di miglioramento, si è deciso di utilizzare lo standard ISO$_G$/IEC$_G$ 15504, anche noto come *Software Process Improvement and Capability Determination* (SPICE). Esso fornisce gli strumenti necessari per valutare l'idoneità dei processi, assegnandovi un indice e ponendolo in relazione a una scala definita dallo standard SPICE. Per applicare correttamente il suddetto modello si deve far uso del ciclo di Deming$_G$, che definisce una metodologia di controllo specifica per i processi nel corso del loro ciclo di vita, con l'intento di migliorarne costantemente la qualità. Per una descrizione più dettagliata del Ciclo di Deming e il PDCA$_G$, si rimanda all'appendice B Ciclo di Deming.

3.2.1.2 Schedule Variance (SV) - MPC2 Tale metrica è utile per valutare se si è in linea con la pianificazione effettuata nel Piano di Progetto. La *Schedule Variance* è un indice che permette di valutare l'efficacia dei processi che si ottiene dalla differenza tra la data pianificata per il termine di un'attività e la data di fine effettiva dell'attività stessa. Se il valore SV>0 significa che il team sta producendo con maggior velocità rispetto alla pianificazione, viceversa se è negativo. Se il valore SV è pari a zero, significa che si è in linea con la pianificazione.
Parametri utilizzati:

- Range di accettazione: [> -(Costo preventivo fase x 5%)];
- Range ottimale: [>=0].

3.2.1.3 Budget Variance (BV) - MPC3 Tale metrica è utile per valutare se al momento del calcolo i costi sono maggiori o minori rispetto a quanto preventivato nel Piano di Progetto. La *Budget Variance* è un indice di efficienza. Se il valore BV>0 significa che il progetto sta consumando il budget più lentamente di quanto pianificato, viceversa se negativo.
Parametri utilizzati:

- Range di accettazione: [> -(Costo preventivo fase x 10%)];
- Range ottimale: [>=0].

3.2.2 Metriche per i prodotti

3.2.2.1 Metriche per i documenti Come metrica per i documenti si è scelto di utilizzare un indice di leggibilità. L'indice utilizzato è specifico per la lingua italiana.

3.2.2.1.1 Indice di Gulpease - MPDD1 L'indice Gulpease$_G$ è un indice di leggibilità tarato sulla lingua italiana. Viene preferito rispetto ad altri poiché utilizza la lunghezza in lettere anziché in sillabe, semplificandone il calcolo. Questo indice evidenzia la complessità dello stile del documento. L'indice è calcolato secondo la seguente formula:

$$89 + \frac{300 * (numero\ frasi) - 10 * (numero\ lettere)}{numero\ parole}$$

I risultati sono compresi tra 0 e 100, con 100 la migliore leggibilità e 0 la peggiore.
Più in generale, è possibile valutare la leggibilità del testo in relazione con il tipo di *target* a cui è mirato il testo:

- Indice inferiore a 80: di difficile lettura per lettori con licenza elementare;
- Indice inferiore a 60: di difficile lettura per lettori con licenza media;
- Indice inferiore a 40: di difficile lettura per lettori con licenza superiore.

In relazione a tali informazioni sono stati stabiliti i range di accettazione che seguono.
Parametri utilizzati:

- Range di accettazione: [40-100];
- Range ottimale: [50-100].

3.2.2.2 Metriche per il software È desiderabile, per poter raggiungere gli obiettivi di qualità del *software*, applicare le metriche a seguire.

3.2.2.2.1 Copertura dei requisiti obbligatori - MPDS1 Tale metrica permette di quantificare lo stato di completamento dei requisiti obbligatori. Essa consente di monitorare il rapporto tra i requisiti obbligatori completati rispetto al totale dei requisiti obbligatori rilevati nell'*Analisi dei Requisiti v2.0.0*. Per maggiore chiarezza, il valore viene calcolato in percentuale:

$$Copertura\ dei\ requisiti\ obbligatori = \frac{numero\ requisiti\ obbligatori\ soddisfatti}{numero\ requisiti\ obbligatori\ totale}$$

Parametri utilizzati:

- **Range di accettazione**: 100%;
- **Range ottimale**: 100%.

3.2.2.2.2 Copertura dei requisiti desiderabili - MPDS2 Tale metrica permette di quantificare lo stato di completamento dei requisiti desiderabili. Essa consente di monitorare il rapporto tra i requisiti desiderabili completati rispetto al totale dei requisiti desiderabili rilevati nell'*Analisi dei Requisiti v2.0.0*. Per maggiore chiarezza, il valore viene calcolato in percentuale:

$$Copertura\ dei\ requisiti\ desiderabili = \frac{numero\ requisiti\ desiderabili\ soddisfatti}{numero\ requisiti\ desiderabili\ totale}$$

Parametri utilizzati:

- Range di accettazione: 70%;
- Range ottimale: 100%.

3.2.2.2.3 Numero di livelli di annidamento - MPDS3 Tale indice rappresenta il numero di livelli di annidamento dei metodi, ossia l'annidamento delle strutture. Un alto livello di questo indice può essere sintomo di un'elevata complessità del codice o di un basso livello di astrazione.

Parametri utilizzati:

- Range di accettazione: [1-6];
- Range ottimale: [1-3].

3.2.2.2.4 Numero di attributi per classe - MPDS4 Tale indice rappresenta il numero di attributi contenuti in una classe. Un valore elevato può indicare la necessità di suddividere la classe in più classi. Pertanto, tale valore potrebbe essere sintomo di errori progettuali.

Parametri utilizzati:

- Range di accettazione: [0-16];
- Range ottimale: [3-8].

3.2.2.2.5 Numero di parametri per metodo - MPDS5 Tale indice rappresenta il numero di parametri contenuti nei metodi. Un valore elevato potrebbe indicare un metodo con funzionalità eccessivamente complesse, sintomo di errori progettuali.

Parametri utilizzati:

- Range di accettazione: [0-8];
- Range ottimale: [0-4].

3.2.2.2.6 Linee di codice per linee di commento - MPDS6 Tale indice viene calcolato come il rapporto tra le linee di commento e le linee di codice. Valori ottimali di questo parametro indicano un codice più manutenibile.

Parametri utilizzati:

- Range di accettazione: [>0.25];
- Range ottimale: [>0.30].

3.2.2.2.7 Copertura di codice - MPDS7 Tale indice indica la percentuale di istruzioni eseguite durante i test. Maggiore è questo parametro, minore sarà la probabilità di errori nel codice. Questo valore può essere abbassato dalla presenza di metodi semplici, come *setter* o *getter*.

Parametri utilizzati:

- Range di accettazione: [42%-100%];
- Range ottimale: [65%-100%].

3.2.2.2.8 Percentuale di test superati - MPDS8 Tale metrica permette di quantificare la percentuale di successo dei test applicati ai requisiti. Il valore ottenuto è utile per stabilire il livello di correttezza del *software*. La percentuale viene calcolata come:

$$Percentuale\ di\ test\ superati = \frac{numero\ test\ superati}{numero\ test\ totale}$$

Si noti che una percentuale di successo superiore al 98% è ritenuta negativa in quanto lo scopo dei test è di trovare problemi. Pertanto, qualora si superasse il 98%, i test previsti verranno modificati.
Parametri utilizzati:

- Range di accettazione: [80%-98%];
- Range ottimale: [90%-98%].

3.2.2.2.9 Failure Avoidance - MPDS9 Tale metrica permette di quantificare la percentuale di situazioni impreviste evitate dal prodotto. La *Failure Avoidance* consente di valutare la robustezza del prodotto in risposta a eventuali situazioni anomale. La percentuale viene calcolata come:

$$Failure\ Avoidance = \frac{numero\ situazioni\ anomale\ evitate}{numero\ situazioni\ anomale\ totale}$$

Parametri utilizzati:

- Range di accettazione: [80%-90%];
- Range ottimale: [>90%].

3.2.2.2.10 Breakdown Avoidance - MPDS10 Tale metrica permette di quantificare la percentuale di interruzioni evitate dal prodotto. La *Breakdown Avoidance* consente di controllare che il prodotto esegua senza interruzioni. La percentuale viene calcolata come:

$$Breakdown\ Avoidance = 1 - \frac{numero\ di\ interruzioni}{numero\ situazioni\ anomale\ presentate}$$

Parametri utilizzati:

- Range di accettazione: [80%-90%];
- Range ottimale: [>90%].

3.3 Tecniche di analisi

3.3.1 Analisi statica

Per analisi statica si intende una tecnica di controllo che permette di effettuare la verifica di quanto prodotto individuando eventuali errori. Essa viene svolta in due modi complementari.

3.3.1.1 Walkthrough Viene svolta una lettura critica di tutto il materiale. Questa tecnica è utile nelle prime fasi di progetto, quando i membri del gruppo non hanno ancora un'adeguata esperienza che permette verifiche più mirate. Grazie a questa tecnica, il *Verificatore* può stilare una lista degli errori più frequenti, in modo da migliorare le attività di analisi future. Questa attività è onerosa e richiede l'intervento di più persone per essere efficace ed efficiente. In seguito alla lettura segue una fase di discussione con il fine di esaminare i difetti e proporre le correzioni. La fase finale consiste nello stilare un rapporto che elenchi le modifiche effettuate.

3.3.1.2 Inspection In questa tecnica viene eseguita un'analisi mirata delle parti del documento o del codice che sono ritenute maggiormente fonte di errore. La lista di controllo, che contiene queste sezioni critiche, è redatta anticipatamente ed è frutto dell'esperienza dei *Verificatori* in seguito all'applicazione del Walkthrough. Questa strategia è più rapida del Walkthrough in quanto riduce il numero di parti da analizzare. Per tale ragione l'Inspection può essere eseguita solamente dai *Verificatori*, che individuano e correggono eventuali errori e redigono il rapporto di verifica per tracciare il lavoro svolto.

3.3.2 Analisi dinamica

L'analisi dinamica viene applicata solamente alla produzione di codice e viene effettuata durante l'esecuzione, mediante l'uso di test utilizzati per verificarne il funzionamento. Per rendere questa attività utile e generare risultati attendibili, è necessario che i test siano ripetibili. Questo significa che il programma da un determinato *input* generi sempre lo stesso *output*. Test di questo tipo sono utili per determinare la correttezza ed evidenziare eventuali problemi in un *software*. Devono quindi essere definiti:

- **Ambiente:** consiste sia del sistema *hardware* che di quello *software* sui quali è pianificato lo sviluppo. Di questi è necessario specificare lo stato iniziale dal quale iniziare i test;
- **Specifica:** consiste nel definire gli *input* e i rispettivi *output* attesi;
- **Procedure:** consiste nella definizione di come i test vengono svolti, con quale ordine e come vengono analizzati i risultati.

Esistono cinque tipi diversi di test: test di unità, test di integrazione, test di sistema, test di regressione e test di validazione.

3.3.2.1 Test di unità Consta nel verificare ogni singola unità del *software* con l'utilizzo di *stub*$_G$, *driver*$_G$ e *logger*$_G$. Con unità è inteso il minimo quantitativo di *software*, che sia utile verificare singolarmente, prodotto da un singolo programmatore. Con questi test si verifica il funzionamento corretto dei moduli per eliminare dal sistema possibili errori di implementazione.

3.3.2.2 Test di integrazione Consta nel verificare i componenti del sistema al fine di controllare che la combinazione di due o più unità *software* funzioni correttamente. L'obiettivo di questo test è di individuare errori residui dalla realizzazione dei moduli, o comportamenti inaspettati da componenti *software* forniti da terze parti. Per effettuare questi test è necessario aggiungere delle componenti fittizie per sostituire quelle ancora non sviluppate, al fine di non influenzare l'esito dell'analisi.

3.3.2.3 Test di sistema Consta nel validare il prodotto *software* nel momento in cui si ritiene che sia giunto ad una versione definitiva. Questo test verifica che tutti i requisiti software stabiliti nell'*Analisi dei Requisiti v2.0.0* vengano rispettati.

3.3.2.4 Test di regressione Consta nell'eseguire nuovamente i test sulle componenti *software* in seguito a nuove modifiche, al fine di controllare che i cambiamenti non alterino il corretto funzionamento di queste componenti o di altre che non sono state aggiornate. Questa operazione viene facilitata dal tracciamento, che permette di individuare e ripetere i test di unità, integrazione e di sistema che sono stati influenzati dalla modifica.

3.3.2.5 Test di validazione Consta nel collaudare il prodotto *software* in presenza del Proponente. Se questo collaudo viene superato, si può procedere al rilascio ufficiale del prodotto sviluppato.

A Standard ISO/IEC 15504

Questo standard descrive come i processi debbano essere controllati costantemente, al fine di rilevare possibili rischi intrinsechi che potrebbero impedire di raggiungere gli obiettivi prefissati. I risultati delle singole valutazioni devono essere ripetibili, oggettivi e comparabili per far sì che contribuiscano al miglioramento effettivo dei processi in esame.

Il modello SPICE$_G$ definisce sei livelli di maturità del processo:

- **Incomplete process:** il processo non viene attuato o non raggiunge i risultati previsti;
- **Performed:** il processo viene completato e raggiunge i risultati previsti
  - **Process performance attribute:** capacità di raggiungere gli obiettivi.
- **Managed process:** il processo viene eseguito in maniera controllata a seconda degli obiettivi
  - **Performance management attribute:** è la capacità di un processo di elaborare un prodotto coerente con gli obiettivi;
  - **Work product management attribute:** è la capacità di un processo di elaborare un prodotto documentato, controllato e verificato.
- **Established process:** il processo viene eseguito in base a principi dell'Ingegneria del Software
  - **Process definition attribute:** l'esecuzione di processo si basa sugli standard per raggiungere gli obiettivi;
  - **Process resource attribute:** capacità del processo di utilizzare le risorse tecniche e umane appropriate, per essere efficaci.
- **Predictable process:** il processo viene eseguito costantemente in limiti predefiniti al fine di raggiungere i risultati attesi
  - **Measurement attribute:** gli obiettivi e le misure dei prodotti e dei processi sono utilizzati per garantire il raggiungimento degli obiettivi;
  - **Process control attribute:** il processo è controllato grazie alle misure di prodotto e processo, per effettuare correzioni migliorative.
- **Optimizing process:** il processo cambia e si adatta costantemente per raggiungere gli obiettivi
  - **Process change attribute:** i cambiamenti strutturali, di gestione e di esecuzione sono gestiti in maniera controllata al fine di raggiungere gli obiettivi;
  - **Continuous improvement attribute:** ogni modifica ai processi è identificata e implementata per garantire il miglioramento nella realizzazione degli obiettivi di business.

Ogni attributo di processo descritto è misurabile e lo standard predispone quattro livelli:

- **Non posseduto:** 0% - 15%;
- **Parzialmente posseduto:** 15% - 50%;

- **Largamente posseduto:** 50% - 85%;
- **Completamente posseduto:** 85% - 100%.

B Ciclo di Deming

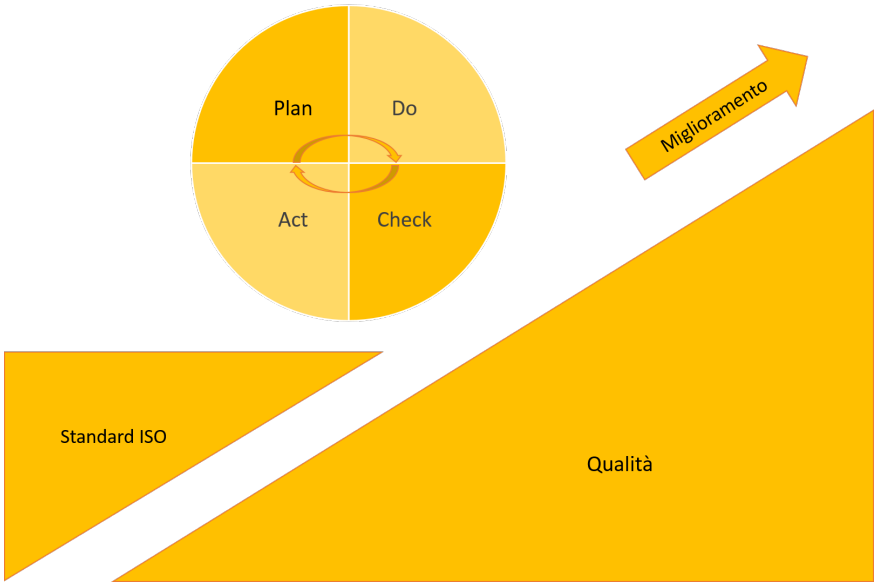

Figura 1: Ciclo di Deming

Il ciclo di Deming, conosciuto anche come ciclo PDCA$_G$, è un modello studiato per il miglioramento continuo della qualità in un ottica a lungo termine.
Questo strumento da la possibilità di fissare obiettivi di miglioramento a partire dagli esiti delle attività di verifica eseguite. Fissati questi obiettivi si procede all'iterazione delle attività previste da questo modello, fino a raggiungere gli obiettivi fissati.
Questi miglioramenti hanno come fine il miglioramento dell'efficienza e dell'efficacia, ossia di essere il più conformi possibile alle aspettative utilizzando la minima quantità possibile di risorse. Vengono ora riportate le 4 attività previste in questo modello:

- **Plan:** attività di pianificazione che definisce:
  - Gli obiettivi da raggiungere;
  - Come svolgere le attività per conseguirli, tenendo in considerazione le risorse disponibili e le scadenze;
  - Le responsabilità di chi si occupa di eseguire;
  - Le metriche per valutare i processi.
- **Do:** attività di implementazione ed esecuzione delle strategie progettate nell'attività precedente;
- **Check:** attività di verifica nella quale vengono controllati i risultati dell'attività precedente e in cui si confrontano i prodotti effettivi con quelli attesi;
- **Act:** attività che prevede il miglioramento dei processi. Nello svolgimento di questa attività viene tenuto conto delle *best practice*$_G$.

Lo scopo di questo ciclo è il continuo incremento qualitativo del processo di sviluppo e, conseguentemente, del prodotto. Le attività quindi devono essere analizzabili, ripetibili e tracciabili. Queste caratteristiche permettono di individuare e correggere eventuali errori.

C Standard ISO/IEC 9126

Lo standard ISO_G/IEC_G 9126 è stato creato con lo scopo di descrivere gli obiettivi qualitativi del prodotto e definire le metriche che possono misurare il raggiungimento di tali obiettivi. Esso si suddivide in quattro parti:

- Modello della qualità del software;
- Metriche per la qualità esterna;
- Metriche per la qualità interna;
- Metriche per la qualità in uso.

Lo standard tratta la qualità del software da tre punti di vista:

- **Qualità in uso:** è la qualità del prodotto *software* visto dall'utilizzatore, che ne fa uso in uno specifico contesto;
- **Qualità esterna:** è la qualità del prodotto *software* visto dall'esterno, quando viene eseguito e testato in un ambiente di prova;
- **Qualità interna:** è la qualità del prodotto *software* visto dall'interno, facendo riferimento a caratteristiche implementative, come l'architettura o il codice che ne deriva.

C.1 Modello della qualità del software

Nello standard sono previste sei caratteristiche qualitative principali, suddivise a loro volta in sottocategorie che possono essere misurate quantitativamente:

- **Funzionalità:** capacità del prodotto di fornire le funzioni richieste
  - **Idoneità:** capacità del prodotto di fornire un insieme di funzioni appropriate per un'attività;
  - **Accuratezza:** capacità del prodotto di fornire risultati coerenti e corretti a seconda del grado di precisione richiesto;
  - **Interoperabilità:** capacità interattiva del prodotto *software* con uno o più sistemi;
  - **Sicurezza:** capacità del prodotto *software* di proteggere i dati e le informazioni, e di consentire solo a sistemi autorizzati di effettuare modifiche;
  - **Conformità funzionale:** capacità del prodotto *software* di essere aderente allo standard in materia di funzionalità.
- **Affidabilità:** capacità del prodotto di garantire un livello di prestazioni adeguato
  - **Maturità:** capacità del *software* di evitare fallimenti causati da errori;
  - **Tolleranza agli errori:** capacità del *software* di mantenere un adeguato livello di prestazioni nonostante errori o violazioni dell'interfaccia;
  - **Capacità di recupero:** capacità del *software* di ristabilire un livello di *performance* adeguato e di recuperare dati in caso di errori;

  - **Conformità di affidabilità:** capacità del prodotto *software* di essere aderente allo standard in materia di affidabilità.

- **Usabilità:** capacità del prodotto di essere facilmente comprensibile ed usabile, e di risultare interessante per l'utente
  - **Intelligibilità:** capacità del *software* di permettere all'utente di capire se il prodotto è adeguato e se possa essere utilizzato per dei compiti particolari;
  - **Apprendibilità:** capacità del *software* di consentire all'utente di imparare ad utilizzare le sue funzionalità;
  - **Operabilità:** capacità del *software* di consentire ai suoi utenti di essere usato;
  - **Attrattività:** capacità del *software* di creare interesse negli utenti;
  - **Conformità di usabilità:** capacità del *software* di essere aderente allo standard in materia di usabilità.

- **Efficienza:** capacità del prodotto di fornire prestazioni adeguate a seconda delle risorse utilizzate
  - **Comportamento temporale:** capacità del *software* di dare una risposta in tempi di elaborazione adeguati, quando svolge le sue funzionalità;
  - **Utilizzo di risorse:** capacità del *software* di utilizzare le risorse adeguate per la sua esecuzione;
  - **Conformità di efficienza:** capacità del *software* di essere aderente allo standard in materia di efficienza.

- **Manutenibilità:** capacità del prodotto di essere modificato e aggiornato. Tali modifiche o aggiornamenti possono rispecchiare correzioni, miglioramenti o adattamenti del *software* nei confronti di cambiamenti ambientali, di specifiche o delle funzionalità
  - **Analizzabilità:** capacità del *software* di poter essere studiato per cercare difetti;
  - **Modificabilità:** capacità del *software* di consentire l'implementazione di una qualche modifica;
  - **Stabilità:** capacità del *software* di evitare errori o fallimenti causati da una o più modifiche;
  - **Testabilità:** capacità del *software* di permettere la validazione di una versione modificata del *software*;
  - **Conformità di manutenibilità:** capacità del prodotto *software* di essere aderente allo standard in materia di manutenibilità.

- **Portabilità:** capacità del *software* di poter essere spostato in vari ambienti di lavoro
  - **Adattabilità:** capacità del *software* di essere adattato a diversi ambienti senza necessitare di modifiche;
  - **Installabilità:** capacità del *software* di poter essere installato in ambienti differenti;
  - **Coesistenza:** capacità del *software* di coesistere con altri *software* indipendenti condividendo delle risorse;
  - **Sostituibilità:** capacità del *software* di poter sostituire un *software* simile che abbia le stesse funzionalità e che sia nello stesso ambiente;
  - **Conformità di portabilità:** capacità del prodotto *software* di essere aderente allo standard in materia di portabilità.

C.2 Modello della qualità in uso

Le categorie presenti nel modello della qualità software in uso sono le seguenti:

- **Efficacia:** la capacità del software di mettere gli utenti nella condizione di raggiungere gli obiettivi specificati con accuratezza e completezza;
- **Produttività:** la capacità di mettere gli utenti nella condizione di spendere una quantità di risorse appropriata in relazione all'efficacia ottenuta in uno specificato contesto d'uso;
- **Soddisfazione:** è la capacità del prodotto software di soddisfare gli utenti;
- **Sicurezza:** rappresenta la capacità del prodotto software di raggiungere accettabili livelli di rischio di danni a persone, al software, ad apparecchiature o all'ambiente operativo d'uso.

C.3 Metriche per la qualità del software

C.3.1 Metriche per la qualità esterna

Le metriche esterne misurano i comportamenti del software sulla base dei test, dall'operatività e dall'osservazione durante la sua esecuzione, in funzione degli obiettivi stabiliti in un contesto tecnico rilevante o di business.

C.3.2 Metriche per la qualità interna

La qualità interna si applica al software non eseguibile (ad esempio il codice sorgente) durante le fasi di progettazione e codifica. Le misure effettuate permettono di prevedere il livello di qualità esterna ed in uso del prodotto finale, poiché gli attributi interni influiscono su quelli esterni e quelli in uso. Le metriche interne permettono di individuare eventuali problemi che potrebbero influire sulla qualità finale del prodotto prima che sia realizzato il software eseguibile.

C.3.3 Metriche per la qualità in uso

La qualità in uso rappresenta il punto di vista dell'utente sul software. Misurano il grado con cui permette di svolgere le attività con efficacia, produttività, sicurezza e soddisfazione nel contesto previsto.

D Pianificazione dei test

In seguito verranno elencati tutti i test di sistema, di integrazione e di validazione prevedendo un aggiornamento futuro per i test di unità.
Nelle tabelle sottostanti lo stato dei test **N.T.** è da intendersi come non testati in quanto tali test saranno eseguiti in un momento successivo, come descritto nel *Piano di Progetto v.2.0.0*.

D.1 Test di Sistema

In questa sezione vengono descritti i test di sistema che permettono di verificare che l'esecuzione del programma avvenga in modo corretto rispetto ai requisiti descritti nell'*Analisi dei Requisiti v.2.0.0*.
I test di sistema riportati sono relativi alla parte più rilevante dei requisiti individuati.

D.1.1 Descrizione dei test di Sistema

Codice	Descrizione	Stato	Requisito
TS1	Viene verificato che il sistema permetta di creare uno sceneggiato	N.T.	R0F1
TS2	Viene verificato che il sistema permetta la creazione di una nuova voce personalizzata	N.T.	R0F2
TS3	Viene verificato che il sistema permetta all'utente non autenticato di potersi registrare al sito di MIVOQ s.r.l.	N.T.	R2F3
TS5	Viene verificato che il sistema permetta di selezionare una voce per i servizi Text-To-Speech$_G$ del sistema	N.T.	R0F5
TS6	Viene verificato che il sistema permetta all'utente autenticato di campionare la propria voce	N.T.	R2F6
TS7	Viene verificato che il sistema permetta ad un'applicazione Android$_G$ di utilizzare per i servizi di Text-To-Speech$_G$ la voce selezionata per il sistema	N.T.	R0F7
TS8.1	Viene verificato che il sistema permetta di creare un personaggio per lo sceneggiato	N.T.	R0F8.1
TS8.2	Viene verificato che il sistema permetta di riordinare i capitoli di uno sceneggiato	N.T.	R0F8.2

TS8.3	Viene verificato che il sistema permetta di salvare uno sceneggiato	N.T.	R0F8.3
TS8.4	Viene verificato che il sistema permetta di cancellare un capitolo esistente dello sceneggiato	N.T.	R0F8.4
TS8.5	Viene verificato che il sistema permetta di importare in uno sceneggiato i personaggi di un altro	N.T.	R0F8.5
TS8.6.1.1	Viene verificato che il sistema permetta di esportare lo sceneggiato in formato audio .wav$_G$	N.T.	R0F8.6.1.1
TS8.6.1.2	Viene verificato che il sistema permetta di esportare lo sceneggiato in formato audio .mp3$_G$	N.T.	R0F8.6.1.2
TS8.6.3	Viene verificato che il sistema permetta di esportare lo sceneggiato in formato video .mp4$_G$	N.T.	R2F8.6.3
TS8.7.7	Viene verificato che il sistema permetta all'utente di assegnare una voce ad un personaggio creato	N.T.	R0F8.7.7
TS8.10	Viene verificato che il sistema permetta la creazione di un capitolo	N.T.	R0F8.10
TS8.11.1	Viene verificato che il sistema permetta l'avvio dell'anteprima di una battuta	N.T.	R0F8.11.1
TS8.11.3.3.1	Viene verificato che il sistema permetta di caricare un file in formato .mp3$_G$ come sottofondo sonoro	N.T.	R1F8.11.3.3.1
TS8.11.3.3.2	Viene verificato che il sistema permetta di caricare un file in formato .wav$_G$ come sottofondo sonoro	N.T.	R1F8.11.3.3.2
TS8.11.4.2	Viene verificato che il sistema permetta di inserire il testo di una battuta	N.T.	R0F8.11.4.2
TS8.11.5.2	Viene verificato che il sistema permetta di scegliere un'emozione da associare ad una battuta	N.T.	R0F8.11.5.2

TS8.11.6	Viene verificato che il sistema permetta di riordinare le battute di un capitolo	N.T.	R0F8.11.6
TS8.11.9	Viene verificato che il sistema permetta di cancellare una battuta esistente	N.T.	R0F8.11.9
TS8.12.1	Viene verificato che il sistema permetta la condivisione di uno sceneggiato in formato audio	N.T.	R1F8.12.1
TS8.12.3	Viene verificato che il sistema permetta la condivisione di uno sceneggiato in formato video	N.T.	R2F8.12.3
TS9.1	Viene verificato che il sistema permetta di ascoltare un'anteprima della voce su un testo predefinito	N.T.	R0F9.1
TS9.2	Viene verificato che il sistema permetta di cancellare una voce personalizzata esistente	N.T.	R0F9.2
TS9.5	Viene verificato che il sistema permetta di impostare gli effetti vocali di una voce esistente	N.T.	R0F9.5
TS15	Viene verificato che il sistema permetta il funzionamento dell'applicazione anche in caso di assenza di connessione	N.T.	R2F15
TS20	Viene verificato che il sistema permetta il funzionamento su dispositivi Android$_G$ a partire dalla versione 4.0	N.T.	R1V20

Tabella 2: Test di Sistema

D.2 Test di Integrazione

In questa sezione vengono descritti i test di integrazione, da utilizzare per i vari componenti descritti nella progettazione ad alto livello, che permettono di verificare la corretta integrazione ed il corretto flusso dei dati all'interno del sistema.

D.2.1 Descrizione dei test di Integrazione

Codice	Descrizione	Stato	Componente
TI1	Test di integrazione finale per le componenti Drama e Libraries	N.T.	Sivodim
TI2	Test di integrazione fra le componenti Model View e Presenter della componente Libraries	N.T.	Sivodim::Libraries
TI3	Test di integrazione fra le componenti Model View e Presenter della componente Drama	N.T.	Sivodim::Drama
TI4	Test di integrazione per le funzionalità delle componenti Screenplay, Chapter, Character e Utilities.	N.T.	Sivodim::Drama::Model
TI5	Verificare che il sistema gestisca correttamente le componenti relative al package Presenter. In particolare che gestisca correttamente l'interazione coi componenti del package Model e del package View	N.T.	Sivodim::Drama::Presenter
TI6	Verificare che il sistema gestisca correttamente le componenti relative al package View. In particolare che gestisca correttamente l'interazione coi componenti del package Presenter	N.T.	Sivodim::Drama::View
TI7	Verificare che il sistema gestisca correttamente le componenti relative al package Presenter. In particolare che gestisca correttamente l'interazione coi componenti del package Model e del package View di Sivodim::Libraries	N.T.	Sivodim::Libraries::Presenter

TI8	Verificare che il sistema gestisca correttamente le componenti relative al package View. In particolare che gestisca correttamente l'interazione con il Presenter di Sivodim::Libraries	N.T.	Sivodim::Libraries::View
TI9	Test di integrazione per le funzionalità delle componenti EngineManager, Mivoq e Voice della componente Libraries	N.T.	Sivodim::Libraries::Model
TI10	Verificare che il sistema gestisca correttamente le componenti relative al package Screenplay. In particolare che gestisca correttamente l'interazione coi componenti Chapter e Character del package Model, e con l'$engine_G$ di Sivodim::Libraries::Model::EngineManager	N.T.	Sivodim::Drama::Model::Screenplay
TI11	Verificare che il sistema gestisca correttamente le componenti relative al package Character. In particolare che gestisca correttamente l'interazione coi componenti Utilities e Character del package Sivodim::Drama::Model e con il componente Sivodim::Libraries::Model	N.T.	Sivodim::Drama::Model::Character
TI12	Verificare che il sistema gestisca correttamente le componenti relative al *package* Utilities. In particolare che gestisca correttamente l'interazione coi componenti Chapter e Charater del *package* Sivodim::Drama::Model	N.T.	Sivodim::Drama::Model::Utilities

TI13	Verificare che il sistema gestisca correttamente le componenti relative al *package* Chapter. In particolare che gestisca correttamente l'interazione coi componenti Utilities, Screenplay e Charater del *package* Sivodim::Drama::Model e con il componente Sivodim::Libraries::Model	N.T.	Sivodim::Drama::Model::Chapter
TI14	Verificare che il sistema gestisca correttamente le componenti relative al package EngineManager. In particolare che gestisca correttamente l'interazione coi componenti Mivoq e Voice di Sivodim::Libraries::Model	N.T.	Sivodim::Libraries::Model::EngineManager
TI15	Verificare che il sistema gestisca correttamente le componenti relative al *package* Mivoq. In particolare che gestisca correttamente l'interazione con le componenti EngineManager e Voice del *package* Sivodim::Libraries::Model	N.T.	Sivodim::Libraries::Model::Mivoq
TI16	Verificare che il sistema gestisca correttamente le componenti relative al *package* Voice. In particolare che gestisca correttamente l'interazione con il componente EngineManager del *package* Sivodim::Libraries::Model	N.T.	Sivodim::Libraries::Model::Voice

Tabella 3: Test di Integrazione

D.2.2 Tracciamento componenti–test di Integrazione

Componente	**Test di Integrazione**
Sivodim	TI1
Sivodim::Libraries	TI2

Sivodim::Drama	TI3
Sivodim::Drama::Model	TI4
Sivodim::Drama::Presenter	TI5
Sivodim::Drama::View	TI6
Sivodim::Libraries::Presenter	TI7
Sivodim::Libraries::View	TI8
Sivodim::Libraries::Model	TI9
Sivodim::Drama::Model::Screenplay	TI10
Sivodim::Drama::Model::Character	TI11
Sivodim::Drama::Model::Utilities	TI12
Sivodim::Drama::Model::Chapter	TI13
Sivodim::Libraries::Model::EngineManager	TI14
Sivodim::Libraries::Model::Mivoq	TI15
Sivodim::Libraries::Model::Voice	TI16

Tabella 4: Tracciamento Componenti - test di Integrazione

D.3 Test di Unità

Di seguito vengono riportati i test di unità previsti per l'applicazione.

D.3.1 Descrizione dei test di Unità

Codice	Descrizione	Metodi	Stato
TU1	Si verifica che, forniti in memoria dei file di salvataggio di test, la lista degli sceneggiati disponibili visualizzata all'avvio non risulti vuota. Si verifica anche che, dato un file di salvataggio di test non valido opportunatamente creato, questo non risulti nella lista dei progetti disponibili	getTitlesAdapter(Context context) getScreenplayTitles(Context context) getScreenplayTitles(Context context) getTitlesAdapter(Context context)	N.T.
TU2	Si verifica che, passato un Context$_G$ e una stringa relativa al titolo di un file sceneggiato di test, venga aperta l'activity$_G$ che mostra l'elenco dei capitoli di tale sceneggiato. Si verifica anche che, data una stringa non relativa ad uno sceneggiato in memoria, l'apertura della lista dei capitoli falisca	goToListChapter(Context context, String selected) goToListChapter(Context context, String selected)	N.T.
TU3	Si verifica che, passata una stringa corrispondente al titolo e un Context$_G$ venga creato un nuovo sceneggiato avente come titolo la stringa passata. Si verifica anche che la creazione fallisca nel caso in cui non vengano passati dei parametri corretti	newScreenplay(String title, Context context) newScreenplay(String title, Context context)	N.T.

TU4	Si verifica che, dato un oggetto della classe ScreenplayImpl creato, si possa salvare nella memoria del telefono un file xml_G di salvataggio relativo alle informazioni dello sceneggiato	saveScreenplay(Screenplay screenplay, Context context) save(Screenplay screenplay, Context context) save(Screenplay screenplay, Context context) getParsedData() parseCharacter(Node node) parseChapter(Node node) parseXml(File file) saveXML(File file, Screenplay screenplay)	N.T.
TU5	Si verifica che, dato un oggetto della classe ScreenplayImpl e dato un formato di esportazione scelto, venga generato un file nel formato desiderato corrispondente alla rappresentazione dello sceneggiato in formato audio o video	export() export(String type) export() export() exec() getCommand() getCommand() getCommand() getCommand() ExportAlgorithm() setScreenplay(Screenplay screenplay) export() AudioExport() finalizeExport() export() VideoExport() exportAudio() finalizeExport()	N.T.
TU5		export()	N.T.

TU6	Si verifica che, dato un oggetto della classe ScreenplayImpl e uno strumento di condivisione, venga generato un file a partire dallo sceneggiato creato e poi condiviso utilizzando lo strumento di condivisione scelto	share() share(String where) share() share() ShareAlgorithm() setFile(File file) share() FacebookShare() share() TwitterShare() share() WhatsappShare() share() TelegramShare() share()	N.T.
TU7	Si verifica che, dato un oggetto della classe ScreenplayImpl e una stringa, sia creato un nuovo capitolo associato all'oggetto avente come titolo la stringa passata come parametro	addChapter(Chapter chapter) addChapter(Chapter chapter)	N.T.
TU8	Si verifica che, creato un oggetto della classe ScreenplayImpl, data una stringa relativa al nome, una stringa relativa al nome della voce e un oggetto della classe Avatar, venga creato un nuovo personaggio da associare allo sceneggiato che sia riutilizzabile in altri progetti futuri	addCharacter(Character character) addCharacter(Character character) addCharacter(Character character) addCharacter(Character character) newCharacter(String name, String voice, Avatar avatar) addCharacter(Character character) add(Character object)	N.T.
TU9	Si verifica che, dato un oggetto della classe Screenplay e un oggetto della classe Character, venga eliminato il personaggio Character dall'oggetto Screenplay	removeCharacter(Character character) removeCharacter(Character character) play()	N.T.

TU10	Si verifica che, dato un oggetto della classe Screenplay creato e un altro passato come parametro, sia possibile importare i personaggi del progetto passato come parametro nell'oggetto corrente. Si verifica inoltre che, se l'oggetto da cui importare sia vuoto o non corretto, l'importazione fallisca	getCharacters() importCharacters(Screenplay screenplay) getCharacters() importCharacters(Screenplay screenplay) importCharacters(String screenplay, Context context) importCharacters(String screenplay, Context context)	N.T.
TU11	Si verifica che, dato un oggetto Chapter e creato un oggetto della classe Speech, quest'ultimo venga aggiunto all'array dell'oggetto Chapter contenente tutti gli oggetti Speech di suo possesso	addSpeech(Speech speech) addSpeech(Speech speech) add(Speech object)	N.T.
TU12	Si verifica che, dato un oggetto di tipo MivoqVoice e una stringa di testo, venga riprodotta la sintesi vocale del contenuto della stringa	speak(MivoqVoice v, String text) speak(String voiceID, String text) speak(String voiceID, String text)	N.T.
TU13	Si verifica che, dato un un percorso dove salvare il file, dato un oggetto di tipo MivoqVoice e una stringa di testo, venga originato un file nel percorso passato come parametro dall'array di byte ricevuto dal server di MIVOQ s.r.l.	synthesizeToFile(String path, MivoqVoice v, String text) synthesizeToFile(String path, String voiceID, String myEmotion, String text, Engine.listener myListener) synthesizeToFile(String path, String voiceID, String myEmotion, String text, Engine.listener myListener)	N.T.

TU14	Si verifica che, dato un oggetto di tipo MivoqVoice e data una stringa di testo, venga generato un array di byte mediante la richiesta al server di MIVOQ s.r.l.. Si verifica che in assenza di connessione dati, la stringa di testo venga sintetizzata dal motore di sintesi vocale di Android$_G$	getResponse() sendRequest(String text) synthesizeText(MivoqVoice v, String text) getResponse() sendRequest(String text) deliverResponse(byte[] response) getHeaders() getParams() parseNetworkResponse(NetworkResponse response)	N.T.
TU15	Si verifica che, data una stringa relativa al nome, una relativa al genere e una relativa alla lingua, venga creato un oggetto di tipo MivoqVoice	createVoice(String name, String gender, String myLanguage) MivoqVoice(String name, String myVoiceName, Language locale) createVoice(String name, String gender, String myLanguage) createMivoqVoice(String s) createMivoqVoice(String s) createVoice(String name, String gender, String myLanguage)	N.T.
TU16	Si verifica che alla voce vengano applicati gli effetti offerti dal server di MIVOQ s.r.l..	getName() getValue() toString() getName() getValue() toString() EffectImpl(String name)	N.T.
TU17	Si verifica che l'utente possa abbinare un'emozione ad una battuta	Emotion(String s) toString()	N.T.
TU18	Si verifica che sia possibile modificare un effetto applicato ad una voce	setValue(String f) setValue(String f)	N.T.
TU19	Si verifica che si possa impostare la lingua desiderata per la voce	Language(String l) toString()	N.T.
TU20	Si verifica che si possa modificare la lingua associata ad una voce	setLanguage(String l) setState(String s) setModifier(String m)	N.T.

TU21	Si verifica che si possano modificare i parametri relativi al sesso, alla lingua, al nome e agli effetti di un oggetto MivoqVoice	setEffect(Effect e) setEmotion(Emotion e) setFemaleDe(boolean b) setGender(String g) setGenderLanguage(String gen, String myLang) setName(String n)	N.T.
TU22	Si verifica che si possa riprodurre un testo di prova differente per ogni lingua	getSampleText(String myLang)	N.T.
TU23	Si verifica che si possa navigare all'interno dell'applicazione di configurazione	HomePresenterImpl(Context context) goToNewVoiceActivity(Context context) goToVoiceListActivity(Context context) goToVoiceListActivity(Context context) goToNewVoiceActivity(Context context) goToEditVoiceActivity(Context context, MivoqVoice mivoqVoice) setActivity(VoiceListActivityInterface voiceListActivityInterface) VoiceListPresenterImpl(Engine engine) goToEditVoiceActivity(Context context, MivoqVoice mivoqVoice) setActivity(VoiceListActivityInterface voiceListActivityInterface) VoicePresenterImpl(Engine engine) VoicePresenterImpl(Mivoq mivoq) setActivity(NewVoiceActivityInterface newVoiceActivityInterface) setActivity(EditVoiceActivityInterface newVoiceActivityInterface) setActivity(NewVoiceActivityInterface newVoiceActivityInterface) setActivity(EditVoiceActivityInterface newVoiceActivityInterface)	N.T.
TU24	Si verifica che le activity NewVoiceActivity e EditVoiceActivity vengano riempite con i dati corretti	loadVoiceGender(Context context) loadVoiceLanguage(Context context) loadVoiceGender(Context context) loadVoiceLanguage(Context context)	N.T.

TU25	Si verifica che si possano caricare le voci presenti in memoria	loadVoiceNames(Context context) getVoicesAdapter(Context context) loadVoiceNames(Context context) getVoicesAdapter(Context context)	N.T.
TU26	Si verifica che si possa navigare all'interno dell'applicazione per la creazione degli sceneggiati	setActivity(ListChapterInterface listChapterInterface) setActivity(NewChapterInterface newChapterInterface) goToListSpeechesActivity(Context context, String selected) goToListCharactersActivity(Context context) goToEditChapterActivity(Context context, String selected) goToNewChapterActivity(Context context) goToNewCharacterActivity(Context context) setActivity(ListChapterInterface listChapterInterface) goToListSpeechesActivity(Context context, String selected) goToListCharactersActivity(Context context) goToEditChapterActivity(Context context, String selected) goToNewChapterActivity(Context context) goToNewCharacterActivity(Context context) setActivity(NewSpeechInterface newSpeechInterface) goToEditSpeechActivity(Context context, Speech selected) goToListCharactersActivity(Context context) goToEditChapterActivity(Context context) goToNewCharacterActivity(Context context)	N.T.

TU26		goToNewSpeechActivity(Context context) setActivity(ListSpeechesInterface listSpeechesInterface) goToEditSpeechActivity(Context context, Speech selected) goToListCharactersActivity(Context context) goToEditChapterActivity(Context context) goToNewCharacterActivity(Context context) goToNewSpeechActivity(Context context) setActivity(EditSpeechInterface editSpeechInterface) setActivity(EditSpeechInterface editSpeechInterface) setActivity(ListCharacterInterface listCharacterInterface) setActivity(NewCharacterInterface newCharacterInterface) setActivity(EditCharacterInterface editCharacterInterface) goToEditCharacterActivity(Context context, Character character) goToNewCharacterActivity(Context context) goToListCharacterActivity(Context context) setActivity(ListCharacterInterface listCharacterInterface) setActivity(EditCharacterInterface editCharacterInterface)	N.T.
TU26		setActivity(NewCharacterInterface newCharacterInterface) goToEditCharacterActivity(Context context, Character selected) goToNewCharacterActivity(Context context) goToListCharacterActivity(Context context)	N.T.

TU27	Dato un oggetto Screenplay e una stringa relativa al nome di un oggetto Chapter creato e contenuto nella lista dell'oggetto Screenplay, si verifica che si sia possibile modificare la posizione di tale oggetto Chapter nella lista. Si verifica inoltre che, se la stringa non corrisponde ad alcun Chapter, l'operazione fallisca	getSpeechIterator() moveUpChapter(String chapterTitle) moveDownChapter(String chapterTitle) moveUpChapter(String chapterTitle) moveDownChapter(String chapterTitle) moveUpSpeech(Speech speech)	N.T.
TU28	Si verifica che si possa modificare l'ordine degli elementi di classe Speech contenute in un oggetto List<Speech> di un oggetto Chapter	moveUpSpeech(ListIterator<Speech> iterator) moveDownSpeech(ListIterator<Speech> iterator) moveUpSpeech(ListIterator<Speech> iterator) moveDownSpeech(ListIterator<Speech> iterator) moveUpSpeech(Speech speech) moveDownSpeech(Speech speech) moveDownSpeech(Speech speech)	N.T.
TU29	Si verifica che, data una stringa relativa al percorso di un file di test salvato in memoria e un oggetto Chapter, si possa settare il file di test come soundtrack dell'oggetto Chapter. Si verifica inoltre che se il file non risulti coerente al formato richiesto o risulti avere una dimensione maggiore di 15 MB non avvenga l'inserimento	setSoundtrack(Soundtrack soundtrack) setSoundtrack(Soundtrack soundtrack) setSoundtrack(Soundtrack soundtrack) setSoundtrack(Soundtrack soundtrack)	N.T.
TU30	Si verifica che si possa eliminare la soundtrack assegnata ad un oggetto Chapter	deleteSoundtrack() deleteSoundtrack()	N.T.

TU31	Si verifica che, data una stringa relativa al percorso di un file di test fornito in memoria e un oggetto Chapter, si possa caricare il file espresso dal path come immagine di sfondo e creare un oggetto Background. Si verifica inoltre che, se il file non corrisponde al formato atteso o se tale file risulti avere una dimensione maggiore di 10 MB, fallisca il caricamento di tale file	setBackground(Background background) setBackground(Background background) setBackground(Background background) setBackground(Background background)	N.T.
TU32	Si verifica che, dato un oggetto Chapter con un oggetto Background creato, si possa eliminare l'oggetto Background	deleteBackground() deleteBackground()	N.T.
TU33	Si verifica che, dato un oggetto Chapter con una List<Speech> di test creata, si possa eliminare un oggetto Speech da tale lista	deleteSpeech(Speech speech) deleteSpeech(Speech speech)	N.T.
TU34	Si verifica che, data una stringa, si possa creare un oggetto della classe Chapter avente titolo tale stringa	setTitle(String title) ChapterImpl(ChapterBuilder builder) setTitle(String title) setTitle(String title) ChapterBuilderImpl() newChapter(String title) newChapter(String title)	N.T.

TU35	Si verifica che, dato un oggetto Speech, si possano modificare il testo, l'emozione assegnata, il personaggio corrispondente e il suono di arricchimento associato	setSoundFx(SoundFx soundFx) setTitle(String title) setSoundFx(SoundFx soundFx) setText(String text) setEmotion(String emotionID) setCharacter(Character character) setSoundFx(SoundFx soundFx) setText(String text) setEmotion(String emotionID) setCharacter(Character character) setSoundFx(SoundFx soundFx) setSpeechText(String text) setSpeechEmotion(String emotion) setSpeechCharacter(Character character) setSpeechText(String text) setSpeechEmotion(String emotion) setSpeechCharacter(Character character) setText(String text)	N.T.
TU35		setEmotion(String emotionID) setCharacter(Character character) setSoundFX(SoundFx soundFx)	N.T.
TU36	Si verifica che, dato un oggetto Chapter, si possano ottenere i suoi campi dati	getTitle() getTitle() getSpeechIterator() getSpeeches(Context context) getChapterTitle() getChapterTitle() getSpeeches()	N.T.

TU37	Si verifica che, dato un oggetto Speech, si possano ottenere i suoi campi dati	getText() getEmotion() getCharacter() getAudio() getText() getEmotion() getCharacter() getAudio() getEmotions(Context context) getSpeechText() getSpeechEmotion() getSpeechCharacter() getSpeechText() getSpeechEmotion() getSpeechCharacter()	N.T.
TU38	Si verifica che, dato un oggetto Screenplay, si possano ottenere i suoi campi dati	getChapter(String title) getCharacterByName(String name) getTitle() getChapterIterator() getTitle() getChapter(String title) getCharacterByName(String name) getScreenplayTitle() getScreenplay() getScreenplay() getScreenplayTitle()	N.T.
TU39	Si verifica che, dato un oggetto File da convertire e un oggetto File corrispondente alla destinazione, si possa convertire il file in formato $mp3_G$. Si verifica inoltre che, se il file da convertire non risulti essere presente in memoria o se non risulti coerente al formato accettato, la conversione fallisca	Mp3Converter(Context context, File fileToConvert, File destination) setFile(File fileToConvert) setDestination(File destination)	N.T.

TU40	Si verifica che, dato un file di testo contenente una lista di nomi di file, si possa concatenare tali file in un unico file audio. Si verifica inoltre che, se il file di testo non esiste o se i file in elenco non esistono, fallisca la concatenazione	AudioConcatenator(Context context, File destination) AudioConcatenator(Context context, Vector<File> files, File destination) addFile(File file) setDestination(File destination) createFileList() concatenateSpeeches()	N.T.
TU41	Si verifica che si possano mixare due file audio di test differenti. Si verifica inoltre che nel caso in cui uno dei due file non esistesse o non fosse conforme al formato richiesto l'operazione fallisca	AudioMixer(Context context, File primaryFile, File secondaryFile, File destination) setPrimaryFile(File primaryFile) setSecondaryFile(File secondaryFile) setDestination(File destination)	N.T.
TU42	Si verifica che, dato un file di testo contenente una lista di nomi di file, si possa concatenare tali file in un unico file video. Si verifica inoltre che, se il file di testo non esiste o se i file in elenco non esistono, fallisca la concatenazione	concatenateImages()	N.T.
TU43	Si verifica che, dato un oggetto Character, si possa settarne o modificarne il campo dati relativo al nome, all'avatar$_G$ e alla voce	setAvatar(Avatar avatar) setName(String name) setVoice(String voiceID) setAvatar(Avatar avatar) setVoice(String voiceID) setName(String name) setAvatar(Avatar avatar) setVoice(String voiceID) setName(String name) setName(String name) setAvatar(Avatar avatar) setVoice(String voiceID)	N.T.

TU44	Si verifica che si possa accedere ai campi dati di un oggetto Character	getAvatar() getName() getVoiceId() getName() getAvatar() getVoiceId()	N.T.
TU45	Si verifica che, data una stringa relativa a del testo, una relativa ad un personaggio e una relativa ad un'emozione, venga creato un nuovo oggetto della classe Speech	newSpeech(String text, String chatacterName, String emotion)	N.T.
TU46	Si verifica che, dato un oggetto di tipo MivoqVoice, si possa accedere ai suoi campi dati	getEffects() getEncodedName(String gen, String myLang) getGender() getLanguage() getName() getStringEffects() getVoiceName() getGenderAdapter(Context context) getLanguageAdapter(Context context) getVoiceName() getGenderAdapter(Context context) getLanguageAdapter(Context context) getVoiceName()	N.T.
TU47	Si verifica che, dato un oggetto MivoqVoice, venga rimosso un effetto	removeEffect(int index)	N.T.
TU48	Si verifica che venga creato un oggetto AudioRequest che permetta la gestione della richiesta della sintesi vocale al server di MIVOQ s.r.l.	AudioRequest(int post, String mUrl, Map<String, String> params, Response.Listener<byte[]> listener, Response.ErrorListener errorListener)	N.T.
TU49	Si verifica che venga creata una connessione con il server di MIVOQ s.r.l.	createConnection() MivoqConnectionFactory() createConnection() MivoqConnectionImpl()	N.T.

TU50	Si verifica che, dato un indice intero, il metodo rimuova dalla lista di voci personalizzate disponibili la voce nella posizione indicata dall'indice.	getVoices() removeVoice(int index) removeVoice(int index) removeVoice(int index)	N.T.
TU51	Si verifica che il metodo ritorni un vettore con tutte le voci personalizzate disponibili per l'utente	getVoices() getVoices()	N.T.

Tabella 5: Test di Unità

D.4 Test di Validazione

In questa sezione vengono descritti i test di validazione che servono per accertarsi che il prodotto realizzato sia conforme alle attese. Per ogni test vengono descritti i vari passi che un utente deve eseguire per testare i requisiti ad esso associati, mentre il tracciamento tra test di validazione e requisiti è riportato nel documento *Analisi dei Requisiti*.

D.4.1 Descrizione dei test di Validazione

Codice	Descrizione	Stato
TV1	L'utente intende creare un nuovo sceneggiato. All'utente è richiesto di: • Aver avviato l'applicazione; • Premere il pulsante Crea nuovo; • Inserire il titolo del nuovo sceneggiato; • Premere il pulsante di conferma creazione; • Verificare che sia stato creato il nuovo sceneggiato.	N.T.
TV2	L'utente intende creare una nuova voce personalizzata. All'utente è richiesto di: • Accedere all'apposita sezione; • Premere il pulsante Crea nuova voce; • Inserire i dati necessari alla creazione di una nuova voce; • Premere il pulsante di conferma per la creazione di una nuova voce; • Verificare che sia stata creata la nuova voce.	N.T.

TV3	L'utente non autenticato intende registrarsi al sito di MIVOQ s.r.l.. All'utente è richiesto di: • Trovarsi nella sezione apposita; • Compilare il *form* di registrazione; • Premere il pulsante di conferma; • Verificare che la registrazione sia avvenuta correttamente attraverso l'autenticazione.	N.T.
TV4	L'utente intende autenticarsi al sito di MIVOQ s.r.l.. All'utente è richiesto di: • Trovarsi nella sezione apposita; • Inserire le credenziali nel *form*; • Premere il pulsante di autenticazione; • Verificare che l'autenticazione sia effettivamente avvenuta.	N.T.
TV5	L'utente intende selezionare una voce per i servizi Text-To-Speech$_G$ del sistema. All'utente è richiesto di: • Trovarsi nella sezione dedicata; • Visualizzare l'elenco delle voci disponibili; • Selezionare la voce desiderata; • Confermare la selezione; • Verificare che il sistema utilizzi la voce selezionata.	N.T.

TV6	L'utente autenticato intende campionare la propria voce. All'utente è richiesto di: • Trovarsi nella sezione dedicata; • Premere il pulsante per avviare il campionamento; • Leggere ad alta voce le frasi che compaiono sullo schermo; • Verificare la creazione della nuova voce ottenuta con il campionamento.	N.T.
TV8	L'utente intende compiere un'operazione su uno sceneggiato esistente. All'utente è chiesto di: • Selezionare lo sceneggiato su cui intende operare; • Effettuare l'operazione desiderata; • Verificare l'effettiva esecuzione dell'operazione scelta.	N.T.
TV8.10	L'utente intende creare un nuovo capitolo per lo sceneggiato. All'utente è richiesto di: • Trovarsi all'interno di uno sceneggiato; • Premere il pulsante Nuovo capitolo; • Inserire il titolo del nuovo capitolo; • Premere il pulsante di conferma per creare il nuovo capitolo; • Verificare che il capitolo sia stato effettivamente creato.	N.T.

TV8.11	L'utente intende operare su un capitolo esistente. All'utente è richiesto di: • Trovarsi all'interno del capitolo da modificare; • Effettuare un'operazione; • Verificare che l'operazione sia stata eseguita correttamente.	N.T.
TV8.1	L'utente intende creare un nuovo personaggio per lo sceneggiato creato. All'utente è richiesto di: • Trovarsi all'interno di uno sceneggiato; • Premere il pulsante Nuovo personaggio; • Inserire il nome del personaggio; • Premere il pulsante di conferma per creare il personaggio; • Verificare l'avvenuta creazione del nuovo personaggio.	N.T.

TV8.5	L'utente intende importare un personaggio creato precedentemente per un altro sceneggiato. All'utente è richiesto di: • Trovarsi all'interno di uno sceneggiato; • Premere il pulsante Importa personaggi; • Visualizzare la lista degli sceneggiati creati; • Selezionare uno degli sceneggiati della lista; • Visualizzare la lista dei personaggi dello sceneggiato selezionato; • Selezionare uno dei personaggi dalla lista; • Premere il pulsante di conferma; • Verificare che il personaggio sia stato importato.	N.T.
TV8.8	L'utente intende rimuovere un personaggio creato o importato nello sceneggiato. All'utente è richiesto di: • Trovarsi nella sezione apposita; • Visualizzare l'elenco di tutti i personaggi inseriti nello sceneggiato; • Selezionare il personaggio da rimuovere; • Premere il pulsante Rimuovi personaggio; • Verificare che il personaggio sia stato rimosso dall'elenco dei personaggi disponibili.	N.T.

TV8.7	L'utente intende effettuare un'operazione su un personaggio esistente. All'utente è richiesto di: • Trovarsi nella sezione appropriata; • Premere il pulsante Modifica personaggio; • Effettuare l'operazione desiderata; • Verificare che l'operazione sia stata effettuata correttamente.	N.T.
TV8.12	L'utente intende condividere uno sceneggiato. All'utente è richiesto di: • Premere il pulsante Condividi Sceneggiato; • Selezionare dove condividere lo sceneggiato; • Verificare che il sistema notifichi la corretta condivisione.	N.T.
TV8.4	L'utente intende cancellare un capitolo creato. All'utente è richiesto di: • Selezionare il capitolo da eliminare; • Premere il pulsante Elimina Capitolo; • Premere il pulsante di conferma; • Verificare che il capitolo eliminato non compaia più nell'elenco dei capitoli disponibili.	N.T.
TV8.3	L'utente intende salvare lo sceneggiato. All'utente è richiesto di: • Premere il pulsante Salva; • Verificare che lo sceneggiato sia stato salvato correttamente.	N.T.

TV8.2	L'utente intende modificare l'ordine dei capitoli dello sceneggiato. All'utente è richiesto di: • Trovarsi nella sezione apposita; • Visualizzare l'elenco dei capitoli; • Tenere premuto il capitolo da spostare; • Trascinarlo nella nuova posizione; • Verificare che l'ordine dei capitoli sia cambiato correttamente.	N.T.
TV8.6	L'utente intende esportare lo sceneggiato. All'utente è richiesto di: • Premere il pulsante Esporta sceneggiato; • Selezionare il formato tra audio o video; – Specificare per il formato audio se esportare in formato .mp3$_G$ o .wav$_G$. • Premere il pulsante di Conferma; • Verificare che il sistema notifichi la corretta esportazione.	N.T.
TV8.9	L'utente intende modificare il titolo di uno sceneggiato. All'utente è richiesto di: • Inserire il nuovo titolo; • Verificare che il titolo corrisponda a quello appena inserito.	N.T.

TV8.11.9	L'utente intende eliminare una battuta creata. All'utente è richiesto di: • Trovarsi dentro un capitolo; • Visualizzare l'elenco delle battute; • Selezionare la battuta da eliminare; • Premere il pulsante di Elimina; • Verificare che la battuta eliminata non sia più presente tra quelle inserite.	N.T.
TV8.11.6	L'utente intende cambiare l'ordine delle battute di un capitolo. All'utente si richiede di: • Trovarsi all'interno di un capitolo; • Visualizzare l'elenco delle battute; • Tenere premuto sulla battuta da spostare; • Trascinarla nella nuova posizione; • Verificare che la battuta sia effettivamente nella nuova posizione;	N.T.
TV8.11.5	L'utente intende operare su una battuta. All'utente è richiesto di: • Trovarsi dentro un capitolo; • Visualizzare l'elenco delle battute; • Selezionare la battuta da modificare; • Eseguire l'operazione desiderata; • Verificare che la modifica sia stata apportata.	N.T.

TV8.11.5.4	L'utente intende associare un'emozione ad una battuta. All'utente è richiesto di: • Selezionare l'emozione desiderata tra quelle nell'elenco; • Verificare che alla battuta sia stata associata tale emozione.	N.T.
TV8.11.4	L'utente intende creare una nuova battuta. All'utente è richiesto di: • Trovarsi dentro un capitolo; • Premere il pulsante Nuova battuta; • Selezionare il personaggio a cui associare la battuta tra quelli in elenco; • Inserire il testo della battuta; • Premere il pulsante di conferma; • Verificare la presenza della nuova battuta nell'elenco delle battute disponibili.	N.T.
TV8.11.3	L'utente intende operare sullo sfondo (audio e video) di un capitolo. All'utente è richiesto di: • Trovarsi dentro un capitolo; • Premere il pulsante Inserisci; – Selezionare immagine o audio; – Visualizzare l'elenco dei file disponibili; – Selezionare il file da inserire. • Verificare che il file sia stato inserito come sfondo.	N.T.

TV8.11.11	L'utente intende modificare il titolo di un capitolo. All'utente è richiesto di: • Trovarsi all'interno di uno sceneggiato; • Selezionare il capitolo desiderato; • Modificare il titolo; • Verificare che il titolo sia stato modificato correttamente.	N.T.
TV9	L'utente intende operare su una voce esistente. All'utente è richiesto di: • Trovarsi nella sezione dedicata; • Selezionare la voce da modificare; • Apportare le modifiche necessarie; • Verificare che le modifiche siano state apportate avviando l'anteprima della voce.	N.T.
TV9.2	L'utente intende eliminare una voce personalizzata. All'utente è richiesto di: • Trovarsi nella sezione dedicata; • Selezionare la voce da eliminare; • Premere il pulsante Elimina voce; • Verificare che la voce eliminata non sia più disponibile dall'elenco delle voci.	N.T.

Tabella 6: Test di Validazione

E Resoconto delle attività di verifica

Di seguito è presente un riassunto delle attività di verifica suddivise in periodi. Per maggiori informazioni sulla pianificazione delle consegne e il raggiungimento delle *milestone*$_G$ si veda il documento *Piano di Progetto v2.0.0*. I processi sono stati verificati utilizzando le metriche di processo descritte nella sezione Metriche per i processi. Sono quindi stati riportati i valori di *Schedule Variance* e *Budget Variance* relativi a tutti i processi di ciascun periodo.

E.1 Periodo di Analisi

Il periodo termina con la consegna della documentazione tecnico-economica descritta nel *Piano di Progetto v2.0.0*. Nel periodo antecedente alla consegna della documentazione sono state eseguite le attività di verifica dei processi e dei documenti redatti. Ogni documento è stato controllato tramite la tecnica del *walkthrough* descritta nella sezione Walkthrough. Si sono riscontrati diversi errori che sono stati segnalati in modo informale ai redattori. Si precisa inoltre che nel corso di questa attività il gruppo non si è occupato di stilare la lista di controllo necessaria per la tecnica dell'*Inspection*, tecnica descritta nella sezione Inspection. Infine, dopo la correzione dei documenti, si è provveduto a calcolare le metriche descritte nella sezione Misure e metriche.
Il tracciamento dei requisiti è avvenuto grazie all'utilizzo di un *software* realizzato appositamente dagli *Amministratori* del gruppo.

E.1.1 Dettaglio della verifica di analisi

Durante il periodo di analisi sono previsti più momenti in cui viene attivato il processo di verifica. Si è cercato di riportare in questa sezione tutti i risultati che sono stati ottenuti durante questi momenti. Ove fosse necessario, si sono tratte delle conclusioni sui risultati ottenuti e su come essi possono essere migliorati.

E.1.1.1 Verifica sui processi Di seguito vengono riportati i valori di *Schedule Variance* (SV) e *Budget Variance* (BV) descritti nella sezione Metriche per i processi, per le attività nel periodo di analisi che hanno portato alla stesura dei documenti.

Attività	SV	BV
Piano di Qualifica v1.0.0	-30 €	-30 €
Piano di Progetto v1.0.0	0 €	-30 €
Norme di Progetto v1.0.0	0 €	-30 €
Analisi dei Requisiti v1.0.0	+50 €	+25 €
Studio di Fattibilità v1.0.0	+30 €	0 €
Glossario v1.0.0	0 €	0 €

Tabella 7: Metriche dei processi

Complessivamente il periodo di analisi ha:

- SV : +50€;
- BV : -65€.

Da questi indici si può notare che:

- Grazie a una pianificazione accurata delle scadenze, le attività sono state svolte nei tempi pianificati, alcune volte terminando in anticipo rispetto alle date di fine prefissate;
- Il costo complessivo si è di poco discostato da quanto pianificato nel *Piano di Progetto v1.0.0*. La causa di questa variazione dei costi è imputabile alla poco esperienza del gruppo nel redigere tale tipologia di documenti.

E.1.1.2 Verifica sui prodotti

E.1.1.2.1 Documenti Riportiamo in questa sezione i valori degli indici Gulpease calcolati per ogni documento. Un documento è valido se e solo se rispetta le metriche descritte nella sezione Metriche per i documenti. La tabella mostra come i documenti rientrino tutti nel *range* ottimale precedentemente definito. Rispettano pertanto il livello di leggibilità minimo desiderato.

Documento	Valore	Indice
Piano di Qualifica v1.0.0	54	Superato
Piano di Progetto v1.0.0	58	Superato
Norme di Progetto v1.0.0	56	Superato
Analisi dei Requisiti v1.0.0	60	Superato
Studio di Fattibilità v1.0.0	54	Superato
Glossario v1.0.0	50	Superato

Tabella 8: Indici Gulpease dei documenti

E.1.1.3 Esito della revisione di analisi Durante lo sviluppo del progetto vi saranno quattro revisioni da parte del Committente. Il Committente segnalerà problematiche che verranno riscontrate valutando globalmente l'andamento del progetto e di ogni singolo documento. Grazie a queste informazioni sarà possibile correggere gli errori commessi, al fine di procedere lungo delle attività lavorative che risultino corrette e verificate.

E.2 Periodo di Analisi di Dettaglio

E.2.1 Dettaglio della verifica di analisi di dettaglio

Durante il periodo di analisi di dettaglio sono previsti più momenti in cui viene attivato il processo di verifica. Ogni documento è stato controllato attraverso il metodo dell'*Inspection*, come descritto nella Metriche per i processi. Si è cercato di riportare in questa sezione tutti i risultati che sono stati ottenuti durante tali momenti. Ove fosse necessario, si sono tratte delle conclusioni sui risultati ottenuti e su come essi possono essere migliorati.

E.2.1.1 Verifica sui processi Di seguito vengono riportati i valori di *Schedule Variance* (SV) e *Budget Variance* (BV), descritti nella sezione Metriche per i processi, per le attività nel periodo di analisi di dettaglio che hanno portato alla revisione dei documenti.

Attività	SV	BV
Piano di Qualifica v1.0.0	0 €	0 €
Piano di Progetto v1.0.0	+30 €	0 €
Norme di Progetto v1.0.0	+40 €	0 €
Analisi dei Requisiti v1.0.0	0 €	-75 €
Glossario v1.0.0	0 €	0 €

Tabella 9: Metriche dei processi

Complessivamente il periodo di analisi di dettaglio ha:

- SV : +70 €;
- BV : -75 €.

Da questi indici si può notare che:

- Grazie alla pianificazione e alle scadenze fissate, le attività sono state svolte nei tempi previsti;
- Il costo complessivo si è di poco discostato da quanto pianificato nel *Piano di Progetto v1.0.0*.

E.2.1.2 Verifica sui prodotti

E.2.1.2.1 Verifica sui documenti Riportiamo in questa sezione i valori degli indici Gulpease calcolati per ogni documento. Un documento è valido se e solo se rispetta le metriche descritte nella sezione Metriche per i documenti. La tabella mostra come i documenti rientrino tutti nel *range* ottimale precedentemente definito. Rispettano pertanto il livello di leggibilità minimo desiderato.

Documento	Valore	Indice
Piano di Qualifica v1.0.0	55	Superato
Piano di Progetto v1.0.0	57	Superato
Norme di Progetto v1.0.0	57	Superato
Analisi dei Requisiti v1.0.0	58	Superato
Glossario v1.0.0	55	Superato

Tabella 10: Indici Gulpease dei documenti

E.3 Periodo di Progettazione Architetturale

Durante il periodo di progettazione architetturale sono previsti più momenti in cui viene attivato il processo di verifica. Si è cercato di riportare in questa sezione tutti i risultati che sono stati ottenuti durante questi momenti. Ove fosse necessario, si sono tratte delle conclusioni sui risultati ottenuti e su come essi possono essere migliorati. Il tracciamento dei *package* e delle classi è avvenuto grazie all'utilizzo di un *software* realizzato appositamente dagli *Amministratori* del gruppo. Il tracciamento di requisiti, *package*, classi e test è avvenuto grazie all'utilizzo di un *software* realizzato appositamente dagli *Amministratori* del gruppo e ampliato durante il periodo di progettazione.

E.3.1 Dettaglio della verifica di progettazione architetturale

E.3.1.1 Verifica sui processi Di seguito vengono riportati i valori di *Schedule Variance* (SV) e *Budget Variance* (BV) descritti nella sezione Metriche per i processi, per le attività nel periodo di analisi di dettaglio che hanno portato alla stesura dei documenti.

Attività	SV	BV
Piano di Qualifica v2.0.0	+25 €	+50 €
Piano di Progetto v2.0.0	0 €	0 €
Norme di Progetto v2.0.0	-30 €	-30 €
Analisi dei Requisiti v2.0.0	-50 €	-150 €
Specifica Tecnica v1.0.0	+30 €	+72 €
Glossario v2.0.0	0 €	0 €

Tabella 11: Metriche dei processi

Complessivamente il periodo di progettazione architetturale ha:

- SV : -25 €;
- BV : -58 €.

Da questi indici si può notare che:

- Complessivamente il periodo di progettazione architetturale ha sforato i tempi e i costi pianificati a causa della mole di correzioni apportate al documento di *Analisi dei Requisiti v1.0.0*.
- Il costo complessivo si è discostato di 83€ da quanto pianificato nel *Piano di Progetto v1.0.0*. La causa di questa variazione dei costi è dovuta all'ingenuità con cui è stata effettuata l'analisi e lo studio dei requisiti.

E.3.1.2 Verifica sui prodotti

E.3.1.2.1 Verifica sui documenti Riportiamo in questa sezione i valori degli indici Gulpease calcolati per ogni documento. Un documento è valido se e solo se rispetta le metriche descritte nella sezione Metriche per i documenti. La tabella mostra come i documenti

Documento	Valore	Indice
Piano di Qualifica v2.0.0	57	Superato
Piano di Progetto v2.0.0	56	Superato
Norme di Progetto v2.0.0	58	Superato
Analisi dei Requisiti v2.0.0	57	Superato
Specifica Tecnica v1.0.0	46	Superato
Glossario v1.0.0	55	Superato

Tabella 12: Indici Gulpease dei documenti

rientrino tutti nel *range* di accettazione definito in sezione Metriche per i documenti. L'unico documento che non rientra nel *range* ottimale è la *Specifica Tecnica v1.0.0*. Il punteggio basso è imputabile all'estremo tecnicismo che caratterizza la struttura del documento molto, ricco di tabelle ed elenchi puntati. Questi due fattori tendono infatti ad abbassare il valore dell'indice. Gli altri documenti rientrano invece nel *range* ottimale, con un indice di leggibilità che supera la sufficienza.

E.4 Periodo di Progettazione di dettaglio e codifica

Durante il periodo di progettazione di dettaglio e codifica sono previsti più momenti in cui viene attivato il processo di verifica. Si è cercato di riportare in questa sezione tutti i risultati che sono stati ottenuti durante questi momenti. Ove fosse necessario, si sono tratte delle conclusioni sui risultati ottenuti e su come essi possono essere migliorati. Il tracciamento dei *requisiti* e delle classi è avvenuto grazie all'utilizzo di un *software* realizzato appositamente dagli *Amministratori* del gruppo.

E.4.1 Dettaglio della progettazione di dettaglio e codifica

E.4.1.1 Verifica sui processi Di seguito vengono riportati i valori di *Schedule Variance* (SV) e *Budget Variance* (BV) descritti nella sezione Metriche per i processi, per le attività nel periodo di analisi di dettaglio che hanno portato alla stesura dei documenti.

Attività	SV	BV
Piano di Qualifica v3.0.0	-25 €	0 €
Piano di Progetto v3.0.0	0 €	0 €
Norme di Progetto v3.0.0	+30 €	+30 €
Analisi dei Requisiti v3.0.0	0 €	+50 €
Specifica Tecnica v2.0.0	+25 €	-22 €
Glossario v3.0.0	0 €	0 €
Definizione di Prodotto v1.0.0	-35 €	-20 €
Manuale utente v1.0.0	10 €	+40 €
Codifica	0 €	+35 €

Tabella 13: Metriche dei processi

Complessivamente il periodo di progettazione architetturale ha:

- SV : +5 €;
- BV : +113 €.

Da questi indici si può notare che:

- Complessivamente il periodo di progettazione architetturale ha sforato i tempi e i costi pianificati a causa della mole di correzioni apportate al documento di *Analisi dei Requisiti v1.0.0*.
- Il costo complessivo si è discostato di -88€ da quanto pianificato nel *Piano di Progetto v2.0.0*. La causa di questa variazione dei costi è dovuta allo sforzo del gruppo di recuperare le eccessive spese del periodo precedente.

E.4.1.2 Verifica sui prodotti

E.4.1.2.1 Verifica sui documenti Riportiamo in questa sezione i valori degli indici Gulpease calcolati per ogni documento. Un documento è valido se e solo se rispetta le metriche descritte nella sezione Metriche per i documenti. La tabella mostra come i documenti

Documento	Valore	Indice
Piano di Qualifica v3.0.0	55	Superato
Piano di Progetto v3.0.0	56	Superato
Norme di Progetto v3.0.0	59	Superato
Analisi dei Requisiti v3.0.0	60	Superato
Specifica Tecnica v2.0.0	45	Superato
Glossario v3.0.0	55	Superato
Definizione di Prodotto v1.0.0	62	Superato
Manuale utente v1.0.0	51	Superato

Tabella 14: Indici Gulpease dei documenti

rientrino tutti nel *range* di accettazione definito in sezione Metriche per i documenti. L'unico documento che non rientra nel *range* ottimale è la *Specifica Tecnica v2.0.0*. Il punteggio basso è imputabile all'estremo tecnicismo che caratterizza la struttura del documento molto ricco di tabelle ed elenchi puntati. Questi due fattori tendono infatti ad abbassare il valore dell'indice. Gli altri documenti rientrano invece nel *range* ottimale, con un indice di leggibilità che supera la sufficienza.

Metrica	Medio	Massimo	Percentuale	Esito
Numero livelli di annidamento	1,30	4		Superato
Numero di attributi per classe	7	20		Superato
Numero di parametri per metodo	1	5		Superato
Linee di codice per commento			5,23%	Non superato

Tabella 15: Metriche sul codice del package Libraries

Metrica	Medio	Massimo	Percentuale	Esito
Numero livelli di annidamento	1,34	6		Superato
Numero di attributi per classe	6,76	26		Superato
Numero di parametri per metodo	0,79	4		Superato
Linee di codice per commento			8,22%	Non superato

Tabella 16: Metriche sul codice del package Drama

E.4.1.2.2 Verifica sul codice

Copertura dei requisiti obbligatori	80%	Non superato
Copertura dei requisiti desiderabili	76%	Superato

Tabella 17: Copertura dei requisiti

F Tracciamento obiettivi di qualità - Metriche

Obiettivo di qualità	**Descrizione**	**Metrica**	**Descrizione**
OQPC1	Miglioramento continuo	MPC1	Software Process Improvement and Capability Determination
OQPC2	Quantificazione dei ritardi	MPC2	Schedule Variance
OQPC3	Quantificazione dei costi	MPC3	Budget Variance
OQPDD1	Indice di qualità dei documenti	MPDD1	Indice di Gulpease
OQPDS1	Copertura dei requisiti obbligatori	MPDS1	Copertura dei requisiti obbligatori
OQPDS2	Copertura dei requisiti desiderabili	MPDS2	Copertura dei requisiti desiderabili
OQPDS3	Manutenibilità e comprensibilità del codice	MPDS3,MPDS4, MPDS5,MPDS6, MPDS7	Numero di livelli di annidamento, Numero di attributi per classe, Numero di parametri per metodo, Linee di codice per linee di commento, Copertura di codice
OQPDS4	Superamento dei test	MPDS8	Percentuale di test superati
OQPDS5	Robustezza	MPDS9	Failure Avoidance
OQPDS6	Funzionamento privo di interruzioni	MPDS10	Breakdown Avoidance

Tabella 18: Tracciamento obiettivi di qualità - Metriche