Sigma18Unipd@gmail.com

Norme di progetto

Responsabile	Pietro Crotti
Redattori	Mirco Borella Alessandro Bernardello Aleena Mathew Matteo Marangon Marco Egidi
Verificatori	Mirco Borella Carmelo Russello Matteo Marangon Alessandro Bernardello Aleena Mathew Pietro Crotti

Versione	0.7.4
Tipo	Documento Interno
Destinatario	*Sigma18*

Abstract dei contenuti:

Documento contenente le Norme di progetto che definiscono le modalità di lavoro del gruppo.

Registro delle versioni

Versione	Data	Autori	Verificatori	Descrizione Modifiche
0.7.4	2025/06/27	Matteo Marangon	Aleena Mathew	Correzione errore di battitura
0.7.3	2025/06/27	Aleena Mathew	Matteo Marangon	Correzioni lessicali e grammaticali
0.7.2	2025/06/06	Aleena Mathew	Matteo Marangon	Correzione di una metrica
0.7.1	2025/06/02	Matteo Marangon	Carmelo Russello	Correzioni lessicali e grammaticali, integrazione link
0.7.0	2025/05/27	Aleena Mathew	Pietro Crotti	Aggiunte metriche di qualità
0.6.0	2025/05/13	Alessandro Bernardello	Mirco Borella	Stesura sezioni da 4.2 (Infrastructure) a 4.4 (Training)
0.5.0	2025/05/08	Mirco Borella	Alessandro Bernardello	Norme analisi dei requisiti e continuazione stesura documento
0.4.0	2025/05/04	Mirco Borella	Carmelo Russello	Aggiunta sezione 3.1.9 (Manutenzione documenti)
0.3.0	2025/04/30	Mirco Borella	Aleena Mathew	Stesura sezioni da 2.3 (Sviluppo) a 3.2 (Gestione della configurazione)
0.2.0	2025/04/26	Marco Egidi	Matteo Marangon	Stesura sezione 2.1 (Fornitura) e

Versione	Data	Autori	Verificatori	Descrizione Modifiche
				2.2 Attività di fornitura
0.1.0	2025/04/22	Mirco Borella	Alessandro Bernardello	Stesura iniziale documento e Introduzione

Indice

1. Introduzione

1.1. Scopo del documento

Questo documento nasce per la necessità di definire e documentare il ***Way of Working***$_{GL}$ che verrà utilizzato dal gruppo di lavoro Sigma18 durante lo svolgimento di tutte le fasi del capitolato.

Ai fini di realizzazione del *«Way of Working»*, i membri del gruppo *Sigma18* prenderanno come riferimento lo standard ***ISO/IEC 12207:1995***$_{GL}$ che definisce i processi di sviluppo del *software* e le relative attività e compiti. In particolare, il gruppo *Sigma18* si concentrerà sui seguenti processi:

- **Primari**
- **Di Supporto**
- **Organizzativi**

Questo documento verrà trattato con un approccio incrementale, ovvero verrà aggiornato e modificato in base all'evoluzione del progetto e delle esigenze del gruppo di lavoro. Pertanto, il documento non è da considerarsi definitivo ma un documento vivo che si evolve con il progetto.

I membri del gruppo di lavoro sono tenuti a rispettare le norme e le modalità di lavoro definite in questo documento. Il gruppo *Sigma18* s'impegna a prendere conoscenza delle eventuali modifiche e/o integrazioni di questo documento.

1.2. Scopo del capitolato

Il capitolato **C3**, proposto dall'azienda **Var Group S.p.A.**, richiede lo sviluppo di un'applicazione che permetta di automatizzare le routine digitali tramite l'intelligenza generativa in cloud.

L'obiettivo è di creare un client per ambiente Windows e/o Mac in grado di generare queste ***routine***$_{GL}$ in modo semplice attraverso il linguaggio naturale. Il client deve essere in grado di interfacciarsi con le varie applicazioni installate sul sistema e con i servizi esposti dalle applicazioni stesse (***API***$_{GL}$). L'utente deve essere in grado di modificare la logica funzionale di un *workflow* tramite un'interfaccia ***drag & drop***$_{GL}$. È prevista la possibilità di salvare i *workflow* generati, in modo da poterli riutilizzare in un secondo momento, e di gestire più automazioni contemporaneamente.

Come dichiarato nella lettera di presentazione, il gruppo si impegna a realizzare il prodotto descritto dal capitolato C3 con un costo stimato di **€12945,00** e con termine previsto di consegna **29 Agosto 2025**.

1.3. Utilizzo del glossario

Al fine di ottenere una lettura chiara e concisa della documentazione prodotta, è stato previsto un glossario dei termini utilizzati nei prodotti del capitolato che possano causare ambiguità o dubbi. Per evidenziare che una parola è presente all'interno del glossario, è stata utilizzata una nomenclatura così definita:

parola$_{GL}$

dove **parola** è il termine che si intende definire.

Il glossario verrà aggiornato ogni qualvolta verrà introdotto un nuovo termine o che verrà modificata la definizione di uno dei termini già esistenti. All'interno dei documenti, verrà utilizzata la

nomenclatura ***parola***$_{GL}$ solamente alla prima occorrenza del termine. Tutti i seguenti richiami al termine non verranno segnalati con la nomenclatura del glossario.

1.4. Riferimenti

1.4.1. Riferimenti normativi

- **ISO/IEC 12207:1995 - Standard internazionale per i processi di ciclo di vita del *software***
  https://www.math.unipd.it/~tullio/IS-1/2009/Approfondimenti/ISO_12207-1995.pdf
- **ISO/IEC 12207**
  https://it.wikipedia.org/wiki/ISO/IEC_12207
- **Capitolato C3 - Automatizzare le routine digitali tramite l'intelligenza generativa**
  https://www.math.unipd.it/~tullio/IS-1/2024/Progetto/C3.pdf
- **Regolamento del progetto didattico**
  https://www.math.unipd.it/~tullio/IS-1/2024/Dispense/PD1.pdf

1.4.2. Riferimenti informativi

- **Glossario**
  https://sigma18unipd.github.io/documentiCompilati/2-RTB/documentidiprogetto/glossario.pdf
- **Processi di ciclo di vita**
  https://www.math.unipd.it/~tullio/IS-1/2024/Dispense/T02.pdf

2. Processi primari

Lo standard ISO/IEC 12207:1995 è nato per fornire un modello di riferimento per la definizione dei processi di ciclo di vita del *software* e per la gestione della qualità del *software*.

Lo standard definisce i processi primari come i processi che sono direttamente coinvolti nella realizzazione del prodotto *software*, tra cui: **acquisizione**, **fornitura**, **sviluppo**, **gestione** e **manutenzione**.

Nello specifico, ai fini del capitolato, tratteremo i seguenti processi primari:

- **Fornitura**
- **Sviluppo**

2.1. Fornitura

Lo standard ISO/IEC 12207:1995 definisce il processo di fornitura come l'insieme delle attività necessarie per garantire che il prodotto *software* soddisfi i requisiti specificati e le aspettative del proponente.

Questo processo è iniziato con uno studio di fattibilità e lo sviluppo di alcuni flussi applicativi insieme al proponente, raccolti nella sessione di ***design thinking***$_{GL}$ di cui è stato redatto il relativo verbale esterno. L'incontro è stato fondamentale per la definizione e la contrattazione dei requisiti del progetto.

Il processo terminerà con la consegna del prodotto al proponente e la successiva verifica della sua conformità ai requisiti specificati.

2.1.1. Strumenti a supporto

La fornitura viene supportata da strumenti di comunicazione, di gestione dei requisiti e di tracciamento delle modifiche. Il gruppo *Sigma18* ha definito di utilizzare i seguenti strumenti:

- **Whatsapp**: per la comunicazione asincrona interna al gruppo;
- **Discord**: per svolgere le riunioni di gruppo e per la comunicazione sincrona tra i vari membri;
- **GitHub**: per la gestione del ***Backlog***$_{GL}$, e per il ***sistema di ticketing***$_{GL}$, utili per avere una panoramica del lavoro svolto e da svolgere;
- **Microsoft Excel**: per la gestione e la tracciabilità delle ore di lavoro svolte da ciascun membro del gruppo.

Gli strumenti scelti per la comunicazione con il proponente sono:

- **Gmail**: per la comunicazione formale asincrona con il proponente, successivamente rimpiazzata da:
- **Slack**: per la comunicazione con il proponente, l'invio di documenti e per il supporto allo sviluppo del prodotto.

2.2. Attività di fornitura

La fornitura è composta da un insieme di attività successivamente descritte:

- **Iniziazione**: In questa fase iniziale, il fornitore analizza le richieste avanzate dal proponente, considerando attentamente eventuali vincoli. È il momento cruciale in cui viene valutata la fattibilità tecnica ed economica del progetto, determinando quali requisiti potrebbero necessitare di ulteriore negoziazione.
- **Preparazione della risposta**: Sulla base dell'analisi effettuata, il fornitore elabora una controproposta per il proponente incorporando le valutazioni emerse durante la fase di iniziazione e presentando modifiche ai requisiti originali quando necessario.
- **Contratto**: Questa fase prevede un incontro con il proponente durante il quale vengono discusse le risposte precedentemente elaborate. L'obiettivo è raggiungere un accordo che soddisfi entrambe le parti.
- **Pianificazione**: Definiti i requisiti contrattuali, il fornitore procede a stabilire l'organizzazione e la metodologia di lavoro più adatte. In questa fase viene selezionato il modello di ciclo di vita del *software*, vengono identificate le risorse necessarie e le tecnologie da impiegare, e viene posta particolare attenzione all'analisi dei potenziali rischi.
- **Esecuzione e controllo**: Una volta documentata la pianificazione, il fornitore avvia l'implementazione di quanto stabilito. Durante questa fase, viene costantemente monitorata sia la qualità del prodotto in sviluppo sia l'avanzamento generale del progetto rispetto alla pianificazione iniziale.
- **Revisione e valutazione**: Parallelamente allo sviluppo, il fornitore mantiene un dialogo costante con il proponente per condividere lo stato di avanzamento e raccogliere feedback. Questo processo di revisione continua permette di valutare la conformità del lavoro svolto rispetto alle aspettative e di apportare eventuali correzioni in corso d'opera.
- **Consegna e completamento**: Al termine del progetto, il fornitore consegna il prodotto finale al proponente, garantendo il supporto post-consegna e risolvendo eventuali problematiche che potrebbero emergere nelle fasi iniziali di utilizzo. (Supporto post-consegna non previsto dal capitolato).

2.2.1. Elenco della documentazione fornita

Segue un elenco dei documenti che il gruppo *Sigma18* fornirà all'azienda proponente *Var Group S.p.A.* e ai committenti del capitolato *Prof. Riccardo Cardin* e *Prof. Tullio Vardanega* (in ordine alfabetico).

2.2.1.1. Analisi dei requisiti

Documento che ha lo scopo di fornire una descrizione dettagliata dei casi d'uso e dei requisiti del progetto che l'azienda proponente *Var Group S.p.A.* ha presentato. Questo documento è pertanto di elevata importanza in quanto costituisce una solida base per lo sviluppo del prodotto finale, permettendo di avere una chiara visione delle funzionalità e dei requisiti richiesti dall'azienda proponente.

Tipo di documento: **Documento Esterno**
Destinatari documento: ***Sigma18, Var Group S.p.A, Prof. Tullio Vardanega, Prof. Riccardo Cardin***

2.2.1.2. Glossario

Documento che contiene la definizione dei termini ritenuti ambigui o non chiari utilizzati nel capitolato e nei documenti del gruppo *Sigma18*.

Tipo di documento: **Documento Interno**
Destinatari documento: ***Sigma18, Prof. Tullio Vardanega, Prof. Riccardo Cardin***

2.2.1.3. Lettera di presentazione candidatura

Lettera con la quale il gruppo *Sigma18* comunica formalmente la propria intenzione a candidarsi per la realizzazione del capitolato.

Tipo di documento: **Documento Esterno**
Destinatari documento: ***Prof. Tullio Vardanega, Prof. Riccardo Cardin***

2.2.1.4. Norme di progetto

Documento che stabilisce le modalità di lavoro, il ***Way of Working***$_{GL}$ del gruppo *Sigma18* e le norme da seguire durante lo svolgimento del progetto. Questo documento è in continuo aggiornamento e verrà modificato in base all'evoluzione del progetto e delle esigenze del gruppo di lavoro.

Tipo di documento: **Documento Interno**
Destinatari documento: ***Sigma18***

2.2.1.5. Piano di progetto

Documento che ha l'obiettivo di definire e tenere traccia di tutte le attività eseguite e pianificate dal team durante lo sviluppo del progetto. In particolare, il piano di progetto si concentra su: analisi dei rischi, modello di sviluppo e pianificazione delle attività.

Tipo di documento: **Documento Esterno**
Destinatari documento: ***Sigma18, Var Group S.p.A, Prof. Tullio Vardanega, Prof. Riccardo Cardin***

2.2.1.6. Piano di qualifica

Documento che ha lo scopo di definire le modalità di verifica e di validazione adottate dal gruppo *Sigma18* al fine di garantire la qualità del prodotto finale. Il documento sarà soggetto ad aggiornamenti durante il ciclo di vita del progetto per riflettere le modifiche apportate per raggiungere una migliore efficacia ed efficienza.

Tipo di documento: **Documento Esterno**
Destinatari documento: ***Sigma18, Var Group S.p.A, Prof. Tullio Vardanega, Prof. Riccardo Cardin***

2.2.1.7. Preventivo dei costi e dichiarazione impegni

Documento che ha lo scopo di definire i ruoli, i costi preventivati e fornire una stima della data per la realizzazione del capitolato. Il documento contiene anche un'analisi dei rischi approssimativa che il gruppo *Sigma18* ha tenuto in considerazione durante la stesura del preventivo.

Tipo di documento: **Documento Esterno**
Destinatari documento: ***Prof. Tullio Vardanega, Prof. Riccardo Cardin***

2.2.1.8. Valutazione dei capitolati

Documento che presenta le motivazioni che hanno portato il gruppo *Sigma18* a scegliere il capitolato *C3 - Automatizzare le routine digitali tramite l'intelligenza generativa* proposto dall'azienda *Var Group S.p.A.* e i relativi pro e contro di ogni capitolato proposto.

Tipo di documento: **Documento Esterno**
Destinatari documento: ***Prof. Tullio Vardanega, Prof. Riccardo Cardin***

2.2.1.9. Verbale Esterno

Documento che contiene i resoconti, le decisioni e le azioni intraprese all'ordine del giorno durante gli incontri con il proponente. Il verbale esterno è redatto dal gruppo *Sigma18* e viene condiviso con il proponente e i committenti del capitolato.

Tipo di documento: **Documento Esterno**
Destinatari documento: ***Var Group S.p.A, Prof. Tullio Vardanega, Prof. Riccardo Cardin***

2.2.1.10. Verbale Interno

Documento che contiene i resoconti, le decisioni e le azioni intraprese all'ordine del giorno durante gli incontri e le riunioni tra i componenti del gruppo *Sigma18*. Il verbale interno è redatto dal gruppo e viene condiviso con il gruppo stesso.

Tipo di documento: **Documento Interno**
Destinatari documento: ***Sigma18***

2.3. Sviluppo

Per lo standard ISO/IEC 12207:1995, il processo di sviluppo è l'insieme delle attività necessarie per realizzare il prodotto *software*, a partire dalla definizione dei requisiti, passando per l'analisi dei requisiti, fino alla consegna del prodotto finale.

2.3.1. Strumenti a supporto dello sviluppo

Tutte le attività racchiuse nel processo di sviluppo sono supportate dai seguenti strumenti:

- **GitHub**: per la gestione del versionamento e per avere una panoramica del lavoro svolto e da svolgere;
- **Visual Studio Code**: per la codifica del *software* e per la scrittura della documentazione;
- **StarUML** e **Draw.io**: per la creazione dei ***diagrammi UML***$_{GL}$

2.4. Attività di sviluppo

In base allo standard ISO/IEC 12207:1995, il processo di sviluppo è composto da un insieme di attività successivamente descritte:

2.4.1. Implementazione del processo

Il processo di sviluppo inizia con la definizione del modello di ciclo di vita del *software* da adottare. Il gruppo *Sigma18* ha deciso di adottare il modello ***Agile***$_{GL}$ in quanto permette una maggiore flessibilità e adattabilità alle esigenze del progetto. Il modello Agile prevede un approccio iterativo

e incrementale, in cui il prodotto viene sviluppato in piccole parti (o *sprint*) e viene costantemente testato e migliorato.

2.4.2. Analisi dei requisiti

L'analisi dei requisiti è l'attività che ha lo scopo di definire e documentare i requisiti richiesti dal proponente del prodotto *software*. Questa attività è fondamentale per garantire che il prodotto finale soddisfi le aspettative e i vincoli del proponente e degli utenti finali.

2.4.3. Progettazione architetturale

La progettazione architetturale è l'attività che ha lo scopo di individuare e definire l'architettura del prodotto *hardware* e *software*. L'architettura deve essere in grado di soddisfare tutte le richieste e i requisiti del sistema come da specifica del proponente.

2.4.4. Analisi dei requisiti *software*

Per ogni componente *software*, vengono stabiliti e documentati i requisiti in modo dettagliato. I requisiti rappresentano il modo in cui il *software* soddisfa le esigenze del cliente finale. Si valutano la coerenza con i requisiti proposti dal proponente, la completezza delle caratteristiche di sicurezza, qualità e usabilità dell'interfaccia grafica.

2.4.5. Progettazione architetturale *software*

I requisiti *software* vengono uniti in un'architettura *software* che definisce le interazioni tra i vari componenti e le loro responsabilità condivise.

2.4.6. Progettazione del dettaglio *software*

Si utilizza un approccio ***top-down***$_{GL}$ per definire le funzionalità di ogni singolo componente *software*, a partire dai componenti più generali fino ad arrivare ai dettagli delle singole unità.

2.4.7. Codifica del *software*

La codifica del *software* è l'attività che ha lo scopo di realizzare le singole unità *software*. Questa attività viene svolta in modo incrementale e iterativo, in modo da garantire che il prodotto finale soddisfi i requisiti richiesti dal proponente.

2.4.8. Testing del *software*

Ciascuna unità *software* codificata viene testata per garantirne il corretto funzionamento e la conformità ai requisiti richiesti. Il testing viene svolto in modo incrementale e iterativo in parallelo all'attività di codifica.

2.4.9. Integrazione del *software*

L'integrazione del *software* è l'attività che ha lo scopo di unire le singole unità *software*, *hardware*, alcune operazioni manuali e tutte le altre componenti necessarie in un unico prodotto finale.

2.4.10. Test di qualifica del *software*

L'intero sistema viene testato per verificarne la conformità ai requisiti richiesti dal proponente e verificarne il funzionamento complessivo.

2.4.11. Installazione del *software*

L'installazione del *software* è l'attività che ha lo scopo di fornire il prodotto finale nell'ambiente di produzione concordato del proponente.

2.4.12. Supporto per l'approvazione del *software*

Il gruppo *Sigma18* fornisce supporto al proponente per la verifica di aver effettivamente soddisfatto i requisiti richiesti e l'approvazione del prodotto finale.

3. Processi di supporto

I processi di supporto definiti dalla norma ISO/IEC 12207:1995 indirizzano le seguenti attività principali e di dettaglio:

- **Documentazione**;
- **Gestione della configurazione**;
- **Qualità**;
- **Verifica**;
- **Validazione**;
- **Revisione congiunta**;
- **Audit**;
- **Risoluzione dei problemi**;
- **Usabilità**;
- **Valutazione del prodotto**;

Ai fini della natura del capitolato, il gruppo *Sigma18* ha deciso di trattare i processi di supporto relativi alla **Documentazione**, **Gestione della configurazione**, **Qualità**, **Verifica** e **Validazione**.

3.1. Documentazione

Il processo di «Gestione della documentazione» è una delle parti fondamentali di tutti i processi primari: garantisce lo sviluppo, la manutenzione e la registrazione delle informazioni prodotte relativamente al prodotto software.

L'obiettivo principale di questo processo è quello di garantire che la documentazione prodotta sia completa, accurata e facilmente accessibile in maniera asincrona a tutti i membri del gruppo di lavoro. Tutto il gruppo di lavoro è tenuto a rispettare le norme e le modalità di lavoro per la stesura dei documenti definite in questa sezione.

3.1.1. Strumenti a supporto della documentazione

Il gruppo *Sigma18* ha deciso di utilizzare i seguenti strumenti per la gestione della documentazione:

- **GitHub**: per la gestione dei sorgenti della documentazione, versionamento dei documenti e sistema di ticketing con ***pull request***$_{GL}$
- **Typst**: per la scrittura della documentazione in modo semplice e veloce. Il gruppo *Sigma18* ha deciso di utilizzare questo linguaggio di markup simile a ***LaTeX***$_{GL}$ per la scrittura della documentazione in quanto permette di generare documenti di alta qualità e di gestire modelli e *template*. Inoltre utilizza costrutti simili a quelli che si trovano nei linguaggi di programmazione e permette al gruppo di generare documenti compilando i sorgenti in modo centralizzato ed automatizzato.
- **Microsoft Excel**: per la gestione e la tracciabilità delle ore di lavoro svolte da ciascun membro del gruppo.

3.1.2. Attività previste

Il processo di documentazione è composto da un insieme di attività successivamente descritte:

- **Produzione**: attività che ha lo scopo di definire le modalità con la quale redigere i documenti. Ulteriori informazioni nella sezione 3.1.8.
- **Manutenzione**: attività che ha lo scopo di definire le modalità con la quale aggiornare o modificare documenti già esistenti. Ulteriori informazioni nella sezione 3.1.9.

3.1.3. Verbali

La redazione di un verbale esterno o interno sfrutta il relativo template ai fini di produrre documenti standardizzati e di alta qualità.

Ogni modifica al template viene discussa e approvata dal gruppo prima di essere applicata e viene testata sui documenti retroattivamente per garantire la compatibilità con i documenti già prodotti.

Un verbale contiene una pagina di copertina così composta:

- **Logo del gruppo *Sigma18* e indirizzo *email* di contatto**;
- **Titolo del documento**;
- **Lista dei responsabili**;
- **Lista dei redattori**;
- **Lista dei verificatori**;
- **Versione del documento**;
- **Tipologia del documento**;
- **Destinatari del documento**;
- **Abstract dei contenuti**.

Successivamente, il verbale contiene una tabella con le versioni e la descrizione delle modifiche del documento. La definizione delle modalità di versionamento dei documenti è riportata nella sezione 3.1.8.2.

Segue poi l'indice del documento, che riporta i titoli delle sezioni e dei paragrafi del documento. L'indice viene generato automaticamente.

Un verbale contiene sempre le sezioni successivamente descritte. La prima riguarda i «Riferimenti generali», che contengono le modalità di svolgimento della riunione (virtuale se svolta in videoconferenza o in presenza se svolta in loco), la data e la durata della riunione. Segue poi la lista dei partecipanti presenti.

La seconda sezione riguarda i punti all'«ordine del giorno», che contiene i punti discussi durante la riunione e le decisioni prese.

La sezione delle «Conclusioni e decisioni prese» contiene le decisioni prese durante la riunione e le azioni da intraprendere. Le attività definite «pendenti» e programmate sono disponibili nella successiva sezione «Attività programmate». Questa sezione contiene una tabella con due colonne: la prima colonna contiene l'identificativo dell'attività (successivamente definito) e la seconda colonna contiene il titolo.

Con identificativo dell'attività si intende un numero a 5 cifre che identifica in modo univoco l'attività. Il numero dell'attività è assegnando seguendo la numerazione interna delle ***issue***$_{GL}$ e delle ***pull request***$_{GL}$ di *GitHub*. Pertanto, ad un'attività con numerazione *N* in un verbale corrisponderà sempre la relativa *issue* o *pull request* su *GitHub*.

I verbali definiti come «esterni» includeranno una sezione per l'aggiunta della firma di verifica da parte dell'azienda proponente.

Ogni pagina del verbale, ad eccezione della copertina, contiene un'intestazione comprendente il logo del gruppo *Sigma18* e il titolo del documento e un piè di pagina con il numero progressivo della pagina.

3.1.4. Glossario

La redazione del glossario sfrutta il relativo template.

Il glossario contiene una pagina di copertina così composta:

- **Logo del gruppo *Sigma18* e indirizzo *email* di contatto**;
- **Titolo del documento**;
- **Lista dei responsabili**;
- **Lista dei redattori**;
- **Lista dei verificatori**;
- **Versione del documento**;
- **Tipologia del documento**;
- **Destinatari del documento**.

Successivamente, il glossario contiene una tabella con le versioni e la descrizione delle modifiche del documento. La definizione delle modalità di versionamento dei documenti è riportata nella sezione 3.1.8.2.

Segue poi l'indice del documento, che riporta i titoli delle sezioni e dei paragrafi del documento. L'indice viene generato automaticamente.

La prima sezione «Introduzione» contiene una breve introduzione al documento e la sua finalità, insieme alla nomenclatura utilizzata per indicare che un termine è definito nel glossario.

Segue poi la lista ordinata in ordine alfabetico dei termini e delle loro definizioni.

Ogni pagina del glossario, ad eccezione della copertina, contiene un'intestazione comprendente il logo del gruppo *Sigma18* e il titolo del documento e un piè di pagina con il numero progressivo della pagina.

Per evidenziare che una parola è presente all'interno del glossario, è stata utilizzata una nomenclatura così definita:

parola$_{GL}$

dove **parola** è il termine che si intende definire. La nomenclatura è stata scelta in modo da non interferire con le altre nomenclature utilizzate nel capitolato.

All'interno dei documenti, verrà utilizzata la nomenclatura ***parola***$_{GL}$ solamente alla prima occorrenza del termine. Tutti i seguenti richiami al termine non verranno segnalati con la nomenclatura del glossario.

3.1.5. Diari di bordo

La redazione dei documenti per il diario di bordo sfrutta il relativo template.

Ogni diario di bordo contiene una pagina di copertina così composta:

- **Logo del gruppo *Sigma18* e indirizzo *email* di contatto**;
- **Titolo del documento**;
- **Lista dei componenti del gruppo**;

Seguono poi generalmente le singole sezioni di riguardo, una per ogni pagina:

- «**Progressi raggiunti e confronto (retrospettiva)**»: una lista puntata con i progressi raggiunti e le attività completate;
- «**Obiettivi e attività programmate (*backlog*)**»: una lista puntata con gli obiettivi e le attività programmate per il periodo successivo;
- «**Analisi delle criticità e soluzioni adottate**»: una lista puntata con le criticità emerse e le soluzioni adottate per affrontarle;

Ogni pagina del glossario, ad eccezione della copertina, contiene un'intestazione comprendente il logo del gruppo *Sigma18* e il titolo del documento e un piè di pagina con il numero progressivo della pagina.

3.1.6. Lettere di presentazione

La redazione delle lettere di presentazione sfrutta il relativo template.

Ogni lettera di presentazione contiene una pagina di copertina così composta:

- **Logo del gruppo *Sigma18* e indirizzo *email* di contatto**;
- **Titolo del documento**;
- **Mittenti**;
- **Destinatari**;

Segue poi il corpo della lettera.

Ogni pagina di una lettera, ad eccezione della copertina, contiene un'intestazione comprendente il logo del gruppo *Sigma18* e il titolo del documento e un piè di pagina con il numero progressivo della pagina. In caso di riferimenti a risorse esterne, il piè di pagina conterrà anche la lista con i *link* per esteso.

3.1.7. Altri documenti

Tutti gli altri documenti redatti dal gruppo *Sigma18* sfruttano il relativo template.

Ogni documento non già descritto contiene una pagina di copertina così composta:

- **Logo del gruppo *Sigma18* e indirizzo *email* di contatto**;
- **Titolo del documento**;
- **Lista dei responsabili**;
- **Lista dei redattori**;
- **Lista dei verificatori**;
- **Versione del documento**;
- **Tipologia del documento**;
- **Destinatari del documento**.
- **Abstract dei contenuti**.

Successivamente, ogni documento contiene una tabella con le versioni e la descrizione delle modifiche. La definizione delle modalità di versionamento dei documenti è riportata nella sezione 3.1.8.2.

Segue poi l'indice, che riporta i titoli delle sezioni e dei paragrafi del documento. L'indice viene generato automaticamente.

Si continua con il corpo del documento, che contiene le informazioni e i contenuti richiesti.

Ogni pagina di un documento, ad eccezione della copertina, contiene un'intestazione comprendente il logo del gruppo *Sigma18* e il titolo del documento e un piè di pagina con il numero progressivo della pagina.

Alcuni documenti, come l'analisi dei requisiti, contengono sezioni o componenti che necessitano di essere normati al fine di garantire la qualità e la coerenza del documento stesso. Seguono quindi le sezioni che necessitano di essere normate, definite come:

TitoloDelDocumento: **Sezione**

3.1.7.1. Analisi dei Requisiti: Casi d'uso

Per la definizione dei casi d'uso è stata utilizzata la seguente nomenclatura:

UC[NUMERO]:TITOLO

dove con «**NUMERO**» s'intende il numero identificativo del caso d'uso, e con «**TITOLO**» s'intende un breve titolo che descrive il caso d'uso.

Ogni caso d'uso è composto da una serie di sezioni che ne descrivono il funzionamento e i scenari principali. Generalmente le sezioni sono le seguenti:

- **Attore principale**: descrive l'attore protagonista del caso d'uso.

- **Pre-condizioni**: rappresentano le condizioni che devono essere vere prima che il caso d'uso possa iniziare ad essere eseguito.
- **Post-condizioni**: specificano le proprietà che devono essere garantite al termine dell'esecuzione del caso d'uso.
- **Scenario principale**: descrive il percorso più comune e desiderato per un utente (attore) nel completare un obiettivo specifico interagendo con il sistema.

Nel caso in cui il caso d'uso preveda più scenari, questi vengono descritti in modo analogo allo scenario principale.

Nel caso in cui il caso d'uso preveda delle estensioni, queste verranno elencate in una sezione «**Estensioni**» indicando i relativi casi d'uso collegati.

3.1.7.2. Analisi dei Requisiti: Requisiti

Ciascun requisito è identificato da un codice univoco composto come segue:

[R[Rilevanza][Tipologia]-[ID]]

dove:

- **R** : indica che si tratta di un requisito.
- **Rilevanza** : indica la rilevanza del requisito, che può essere:
  - **O** : requisito obbligatorio, pertanto indispensabile per lo sviluppo del progetto;
  - **D** : requisito desiderabile, pertanto non necessario ma fornisce un valore aggiunto al progetto;
  - **F** : requisito facoltativo, pertanto non necessario ma può essere implementato in accordo con l'azienda se vi sono le condizioni appropriate.
- **Tipologia** : indica la tipologia del requisito, che può essere:
  - **F** : requisito funzionale, che descrive una funzionalità del sistema;
  - **Q** : requisito qualitativo, che descrive un aspetto di qualità del prodotto per soddisfare esigenze specifiche;
  - **V** : requisito di vincolo, che descrive un vincolo imposto dal proponente nel capitolato e non può essere trascurato.
- **ID** : numero progressivo del requisito, univoco all'interno della rispettiva categoria.

3.1.7.3. Piano di progetto: Rischi

Ogni rischio nel piano di progetto viene identificato con la seguente nomenclatura:

R + [Tipo di Rischio] + [Indice]

dove «**R**» indica il fatto che si tratti di un rischio. Il **Tipo di Rischio** si divide in due categorie:

- Rischi legati alla tecnologia utilizzata (**T**).
- Rischi legati all'organizzazione (**O**).

L'**Indice** è un numero incrementale che serve ad identificare univocamente i vari rischi per ogni tipo di Rischio.

3.1.8. Produzione dei documenti

La produzione dei documenti segue una serie di passi ben definiti e successivamente descritti:

Creazione della *issue* su *GitHub*: Come primo passo viene aperta una *issue* che conterrà come titolo una breve descrizione del documento che si intende produrre.

Assegnazione della *issue*: La *issue* viene assegnata al redattore del documento, che dovrà occuparsi della sua stesura. Sarà compito del redattore creare la relativa ***branch***$_{GL}$ sul quale andare a lavorare. La *branch* segue la seguente nomenclatura:

Numero-Titolo

Dove con «Numero» si intende il numero della *issue* e con «Titolo» si intende il titolo della *issue* stessa, tutto in minuscolo e con gli spazi sostituiti da «-».

Ad esempio la *issue* con numero 1234 e titolo «Analisi dei requisiti» avrà come branch «1234-analisi-dei-requisiti».

Creazione del documento: Il redattore crea il documento utilizzando il template fornito dal gruppo *Sigma18* e lo salva con la seguente nomenclatura descritta nella sezione 3.1.8.2. Il redattore segnerà il termine della stesura segnando come messaggio nel ***Commit***$_{GL}$ «closes N» dove con «N» si intende il numero della *issue* aperta con # come suffisso. In questo modo, il sistema di gestione delle *issue* di *GitHub* andrà a chiudere automaticamente la *issue* una volta che la *pull request* sarà stata accettata e il ***merge***$_{GL}$ effettuato.

Verifica del documento: Terminata la stesura, il redattore andrà ad aprire una *pull request* andando ad assegnare i verificatori.

I verificatori sono i membri del gruppo che si occuperanno di verificare il documento e di fornire feedback al redattore. In caso di esito negativo, i verificatori andranno a rifiutare la *pull request* lasciando un commento e il redattore dovrà apportare le modifiche richieste. In caso di esito positivo, i verificatori andranno ad approvare la *pull request* e il documento potrà essere considerato pronto per la pubblicazione.

Pubblicazione del documento: Una volta approvato il documento da parte dei verificatori, il responsabile andrà a fare il merge della *pull request* e il documento sarà considerato pubblicato.

I documenti in fase di stesura non sono considerati pubblici e pertanto saranno accessibili solamente all'interno delle loro *branch* di lavoro. I verbali che richiedono firme esterne, verranno pubblicati senza firma per permetterne la visione in fase di validità da parte dell'azienda proponente. A firma ottenuta, il documento verrà aggiornato e pubblicato direttamente nel sito web dedicato alla documentazione.

Tutti i documenti all'interno di una *pull request* scateneranno l'avvio di una *GitHub Action* che andrà a verificare la presenza effettiva dei termini contrassegnati come glossario nel glossario stesso. In caso di errore, la *pull request* non potrà essere accettata e il redattore dovrà provvedere a integrare il glossario con i termini mancanti.

Tutti i documenti pubblicati nella *branch* «main» scateneranno l'avvio di una *GitHub Action* che andrà ad effettuare la compilazione e la successiva pubblicazione dei sorgenti nel sito web dedicato alla documentazione.

3.1.8.1. Denominazione dei documenti

Come deciso nel verbale interno in data 2025/03/07 i documenti seguiranno una nomenclatura ben definita, in modo tale da poter far effettuare delle azioni alle *GitHub Action* basandosi sul nome del file. Tutti i sorgenti della documentazione termineranno con l'estensione *Typst «.typ»*.

3.1.8.1.1. Verbali

I verbali seguiranno una nomenclatura così definita:

TIPO_DATA_VERSIONE

dove «TIPO» è una sigla e varia per «vi» in caso di un verbale interno e «ve» per un verbale esterno. «DATA» rappresenta la data del verbale in formato *AAAAMMGG*.
«VERSIONE» rappresenta l'ultima versione del file con la nomenclatura descritta nella sezione 3.1.8.2.

3.1.8.1.2. Diari di bordo

I diari di bordo seguiranno una nomenclatura così definita:

ddb_DATA

dove «DATA» rappresenta la data del verbale in formato *AAAAMMGG*.

3.1.8.1.3. Glossario

Il glossario sarà definito come «glossario»

3.1.8.1.4. Altri documenti

Gli altri documenti seguiranno una nomenclatura così definita:

TITOLOBREVE_VERSIONE

dove «TITOLOBREVE» è il titolo abbreviato del documento in minuscolo e senza spazi. «VERSIONE» rappresenta l'ultima versione del file con la nomenclatura descritta nella sezione 3.1.8.2.

3.1.8.2. Versionamento dei documenti

Il gruppo ha optato per l'utilizzo di un sistema di versionamento così definito:

X.Y.Z

dove:

- **X**: Modifiche sostanziali e ristrutturazioni importanti
- **Y**: Aggiunta di nuovi paragrafi e correzioni concettuali
- **Z**: Correzioni sintattiche e grammaticali, modifiche minori e cambiamenti al template

3.1.8.3. Riferimenti e formati

3.1.8.3.1. Date

Ogni data nei documenti verrà scritta utilizzando il formato «**AAAA/MM/GG**» derivante dallo standard *ISO 8601* dove **AAAA** indica l'anno in numero a 4 cifre, **MM** indica il mese in numero a 2 cifre e **GG** indica il giorno in numero a 2 cifre, come stabilito nel verbale interno del 2025/03/07.

Nel caso di date scritte per intero, si utilizzerà il formato **GG MESEINLETTERE AAAA**, dove **GG** indica il giorno in numero a 2 cifre, **MESEINLETTERE** indica il mese in lettere (ad esempio «Aprile») e **AAAA** indica l'anno in numero a 4 cifre.

Esempio di data: **2025/05/01** oppure **01 Maggio 2025**.

3.1.9. Manutenzione dei documenti

Con manutenzione dei documenti s'intende l'attività di modifica di un documento già pubblicato in precedenza. Questa attività differisce di poco e riprende i passaggi già descritti nella sezione 3.1.8 con al posto della creazione di un nuovo documento, la modifica di un sorgente già esistente. I restanti passi rimangono invariati.

3.2. Gestione della configurazione

Per lo standard ISO/IEC 12207:1995, la gestione della configurazione è l'insieme delle attività che hanno lo scopo di identificare e controllare le modifiche apportate al prodotto *software* e alla documentazione associata.

3.2.1. Strumenti a supporto della gestione della configurazione

Il gruppo *Sigma18* ha deciso di utilizzare i seguenti strumenti per la gestione della configurazione:

- **GitHub**: per la gestione del versionamento, del ticketing e per avere una panoramica del lavoro svolto e da svolgere;
- **GitHub Actions**: per l'automazione di alcune attività di gestione della configurazione, come la compilazione automatica della documentazione.

3.2.2. Attività previste

Il processo di gestione della configurazione è composto da un insieme di attività così descritte:

- **Identificazione della configurazione**: attività che ha lo scopo di identificare e definire gli elementi di configurazione del prodotto *software* e della documentazione associata.
- **Controllo della configurazione**: attività che ha lo scopo di controllare le modifiche apportate agli elementi di configurazione e garantire la tracciabilità delle modifiche.
- **Registrazione della configurazione**: attività che ha lo scopo di registrare e documentare le modifiche apportate.
- **Valutazione della configurazione**: attività che ha lo scopo di valutare la conformità degli elementi di configurazione agli standard e alle specifiche richieste dal proponente.

3.2.3. Identificazione della configurazione

Data la natura del capitolato, il gruppo *Sigma18* ha deciso di identificare la documentazione prodotta in 4 categorie, definite da:

N-NOMEFASE

dove la coppia **N-NOMEFASE** rappresenta una delle 4 fasi del capitolato:

- *0-diaridibordo*
- *1-candidatura*
- *2-RTB*
- *3-PB*

Anche se non parte integrante per il proponente, è presente una cartella «**0-diaridibordo**» che contiene le presentazioni per i diari di bordo prodotti dal gruppo.

Ogni fase viene rappresentata da una cartella nella ***root directory***$_{GL}$ del repository della documentazione. All'interno di ogni cartella, sono presenti le sottocartelle i relativi documenti prodotti dal gruppo.

Esempio di struttura del repository della documentazione come definita nel verbale interno del 2025/03/07:

```
docs
├── assets
│   └── img
│       ├── LogoSlim.svg
│       └── Logo.svg
├── readme.md
├── templates
│   └── template.typ
├── 0-diaridibordo
│   └── ...
├── 1-candidatura
│   ├── ...
│   └── verbali
│       ├── interni
│       │   └── vi_20250305_1.0.0.typ
│       └── ...
├── 2-RTB
│   └── ...
└── ...
```

3.2.4. Controllo della configurazione

Il controllo della configurazione è l'attività che ha lo scopo di controllare le modifiche apportate agli elementi di configurazione e garantire la tracciabilità delle modifiche. Il gruppo *Sigma18* ha deciso di utilizzare il sistema di gestione delle *branch*, delle *issue* e delle *pull request* di *GitHub* per

garantire la tracciabilità delle modifiche apportate alla documentazione e al codice sorgente del prodotto *software*.

Come descritto nella sezione 3.1.8, ogni modifica o aggiunta di file inizia con la creazione di una *issue* dedicata.

Ogni issue contiene:

- **Titolo**: che descrive brevemente il contenuto della *issue*;
- **Descrizione**: che descrive il contenuto e le motivazioni della *issue*;
- **Assegnatari**: che sono i membri del gruppo che si occuperanno della *issue*;
- **Labels (dall'inglese: etichette)**: che permettono di categorizzare le *issue* in base al tipo di attività da svolgere;
- **Tipo**: *non utilizzato*;
- **Progetto**: *non utilizzato*;
- ***Milestone***: che permette di raggruppare le *issue* in base alla fase del capitolato a cui appartengono;

Inoltre, per ogni *issue* o *pull request* creata, *GitHub* crea automaticamente un numero identificativo univoco (**ID**) all'interno del repository.

Successivamente alla creazione di una *issue*, l'assegnatario dovrà creare la relativa *branch* di sviluppo. La nomenclatura delle *branch* è definita nella sezione 3.1.8.

Ogni modifica apportata ai file presenti nella *branch* dovrà essere documentata tramite un $\boldsymbol{commit}_{GL}$ che descrive brevemente la modifica apportata. In caso di risoluzione di una *issue*, il messaggio di *commit* dovrà contenere la dicitura «closes #N» dove con «N» si intende il numero della *issue* aperta. In questo modo, il sistema di gestione delle *issue* di *GitHub* andrà a chiudere automaticamente la *issue* una volta che la *pull request* sarà stata accettata e il *merge* effettuato.

Al termine del lavoro sulla *branch*, l'assegnatario dovrà creare una *pull request* per richiedere l'integrazione delle modifiche nella *branch* principale «main». La *pull request* dovrà contenere:

- **Titolo**: che descrive brevemente il contenuto della *pull request*;
- **Descrizione**: che descrive il contenuto e le motivazioni della *pull request*;
- **Verificatori** : che sono i membri del gruppo che si occuperanno della verifica della *pull request*;
- **Assegnatari**: che sono i membri del gruppo che hanno lavorato sulla *branch*;
- **Etichette**: che permettono di categorizzare le *pull request* in base al tipo di attività svolta;

I verificatori designati dovranno esaminare le modifiche apportate e fornire feedback. In caso di esito negativo, i verificatori andranno a rifiutare la *pull request* lasciando un commento specifico sulle problematiche riscontrate. L'assegnatario dovrà quindi apportare le modifiche richieste. In caso di esito positivo, i verificatori approveranno la *pull request* e il contenuto sarà pronto per l'integrazione.

Grazie al sistema di *GitHub Actions*, alla creazione di una *pull request* vengono attivati automaticamente dei controlli sulla validità della documentazione, tra cui la verifica della presenza dei termini del glossario.

Il gruppo *Sigma18* ha anche configurato un sistema di protezione del *branch «main»* che impedisce il *merge* di una *pull request* se non sono stati soddisfatti i requisiti di verifica e approvazione, e il *commit* diretto di modifiche di file senza passare per una *pull request.*

3.2.5. Registrazione della configurazione

Per soddisfare la necessità di registrare e tracciare i cambiamenti apportati alla documentazione, il gruppo *Sigma18* ha deciso di utilizzare un sistema di nomenclatura dei file e di versionamento dei documenti definito nella sezione 3.1.8.2 (Versionamento dei documenti).

3.2.6. Valutazione della configurazione

La valutazione della configurazione è l'attività che ha lo scopo di valutare la conformità degli elementi di configurazione agli standard e alle specifiche richieste dal proponente.

Il gruppo *Sigma18* s'impegna a raggiungere in modo efficace ed efficiente gli obiettivi di qualità definiti nel capitolato e a garantire la conformità del prodotto *software* agli standard e alle specifiche richieste dal proponente.

3.3. Accertamento della qualità

Il processo di accertamento della qualità è l'insieme delle attività che hanno lo scopo di garantire la qualità del prodotto *software* e della documentazione associata seguendo gli obiettivi fissati dal proponente.

3.3.1. Strumenti a supporto della qualità

Il gruppo *Sigma18* ha deciso di utilizzare delle metriche oggettive per la valutazione della qualità del prodotto *software* e della documentazione associata. Le metriche utilizzate sono descritte in dettaglio nel piano di qualifica.

3.3.2. Attività previste

Secondo lo standard ISO/IEC 12207:1995, il processo di accertamento della qualità è composto da un insieme di attività così descritte:

- **Implementazione del processo**: attività che ha lo scopo di implementare il processo di raccoglimento dei dati da valutare e l'individuazione degli standard di qualità insieme al proponente;
- **Accertamento della qualità del prodotto**: attività che ha lo scopo di valutare il prodotto *software* e la documentazione associata;
- **Accertamento della qualità del processo**: attività che ha lo scopo di valutare i processo del gruppo *Sigma18* e la loro conformità agli standard e alle specifiche richieste dal proponente.

3.4. Verifica

Il processo di verifica è l'insieme delle attività che hanno lo scopo di garantire la conformità e la correttezza del prodotto *software* e della documentazione associata agli standard e alle specifiche richieste dal proponente.

In altre parole risponde alla domanda: «Costruiamo il prodotto correttamente?».

A partire dai primi verbali prodotti in fase di candidatura, il gruppo *Sigma18* si è avvalso del meccanismo di verifica delle *pull request* di *GitHub* dove attraverso il controllo dei verificatori, viene garantita la correttezza grammaticale, sintattica e del contenuto della documentazione prodotta.

In caso di modifiche corpose ai documenti, vengono assegnati verificatori aggiuntivi di supporto.

Tutte le attività di verifica riguardanti il prodotto *software* sono registrate nel piano di qualifica e riguarderanno principalmente lo sviluppo nella terza fase del capitolato (*3-PB, Product Baseline*).

In generale, il processo di verifica è composto da un insieme di due attività successivamente descritte.

3.4.1. Analisi statica

L'analisi statica è l'attività che ha lo scopo di analizzare il prodotto *software* senza eseguirne il codice sorgente. Questa attività viene svolta in modo automatico tramite strumenti di analisi del codice. In altre parole l'analisi statica si concentra nella verifica della correttezza sintattica e semantica del codice sorgente.

Per realizzare ciò, l'analisi statica si può effettuare in due tipologie:

- **Metodi formali**: metodi matematici che permettono di dimostrare la correttezza del codice sorgente. Questi metodi sono molto complessi e richiedono una conoscenza approfondita della teoria dei linguaggi.
- **Metodi di lettura**, a loro volta suddivisi in:
  - ***Walkthrough***: metodi di lettura del codice sorgente che prevede l'esistenza di un problema, e attraverso un analisi rigorosa di tutto il prodotto si va alla ricerca di errori. Non sapendo a priori di che cosa e dove si tratta l'errore, il *walkthrough* diventa un metodo molto costoso, poco efficace e non applicabile al nostro caso d'uso.
  - ***Inspection***: metodi di lettura del codice sorgente basata in una *checklist* (lista di controllo) dove, in base alla conoscenza pregressa degli errori più probabili, va ad analizzarli in modo lineare. In caso di errori non presenti e previsti nella *checklist*, essa verrà aggiornata e il codice sorgente verrà analizzato nuovamente. Questo metodo è molto efficace e soprattutto automatizzabile basandosi nella lista di controllo.

Il gruppo *Sigma18* ha deciso di utilizzare il metodo di analisi statica basato su *inspection* in quanto stimiamo di riuscire con buona probabilità a creare una lista di controllo efficace e completa per il nostro prodotto.

3.4.2. Analisi dinamica

L'analisi dinamica è l'attività che ha lo scopo di analizzare il prodotto *software* eseguendo il codice sorgente e analizzando il comportamento in esecuzione. Lo scopo dell'analisi dinamica è quello di

verificare il comportamento del prodotto *software* in fase di esecuzione e di individuare eventuali errori o anomalie di funzionamento non previste.

Per fare ciò, l'analisi dinamica fa uso dei ***Test***$_{GL}$ che devono essere ripetibili, in quanto devono essere eseguiti più volte (ad esempio una volta per riscontrare un errore, e una seconda volta per verificare che l'errore sia stato corretto) e automatizzabili, in quanto verranno eseguiti in modo automatico per garantire la ripetibilità e la correttezza dei risultati. Per riuscire a testare il prodotto nelle sue parti, vengono utilizzati degli elementi a supporto definiti come:

- **Stub**: oggetti che sostituiscono il comportamento di una componente reale per permettere il testing dell'oggetto chiamante;
- **Driver**: oggetti che richiamano parti di codice non direttamente eseguibili;
- **Logger**: oggetti che registrano gli esiti dei test eseguiti del prodotto *software* per permettere l'analisi dei risultati e la loro valutazione.

Tutti i test previsti dal gruppo *Sigma18* sono sviluppati per essere eseguiti in modo indipendente dall'ambiente di esecuzione e in modo deterministico, ovvero che il risultato del test non deve dipendere da fattori esterni al prodotto *software* (in altre parole, a partire da uno stesso input, il test deve sempre restituire lo stesso output).

Le tipologie di test previste dal gruppo *Sigma18* sono:

- **Test di unità**: test che verificano il corretto funzionamento di una singola unità del prodotto *software* (ad esempio una singola funzione);
- **Test di integrazione**: test che verificano il corretto funzionamento di più unità del prodotto *software* e l'integrazione tra loro (ad esempio due oggetti che comunicano tra loro);
- **Test di sistema**: test che verificano il funzionamento completo dell'applicazione in un ambiente che simula quello reale di utilizzo;
- **Test di regressione**: test che verificano il corretto funzionamento del prodotto *software* dopo una modifica al codice sorgente.
- **Test di accettazione**: test che verificano il corretto funzionamento del prodotto *software* in base alle specifiche richieste dal proponente.

3.5. Validazione

Secondo lo standard ISO/IEC 12207:1995, il processo di validazione è il processo che conferma che il prodotto soddisfi i requisiti specificati del sistema, attraverso dimostrazioni oggettive e misurabili. Valuta concretamente la qualità raggiunta del prodotto.

In altre parole, la validazione ci fa rispondere alle domande «Il prodotto è quello che ci aspettavamo?», «Abbiamo realizzato il sistema giusto?».

Il gruppo *Sigma18* applica questo processo in parallelo al processo di verifica.

Tutte le esigenze del proponente «Var Group S.p.A.» sono state raccolte e analizzate in modo da definire i requisiti del prodotto *software* e della documentazione associata nel documento analisi di requisiti.

Nel processo di validazione, vengono in aiuto anche le attività di tracciamento dei requisiti sopra descritte, che permetteranno di validare se il prodotto funziona nel modo in cui il proponente si aspetta e se è conforme a quanto stabilito.

Un altro strumento a supporto sono i test di accettazione che rappresentano l'ultimo punto di controllo del sistema prima del rilascio del prodotto. Questi test vengono eseguiti in un ambiente che simula quello reale di utilizzo e verificano il corretto funzionamento del prodotto *software* assieme al proponente.

4. Processi organizzativi

I processi organizzativi sono l'insieme delle attività volte ad abilitare, controllare e supportare il ciclo di vita del sistema e dei progetti collegati.

La normativa ISO/IEC 12207:1995 indirizza i seguenti sotto processi:

- **Management**: il processo garantisce lo sviluppo e la manutenzione delle informazioni prodotte e registrate relativamente al prodotto software.
- **Infrastructure**: Il processo ha lo scopo di definire e mantenere l'integrità di tutti i componenti della configurazione (Configuration Items) e di renderli accessibili a chi ne ha diritto.
- **Improvement**: Il processo ha lo scopo di stabilire, valutare, misurare, controllare e migliorare il ciclo di vita del software.
- **Training**: Il processo ha lo scopo di fornire all'organizzazione risorse umane adeguate e di mantenere le loro competenze consistenti con le necessità del business.

4.1. Management (Gestione dei processi)

Secondo la normativa ISO/IEC 12207:1995, il processo di gestione dei processi ha lo scopo di individuare i processi e attribuirli ai relativi ruoli nella gestione del ciclo di vita del software. L'obiettivo è quello di garantire che i processi siano definiti, implementati e mantenuti in modo da soddisfare le esigenze del proponente e garantire la massima efficienza de gruppo di lavoro.

I risultati del processo dell'attività di gestione dei processi si concretizzano nel piano di progetto.

4.1.1. Strumenti a supporto della gestione dei processi

Il gruppo *Sigma18* ha deciso di utilizzare i seguenti strumenti per la gestione dei processi:

- **GitHub**: per la gestione delle attività in corso, l'assegnazione di attività ai singoli membri del gruppo di lavoro e pianificazione delle ***milestone***$_{GL}$
- **Whatsapp**: per la comunicazione tra i membri del gruppo di lavoro e l'organizzazione delle riunioni;
- **Discord**: per le riunioni tra i membri del gruppo di lavoro;
- **Slack**: per la comunicazione con l'azienda proponente.;

4.1.2. Attività previste

Il processo di gestione dei processi è composto da un insieme di attività così descritte:

- **Definizione dei processi**: attività che ha lo scopo di definire i processi e le attività da svolgere;
- **Pianificazione dei processi**: attività che ha lo scopo di pianificare le attività da svolgere e le risorse necessarie;
- **Esecuzione dei processi**: attività che ha lo scopo di eseguire le attività pianificate;
- **Controllo e monitoraggio dei processi**: attività che ha lo scopo di controllare l'andamento delle attività e garantire il rispetto delle scadenze e delle risorse assegnate;

- **Chiusura dei processi**: attività che ha lo scopo di chiudere le attività e verificare che quello che è stato prodotto sia coerente con l'obiettivo iniziale.

4.1.3. Ruoli dei membri

Il gruppo *Sigma18* ha deciso di utilizzare dei ruoli per la gestione dei processi descritti nel preventivo dei costi e della dichiarazione degli impegni.

4.1.3.1. Amministratore

L'amministratore si occupa della gestione degli strumenti IT necessari all'avanzamento dello sviluppo. La figura, che conosce e comprende approfonditamente il *way of working*, offre supporto agli altri ruoli. Si ritiene che la figura abbia maggior importanza nelle fasi iniziali dello sviluppo, con alcuni picchi nei momenti di rilascio del prodotto.

4.1.3.2. Analista

L'analista è la figura che svolge l'attività di analisi dei requisiti del *software*, i quali potranno essere funzionalità in accordo con il proponente o specifiche minime di funzionamento.

La figura produrrà il documento di analisi dei requisiti.

4.1.3.3. Progettista

Il progettista è responsabile della definizione dell'architettura del sistema e della progettazione delle componenti *software*. Deve essere in grado di comunicare efficacemente con gli altri membri del team, tra cui programmatori, analisti e verificatori, per assicurarsi che la visione progettuale sia compresa e implementata correttamente e che i requisiti siano tradotti in una soluzione tecnica adatta.

4.1.3.4. Programmatore

È la figura incaricata a trasformare le direttive di *design* fornite dal progettista in codice. Il ruolo richiede che la persona abbia familiarità e padronanza delle tecnologie che andranno ad essere implementate.

4.1.3.5. Responsabile

Il responsabile è la figura che si occupa di coordinare e supervisionare l'intero progetto, assicurandosi che tutte le attività siano svolte nei tempi e nei modi previsti. Il ruolo si occupa anche del dialogo tra l'interno e l'esterno del gruppo garantendo una comunicazione chiara e tempestiva.

4.1.3.6. Verificatore

È la figura che garantisce che la documentazione prodotta ed il *software* sviluppato mantengano alti standard di qualità. Il verificatore esegue test approfonditi per identificare *bug* e problemi, assicurandosi che il prodotto finale sia conforme ai requisiti specificati.

Ai fini di rispettare la richiesta di condividere il *bug reporting* avvenuto durante la fase di sviluppo, il verificatore dovrà gestire la piattaforma ITS (*Issue Tracking System*) utilizzata (*GitHub*).

4.1.4. Comunicazioni

Il gruppo ha scelto di utilizzare le piattaforme citate nella sezione 4.1.1 per le comunicazioni interne. Non sono stati previsti incontri di allineamento con l'azienda proponente. In particolare, il gruppo

ha deciso di utilizzare *WhatsApp* per la comunicazione rapida asincrona e *Discord* per le riunioni interne.

L'azienda proponente è disponibile, in caso di necessità, per chiarimenti o consigli tramite la piattaforma di messaggistica asincrona *Slack*.

Per quanto riguarda invece le comunicazioni non sopra citate, l'indirizzo email ufficiale del gruppo *Sigma18* è:

sigma18unipd@gmail.com

4.1.5. Coordinamento delle attività

Il gruppo riesce, in modo efficace, a coordinare le attività e a riportare in modo efficiente gli esiti e l'avanzamento del lavoro svolto nella riunione dedicata alla retrospettiva degli *sprint* e di pianificazione delle attività future svolta ogni due settimane.

Durante questa riunione, il piano di progetto viene aggiornato per riflettere l'avanzamento delle attività, le modifiche ai processi e le nuove esigenze emerse durante lo sviluppo. Ogni cambiamento significativo viene riportato, garantendo così una documentazione sempre allineata allo stato attuale del lavoro del gruppo *Sigma18*.

Ciò nonostante, in caso di necessità, il gruppo è disponibile a organizzare riunioni straordinarie per discutere di problematiche o chiarimenti.

4.2. Infrastructure (Infrastruttura)

Il processo di infrastruttura è l'insieme delle attività che hanno lo scopo di garantire la disponibilità e l'integrità degli elementi necessari per lo sviluppo del prodotto *software* e della documentazione associata.

4.2.1. Attività previste

Il processo di infrastruttura è composto da un insieme di attività così descritte:

- **Definizione**
- **Implementazione**
- **Manutenzione**

4.2.2. Definizione

Il gruppo *Sigma18* ha scelto durante lo svolgimento del progetto di utilizzare strumenti che consentano ai suoi vari membri di lavorare in modalità efficace ed efficiente.

Di seguito sono elencati tutti gli strumenti individuati dal gruppo:

- **Git**: Utilizzato per il versionamento del codice sorgente e della documentazione associata;
- **GitHub**: Utilizzato per la sincronizzazione degli sviluppi e come *Issue Tracking System* (*ITS*) per tenere traccia del *backlog* tramite l'utilizzo delle *issue*;
- **Typst**: Utilizzato per la scrittura della documentazione e per la generazione dei documenti in formato PDF;

- **Whatsapp**: Utilizzato per la comunicazione tra i membri del gruppo di lavoro e l'organizzazione delle riunioni;
- **Discord**: Utilizzato per le riunioni tra i membri del gruppo di lavoro;
- **Slack**: Utilizzato per la comunicazione con l'azienda proponente;
- **Microsoft Teams**: Utilizzato per la comunicazione con l'azienda proponente;
- **Google Calendar**: Utilizzato per la pianificazione delle riunioni con l'azienda proponente;
- **Script in Python e Bash**: Utilizzati per l'aggiornamento automatico del sito web, le automazioni di verifica della versione nel nome di un file e di presenza dei termini del glossario.

4.2.3. Implementazione

4.2.3.1. Git

Il gruppo *Sigma18* ha deciso di utilizzare *Git* come sistema di versionamento del codice sorgente e della documentazione. Il sistema di versionamento è stato scelto in quanto permette di tenere traccia delle modifiche, garantendo la possibilità di tornare indietro nel tempo in caso di necessità.

Uno dei maggiori vantaggi di *Git* sono le sue funzionalità di creazione di *branch*. A differenza dei sistemi di controllo delle versioni centralizzati, i branch *Git* sono facili da sottoporre a *merge* e questo agevola il flusso di lavoro parallelo scelto dal gruppo.

4.2.3.2. GitHub

Il gruppo *Sigma18* ha deciso di utilizzare *GitHub* come piattaforma di hosting per il codice sorgente e la documentazione. *GitHub* è stato scelto in quanto offre funzionalità avanzate per la gestione delle *issue*, delle *pull request* e delle *branch* (già descritte nella sezione 3.1.8).

4.2.3.3. Typst

Già descritto nella sezione 3.1 - Documentazione.

4.2.3.4. Whatsapp

Il gruppo *Sigma18* ha deciso di utilizzare *WhatsApp* come piattaforma di messaggistica per la comunicazione tra i membri del gruppo di lavoro. *WhatsApp* è stato scelto per la sua semplicità d'uso e ampia diffusione tra i membri del gruppo.

4.2.3.5. Discord

Il gruppo *Sigma18* ha deciso di utilizzare *Discord* come piattaforma di videoconferenza per le riunioni tra i membri del gruppo di lavoro. *Discord* è stato scelto per la sua semplicità d'uso e ampia diffusione tra i membri del gruppo.

Sono stati creati dei canali dedicati per permettere il lavoro simultaneo di più membri del gruppo in parallelo, senza che le comunicazioni si sovrappongano, e dei canali testuali dedicati allo scambio di *link* e riferimenti o credenziali.

4.2.3.6. Slack

L'azienda proponente ha deciso di utilizzare *Slack* come piattaforma di messaggistica per la comunicazione tra il gruppo *Sigma18* e il proponente. *Slack* è stato scelto per la sua ampia diffusione tra i membri dell'azienda proponente, in modo da poter fornire supporto in caso di necessità al gruppo di lavoro da varie figure aziendali.

4.2.3.7. Microsoft Teams

L'azienda proponente ha organizzato un *workshop* di presentazione delle tecnologie di *Gen AI* in modalità virtuale. Per fare ciò, l'azienda ha scelto di utilizzare *Microsoft Teams* come piattaforma di videoconferenza tra il gruppo *Sigma18* e la figura esperta associata.

Microsoft Teams è stato utilizzato anche nei workshop precedenti per consentire ai membri del gruppo di seguire le presentazioni in modalità virtuale, qualora non avessero la possibilità di partecipare in presenza.

4.2.3.8. Google Calendar

Il gruppo *Sigma18* ha deciso di utilizzare *Google Calendar* come piattaforma di pianificazione delle riunioni con l'azienda proponente.

4.2.3.9. Script in Python e Bash

Il gruppo *Sigma18* ha deciso di sviluppare degli script in *Python* e *Bash* per automatizzare alcune attività di gestione della configurazione, come la compilazione automatica della documentazione e l'aggiornamento del sito web. In particolare, sono stati creati degli script per:

- Aggiornare automaticamente il sito web con la documentazione prodotta;
- Verificare automaticamente la presenza dei termini del glossario nei documenti prodotti;
- Verificare automaticamente la versione del documento in base al nome del file;

4.2.4. Manutenzione

Il gruppo *Sigma18* ha deciso di assegnare alla figura dell'amministratore la responsabilità della manutenzione degli strumenti utilizzati e dell'infrastruttura.

4.3. Improvement (Miglioramento)

Il processo di miglioramento secondo lo standard ISO/IEC 12207:1995 consiste nella consolidazione, controllo e miglioramento continuo dei processi utilizzati durante il ciclo di sviluppo del prodotto *software* e della documentazione associata.

Come specificato in precedenza nella sezione 2.4.1, il gruppo *Sigma18* ha deciso di utilizzare il modello di lavoro *Agile* che prevede un miglioramento continuo del prodotto e dei processi utilizzati.

4.3.1. Attività previste

Il processo di miglioramento è composto da un insieme di attività così descritte:

- **Implementazione del processo**
- **Individuazione criticità**

- **Azioni correttive**

4.3.2. Implementazione del processo

Un processo viene stabilito, documentandolo in questo stesso documento.

4.3.3. Individuazione criticità

Una volta stabilito un processo, è fondamentale controllarne l'andamento e l'efficacia. È quindi essenziale individuare delle misurazioni appropriate ed effettuare controlli periodici su di esse.

4.3.4. Azioni correttive

Una volta revisionate le misurazioni è necessario individuare i processi problematici e stabilire soluzioni che possano portare a miglioramenti. Sarà poi necessario aggiornare la documentazione per riflettere le modifiche apportate.

4.4. Training (Formazione)

Il processo di formazione secondo lo standard ISO/IEC 12207:1995 ha lo scopo di fornire e mantenere personale formato e competente per le attività di sviluppo del prodotto *software* e della documentazione associata.

Il gruppo *Sigma18* si impegna a garantire che i membri del gruppo siano formati e competenti per le attività di sviluppo del prodotto *software* e in particolar modo sulle tecnologie proposte per lo sviluppo.

In caso di difficoltà, i vari membri del gruppo sono disponibili a fornire supporto e formazione ai componenti che lo richiedano. Questo meccanismo di collaborazione reciproca rafforza il lavoro di squadra e la condivisione delle conoscenze, oltre che rappresentare un'opportunità di conoscenza e coesione tra le persone.

5. Standard per la qualità

Il gruppo ha deciso di definire le metriche e i criteri che determinano la qualità del software sviluppato adottando le linee guida dello standard ISO/IEC 9126. Questo standard definisce un modello di qualità del software in termini di sei caratteristiche generali e venticinque sotto-caratteristiche, successivamente descritte.

5.1. Funzionalità

La funzionalità misura la capacità di un prodotto software di fornire servizi e strumenti che soddisfano le esigenze esplicite ed implicite del proponente. Questa caratteristica è composta dalle seguenti sotto-caratteristiche:

- **Adeguatezza**: capacità di offrire funzioni appropriate per svolgere i compiti specifici previsti;
- **Accuratezza**: capacità di fornire risultati o effetti attesi in accordo con i requisiti;
- **Interoperabilità**: capacità di interagire con altri sistemi specificati;
- **Sicurezza**: capacità di proteggere le funzioni e i dati da accessi non autorizzati e potenziali minacce;
- **Conformità**: capacità di aderire agli standard e alle normative stabilite.

5.2. Affidabilità

L'affidabilità misura la capacità di un prodotto software di mantenere un determinato livello di prestazioni richieste in condizioni specifiche per un periodo di tempo definito.
In particolare è composta dalle seguenti sotto-caratteristiche:

- **Maturità**: capacità del prodotto di evitare errori, malfunzionamenti o arresti inaspettati;
- **Tolleranza agli errori**: capacità del prodotto di mantenere un livello prestabilito di prestazioni anche in presenza di errori;
- **Recuperabilità**: capacità del prodotto di ripristinare i livelli di prestazione determinati e di recuperare i dati a seguito di malfunzionamenti o guasti;
- **Aderenza**: capacità del prodotto di aderire a standard riguardanti l'affidabilità.

5.3. Efficienza

L'efficienza misura la capacità di un prodotto software di fornire prestazioni adeguate in relazione alle risorse utilizzate.
Nel dettaglio, è composta dalle seguenti sotto-caratteristiche:

- **Comportamento rispetto al tempo**: misura i tempi di risposta e di elaborazione richiesti per eseguire le funzioni richieste in condizioni specifiche;
- **Utilizzo delle risorse**: misura la quantità e la tipologia di risorse utilizzate per eseguire le funzioni richieste come memoria, CPU e spazio su disco;
- **Conformità**: capacità del prodotto di aderire a standard riguardanti l'efficienza.

5.4. Usabilità

L'usabilità misura la facilità con cui un prodotto software può essere compreso, appreso e utilizzato dagli utenti finali.
In particolare è composta dalle seguenti sotto-caratteristiche:

- **Comprensibilità**: misura l'impegno richiesto agli utenti per comprendere il funzionamento del prodotto e le sue applicabilità;
- **Apprendibilità**: misura l'impegno richiesto agli utenti per imparare ad usare il software;
- **Operabilità**: misura la capacità del prodotto ad essere utilizzato con semplicità dagli utenti per i propri scopi;
- **Attrattività**: capacità del prodotto di essere piacevole da utilizzare per l'utente;
- **Conformità**: capacità del prodotto di aderire a standard riguardanti l'usabilità.

5.5. Manutenibilità

La manutenibilità misura la facilità con cui un prodotto software può essere modificato, corretto e migliorato nel tempo.
È composta dalle seguenti sotto-caratteristiche:

- **Analizzabilità**: misura l'impegno richiesto per analizzare il prodotto per identificare carenze, cause di fallimento o per apportare miglioramenti;
- **Modificabilità**: misura l'impegno richiesto per apportare modifiche al prodotto, come correzioni di errori o modifiche di componenti;
- **Stabilità**: misura la capacità del prodotto di ridurre il rischio di comportamenti indesiderati a seguito di modifiche;
- **Testabilità**: misura la facilità con cui il prodotto può essere testato per validare le modifiche apportate.

5.6. Portabilità

La portabilità misura la capacità del software di essere trasferito e utilizzato in ambienti di esecuzione diversi senza problemi.
In particolare, è composta dalle seguenti sotto-caratteristiche:

- **Adattabilità**: misura la facilità con cui il software può essere adattato a nuovi ambienti di esecuzione;
- **Installabilità**: misura la facilità con cui il software può essere installato in un particolare ambiente;
- **Coesistenza**: misura la capacità del software di coesistere con altri prodotti software nello stesso ambiente, condividendo risorse come CPU e memoria;
- **Sostituibilità**: valuta la capacità con cui il software può svolgere gli stessi compiti di un altro software nello stesso ambiente.

6. Metriche per la qualità di processo

6.1. Nomenclatura delle metriche

Il gruppo *Sigma18* ha deciso di utilizzare la seguente sigla per identificare le metriche relative alla qualità di processo:

MPC (*Minimum Predictive Capability*)

Ogni metrica sarà identificata da un codice univoco composto dalla sigla *MPC* seguita da un acronimo che identifica la metrica specifica: **MPC-AcronimoMetrica**.

6.2. Processi primari

6.2.1. Fornitura

6.2.1.1. Earned Value

- **Codice**: MPC-EV;
- **Formula**: EV= Budget at Completion (BAC) * Percentuale di lavoro completato;
- **Descrizione**: misura il valore del lavoro completato in un progetto fino a quel momento rispetto al budget pianificato.

6.2.1.2. Planned Value

- **Codice**: MPC-PV;
- **Formula**: $\text{PV} = \text{Budget at Completion (BAC)}$ * Percentuale di lavoro pianificato;
- **Descrizione**: misura il valore del lavoro pianificato in un progetto in un determinato momento secondo il piano di progetto rispetto al budget pianificato.

6.2.1.3. Actual Cost

- **Codice**: MPC-AC;
- **Formula**: $\text{AC} = \text{Costo effettivo del lavoro svolto}$;
- **Descrizione**: misura il costo effettivo del lavoro svolto in un progetto fino a quel momento.

6.2.1.4. Estimated At Completion

- **Codice**: MPC-EAC;
- **Formula**: $\text{EAC} = \frac{\text{Budget at Completion (BAC)}}{\text{Cost Performance Index (CPI)}}$;
- **Descrizione**: stima del costo totale del progetto al suo completamento basata sui costi sostenuti fino a quel momento.

6.2.1.5. Estimated To Complete

- **Codice**: MPC-ETC;
- **Formula**: $\text{ETC} = \text{Budget at Completion (BAC)} - \text{Actual Cost (AC)}$;
- **Descrizione**: stima il costo del lavoro rimanente per completare il progetto.

6.2.1.6. Cost Variance

- **Codice**: MPC-CV;
- **Formula**: $\text{CV} = \text{Earned Value (EV)} - \text{Actual Cost (AC)}$;
- **Descrizione**: misura la differenza tra il budget disponibile e quello usato effettivamente. Rappresenta la differenza tra il valore del lavoro completato e il costo sostenuto.

6.2.1.7. Schedule Variance

- **Codice**: MPC-SV;
- **Formula**: $\text{SV} = \text{Earned Value (EV)} - \text{Planned Value (PV)}$;
- **Descrizione**: misura la differenza tra il valore del lavoro completato e il valore del lavoro pianificato. Identifica eventuali ritardi o anticipi rispetto al piano di progetto.

6.2.1.8. Cost Performance Index

- **Codice**: MPC-CPI;
- **Formula**: $\text{CPI} = \frac{\text{Earned Value (EV)}}{\text{Actual Cost (AC)}}$;
- **Descrizione**: misura l'efficienza del costo del lavoro svolto fino a quel momento. Un valore inferiore a 1 indica che si sta spendendo più del previsto.

6.2.2. Sviluppo

6.2.2.1. Requirements Stability Index

- **Codice**: MPC-RSI;
- **Formula**: $\text{RSI} = \left(1 - \frac{\text{CR} + \text{AR} + \text{DR}}{\text{OR}}\right) * 100$;

  dove:

  - **OR**: Numero di requisiti originali;
  - **CR**: Numero di requisiti cambiati;
  - **AR**: Numero di requisiti aggiunti;
  - **DR**: Numero di requisiti rimossi.
- **Descrizione**: misura la stabilità dei requisiti del progetto. Un valore alto indica che i requisiti sono stabili e non soggetti a modifiche frequenti.

6.2.2.2. Technical Debt Ratio

- **Codice**: MPC-TD;
- **Formula**: $\text{TD} = \left(\frac{\text{Tempo per risolvere problemi tecnici}}{\text{Tempo per sviluppare nuove funzionalità}}\right) * 100$;
- **Descrizione**: rapporto tra il tempo necessario per risolvere problemi tecnici e il tempo necessario per sviluppare nuove funzionalità. Un valore basso indica che il codice è ben strutturato.

6.3. Processi di supporto

6.3.1. Documentazione

6.3.1.1. Correttezza ortografica

- **Codice**: MPC-CO;
- **Descrizione**: indica la presenza di errori ortografici e grammaticali nel documento, in particolare è considerato accettabile un valore inferiore o pari a 2.

6.3.2. Verifica

6.3.2.1. Code Coverage

- **Codice**: MPC-CCO;
- **Formula**: $\mathrm{CCO} = \left(\frac{\text{Numero di linee di codice testate}}{\text{Numero totale di linee di codice}}\right) * 100$;
- **Descrizione**: misura la percentuale di codice sorgente coperto dai test. Un valore alto indica che il codice è ben testato e che le funzionalità sono verificate in modo adeguato.

6.3.2.2. Test superati in percentuale

- **Codice**: MPC-TSP;
- **Formula**: $\mathrm{TSP} = \left(\frac{\text{Numero di test superati}}{\text{Numero totale di test}}\right) * 100$;
- **Descrizione**: misura la percentuale di test superati rispetto al numero totale di test previsti.

6.3.3. Gestione della qualità

6.3.3.1. Satisfaction of Quality Metrics

- **Codice**: MPC-SQM;
- **Formula**: $\mathrm{SQM} = \left(\frac{\text{Numero di metriche soddisfatte}}{\text{Numero totale di metriche}}\right) * 100$;
- **Descrizione**: misura la percentuale di metriche che soddisfano gli obiettivi minimi di qualità stabiliti.

6.4. Processi organizzativi

6.4.1. Gestione dei processi

6.4.1.1. Efficienza temporale

- **Codice**: MPC-ET;
- **Formula**: $\mathrm{ET} = \left(\frac{\text{Ore produttive}}{\text{Ore totali}}\right) * 100$;
- **Descrizione**: misura la percentuale di tempo effettivamente dedicato alle attività produttive rispetto al tempo totale disponibile.

7. Metriche per la qualità di prodotto

7.1. Nomenclatura delle metriche

Il gruppo *Sigma18* ha deciso di utilizzare la seguente sigla per identificare le metriche relative alla qualità di prodotto:

MPD (*Minimum Product Delivery*)

Ogni metrica sarà identificata da un codice univoco composto dalla sigla *MPD* seguita da un acronimo che identifica la metrica specifica: **MPD-AcronimoMetrica**.

7.2. Funzionalità

7.2.1. Requisiti obbligatori soddisfatti

- **Codice**: MPD-RO;
- **Formula**: $\mathrm{RO} = \left(\frac{\text{Numero di requisiti obbligatori soddisfatti}}{\text{Numero totale di requisiti obbligatori}}\right) * 100$;
- **Descrizione**: misura la percentuale di requisiti obbligatori soddisfatti rispetto al numero totale di requisiti obbligatori definiti nel documento di analisi dei requisiti.

7.2.2. Requisiti desiderabili soddisfatti

- **Codice**: MPD-RD;
- **Formula**: $\mathrm{RD} = \left(\frac{\text{Numero di requisiti desiderabili soddisfatti}}{\text{Numero totale di requisiti desiderabili}}\right) * 100$;
- **Descrizione**: misura la percentuale di requisiti desiderabili soddisfatti rispetto al numero totale di requisiti desiderabili definiti nel documento di analisi dei requisiti.

7.2.3. Requisiti facoltativi soddisfatti

- **Codice**: MPD-RF;
- **Formula**: $\mathrm{RF} = \left(\frac{\text{Numero di requisiti facoltativi soddisfatti}}{\text{Numero totale di requisiti facoltativi}}\right) * 100$;
- **Descrizione**: misura la percentuale di requisiti facoltativi soddisfatti rispetto al numero totale di requisiti facoltativi definiti nel documento di analisi dei requisiti.

7.3. Affidabilità

7.3.1. Code Coverage

- **Codice**: MPD-CCO;
- **Formula**: $\mathrm{CCO} = \left(\frac{\text{Numero di righe di codice testate}}{\text{Numero totale di righe di codice}}\right) * 100$;
- **Descrizione**: misura la percentuale di codice sorgente coperto dai test.

7.3.2. Branch Coverage

- **Codice**: MPD-BC;
- **Formula**: $\mathrm{BC} = \left(\frac{\text{Numero di rami decisionali di codice testati}}{\text{Numero totale di rami decisionali di codice}}\right) * 100$;
- **Descrizione**: misura la percentuale di rami decisionali del codice coperti dai test.

7.3.3. Statement Coverage

- **Codice**: MPD-SC;
- **Formula**: $\mathrm{SC} = \left(\frac{\text{Numero di istruzioni di codice testate}}{\text{Numero totale di istruzioni di codice}}\right) * 100$;
- **Descrizione**: misura la percentuale di istruzioni del codice sorgente coperte dai test.

7.3.4. Passed Test Cases Percentage

- **Codice**: MPD-PTCP;
- **Formula**: $\mathrm{PTCP} = \left(\frac{\text{Numero di test superati}}{\text{Numero totale di test}}\right) * 100$;
- **Descrizione**: misura la percentuale di test superati rispetto al numero totale di test previsti.

7.3.5. Failure Tolerance

- **Codice**: MPD-FT;
- **Descrizione**: capacità del software di mantenere un livello di prestazioni accettabile anche in caso di guasti o malfunzionamenti.

7.3.6. Failure Frequency

- **Codice**: MPD-FF;
- **Descrizione**: indica la frequenza con cui si verificano guasti o malfunzionamenti nel prodotto.

7.4. Usabilità

7.4.1. Tempo di apprendimento

- **Codice**: MPD-TA;
- **Descrizione**: misura il tempo necessario per un utente per imparare ad utilizzare il software.

7.4.2. Error Rate

- **Codice**: MPD-ER;
- **Formula**: $\mathrm{ER} = \left(\frac{\text{Numero di errori commessi dagli utenti}}{\text{Numero totale di azioni degli utenti}}\right) * 100$;
- **Descrizione**: misura il numero di errori commessi dagli utenti durante l'utilizzo del software.

7.5. Efficienza

7.5.1. Utilizzo risorse

- **Codice**: MPD-UR;
- **Descrizione**: misura l'utilizzo delle risorse del sistema, come CPU, memoria e spazio su disco, durante l'esecuzione del software.

7.5.2. Tempi di risposta delle API

- **Codice**: MPD-TRA;
- **Descrizione**: misura il tempo medio di risposta delle API del sistema.

7.6. Manutenibilità

7.6.1. Complessità ciclomatica

- **Codice**: MPD-CC;
- **Formula**: $\mathrm{CC} = \mathrm{E} - \mathrm{N} + \mathrm{P}$

  dove:

  - **E**: Numero di archi del grafo di controllo;
  - **N**: Numero di nodi del grafo di controllo;
  - **P**: Numero di componenti connessi ad ogni arco.

7.6.2. Code Smell

- **Codice**: MPD-CS;
- **Descrizione**: indica la presenza di potenziali problemi di progettazione o codice che potrebbero richiedere manutenzione.

7.6.3. Coefficient of Coupling

- **Codice**: MPD-COC;
- **Formula**: $\mathrm{COC} = \frac{\text{Numero di dipendenze tra moduli}}{\text{Numero totale di moduli}}$;
- **Descrizione**: indica il grado di dipendenza tra i moduli o le componenti di un sistema.

7.6.4. Tempo per risolvere i bug

- **Codice**: MPD-TRB;
- **Descrizione**: misura il tempo medio necessario per risolvere i bug identificati nel software.