S.W.A.T.
ENGINEERING

Norme di Progetto

Contatti: swateng.team@gmail.com

Versione: 1.0

Registro delle Modifiche

Versione	Data	Descrizione	Autore	Ruolo
1.0	29-11-2023	Aggiunta sezione "Progettazione" "Codifica" "Gestione Qualità" "Procedure"	Riccardo Costantin	Amministratore
0.9	27-11-2023	Ampliata la sezione riguardante la documentazione	Riccardo Costantin	Amministratore
0.8	26-11-2023	Aggiunta sezione "Fornitura"	Riccardo Costantin	Amministratore
0.7	24-11-2023	Aggiunta sezione "Ruoli progetto"	Riccardo Costantin	Amministratore
0.6	24-11-2023	Aggiunta sezione "Analisi requisiti"	Riccardo Costantin	Amministratore
0.5	17-11-2023	Aggiunta specifica inserimento termini nel Glossario	Matteo Rango	Amministratore
0.4	14-11-2023	Aggiunta sezione "Processi Primari"; Chiarito workflow documentazione e codice	Matteo Rango	Amministratore
0.3	13-11-2023	Aggiunte sezioni "Oggetti Esterni al Repository" e "Tracciamento del Tempo Speso"; modificata organizzazione logica	Matteo Rango	Amministratore
0.2	08-11-2023	Aggiunta sezione "Processi di Supporto" e "Processi Organizzativi"	Matteo Rango	Amministratore
0.1	07-11-2023	Aggiunta sezione "Introduzione"	Matteo Rango	Amministratore

Indice

1 Introduzione

1.1 Glossario

Al fine di evitare possibili ambiguità relative al linguaggio utilizzato nei documenti, viene fornito il *Glossario*, nel quale sono presenti tutte le definizioni di termini aventi uno specifico significato che vuole essere disambiguato. Tali termini, sono scritti in *corsivo* e marcati con una $_G$ a pedice. Un'attività che comprende l'inserimento di un termine di glossario può considerarsi conclusa, nella sua interezza, solo nel momento in cui il termine viene scritto e spiegato nel *Glossario*.

1.2 Scopo del Documento

Il presente documento ha come scopo la definizione delle *best practices*$_G$ e del *way of working*$_G$ che ogni componente del *SWAT Engineering* ha l'obbligo di rispettare durante l'intero svolgimento del progetto. In questo modo si prova a garantire un metodo di lavoro omogeneo, verificabile e miglorabile nel tempo.

2 Processi Primari

2.1 Fornitura

2.1.1 Descrizione e scopo

Il processo di fornitura si propone di dettagliare le attività del fornitore per comprendere e soddisfare le richieste della Proponente. Dopo la completa comprensione delle esigenze, il fornitore, in collaborazione con la Proponente, stabilisce, tramite contratto, la data di consegna del prodotto.
Successivamente, viene redatto il Piano di Progetto per scaglionare le varie attività da svolgere, garantendo un chiaro processo di sviluppo del prodotto finale. L'obiettivo principale è soddisfare in modo chiaro le richieste della Proponente, evitando possibili incomprensioni, attraverso una collaborazione continua.

Il processo è caratterizzato da varie fasi:

- Determinare chiaramente i bisogni soddisfatti dal prodotto finale;
- Contrattazione:
  1. Ottenere feedback sulle tecnologie utilizzate;
  2. Chiarire dubbi o difficoltà riscontrate;
  3. Richiedere incontri di formazione sulle tecnologie consigliate per ottimizzare lo sviluppo.
- Stima dei costi;
- Continua verifica e controllo;
- Completamento e consegna.

2.1.2 Documenti

2.1.2.1 Piano di progetto

Documento redatto dal responsabile che offre una visione dettagliata dell'intero processo di sviluppo di un progetto, fornendo al team coinvolto una guida approfondita per mantenere l'allineamento con gli obiettivi, gestire le risorse in modo efficace e affrontare le sfide che potrebbero emergere durante lo sviluppo del progetto.

Si compone di:

- **Analisi dei rischi**: identifica, valuta e gestisce i potenziali rischi che potrebbero influenzare il successo di un progetto o di un'attività. È essenziale per anticipare e affrontare le sfide che potrebbero emergere durante lo sviluppo di un progetto, consentendo al team di adottare misure preventive o di pianificare strategie di mitigazione;

- **Modello di sviluppo**: approccio metodologico che viene scelto per guidare il processo di sviluppo del prodotto. Definisce la struttura di lavoro che sarà seguita durante l'intero ciclo di vita del progetto;
- **Pianificazione**: fornisce una roadmap dettagliata delle attività, delle risorse, e delle scadenze associate al progetto;
- **Preventivo**: stima dei costi per portare a termine il progetto.

2.1.2.2 Piano di Qualifica

Documento che descrive le strategie e gli approcci adottati per garantire la qualità del prodotto o del servizio che si sta sviluppando. Il suo scopo principale è quello di definire le modalità di verifica e validazione, nonché gli standard e le procedure di qualità che verranno seguite durante l'intero ciclo di vita del progetto.

È caratterizzata da:

- **Qualità del processo**: standard e procedure adottate per garantire la qualità durante lo sviluppo del progetto. Informazioni sulle attività di gestione della qualità, le metodologie utilizzate, e come vengono misurati e migliorati i processi stessi;
- **Qualità di prodotto**: standard, le specifiche e le caratteristiche che il prodotto deve soddisfare per essere considerato di alta qualità. Include anche le metriche e i criteri di valutazione utilizzati per misurare la qualità del prodotto;
- **Specifiche dei test**: specifiche dettagliate dei test che verranno condotti durante lo sviluppo del progetto;
- **Resoconto e valutazioni**: resoconto delle attività svolte e delle valutazioni effettuate durante il progetto. Utili per capire come il progetto si sta sviluppando rispetto agli obiettivi e alle aspettative, e per identificare eventuali azioni correttive necessarie.

2.2 Sviluppo

2.2.1 Codifica e Verifica

Come sistema di controllo di versione si utilizza *Git* all'interno del servizio online *Github*.

All'interno dei repository si utilizza il *Rebase workflow*$_G$ come metodo di gestione, con piccoli accorgimenti, segnalati nelle apposite sezioni.

In particolare si nota la differenza tra il repository di documentazione, chiamato *Docs* e quello di progetto (codice) chiamato *InnovaCity*: per entrambi si utlizza il *Rebase workflow*, con l'utilizzo dei *Feature branch* per

separare logicamente il lavoro da svolgere. Nel primo repository però, i *Feature branch* si derivano direttamente dal *main*, mentre nel secondo si derivano dal branch *dev*. Questo impone quindi che, prima di andare ad effettuare la chiusura di una *Pull request*, si vada ad effettuare un *rebase* del branch di derivazione, per rendere il nostro branch di sviluppo aggiornato rispetto alla base.

I *Feature branch* vengono aperti a partire dalle issue create nell' *Issue Tracking System* (vedi Sezione 3.2.1). Si procede poi ad associare una *Pull request*, a una o più issue collegate tra loro, per effettuare la verifica.

Nel caso del repository *InnovaCity* il branch *main* viene utilizzato per la pubblicazione di cambiamenti *major*, cioè quando si è implementato un numero di funzionalità significativo all'avanzamento del progetto. In quel caso è il responsabile che esegue l'approvazione finale.

2.2.2 Analisi dei Requisiti

L'analisi dei requisiti, condotta dagli Analisti, rappresenta un'attività preliminare nello sviluppo di un sistema software. Questa attività mira a definire le funzionalità che il nuovo prodotto deve offrire. L'obiettivo principale è quello di identificare, in modo chiaro e completo, i requisiti che il software sviluppato deve soddisfare, al fine di rispondere adeguatamente alle esigenze degli utenti e della Proponente.

L'analisi si compone di varie fasi:

- Scopo del prodotto: si devono soddisfare le esigenze della Proponente;
- Definizione degli attori: entità o persone che interagiscono con il *sistema*$_G$
- Definizione dei casi d'uso: rappresentazione di uno scenario specifico che descrive come un attore interagisce con il sistema;
- Definizione di requisiti obbligatori, opzionali, di vincolo;
- Confronti interni ed esterni.

La collaborazione con la Proponente è essenziale per comprendere in modo accurato e approfondire dettagliatamente il processo di realizzazione del prodotto.

2.2.2.1 Identificazione caso d'uso

I casi d'uso sono identificati nel seguente modo:

UC[Numero].[Numero sottocaso] [Titolo]

legenda:
- **Numero**: numero del caso d'uso;

- **Numero sottocaso**: numero del sottocaso del caso d'uso generale se presente;
- **Titolo**: breve titolo che descrive il contesto del caso d'uso in questione.

2.2.2.2 Struttura dei casi d'uso

I casi d'uso sono strutturati nel seguente modo:

- Attore;
- Precondizioni;
- Postcondizioni;
- Scenario Principale;
- Scenari Secondari (ove necessario);
- Estensioni, se presenti;
- Specializzazioni, se presenti.

2.2.2.3 Requisiti

I requisiti trovati vengono classificati nei seguenti modi:

- **Funzionali**

  un requisito funzionale specifica una funzionalità che il sistema deve essere in grado di svolgere;

- **Qualità**

  Un requisito di qualità stabilisce gli standard e i criteri che il sistema deve soddisfare per garantire prestazioni, affidabilità, sicurezza e altri aspetti relativi alla qualità;

- **Vincolo**

  Un requisito di vincolo è una restrizione o una condizione imposta al progetto.

2.2.2.4 Identificazione Requisiti

I requisiti trovati avranno un codice univoco con la seguente sintassi:

R[Importanza][Tipo][Numero]

legenda:

- **Importanza**:
  - O -> se requisito obbligatorio;
  - D -> se requisito desiderabile;
  - P -> se requisito opzionale.
- **Tipo**:
  - F -> se funzionale;

  - Q -> se qualità;
  - V -> se di vincolo.
- **Numero**: per ogni requisito aggiunto il numero viene incrementato.

2.2.3 Progettazione

2.2.3.1 Scopo

L'attività di progettazione verrà svolta dai Progettisti, i quali avranno il compito di definire le caratteristiche del prodotto finale, basandosi sui requisiti specificati nel documento *Analisi dei Requisiti*. La fase di progettazione segue l'analisi dei requisiti, nella quale sono definite le necessità e le aspettative per il prodotto. I Progettisti traducono queste informazioni in una struttura architetturale definita, organizzando il sistema in componenti specifici e definendo le interazioni tra di essi. In tal modo, la progettazione costituisce un passo essenziale nel percorso di sviluppo, contribuendo a trasformare i requisiti in un piano tangibile per la creazione del prodotto finale.

2.2.3.2 Descrizione

Si definiscono tre sottoattività:

1. ***Technology Baseline***: scelta e definizione delle tecnologie di base che saranno utilizzate per la realizzazione del sistema, comprende decisioni riguardanti linguaggi di programmazione, librerie e *framework*$_G$. Tale processo porterà alla creazione di un Proof of Concept(*PoC*$_G$);
2. ***Progettazione Architetturale***: definizione ad alto livello dell'architettura del prodotto, include test di integrazione;
3. ***Product Baseline***: segna un punto stabile nel processo di progettazione, in cui le specifiche tecniche, le funzionalità principali e l'architettura del prodotto sono definite in modo dettagliato e accettate dalle parti coinvolte. Include tutti gli elementi essenziali e i requisiti chiave del prodotto che devono essere soddisfatti, fornendo una base solida per lo sviluppo continuo del prodotto. Questo processo porterà infine alla realizzazione di un *Minimum Viable Product* (*MVP*$_G$); Include:
   - *Design Patterns*$_G$ e Package;
   - Definizione classi;
   - Diagrammi UML che includono:
     - classi;
     - package;
     - sequenze: utilizzato per descrivere uno scenario che costituisce una determinata sequenza di azioni in cui tutte le scelte sono state già effettuate;

    - attività: diagramma comportamentale che illustra il flusso delle attività attraverso un sistema.
- Test di Unità su ogni componente.

2.2.4 Codifica

2.2.4.1 Scopo

L'attività di codifica viene svolta dai Programmatori, che hanno come compito la traduzione delle decisioni progettuali in codice sorgente. I programmatori seguono le linea guida e le *best practices* stabilite durante la fase di progettazione architetturale.

2.2.4.2 Aspettative

Ci si aspetta che il codice sviluppato rispetti determinate caratteristiche:

- Conformità alle specifiche;
- Chiarezza e comprensibilità;
- Ottimizzazione delle prestazioni;
- Supplemento di test per verificare la correttezza e il funzionamento;
- Conformita' agli standard di qualita' caratteristici del prodotto.

2.2.4.3 Stili di codifica

[Verrano scritti in seguito.]

3 Processi di Supporto

3.1 Documentazione

3.1.1 Descrizione

La documentazione è l'insieme di informazioni rappresentate sotto forma di testo scritto che accompagna un prodotto software, svolgendo un ruolo essenziale nella descrizione del prodotto per coloro che lo sviluppano, lo distribuiscono e lo utilizzano. Il suo obiettivo primario è facilitare l'attività di sviluppo ai membri del team durante l'intero ciclo di vita del progetto e garantirne la coerenza,tracciando tutti i processi e le attività coinvolte per migliorare la qualità del risultato finale e semplificare la manutenzione. L'implementazione di regole chiare e di una struttura uniforme non solo migliora la fruibilità e la comprensione, ma favorisce anche la collaborazione all'interno del team, contribuendo in modo significativo al successo complessivo del progetto software.

3.1.2 Lista Documenti

I documenti che verranno prodotti sono:

- *Norme di Progetto*;
- *Piano di Progetto*;
- *Piano di Qualifica*;
- *Analisi dei Requisiti*;
- *Glossario*;
- *Verbali*:
  1. *Interni*;
  2. *Esterni*.

3.1.3 Template

Per la stesura dei documenti viene usato un template in formato *Typst*. Il template fornisce una struttura e un formato predefinito per semplificare la creazione di documenti. Serve a garantire coerenza, risparmiare tempo, standardizzare la presentazione e contribuire a una produzione di documenti più efficiente e professionale. Sono stati sviluppati quattro template distinti per adattarsi alle diverse esigenze di documentazione:

- documentazione ufficiale;
- lettere di presentazione;
- verbali per incontri interni ed esterni.

Ogni template è progettato per garantire coerenza e facilità d'uso, con piccole modifiche per rispecchiare le specificità di ciascun tipo di documento.

3.1.4 Nomenclatura

La consueta nomeclatura per i documenti si ottiene unendo, attraverso un underscore (_), il nome del file in *CamelCase* senza spazi (`NomeDelFile`) e la sua versione (`3.5`). Ad esempio `NormeDiProgetto_2.6.pdf`. Nel caso di documenti il cui nome contiene una data, essa si inserisce dopo il nome, ma prima della versione, sempre separandolo con gli underscores, nella forma `ggmmaa` senza separatori tra i singoli componenti della data: `gg` rappresenta il giorno, sempre scritto in due cifre, allo stesso modo `mm` rappresenta il mese, mentre l'anno è rappresentato da `aa`, corrispondente alle ultime due cifre dell'anno corrente.

3.1.5 Versionamento

Il versionamento avviene secondo il seguente formato **x.y**:

- **y** si incrementa una volta effettuata una modifica e la sua conseguente verifica;
- **x** si incrementa quando si effettua la modifica definitiva in vista di una verifica di avanzamento, questo comporta l'azzeramento di **y**.

Due modifiche, fatte in momenti diversi, differiscono l'una dall'altra solo se hanno scopi diversi. Ad esempio non è necessario incrementare la versione se viene fatta una modifica alla stessa sezione in due giorni differenti; anche se faccio una modifica, ed essa non viene approvata, non è necessario incrementare la versione con le nuove modifiche proposte dal/dai verificatore/i, dal momento che modifica e verifica "viaggiano" parallelamente.

3.1.6 Creazione e Modifica di un Documento

Ogni documento segue le fasi del seguente *workflow*$_G$:

1. Si crea un branch per lo sviluppo del documento nell'apposita repository *Docs* e si mette in uso.
2. Si copia dall'apposita repository `Templates` il template relativo al file che si deve redigere, e lo si inserisce nella cartella appropriata.
3. Si redige il documento o una sua sezione. Nel caso di documenti nuovi, in cui è necessario un elevato parallelismo di lavoro, è possibile usare *Google Drive* per la prima stesura e successivamente caricare il documento all'interno del branch.
4. Nel file `changelog.typ` si aggiunge una riga **in coda**, secondo il seguente formato: `<versione>,<data-modifica>,<descrizione-modifica>,<nome-autore>,<ruolo-autore>`; la versione segue le regole descritte nella Sezione 3.1.5.
5. Si esegue la commit sul branch creato.

6. Si apre una *pull request* dal branch appena creato verso il branch `main`: se il documento non è pronto per la verifica, ma ha bisogno di ulteriori modifiche, si apre la *pull request* in modalità `draft`, altrimenti in modalità normale, spostando la issue nell'apposita corsia *Ready to Review*.
7. Per ulteriori modifiche richieste dal/dai verificatore/i si ripetono i punti, **in ordine**, dal punto *3* al punto *5*.
8. Si elimina, **solo quando la pull request viene chiusa o risolta**, il branch creato.

La modifica di un documento avviene allo stesso modo, saltando il passo 2. Ogni cambiamento di stato è accompagnato dal conseguente movimento della issue, associata allo sviluppo, attraverso le diverse corsie dell'issue tracking system.

3.1.7 Struttura

3.1.7.1 Prima Pagina

- **logo team**: situato in alto a destra;
- **Titolo**:
  - Nome del documento, se diverso dai verbali;
  - per i verbali interni: Verbale Interno;
  - per i verbali esterni: Verbale Esterno;
- **Data**: solo se si tratta di verbali, vedere Sezione 3.1.7.2;
- **Contatti**: l'email del team;
- **Versione**: l'ultima versione del documento;
- **logo università**: in basso a destra.

3.1.7.2 Formato data

Si seguirà il formato GG-MM-AAAA.

3.1.7.3 Intestazione

Su ogni pagina del documento, eccetto la prima, si trova il titolo del documento seguito dalla sua versione e il logo del team.

3.1.7.4 Registro delle modifiche

Tabella con l'intestazione:

- **Versione**: versione del documento;
- **Data**: data della modifica apportata;
- **Descrizione**: cosa è stato modificato o aggiunto al file;
- **Autore**: l'autore della modifica;
- **Ruolo**: ruolo dell'autore al momento della modifica.

3.1.7.5 Indice
In una nuova pagina deve essere presente l'indice, utile per facilitare la ricerca e la navigazione all'interno del documento.

3.1.7.6 Verbali
I verbali differiscono leggermente da un documento ufficiale, in quanto non evolvibili nel tempo.

Si compongono principalmente di 2 sezioni:
- **Partecipanti**: si indica data di inizio e fine incontro e il luogo in cui si è svolto. A seguire, i nomi dei partecipanti del team e la durata della presenza di ciascuno vengono rappresentati in forma tabellare. Se il verbale è esterno si indicano anche i partecipanti della Proponente;
- **Sintesi dell'incontro**: Riassunto degli argomenti trattati durante la riunione.

Il verbale esterno oltre alle sezioni sopra elencate ha una pagina per la convalida, attraverso firma, del documento.

3.1.7.7 Norme tipografiche

3.1.7.7.1 Stili
- **Grassetto**: titoli delle sezioni o parole rilevanti;
- **Corsivo**: per enfatizzare le parole, per i termini inglesi e per specificare parole destinate ad essere aggiunte al glossario per definirne la definizione.

3.1.7.7.2 Elenchi puntati
Le voci di ogni elenco iniziano con lettera maiuscola e terminano con punto e virgola ';', eccetto l'ultima voce che termina con punto normale '.'.

3.1.8 Strumenti
Il gruppo utilizza:

- **Typst**: linguaggio per la stesura dei documenti, valida alternativa a Latex;
- **Google Docs**: per la stesura di bozze e appunti;
- **PlantUML**: linguaggio usato per la creazione dei diagrammi UML.

3.2 Gestione della Configurazione

3.2.1 Issue Tracking System
Come ITS si utilizza *Github* che, attraverso le funzioni di *Project*, *Issues* e *Pull requests*, garantisce una struttura all'organizzazione di progetto.

Le *Corsie di Stato*$_G$ descrivono lo stato attuale delle attività, all'interno del *Project* nell'*ITS* sono presenti:
Backlog attività individuate da svolgere.

Ready attività individuate per il completamento durante il prossimo $sprint_G$.
In Progress attività che sono in corso d'opera da parte dei redattori.
Ready to Review attività svolte che sono pronte per essere verificate.
In Review attività in corso di verifica da parte dei verificatori.
Done attività le cui modifiche sono state verificate e accettate.

3.2.2 Strumento di Condivisione

Per la condivisione veloce o la creazione di bozze si utilizza *Google Drive*. Uno dei suoi principali casi d'uso consiste nella collaborazione in tempo reale nella stesura di sezioni testuali ampie, da inserire successivamente nella documentazione (questo risulta particolarmente utile nel momento in cui il documento è alla sua prima stesura). Viene inoltre utilizzato come sistema per l'immagazzinamento di conoscenze acquisite durante lo svolgimento del progetto.

3.2.3 Tracciamento del Tempo Speso

Al fine di tracciare il tempo speso nel corso del progetto, nei diversi ruoli, si userà uno spreadsheet appositamente creato, disponibile all'interno di *Google Drive* dove, a fine giornata, ogni membro del team andrà ad inserire le proprie ore **produttive** svolte quel giorno, secondo la sua miglior stima del rapporto tra ore di orologio e ore produttive. Si inserisce *una sola* riga per ogni giornata e nella descrizione si andranno ad inserire dei brevi titoli rappresentativi delle attività svolte.

3.3 Verifica

3.3.1 Elementi Interni al Repository

La verifica del documento avviene tramite apposito metodo nell'ITS, attraverso la *Pull request*, indicando i punti in cui si richiede la modifica, il motivo della richiesta e una proposta se necessario. Anche in questo caso i verificatori si occupano di spostare la issue di riferimento nelle corsie appropriate, chiudendola se la verifica è terminata con successo, o spostandola nuovamente nella corsia *In Progress* in caso vengano richieste altre modifiche.

3.3.2 Elementi Esterni al Repository

Potrebbero esservi delle issue aperte all'interno dell'*ITS* che non hanno un corrispondente documento o prodotto in generale, all'interno del repository, ma che fungono come attività di gestione. Per queste, il ciclo di vita segue il normale flusso attraverso i diversi stati elencati nella Sezione 3.2.1. La verifica viene effettuata attraverso i commenti della issue stessa, che avranno la seguente forma:

- caso *richiesta cambiamenti*:

```
[REV]
- richiesta 1;
- richiesta 2;
```

- caso *approvazione*:

```
[REV] done
```

3.4 Gestione della Qualità

3.4.1 Descrizione

La gestione della qualità è un insieme di processi e attività volte a garantire che un prodotto soddisfi gli standard di qualità definiti, assicurando che il prodotto sviluppato sia affidabile ed efficiente. È fondamentale garantire che il prodotto soddisfi a pieno i requisiti funzionali e non, stabiliti durante la fase di progettazione.

3.4.2 Obiettivi

- Controllo continuo della qualità del prodotto, verificando che rispetti le aspettative della Proponente;
- Minimizzare la presenza di errori o anomalie nel prodotto;
- Riduzione dei rischi che potrebbero influenzare la qualità;
- Consegna del progetto rispettando il budget preventivato inizialmente e i requisiti individuati assieme alla Proponente.

Per la valutazione della qualità viene fornito il documento *Piano di Qualifica*, in cui sono presenti varie metriche con le relative soglie di valori accettabili ed ideali.

3.4.3 Denominazione Metriche

Per identificare le metriche si usa la seguente formattazione:

M[Tipo]-[Nome]

legenda:

- **Tipo**: specifica la tipologia di metrica:
  1. **PC** se si tratta di qualità di processo;
  2. **PD** se si tratta di qualità di prodotto;
- **Nome**: abbreviazione del nome della metrica.

4 Processi Organizzativi

4.1 Gestione Organizzativa

4.1.1 Decisioni

Per poter prendere una qualsiasi decisione è necessario vi siano due condizioni:

1. Si deve raggiungere il $quorum_G$ di quattro persone su sei;
2. La decisione deve essere verbalizzata e motivata.

4.1.2 Ruoli Progetto

Durante il periodo di sviluppo del progetto, ogni membro assumerà 6 ruoli distinti con compiti diversificati, al fine di garantire una gestione completa ed efficace delle diverse fasi e aspetti del lavoro. Questi ruoli contribuiranno a promuovere la collaborazione sinergica e l'ottimizzazione delle risorse all'interno del team.

I ruoli assunti sono i seguenti:

4.1.2.1 Responsabile

Figura chiave che supervisiona, coordina e gestisce le attività del team. Si occupa di garantire l'allineamento tra gli obiettivi del progetto e le azioni intraprese, gestisce le risorse disponibili, stabilisce e mantiene le relazioni esterne con la Proponente e assicura il flusso efficace delle informazioni all'interno del team e al di fuori di esso.

I suoi compiti sono:

- Gestione della comunicazione con la Proponente (si fa uso della piattaforma Element);
- Preparazione dell'ordine del giorno per la successiva riunione, anche sulla base dei punti individuati dagli altri componenti;
- Redazione dei verbali interni ed esterni;
- Stesura e e avanzamento del documento "Piano di progetto";
- Creazione dei diari di bordo;
- Upload dei verbali esterni convalidati, tramite firma, dalla Proponente in una cartella apposita su Google Drive.

4.1.2.2 Amministratore

Figura professionale con la responsabilità della creazione, manutenzione e ottimizzazione degli strumenti, delle risorse e dei processi necessari per il corretto svolgimento del progetto.

I suoi compiti sono:

- Configurazione e gestione gli ambienti di sviluppo;
- Implementazione delle procedure operative;

- Assicurare la disponibilità degli strumenti necessari per la collaborazione e la comunicazione all'interno del team;
- Creazione e assegnazione delle *issue* ai membri del team;
- Stesura e avanzamento del documento "Norme di Progetto";
- Implementazione di script dedicati per automatizzare processi nell'ambiente di lavoro;
- Gestione del versionamento dei documenti.

4.1.2.3 Analista

Figura professionale che si occupa di analizzare, comprendere e definire i requisiti e le specifiche di un sistema software prima che venga sviluppato. Questa figura svolge un ruolo fondamentale nel processo di sviluppo del software, contribuendo a garantire che il prodotto finale soddisfi le esigenze e le aspettative degli utenti e della Proponente.

I suoi compiti:

- Studio del contesto applicativo e relativa complessità;
- Specifica dei casi d'uso per comprendere in dettaglio i requisiti funzionali del sistema;
- Raccolta dei requisiti per definire le necessità e le funzionalità richieste;
- Stesura del documento Analisi dei Requisiti;
- Creazione diagrammi UML;

4.1.2.4 Progettista

Figura professionale specializzata nella progettazione architetturale e strutturale di sistemi. La sua responsabilità principale è definire la configurazione, la disposizione e l'organizzazione dei vari componenti del sistema, concentrandosi su come questi elementi interagiscono tra loro per raggiungere determinati obiettivi di funzionalità, prestazioni e scalabilità.

I suoi compiti sono:

- Scelta degli aspetti tecnici e tecnologici;
- Progettazione architetturale che miri all'economicità e alla manutenibilità del sistema;
- Ottimizzazione delle prestazioni usando algoritmi efficienti e gestione memoria;
- Gestione dei rischi: cerca di mitigare problemi che possono sorgere durante lo sviluppo;
- Redazione della Specifica tecnica e di una parte del documento Piano di Qualifica.

4.1.2.5 Programmatore

Figura professionale incaricata di trasformare le specifiche tecniche in codice eseguibile, garantendo un'implementazione efficiente e accurata delle funzionalità richieste dal progetto.

I suoi compiti:

- Traduzione delle specifiche tecniche in codice funzionante;
- Scrittura di codice chiaro, leggibile e mantenibile;
- Creazione di test per la verifica del software;
- Ampliamento delle Specifiche Tecniche conforme alle esigenze del progetto.
- Risoluzione di bug e problemi di performance;
- Realizzazione del Manuale Utente;
- Collaborazione con il team per l'integrazione del codice e il mantenimento della coerenza del progetto.

4.1.2.6 Verificatore

Figura professionale che si occupa di esaminare il lavoro prodotto dagli altri membri del team.

I suoi compiti:

- Revisione e valutazione della documentazione prodotta dal team;
- Analisi critica del codice per individuare errori, discrepanze o possibili miglioramenti;
- Identificazione e segnalazione di problemi;
- Collaborazione con il team per garantire che il lavoro sia conforme alle linee guida e agli standard richiesti.

4.1.3 Cambio dei ruoli

Per fare in modo che ogni membro svolga almeno 1 volta tutti i ruoli di cui sopra, il team si impegna a cambiarli ogni settimana.

4.1.4 Procedure

4.1.4.1 Gestione delle comunicazioni

4.1.4.1.1 Comunicazioni Interne

Riguardano esclusivamente i membri del Team e si svolgono tramite:

- ***Whatsapp***: utilizzato per messaggistica istantanea e una comunicazione veloce;
- ***Discord***: piattaforma utilizzata per:

1. Creare server suddivisibili in vari canali testuali o vocali, dove verranno svolte le riunioni;
2. Supplementare la comunicazione all'interno della piattaforma con funzionalita' offerte da servizi esterni quali GitHub

4.1.4.1.2 Comunicazioni Esterne

Le comunicazioni esterne vengono affidate al Responsabile attraverso i seguenti mezzi:

- *Email* : si usa l'email di gruppo swateng.team@gmail.com;
- *Element*: si usa il canale creato appositamente dalla Proponente per avere una comunicazione diretta.

4.1.4.2 Gestione degli Incontri

4.1.4.2.1 Incontri Interni

Negli incontri interni possono partecipare solamente i membri del gruppo. Si svolgono principalmente una volta a settimana, il giorno può variare in caso di imprevisti, ma solitamente si tiene il venerdì mattina in modalità sincrona.

Le linee guida per le riunioni:
1. Prima dell'incontro avere un ordine del giorno, ovvero i punti eventuali da discutere;
2. Discussione dei punti;
3. Pianificazione attività per la settimana e assegnazione issue.

Alla fine dell'incontro:
1. Il responsabile ha il compito della stesura del verbale interno, fornendo una sintesi dei punti salienti dell'incontro.

Gli incontri hanno due modalità:

- **Fisici**: per gli stand-up meeting quotidiani (*Daily Scrum*$_G$) di 5 minuti in cui si discutono brevemente le attività completate il giorno precedente e si espongono le attività pianificate per il futuro;
- **Virtuali**: si svolgono chiamate o video di gruppo in cui si discutono eventuali dubbi o difficoltà riscontrate. Lo strumento adatto per questo scopo è Discord.

4.1.4.2.2 Incontri Esterni

Negli incontri esterni i partecipanti sono i membri del team e i referenti della Proponente. Questi incontri sono pianificati ogni due settimane in concomitanza con l'inizio e fine sprint. Durante queste sessioni, i partecipanti del team hanno l'opportunità di presentare gli sviluppi recenti, condividere i

progressi raggiunti e discutere eventuali sfide o questioni emerse nel corso del lavoro. Inoltre si possono richiedere incontri di formazione specifici su particolari tecnologie, il che offre al team l'opportunità di approfondire la comprensione di una tecnologia specifica, imparare le *best practice* e acquisire competenze più avanzate.

Il responsabile ha il compito della stesura del verbale esterno, che viene successivamente convalidato, con firma, dalla Proponente.