Norme di Progetto

Versione: 0.6.0 23/11/2024

Redattori	Malik Giafar Mohamed
Verifica	Maria Fuensanta Trigueros Hernandez Ion Cainareanu Stefano Baso
Approvazione	
Uso	Interno

nextsoftpadova@gmail.com

Registro dei cambiamenti

Versione	Data	Autore	Descrizione	Verifica
0.6.0	01/03/2025	Ion Cainareanu	Miglioramenti per varie sezioni	Malik Giafar Mohamed
0.5.0	28/02/2025	Malik Giafar Mohamed, Stefano Baso	Modifiche generali alle sezioni del documento	Ion Cainareanu
0.4.1	11/02/2025	Malik Giafar Mohamed	Integrazione di alcune specifiche relative al way of working	Ion Cainareanu
0.4.0	11/01/2025	Malik Giafar Mohamed	Integrazione parti mancanti del documento	Marco Perazzolo, Stefano Baso
0.3.0	05/01/2025	Malik Giafar Mohamed	Stesura sezione Documentazione fino a Norme Tipografiche	Ion Cainareanu, Stefano Baso
0.2.0	15/12/2024	Malik Giafar Mohamed	Stesura fino a sezione 2.1.2.2	0.1.0
23/11/2024	Malik Giafar Mohamed	Stesura Prima Versione	Maria Fuensanta Trigueros Hernandez, Ion Cainareanu	

Indice

1 Introduzione 4
1.1 Scopo del documento 4
1.2 Scopo del prodotto 4
1.3 Glossario 4
1.4 Riferimenti 4
2 Processi Primari 5
2.1 Fornitura 5
2.1.1 Scopo e aspettative 5
2.1.2 Documentazione Fornita 5
2.2 Sviluppo 5
2.2.1 Scopo e aspettative 5
2.2.2 Analisi dei Requisiti 6
2.2.3 Progettazione 7
2.2.4 Codifica 7
3 Processi di Supporto 8

3.1 Documentazione ... 8
3.1.1 Scopo e aspettative ... 8
3.1.2 Ciclo di vita del documento ... 8
3.1.3 Tipologie di documenti ... 9
3.1.4 Template ... 9
3.1.5 Struttura del documento ... 9
3.1.6 Norme Tipografiche ... 10
3.1.7 Strumenti ... 11
3.2 Gestione della configurazione ... 12
3.2.1 Scopo e aspettative ... 12
3.2.2 Versionamento ... 12
3.2.3 Strumenti ... 12
3.2.4 Struttura delle repository ... 12
3.2.5 Branch ... 13
3.3 Gestione della Qualità ... 13
3.3.1 Scopo ... 13
3.3.2 Piano di Qualifica ... 13
3.3.3 Ciclo di Deming ... 14
3.3.4 Denominazione Metriche ... 14
3.4 Verifica ... 14
3.4.1 Scopo e aspettative ... 14
3.4.2 Analisi statica ... 14
3.4.3 Analisi dinamica ... 15
3.4.4 Strumenti ... 15
3.4.5 Codice identificativo dei test ... 15
3.4.6 Verifica della Documentazione ... 16
3.5 Validazione ... 16
3.5.1 Scopo e aspettative ... 16
4 Processi Organizzativi ... 16
4.1 Gestione dei Processi ... 16
4.1.1 Scopo e aspettative ... 16
4.1.2 Ruoli di progetto ... 16
4.1.3 Issue Tracking System ... 17
4.1.4 Riunioni ... 18
4.2 Strumenti ... 19
4.3 Formazione del Personale ... 19

1 Introduzione

1.1 Scopo del documento

Il presente documento ha lo scopo di definire in modo dettagliato le *best practice*[G] e il *way of working*[G] del nostro gruppo per il progetto al fine di garantire qualità e coerenza nel lavoro svolto. Il documento sarà quindi soggetto a modifiche e integrazioni durante il corso del progetto, in particolare durante le fasi di analisi e progettazione, e quindi non può essere considerato come definitivo.

1.2 Scopo del prodotto

Il prodotto, un plug-in per Visual Studio Code chiamato "Requirement Tracker", è progettato per automatizzare il tracciamento dei *requisiti*[G] nei progetti software complessi, con un focus particolare sull'ambito embedded. L'obiettivo principale è migliorare la qualità e la chiarezza dei requisiti, fornendo suggerimenti basati sull'analisi di un'intelligenza artificiale, riducendo al contempo i tempi e gli errori legati alla verifica manuale dell'implementazione nel codice sorgente. Il plug-in adotta un'architettura modulare che consente un'estensibilità semplice, rendendolo facilmente adattabile a nuove funzionalità o esigenze future. Inoltre, supporta gli sviluppatori avendo la capacità di utilizzare documenti tecnici come knowledge, ad esempio datasheet e manuali, permette di garantire una corretta copertura dei requisiti.

1.3 Glossario

I termini ambigui che necessitano di una spiegazione sono contrassegnati da una [G] come apice alla loro prima occorrenza nei documenti. Tutti i termini da glossario sono riportati in ordine alfabetico nel documento Glossario.

1.4 Riferimenti

Riferimenti Normativi

- Presentazione del capitolato **Requirement Tracker - Plug-in VS Code**
  - https://www.math.unipd.it/~tullio/IS-1/2024/Progetto/C8.pdf
- Standard ISO/IEC 12207:1995
  - https://www.math.unipd.it/~tullio/IS-1/2009/Approfondimenti/ISO_12207-1995.pdf
- Documento Piano di Qualifica

Riferimenti Informativi

- Materiale didattico del corso di Ingegneria del Software
  - https://www.math.unipd.it/~tullio/IS-1/2024/Dispense/
- Documentazione GitHub:
  - https://help.github.com/en/github
- Documentazione git:
  - https://git-scm.com/docs
- Documentazione Typst:
  - https://typst.app/docs/
- Documentazione Visual Studio Code:
  - https://code.visualstudio.com/docs
  - https://code.visualstudio.com/api
- Documentazione Ollama:
  - https://github.com/ollama/ollama/tree/main/docs
- Documentazione NodeJS:
  - https://nodejs.org/en/docs/

2 Processi Primari

2.1 Fornitura

2.1.1 Scopo e aspettative

Il processo di fornitura determina le procedure e le risorse necessarie per gestire e garantire la consegna del prodotto. Tale processo è iniziato con l'aggiudicazione dell'appalto da parte di *NextSoft*[G]. Il nostro gruppo si aspetta quindi una comunicazione costante ed efficace con il *proponente*[G] al fine di garantire il rispetto dei vincoli progettuali definiti nella presentazione del *capitolato*[G] e di produrre un applicativo che soddisfi le esigenze del proponente.

2.1.2 Documentazione Fornita

Il processo di fornitura prevede la produzione dei seguenti documenti:

- **Analisi dei Requisiti**: descrive dettagliatamente i requisiti funzionali e non funzionali del progetto. Include una classificazione dei requisiti, i casi d'uso e le specifiche tecniche necessarie per soddisfare le esigenze del proponente. Questo documento è fondamentale per comprendere a fondo le necessità del proponente e per fornire una base solida per la progettazione e lo sviluppo del prodotto.
- **Piano di Qualifica**: definisce le strategie e le metodologie di verifica e validazione adottate dal gruppo. Include le metriche di qualità, i test pianificati e le procedure per garantire che il prodotto finale soddisfi i requisiti. Questo documento è essenziale per assicurare che il prodotto sia conforme agli standard di qualità e che tutte le funzionalità richieste siano implementate correttamente.
- **Piano di Progetto**: descrive la pianificazione temporale e le risorse necessarie per completare il progetto. Include il calendario delle attività, la suddivisione dei compiti tra i membri del gruppo e le milestone principali. Questo documento è cruciale per garantire che il progetto sia completato nei tempi previsti e che tutte le risorse siano utilizzate in modo efficiente.
- **Proof of Concept**: un prodotto software temporaneo creato per dimostrare la fattibilità del progetto. Viene utilizzato per verificare che le idee e le tecnologie proposte siano in grado di soddisfare i requisiti del progetto prima di procedere con lo sviluppo del prodotto finale.
- **Glossario**: documento che contiene la definizione dei termini tecnici e delle sigle utilizzate nel progetto. Questo documento è essenziale per garantire che tutti i membri del gruppo e le parti interessate abbiano una comprensione comune dei termini utilizzati nel progetto.

2.2 Sviluppo

2.2.1 Scopo e aspettative

Il processo di sviluppo ha lo scopo di delineare i compiti e le attività necessarie per la creazione del prodotto software. In questa sezione vengono descritte le attività, le norme e le convenzioni adottate per tale processo. Il gruppo si aspetta di:

- Definire vincoli e obiettivi di sviluppo chiari e strutturati
- Garantire la qualità del prodotto finale
- Consegnare il prodotto entro i tempi stabiliti senza sprecare risorse
- Soddisfare pienamente le richieste del proponente

Il processo di sviluppo prevede le seguenti attività:

- **Analisi dei Requisiti**
- **Progettazione**
- **Codifica**

2.2.2 Analisi dei Requisiti

Durante l'attività di analisi dei requisiti vengono definiti e documentati i requisiti e i casi d'uso del sistema. I requisiti rappresentano le caratteristiche e le funzionalità che il sistema deve possedere per soddisfare le esigenze degli utenti e degli stakeholder. I casi d'uso, invece, descrivono le interazioni tra gli utenti e il sistema, specificando come il sistema deve comportarsi in risposta a determinate azioni degli utenti.

L'analisi dei requisiti è compito degli *Analisti*[G], consiste nell'analizzare ogni singolo requisito e caso d'uso del progetto. Ha lo scopo di:

- Comprendere a fondo le necessità del proponente
- Aiutare i *progettisti*[G]
- Fornire riferimenti utili ai *verificatori*[G]

2.2.2.1 Casi d'uso

Ogni caso d'uso verrà riportato con una denominazione chiara e univoca che identifichi l'azione principale e l'attore coinvolto, il formato adottato sarà il seguente:

UC_[Numero caso d'uso](.[Numero sottocaso o scenario alternativo])* - Titolo

Mentre saranno strutturati nel seguente modo :

- **Denominazione**: codice identificativo del caso d'uso, stabilito come enunciato sopra
- **Diagramma UML**: diagramma per rappresentare graficamente il caso d'uso (opzionale?)
- **Attori Principali**: entità esterne al sistema che interagiscono con esso
- **Precondizioni**: descrivono lo stato del sistema prima del verificarsi del caso d'uso
- **Postcondizioni**: descrivono lo stato del sistema dopo che si è verificato il caso d'uso
- **Descrizione**: breve descrizione del caso d'uso
- **Scenario principale**: elenco puntato che descrive il flusso degli eventi del caso d'uso
- **Scenario secondario/alternativo**: elenco puntato che descrive il flusso degli eventi del caso d'uso dopo un evento imprevisto che lo ha deviato dal caso principale. Può non esserci o possono essercene più di uno

2.2.2.2 Denominazione dei requisiti

Ogni requisito verrà riportato nel seguente formato:

R[Classificazione][Tipologia][Identificativo]

Dove:

- **R**: indica che si tratta di un requisito
- **Classificazione**: specifica la categoria del requisito
  - **F**: requisito funzionale
  - **P**: requisito prestazionale
  - **Q**: requisito qualitativo
  - **V**: vincolo
- **Tipologia**: descrive la natura del requisito
  - **D**: requisito desiderabile
  - **O**: requisito obbligatorio
  - **F**: requisito facoltativo
- **Identificativo**: numero intero progressivo e univoco del requisito

2.2.3 Progettazione

2.2.3.1 Scopo e aspettative

L'attività di progettazione è legata al ruolo dei *Progettisti*[G], consiste nel definire l'architettura della soluzione considerando i requisiti derivanti dall'analisi fatta in precedenza.

Il gruppo si aspetta di:

- Definire un'architettura del sistema adeguata alle esigenze del proponente
- Assicurare che tutte le scelte progettuali siano documentate e giustificate
- Comunicare con gli stakeholders al fine di garantire che l'architettura soddisfi le aspettative

2.2.3.2 Requirements and Technology Baseline

Lo scopo della Requirements and Technology Baseline è di fornire una base solida per la progettazione e lo sviluppo del prodotto, dimostrando la fattibilità delle soluzioni proposte e garantendo che le tecnologie scelte siano adeguate per soddisfare i requisiti del proponente.

In questa *baseline*[G] dovranno essere forniti:

- ***Proof of Concept***[G]: una raccolta di implementazioni di alcune funzionalità che il prodotto dovrà avere. L'obiettivo è dimostrare la fattibilità delle richieste del proponente
- **Scelte tecnologiche**: verranno stabilite le tecnologie da utilizzare e le motivazioni legate a tali scelte
- **Diagrammi dei casi d'uso**: verranno inclusi i diagrammi dei casi d'uso individuati durante l'analisi dei requisiti
- **Test**: verranno definiti i test da eseguire sul prodotto.

2.2.3.3 Product Baseline

Lo scopo della Product Baseline è di fornire un prodotto finale che verrà valutato come *MVP*[G], dimostrando che esso soddisfi i requisiti del proponente e che l'architettura definita nella RTB è stata implementata correttamente.

Questa baseline si differenzia dalla prima per le seguenti informazioni raccolte dai progettisti:

- **Diagrammi delle classi**: verranno utilizzati per descrivere l'architettura delle classi del prodotto finale
- ***Design Pattern***[G]: la definizione dell'architettura può essere basata sull'utilizzo di design pattern, che consentono di risolvere problemi ricorrenti in modo rapido ed efficace. I design pattern sono schemi riutilizzabili di progettazione illustrati con diagrammi che ne mostrano la struttura;
- ***Test di unità***[G]: ovvero i risultati dei test eseguiti per verificare che il funzionamento delle classi e dei singoli moduli che implementano il sistema siano corretti e conformi ai requisiti.

2.2.4 Codifica

2.2.4.1 Scopo

L'attività di codifica è legata al ruolo dei *Programmatori*[G], consiste nell'implementazione del prodotto software tenendo conto della progettazione.

2.2.4.2 Strumenti

Per la codifica del prodotto, i principali strumenti adottati saranno:

- **Visual Studio Code**: in quanto è l'IDE più consigliato per realizzare proprie estensioni.
- **StarUML**: per la creazione dei diagrammi UML

2.2.4.3 Metodi

I metodi di un progetto verranno considerati accettabili solamente se brevi. Risulta essere una buona pratica in quanto porta notevoli vantaggi quali:

- **Manutenibilità**: sono più facili da mantenere rispetto a metodi lunghi e complessi in quanto il codice è più leggibile e comprensibile. Si ottiene così un codice robusto e meno suscettibile a errori.
- **Leggibilità**: il codice è reso più accessibile a tutti gli sviluppatori che potrebbero doverlo leggere, comprendere o modificare in futuro.
- **Debugging**: se un metodo è breve e semplice, è più facile identificare eventuali bug o problemi nel codice. Questo rende il processo di debugging più efficiente.

Per la dichiarazione dei metodi si fa riferimento alla convenzione "*CamelCase*[G]" per la scelta del nome del metodo.

2.2.4.4 Univocità dei nomi

Tutte le variabili, i metodi e le classi dovranno avere un nome che li distingua univocamente, per evitare di creare confusione durante la stesura e l'analisi.

2.2.4.5 Ricorsione

La ricorsione può essere un approccio utile per lo sviluppo, però gli svantaggi che porta possono essere maggiori dei vantaggi. È importante valutare attentamente l'utilizzo della ricorsione perché potrebbe comportare alcune sfide e limitazioni, tra cui:

- **Performance**: la ricorsione può comportare una maggiore complessità computazionale rispetto ad altre soluzioni, poiché ogni chiamata ricorsiva comporta l'aggiunta di un nuovo livello alla pila delle chiamate. Quindi aumenta la quantità di memoria utilizzata e ridurre la velocità di esecuzione del programma.
- **Debugging**: la ricorsione richiede maggior attenzione durante l'attività di debugging, questo può rendere più complesso capire dove si verifica un errore o dove un problema sta influenzando il comportamento del programma.

Qualora si dovesse ricorrere alla di ricorsione, la decisione dovrà essere adeguatamente giustificata tramite commenti.

3 Processi di Supporto

3.1 Documentazione

3.1.1 Scopo e aspettative

Il processo di documentazione ha lo scopo di stabilire le linee guida per la redazione dei documenti di progetto.

3.1.2 Ciclo di vita del documento

Il ciclo di vita di un documento è suddiviso nelle seguenti fasi:

- **Creazione**: in questa fase viene definita la struttura del documento e vengono raccolte le informazioni necessarie per la sua stesura. Viene inoltre assegnato un responsabile per la redazione del documento.
- **Stesura**: redazione o modifica del documento secondo le linee guida e gli standard definiti.
- **Verifica**: una volta completata la stesura, il documento viene sottoposto a verifica da parte di uno o più verificatori. Quest'ultimi controllano la correttezza e la completezza del contenuto, segnalando eventuali errori o incongruenze. Eventualmente è possibile un ritorno alla fase di stesura in caso il documento non venga aprrovato.
- **Approvazione**: una volta completato il documento, l'approvazione finale deve essere effettuata dal Responsabile di Progetto. Una volta approvato, il documento viene considerato ufficiale e può essere distribuito o utilizzato come riferimento.

Ogni fase del ciclo di vita del documento è fondamentale per garantire la qualità e l'accuratezza delle informazioni contenute. È importante seguire rigorosamente queste fasi per assicurare che il documento finale soddisfi tutti i requisiti e gli standard previsti.

3.1.3 Tipologie di documenti

3.1.3.1 Documenti ad uso Interno

I documenti ad uso interno sono destinati all'uso esclusivo dei membri del gruppo di progetto e del committente. I principali documenti ad uso interno sono:

- Norme di Progetto
- Verbali interni

3.1.3.2 Documenti ad uso Esterno

I documenti ad uso esterno sono destinati all'uso del committente e agli stakeholders. I principali documenti ad uso esterno sono:

- Analisi dei Requisiti
- Piano di Qualifica
- Piano di Progetto
- Glossario
- Verbali esterni

3.1.4 Template

Sono stati creati dei template per facilitare la stesura dei documenti. Ogni template contiene uno stile unico utilizzato da tutti i documenti che lo importano. Possono esistere molteplici template contemporaneamente; attualmente i principali template sono quello della Candidatura e quello dell'RTB.

3.1.5 Struttura del documento

3.1.5.1 Frontespizio

La prima pagina di ogni documento contiene le seguenti informazioni:

- Logo del gruppo
- Titolo del documento
- Sottotitolo del documento (opzionale)
- Data di creazione del documento
- Versione del documento
- Nome e cognome dei redattori
- Nome e cognome dei verificatori
- Nome e cognome del responsabile di progetto che ha approvato quel documento
- Uso del documento (interno/esterno)
- Email del gruppo

Tutti i seguenti elementi sono disposti al centro del documento, versione e data di creazione sono disposti in linea; mentre l'approvatore, i redattori e i verificatori sono disposti in righe separate di una tabella creata ad hoc.

3.1.5.2 Registro dei cambiamenti

Il registro delle modifiche contiene la cronologia delle modifiche apportate ai documenti scritti in maniera incrementale. E' situato alla seconda pagina del documento sopra all'indice e contiene le seguenti informazioni:

- Versione del documento

- Data della modifica
- Nome e cognome del redattore
- Breve descrizione della modifica
- Nome e cognome dei verificatori

Le informazioni sono disposte in colonne separate di una tabella creata ad hoc.

3.1.5.3 Indice

L'indice del documento è disposto dopo il frontespizio e prima del corpo del documento, contiene la lista dei capitoli e delle sezioni presenti nel documento, con i relativi numeri di pagina.

3.1.5.4 Corpo del documento

In ogni pagina del documento sono presenti i seguenti elementi:

- **Intestazione**
  - Nome del gruppo: posizionato in alto a sinistra della pagina
  - Titolo del documento: posizionato in alto al centro della pagina
  - Versione del documento: posizionato in alto a destra della pagina
- **Contenuto**
  - Testo del documento
- **Piè di pagina**
  - Numero di pagina: posizionato in basso a destra della pagina

3.1.5.5 Verbali

I verbali sono documenti che riportano le decisioni prese durante le riunioni del gruppo di progetto. I verbali differiscono leggermente nello stile grafico del frontespizio degli altri documenti, e non contengono il registro dei cambiamenti.

3.1.6 Norme Tipografiche

3.1.6.1 Nome del documento

Ogni documento ha una denominazione omogenea. I verbali come nome file hanno la data del relativo incontro in formato YYYY-MM-DD, mentre gli altri documenti sarano denominati nel seguente modo:

`Nome_File-vX.Y.Z`

Per la denominazione dei file si fa riferimento alla convenzione "CamelCase[G]", le parole verranno separate dal carattere "_" (underscore) e la versione sarà indicata con la lettera "v" seguita dal numero di versione, specificato a sua volta nel paragrafo *Versionamento*.

Per questioni di praticità, questo tipo di nomenclatura vale solo per i documenti ad uso interno ed esterno, non viene applicata ai template, alle immagini e altri tipi di documento. La nomenclatura dei documenti senza versionamento utilizza sempre gli underscore al posto degli spazi, ma senza utilizzare il CamelCase.

3.1.6.2 Stile del testo

Il testo dei documenti deve essere scritto in lingua italiana, e utilizzerà i seguenti formati di testo:

- **Grassetto**: per evidenziare i titoli di sezioni
- **Corsivo**: per far riferimento alla prima occorrenza di un termine del glossario
- **Monospace**: per riportare nomi di file, cartelle o elementi che richiamano alla stesura di codice

3.1.6.3 Elenchi

Gli elenchi puntati seguono le seguenti norme:

- Il simbolo che scandisce ogni elemento dell'elenco è il pallino (•), al secondo e terzo livello si trovano rispettivamente il trattino(-) e il triangolo nero (‣)

- Le voci iniziano per lettera maiuscola
- Le liste del tipo “Termine: descrizione” presentano il termine in grassetto con la prima lettera in maiuscolo

3.1.6.4 Sigle

Nella documentazione verranno utilizzate delle sigle per facilitare l’identificazione di un documento, ruolo o revisione senza doverne scrivere il nome per intero. Queste si sono rivelate utili soprattutto nell’utilizzo di tabelle.

Le sigle utilizzate sono le seguenti:

- Sigle relative ai documenti:
  - **AdR**: Analisi dei Requisiti
  - **NdP**: Norme di Progetto
  - **PdQ**: Piano di Qualifica
  - **PdP**: Piano di Progetto
  - **VI**: Verbale Interno
  - **VE**: Verbale Esterno
- Sigle relative alle revisioni:
  - **RTB**: Requirement and Technology Baseline
  - **PB**: Product Baseline
  - **CA**: Customer Acceptance
- Sigle relative ai ruoli:
  - **Re**: Responsabile
  - **Am**: Amministratore
  - **An**: Analista
  - **Prj**: Progettista
  - **Ve**: Verificatore
  - **Prg**: Programmatore

3.1.6.5 Elementi grafici

Vengono seguite le seguenti norme per utilizzare immagini, grafici e tabelle:

- **Immagini**: sono centrate e accompagnate da una didascalia
- **Diagrammi dei casi d’uso**: anch’essi verranno inseriti come immagine nel documento; verrano centrati, numerati e accompagnati da una didascalia
- **Tabelle**: come per i diagrammi, anch’esse sono centrate e numerate, con una didascalia che le descrive. Non è stato definito uno stile grafico standard per le tabelle, dunque è possibile che esso possa essere diverso per i vari documenti
- **Collegamenti ipertestuali**: sono riportati nel documento di colore blu, più precisamente del colore 005d88

3.1.7 Strumenti

Per la stesura dei documenti finora sono stati utilizzati i seguenti strumenti:

- **Typst**: linguaggio di markup simile a Markdown utilizzato per la stesura di documenti
- **Visual Studio Code**: IDE utilizzato per la scrittura del codice sorgente dei documenti
- **Typst.app**: sito web utilizzato come alternativa per la stesura dei document

- **GitHub Actions**: utilizzate per la generazione automatica dei file pdf derivanti dal codice sorgente dei documenti

3.2 Gestione della configurazione

3.2.1 Scopo e aspettative

Il processo di gestione della configurazione ha lo scopo di gestire in modo ordinato e sistematico la produzione di documenti e codice. Per ogni oggetto sottoposto a configurazione viene garantito il versionamento e il controllo sulle modifiche per permettere il mantenimento dell'integrità del prodotto. Il gruppo si aspetta di mantenere una *repository*[G] organizzata e ben strutturata.

3.2.2 Versionamento

Per poter capire lo stato di avanzamento di un prodotto derivante delle attività del progetto è necessario un identificatore. Il formato del numero di versione utilizzato è il seguente:

X . Y . Z

dove:

- **X**, **Y** e **Z** sono numeri interi positivi
- **X** corrisponde ad una versione approvata dal Responsabile di Progetto. La numerazione parte da 0.
- **Y** indica l'aggiunta di un incremento, senza però essere arrivati ad avere una versione rilasciabile del prodotto. La numerazione parte da 0 e si azzera ad ogni incremento di **X**.
- **Z** viene incrementato ad ogni piccola modifica o correzione, questo tipo di versione assume il nome di *minor*. La numerazione parte da 0 e si azzera ad ogni incremento di **X** o **Y**.
- Per evitare un eccessivo overhead burocratico, le versioni **Z** possono essere verificate insieme alle **Y**, mentre le versioni **X** devono sempre essere verificate prima di nuove modifiche.

3.2.3 Strumenti

Per il versionamento si è scelto di utilizzare un repository GitHub, che, a sua volta, implementa il software di controllo versione distribuito Git.

3.2.4 Struttura delle repository

3.2.4.1 Documenti

La repository utilizzata dal gruppo per la creazione dei documenti è strutturata nel seguente modo:

- `Candidatura`: contiene i documenti relativi alla candidatura del gruppo per il capitolato.
  - `src`: contiene il codice sorgente dei file pdf.
  - `Verbali`: contiene i verbali esterni ed interni delle riunioni relativi alla candidatura.
- `RTB`: contiene i documenti relativi alla milestone RTB, quindi le norme di progetto, il piano di progetto e di qualifica, l'analisi dei requisiti e il glossario.
  - `Documentazione Esterna`: contiene i documenti ad uso interno relative alla milestone RTB.
    - `src`: contiene il codice sorgente dei file pdf.
    - `Verbali`: contiene i verbali esterni fatti con il proponente per discutere del capitolato e del PoC.
    - `img`: contiene i file delle immagini a supporto dei documenti, ogni insieme di immagini è raggruppato in una cartella denominata con la sigla del documento al quale essa è associata.
  - `Documentazione Interna`: contiene i verbali delle riunioni relative alla milestone RTB.
    - `src`: contiene il codice sorgente dei file pdf.
    - `Verbali`: contiene i verbali delle riunioni relative alla milestone RTB.

  `assets`: contiene tutto ciò che è di supporto alla documentazione, come:
  - file di template per i documenti.

- il logo del gruppo, utilizzato per tutta la repository.
- `utils`: creata contiene script di utility, attualmente contiene solo uno script che agevola la generazione del sito web del gruppo.

La repository è pubblica e si può facilmente trovare al seguente link:
- https://github.com/nextsoftPD/Documenti

3.2.4.2 PoC
La repository del PoC è stata creata come repository privata per garantire la riservatezza del codice e dei dati di sviluppo. Trattandosi di una demo, la sua struttura è volutamente semplice e focalizzata sulla dimostrazione delle funzionalità chiave del progetto. All'interno della repository è presente un file README che fornisce istruzioni dettagliate su come configurare e lanciare il PoC, includendo eventuali dipendenze richieste, comandi di avvio e linee guida per la corretta esecuzione.

Il Proof of Concept è stato mostrato e condiviso con l'azienda proponente, che ha avuto modo di testarlo e fornire feedback.

3.2.5 Branch
Tutte le repository del gruppo si compongono di più *branch*[G], suddivisi ad hoc, in modo da garantire una separazione da ciò che è stabile e verificato e ciò che è in fase di sviluppo. Ogni membro può fare delle modifiche in un branch specifico a patto che esse siano relative solo ai configuration items[G] modificabili in quello specifico branch.

La suddivisione dei branch varia in base allo scopo della repository.

In nessuna repository è consentito modificare direttamente il branch principale, poiché porterebbe ad un elevato rischio di incongruenze e merge conflicts[G]. Si potranno applicare modifiche solo tramite il meccanismo di pull request[G], con verifica obbligatoria da parte di un verificatore, in modo da garantire che sia sempre presente una versione verificata e corretta del documento, anche se incompleta. Nel caso di una minor invece, la verifica può essere svolta nel momento stesso in cui viene aggiunta una versione "stabile".

Per quanto riguarda cambiamenti minimali (punteggiatura, errori ortografici, ecc.) è permessa la modifica autonoma da parte dei verificatori, a patto che sia solo allo scopo di risolvere errori ortografici o per migliorare la comprensibilità alcune frasi senza modificarne il significato logico. Questa decisione è stata presa al fine di evitare la creazione di numerose minor e di poter proseguire più velocemente con la stesura dei documenti.

La repository dei documenti è suddivisa in più branch[G] così definiti:

- **`main`**: il branch principale, che contiene l'ultima versione verificata di ogni documento
- **`nome_documento`**: branch che assume il nome del documento a cui fa riferimento, qui sarà possibile modificare solo il documento in questione.
- **`bugfix`**: branch secondario utilizzato per la correzione di errori di vario genere.

3.3 Gestione della Qualità

3.3.1 Scopo
La gestione della qualità di progetto è l'insieme delle attività che vengono eseguite all'interno di un progetto per garantire qualità su prodotti e processi in modo che il prodotto finale possa soddisfare i requisiti e le aspettative degli stakeholders.

3.3.2 Piano di Qualifica
Il documento Piano di Qualifica[G] definisce le strategie e le metodologie di verifica e validazione adottate dal gruppo, verrà quindi utilizzato come riferimento per garantire che il processo di gestione della qualità raggiunga l'economicità.

3.3.3 Ciclo di Deming

Per mantenere un'alta qualità di lavoro si è stabilito l'utilizzo del ciclo di Deming[G] (o PDCA),un approccio continuo e costante al miglioramento della qualità dei processi e dei prodotti, basandosi su uno schema sistematico e iterativo che consiste di quattro punti:

- **Plan**: Si identificano gli obiettivi di miglioramento e si pianificano le azioni necessarie per raggiungerli. Si analizzano i processi attuali, si raccolgono dati e si individuano le aree che necessitano di miglioramenti.
- **Do**: si implementano le azioni pianificate nella fase precedente. Si eseguono le attività necessarie per apportare i miglioramenti, raccogliendo dati e documentando i risultati ottenuti.
- **Check**: si verificano i risultati ottenuti confrontandoli con gli obiettivi stabiliti nella fase di pianificazione. Si analizzano i dati raccolti per determinare se le azioni intraprese hanno portato ai miglioramenti desiderati.
- **Act**: si decide se standardizzare le nuove pratiche o apportare ulteriori modifiche. Se i risultati della fase di verifica dimostrano che le azioni intraprese hanno portato a miglioramenti significativi, queste diventano il nuovo standard. Altrimenti, si ritorna alla fase di pianificazione per individuare nuove azioni correttive.

3.3.4 Denominazione Metriche

La denominazione delle metriche segue il seguente formato:

M[Categoria][TipoProdotto][Numero]

dove:

- **M**: indica che si tratta di una metrica di qualità
- **[Categoria]**: indica a quale categoria appartiene la metrica, e può assumere i seguenti valori:
  - **PD**: per indicare i prodotti
  - **PC**: per indicare i processi
  - **TS**: per indicare i test
- **[TipoProdotto]**: indica se si riferisce a documenti o software ed è presente solo nel caso sia una metrica di prodotto. Può assumere i seguenti valori:
  - **D**: per indicare i documenti
  - **S**: per indicare i prodotti software
- **Numero**: rappresenta il codice numerico identificativo della metrica, inizia da 01

3.4 Verifica

3.4.1 Scopo e aspettative

Il processo di verifica ha lo scopo di garantire che i prodotti e i processi siano conformi agli standard e ai requisiti definiti. Il gruppo si aspetta di verificare costantemente i prodotti e i processi per garantire la qualità del lavoro svolto.

3.4.2 Analisi statica

L'analisi statica è una tecnica di verifica del software che si effettua senza eseguire il codice. Si basa sull'esame del codice sorgente, della documentazione e di altri artefatti del progetto per individuare errori, violazioni di standard e altre problematiche. Nell'analisi statica si distinguono due tecniche principali:

- **Walkthrough**: consiste in una ricerca generale di errori, analizzando l'insieme di tutti i configuration items. Durante un walkthrough, i membri del team esaminano il codice o i documenti per identificare problemi evidenti, errori di logica, violazioni degli standard di codifica e altre anomalie.
- **Inspection**: consiste nell'eseguire una lettura mirata di una specifica unità di testing. Le inspection sono solitamente più brevi rispetto ai walkthrough. Durante un'inspection, un team di revisori

esamina il codice o alcuni paragrafi di documenti, utilizzando una checklist di criteri predefiniti per identificare difetti. Ogni difetto trovato viene documentato e discusso, e vengono proposte soluzioni per correggerlo.

3.4.3 Analisi dinamica

L'analisi dinamica è applicabile solo al prodotto software in quanto prevede l'esecuzione di test, cioè prove sul codice in esecuzione. Un test definito correttamente deve:

- Essere ripetibile: dato un determinato input si deve ottenere sempre lo stesso output per ogni prova effettuata
- Specificare l'ambiente di esecuzione
- Identificare input e output richiesti
- Fornire informazioni utili sui risultati dell'esecuzione

Esistono diverse categorie di test, ognuno con uno scopo e oggetto di verifica diverso.

3.4.3.1 Test di unità

I test di unità verificano la correttezza di una piccola parte di software testabile, chiamata unità, per stabilirne il corretto funzionamento rispetto alle attese. Le unità vengono testate con l'ausilio di *driver*[G] e *stub*[G] che simulano rispettivamente un'unità chiamante e un'unità chiamata che però non sono state ancora implementate del tutto.

3.4.3.2 Test di Sistema

I test di sistema verificano il sistema completo del prodotto software, prendendo in considerazione tutti i componenti e interfacce con altri sistemi. Questi test controllano che il software rispetti tutti i requisiti prestabiliti e che sia adatto all'uso in produzione.

3.4.3.3 Test di Regressione

I test di regressione servono a testare che gli aggiornamenti o modifiche rilasciati nel software non incidano negativamente sulle funzioni già presenti. Ciò consiste nella ripetizione di test di unità, integrazione e sistema.

3.4.3.4 Test di Accettazione

Il test di accettazione o collaudo è un'attività esterna, supervisionata dal committente e consiste nel dimostrare il soddisfacimento dei requisiti. Al collaudo segue il rilascio del prodotto.

3.4.4 Strumenti

Non sono ancora stati individuati strumenti per la verifica del codice.

3.4.5 Codice identificativo dei test

I test vengono identificati dal codice

T[Tipo]-[ID]

dove:

- **T**: indica che si tratta di un test;
- **Tipo**: indica il tipo di test e può assumere i seguenti valori:
  - **U** per i test di unità;
  - **I** per i test d'integrazione;
  - **S** per i test di sistema;
  - **R** per i test di regressione;
  - **A** per i test di accettazione.
- **ID**: codice numerico progressivo che inizia da 1.

3.4.6 Verifica della Documentazione

La documentazione è considerata valida quando:

- E' coerente e rispetta i criteri specificati nel seguente documento.
- E' completa ed esaustiva, coprendo tutti gli aspetti richiesti.
- La gestione della configurazione dei documenti segue le procedure specificate nei paragrafi addietro.

3.5 Validazione

3.5.1 Scopo e aspettative

Il processo di validazione è necessario per determinare se il prodotto finale è pronto per l'utilizzo, sono quindi necessari vari test per assicurare che il prodotto contenga tutte le funzionalità richieste dal proponente. Le aspettative del gruppo sono di avere un prodotto che soddisfi i requisiti del proponente e che sia pronto per l'utilizzo.

4 Processi Organizzativi

4.1 Gestione dei Processi

4.1.1 Scopo e aspettative

Il processo di gestione dei processi identifica le attività e i compiti di progetto necessari per gestire efficacemente i processi utilizzati dal team. Gli obiettivi principali del nostro gruppo per questo processo sono:

- Pianificare con precisione le attività da svolgere
- Comunicare in modo efficiente tra i membri del gruppo
- Evitare il più possibile conflitti di interesse

4.1.2 Ruoli di progetto

I ruoli di progetto non sono assegnati a periodo ma ad attività, in modo da garantire una maggiore flessibilità e adattabilità alle esigenze del progetto. Questo significa che ogni membro del gruppo avrà un ruolo definito in base ad accordi presi sulla ripartizione di determinate task.

I ruoli che ciascun membro dovrà ricoprire sono i seguenti:

Responsabile di Progetto
Rappresenta l'intero progetto, punto di riferimento sia per il committente sia per il proponente, con lo scopo di mediare tra le due parti. Si assume la responsabilità delle scelte del gruppo dopo averle approvate. In particolare, si occupa di:

- Approvare l'emissione della documentazione
- Approvare l'offerta economica sottoposta al committente
- Pianificare e coordinare le attività di progetto
- Gestire le risorse umane

Amministratore di Progetto
Esegue delle procedure di controllo e amministrazione dell'ambiente di lavoro. In particolare, si occupa di:

- Risolvere problemi legati alla gestione dei processi
- Gestire la documentazione di progetto
- Automatizzare i processi
- Individuare punti di miglioramento nei processi
- Redigere e attuare i piani e le procedure per la gestione della qualità

Analista
Possiede maggiori competenze riguardo il dominio applicativo del problema. Si occupa di:
- Studiare il problema e il relativo contesto applicativo
- Comprendere il problema e definire la complessità e i requisiti
- Svolgere l'Analisi dei Requisiti e redigere il relativo documento

Progettista
Gestisce gli aspetti tecnologici e tecnici del progetto. In particolare si occupa di:
- Effettuare scelte riguardanti gli aspetti tecnici e tecnologici del progetto, favorendone l'efficacia e l'efficienza
- Definire un'architettura del prodotto da sviluppare che miri all'economicità e alla manutenibilità a partire dal lavoro svolto dall'analista
- Redigere la parte pragmatica del Piano di Qualifica

Verificatore
Verifica il lavoro svolto dagli altri componenti del gruppo, sulla base delle proprie competenze tecniche, esperienza e conoscenza delle norme. In particolare si occupa di:
- Esaminare i prodotti in fase di revisione, con l'ausilio delle tecniche e degli strumenti definiti nelle Norme di Progetto
- Verificare la conformità dei prodotti ai requisiti funzionali e di qualità
- Segnalare eventuali errori

Programmatore
Svolge la fase di codifica del progetto e delle componenti di supporto che verranno utilizzate per eseguire prove di verifica e validazione sul prodotto. In particolare, si occupa di:
- Scrivere un codice ordinato e facile da mantenere
- Realizzare gli strumenti per la verifica e la validazione del software

4.1.3 Issue Tracking System

Il gruppo adotta GitHub Projects come Issue Tracking System (ITS). GitHub Projects permette una gestione semplice e chiara dei compiti da svolgere. Lo stato di avanzamento è segnato tramite le seguenti colonne:
- `To do`: contiene la lista delle task create dal responsabile.
- `In progress`: contiene le task che sono in fase di svolgimento.
- `In review`: contiene le task completate e in attesa di verifica.
- `Done`: contiene le task completate e verificate.

Ogni volta che è necessario portare a termine un compito, si segue questa procedura:
1. Il responsabile crea e assegna la task su GitHub Issues
    1. una task può essere creata da chiunque, ma deve essere assegnata al responsabile per essere inserita nel progetto
    2. una issue può corrispondere a più task o ad una singola parte di una task
2. L'incaricato si assegna la task spostando la issue nella colonna `In progress`, segnando l'inizio del lavoro di produzione
    1. Sono possibili spostamenti in avanti di più colonne, aggiungendo un commento che ne spiega il motivo.
3. Finito il lavoro di produzione, viene aperta la pull request su GitHub per fare il merge dal branch sul quale si stava lavorando al branch principale
    1. Una pull request può corrispondere a più issues e quindi incorporare più task purché siano correlate tra loro

4. La task viene marcata come `In review` su GitHub Issues.
5. Il verificatore verifica la presenza di errori, incongruenze o bug all'interno del lavoro svolto.
    1. Se la verifica ha esito positivo:
        1. Il verificatore approva la pull request su GitHub con una review, che può essere richiesta da chi ha aperto la pull request
        2. Il cambiamento viene integrato sul branch principale, facendo il merge della pull request
        3. La issue viene marcata come `Done` su GitHub Issues
    2. Se la verifica ha esito negativo:
        1. Il verificatore richiede dei cambiamenti sul commit della pull request
        2. Si ritorna al punto precedente

Considerazioni aggiuntive:

- Una pull request può essere approvata dalla persona che l'ha creata, oppure dall'amministratore di progetto in caso sia rimasta aperta per troppo tempo.
- Nel caso della verifica dei documenti, il verificatore aggiunge il suo nome nella lista dei verificatori del documento o nella colonna di verifica del versionamento
- Nel caso un branch contenga troppi errori, o ci siano state molte modifiche ad una pull request molto vecchia, il verificatore può richiedere all'amministratore di chiudere quella pull-request con un commento che ne spiega il motivo
- Nel caso in cui la verifica abbia esito negativo, quando verranno effettuate le modifiche, non sarà necessario aumentare di una versione, poiché il configuration item verrà ritenuto valido quando "stabile" e di conseguenza privo di errori o incongruenze
- I cambiamenti possono essere richiesti tramite un commento su github o possono essere accordati in una riunione o una breve chiamata

4.1.4 Riunioni

Le riunioni sono un momento fondamentale per coordinare le attività del gruppo e prendere decisioni importanti. Esse si dividono in: **Riunioni interne**
Riservate ai membri del gruppo di progetto e con lo scopo di:

- Coordinare le attività del gruppo
- Discutere e risolvere eventuali problemi
- Pianificare le attività future
- Verificare lo stato di avanzamento del progetto

Riunioni esterne
Che coinvolgono anche il Proponente o altre parti interessate e hanno lo scopo di:

- Presentare lo stato di avanzamento del progetto
- Discutere e approvare le decisioni importanti
- Ricevere feedback dal Proponente

La frequenza delle riunioni dipende dalle esigenze del progetto e può variare nel tempo. In generale, si prevede di tenere una riunione interna settimanale per discutere lo stato di avanzamento e pianificare le attività future. E' prevista, ma ancora da definire, la frequenza delle riunioni esterne con il proponente.

Per ogni riunione viene redatto un verbale che riporta:

- Data e ora della riunione
- Argomenti trattati
- Decisioni prese

- Azioni da intraprendere
- Analisi delle criticità riscontrate

I verbali delle riunioni vengono archiviati e resi disponibili a tutti i membri del gruppo per garantire la trasparenza e la tracciabilità delle decisioni prese.

4.1.4.1 Comunicazioni

Le comunicazioni all'interno del gruppo avvengono principalmente attraverso i seguenti canali:

- **Whatsapp**: per fissare le riunioni
- **Discord**: per prendere discutere eventuali problemi e definire delle azioni da intraprendere di conseguenza

Per le comunicazioni formali con il proponente o altre parti interessate, si utilizzano principalmente:

- **Email**: per fissare riunioni o per lo scambio di risorse e documenti
- **Zoom**: per ottenere feedback sull'avanzamento del progetto e discutere eventuali dubbi

4.2 Strumenti

Per la gestione dei rischi sono stati utilizzati i seguenti strumenti:

- **GitHub**: per la gestione delle issue e dei pull request
- **Typst**: per la stesura dei documenti
- **Discord**: per le comunicazioni interne
- **WhatsApp**: per le comunicazioni informali e rapide
- **Zoom**: per le riunioni online
- **Visual Studio Code**: per la scrittura del codice sorgente

4.3 Formazione del Personale

I componenti di NextSoft si diveranno in due macrogruppi, uno dedito alla ricerca delle tecnologie necessarie per il Capitolato e l'altro dedito al processo di stesura della documentazione e alla discussione delle scelte progettuali proposte. L'assegnazione ad un macrogruppo avviene in base alle preferenze dei singoli membri e sulle eventuali competenze pregresse sui requisiti obbligatori del capitolato. Tale approccio permette una divisione del lavoro equa e una maggiore efficienza nella ricerca delle tecnologie, mantenendo comunque un alto ritmo di tracciabilità delle scelte progettuali.